

**Oracle® Application Server 10g**

High Availability Guide

10g (10.1.2)

**Part No. B14003-01**

December 2004

Oracle Application Server 10g High Availability Guide 10g (10.1.2)

Part No. B14003-01

Copyright © 2004, Oracle. All rights reserved.

Contributing Author: Thomas Van Raalte, Rodney Ward, Richard Delval, Xiang Liu, Kai Li

Contributor: Pradeep Bhat, Fermin Castro, Jay Feenan, Luk Ho, Susan Kornberg, Paul Mackin, David Rowlands, Shari Yamaguchi

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States 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, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

---

---

# Contents

<b>Send Us Your Comments</b> .....	xiii
<b>Preface</b> .....	xv
Intended Audience .....	xv
Documentation Accessibility .....	xv
Organization .....	xvi
Related Documents .....	xvii
Conventions .....	xvii
<b>Part I Overview</b>	
<b>1 Introduction</b>	
1.1 What is High Availability .....	1-1
1.1.1 High Availability Problems.....	1-1
1.1.2 High Availability Solutions.....	1-2
1.2 Oracle Application Server High Availability Concepts .....	1-4
1.2.1 Terminology .....	1-4
1.2.2 Oracle Application Server Base Architecture .....	1-8
1.2.3 Oracle Application Server High Availability Architectures .....	1-11
1.2.4 Choosing the Best High Availability Architecture .....	1-12
1.3 Organization of this Guide .....	1-13
1.4 High Availability Information in Other Documentation.....	1-13
<b>2 Oracle Application Server High Availability Framework</b>	
2.1 Redundant Architectures.....	2-1
2.1.1 Oracle Application Server Active-Active Configurations: Oracle Application Server Clusters .....	2-1
2.1.2 Oracle Application Server Active-Passive Configurations: Oracle Application Server Cold Failover Clusters .....	2-2
2.2 High Availability Services in Oracle Application Server.....	2-3
2.2.1 Process Death Detection and Automatic Restart .....	2-4
2.2.1.1 Process Management with Oracle Process Manager and Notification Server ....	2-4
2.2.1.1.1 Automated Process Management with OPMN .....	2-4
2.2.1.1.2 Distributed Process Control with OPMN.....	2-5
2.2.2 Configuration Management.....	2-5

2.2.2.1	Configuration Management with Distributed Configuration Management .....	2-5
2.2.2.1.1	Configuration Synchronization and Management with DCM.....	2-6
2.2.2.1.2	Distributed Application Deployment with DCM.....	2-6
2.2.3	State Replication.....	2-6
2.2.4	Server Load Balancing and Failover .....	2-7
2.2.4.1	Internal Load Balancing Mechanism Provided in Oracle Application Server ....	2-7
2.2.4.2	External Load Balancers .....	2-8
2.2.5	Backup and recovery.....	2-10
2.2.5.1	Oracle Application Server Backup and Recovery Tool.....	2-10
2.2.6	Disaster Recovery .....	2-10
2.2.6.1	Oracle Application Server Guard.....	2-11

## Part II Middle-tier High Availability

### 3 Middle-tier High Availability

3.1	Redundancy .....	3-1
3.1.1	Active-Active .....	3-1
3.1.1.1	Oracle Application Server Web Cache Sub-Tier .....	3-2
3.1.1.2	Oracle HTTP Server Sub-Tier .....	3-3
3.1.1.2.1	Oracle HTTP Server High Availability Summary .....	3-4
3.1.1.2.2	OC4J Load Balancing Using mod_oc4j .....	3-4
3.1.1.2.3	Database Load Balancing with mod_plsql .....	3-6
3.1.1.3	Oracle Application Server Containers for J2EE Sub-Tier .....	3-7
3.1.1.3.1	OracleAS Cluster (OC4J) .....	3-7
3.1.1.3.2	OC4J Distributed Caching Using Java Object Cache .....	3-11
3.1.1.3.3	JMS High Availability.....	3-11
3.1.2	Active-Passive .....	3-15
3.1.2.1	Oracle Application Server Cold Failover Cluster (Middle-Tier) .....	3-15
3.1.2.1.1	Managing Failover .....	3-15
3.1.2.1.2	OracleAS JMS in an OracleAS Cold Failover Cluster (Middle-Tier) Environment .....	3-16
3.2	Highly Available Middle-tier Configuration Management Concepts .....	3-16
3.2.1	Oracle Application Server Clusters Managed Using DCM.....	3-17
3.2.1.1	What is a DCM-Managed OracleAS Cluster? .....	3-17
3.2.1.2	Oracle Application Server DCM Configuration Repository Types .....	3-18
3.2.2	Manually Managed Oracle Application Server Clusters .....	3-19
3.3	Middle-tier Backup and Recovery Considerations.....	3-19
3.4	Middle-tier Applications.....	3-20
3.4.1	Oracle Application Server Portal .....	3-20
3.4.1.1	Enabling Redundancy for OracleAS Portal .....	3-22
3.4.1.2	Configuring Load Balancer Routers for OracleAS Portal.....	3-22
3.4.1.3	Session Binding for Web Clipping Portlet .....	3-23
3.4.1.4	OracleAS Portal and OracleAS Web Cache .....	3-23
3.4.2	Oracle Application Server Wireless .....	3-24
3.4.2.1	OracleAS Wireless Clustering Architecture .....	3-24
3.4.3	OracleAS Integration B2B.....	3-24
3.4.4	Oracle Application Server Integration InterConnect.....	3-27

3.4.5	Oracle Business Intelligence Discoverer.....	3-34
3.4.5.1	Oracle Business Intelligence Discoverer Preferences Server.....	3-34

## 4 Managing and Operating Middle-tier High Availability

4.1	Middle-tier High Availability Configuration Overview .....	4-1
4.1.1	DCM-Managed Oracle Application Server Clusters .....	4-1
4.1.2	Manually Managed Oracle Application Server Clusters .....	4-2
4.2	Using DCM-Managed OracleAS Clusters.....	4-2
4.2.1	Creating DCM-Managed OracleAS Clusters.....	4-2
4.2.1.1	Associating An Instance With An OracleAS Database-based Farm .....	4-3
4.2.1.2	Associating An Instance With An OracleAS File-based Farm.....	4-3
4.2.1.2.1	Creating An OracleAS File-based Farm Repository Host .....	4-3
4.2.1.2.2	Adding Instances To An OracleAS File-based Farm .....	4-5
4.2.1.3	Using the Application Server Control Console Create Cluster Page .....	4-6
4.2.2	Adding Instances To DCM-Managed OracleAS Clusters .....	4-7
4.2.3	Removing Instances from DCM-Managed OracleAS Clusters.....	4-9
4.2.4	Starting Stopping and Deleting DCM-Managed OracleAS Clusters .....	4-9
4.2.5	Configuring Oracle HTTP Server Options for DCM-Managed OracleAS Clusters	4-10
4.2.5.1	Using and Configuring <code>mod_oc4j</code> Load Balancing .....	4-10
4.2.5.2	Configuring Oracle HTTP Server Instance-Specific Parameters .....	4-12
4.2.5.3	Configuring <code>mod_plsql</code> With Real Application Clusters .....	4-12
4.2.5.3.1	Configuring Detection and Cleanup of Dead Connections .....	4-12
4.2.5.3.2	Using Oracle Directory for Lookups .....	4-13
4.2.6	Understanding DCM-Managed OracleAS Cluster Membership .....	4-13
4.2.6.1	How the Common Configuration is Established.....	4-13
4.2.6.2	Parameters Excluded from the Common Configuration: Instance-Specific Parameters .....	4-14
4.3	Availability Considerations for the DCM Configuration Repository .....	4-16
4.3.1	Availability Considerations for DCM-Managed OracleAS Cluster (Database) .....	4-16
4.3.2	Availability Considerations for DCM-Managed OracleAS Cluster (File-based) ....	4-16
4.3.2.1	Selecting the Instance to Use for a OracleAS File-based Farm Repository Host .....	4-17
4.3.2.2	Protecting Against The Loss of a Repository Host.....	4-17
4.3.2.3	Impact of Repository Host Unavailability .....	4-17
4.3.2.4	Impact of Non-Repository Host Unavailability .....	4-18
4.3.2.5	Updating and Checking the State of Local Configuration .....	4-18
4.3.2.6	Performing Administration on a DCM-Managed OracleAS Cluster.....	4-19
4.3.2.7	Best Practices for Repository Backups.....	4-21
4.3.2.8	Best Practices for Managing Instances In OracleAS File-based Farms .....	4-21
4.4	Using Oracle Application Server Clusters (OC4J) .....	4-21
4.4.1	Overview of OracleAS Cluster (OC4J) Configuration .....	4-22
4.4.2	Cluster-Wide Configuration Changes and Modifying OC4J Instances .....	4-23
4.4.2.1	Creating or Deleting OC4J Instances In An OracleAS Cluster (OC4J) .....	4-23
4.4.2.2	Deploying Applications On An OracleAS Cluster (OC4J) .....	4-24
4.4.2.3	Configuring Web Application State Replication With OracleAS Cluster (OC4J) .....	4-24

4.4.2.4	Configuring EJB Application State Replication With OracleAS Cluster (OC4J-EJB) .....	4-26
4.4.2.5	Configuring Stateful Session Bean Replication for OracleAS Cluster (OC4J-EJB)s .....	4-27
4.4.2.5.1	End of Call Replication .....	4-28
4.4.2.5.2	JVM Termination Replication.....	4-28
4.4.3	Configuring OC4J Instance-Specific Parameters.....	4-28
4.4.3.1	Configuring OC4J Islands and OC4J Processes .....	4-28
4.4.3.2	Configuring Port Numbers and Command Line Options .....	4-29
4.5	Using Oracle Application Server Clusters (Portal) .....	4-30
4.6	Using Oracle Application Server Clusters (Web Cache).....	4-30
4.7	Managing OracleAS Cold Failover Cluster (Middle-Tier).....	4-31
4.7.1	Managing Configuration and Deployment for OracleAS Cold Failover Cluster (Middle-Tier) .....	4-31
4.7.1.1	Configuration and Deployment Changes for OracleAS Cold Failover Cluster (Middle-Tier) .....	4-31
4.7.1.2	Backup and Recovery for OracleAS Cold Failover Cluster (Middle-Tier).....	4-32
4.7.1.3	Using Application Server Control Console for OracleAS Cold Failover Cluster (Middle-Tier) .....	4-32
4.7.2	Managing Failover for OracleAS Cold Failover Cluster (Middle-Tier).....	4-32
4.7.2.1	Manual Failover for OracleAS Cold Failover Cluster (Middle-Tier).....	4-33
4.7.2.2	Manual Failover for the Virtual IP in OracleAS Cold Failover Cluster (Middle-Tier) .....	4-33
4.7.2.3	Manual Failover of Components for OracleAS Cold Failover Cluster (Middle-Tier) .....	4-34
4.7.2.4	Manual Failover of OracleAS Cluster (OC4J-JMS) .....	4-35
4.8	Managing Custom Processes With OPMN .....	4-35
4.9	Managing Oracle Application Server Middle-tier Upgrades .....	4-36
4.9.1	Upgrading Oracle Application Server Instances .....	4-36
4.9.2	Upgrading DCM-Managed OracleAS Clusters.....	4-37
4.9.3	Upgrading Stateful OC4J Applications .....	4-37
4.10	Using OracleAS Single Sign-On With OracleAS Cluster (Middle-Tier) .....	4-37

## Part III OracleAS Infrastructure High Availability

### 5 Oracle Application Server Infrastructure High Availability

5.1	Oracle Application Server Infrastructure Overview .....	5-1
5.2	Oracle Application Server Infrastructure Components .....	5-1
5.2.1	Oracle Application Server Metadata Repository .....	5-2
5.2.1.1	When to Use Oracle Application Server Metadata Repository .....	5-2
5.2.2	Oracle Identity Management .....	5-3
5.2.2.1	Oracle Internet Directory.....	5-3
5.2.2.2	Oracle Application Server Single Sign-On .....	5-3
5.2.2.3	Oracle Delegated Administration Services.....	5-4
5.2.2.4	Oracle Directory Integration and Provisioning.....	5-4
5.2.2.5	Oracle Application Server Certificate Authority .....	5-5
5.2.3	Oracle HTTP Server.....	5-5
5.2.4	Oracle Application Server Containers for J2EE (OC4J).....	5-5

5.2.5	Oracle Enterprise Manager 10g Application Server Control Console .....	5-6
5.3	High Availability Configurations for Infrastructure .....	5-6
5.3.1	Active-Active High Availability Solutions .....	5-8
5.3.1.1	OracleAS Cluster (Identity Management) .....	5-9
5.3.1.2	Distributed OracleAS Cluster (Identity Management) .....	5-11
5.3.2	Active-Passive High Availability Solutions.....	5-13
5.3.2.1	OracleAS Cold Failover Cluster (Infrastructure) .....	5-15
5.3.2.1.1	Installation .....	5-16
5.3.2.1.2	Runtime.....	5-17
5.3.2.1.3	Failover .....	5-17
5.3.2.1.4	Summary of High Availability Capabilities .....	5-19
5.3.2.1.5	OracleAS Cold Failover Cluster (Infrastructure) Solution for Windows Using Oracle Fail Safe.....	5-19
5.3.2.1.6	OracleAS Cold Failover Cluster (Infrastructure) with Co-located OracleAS Cold Failover Cluster (Middle-Tier) .....	5-21
5.3.2.2	Distributed OracleAS Cold Failover Cluster (Infrastructure).....	5-22
5.3.2.3	OracleAS Cold Failover Cluster (Identity Management) .....	5-25
5.3.2.4	Distributed OracleAS Cold Failover Cluster (Identity Management).....	5-27
5.4	OracleAS Infrastructure Backup and Recovery Considerations.....	5-29
5.4.1	OracleAS Cold Failover Cluster .....	5-29
5.4.2	Oracle Identity Management .....	5-29

## 6 Managing and Operating Infrastructure High Availability

6.1	Managing Oracle Application Server Cluster (Identity Management).....	6-1
6.1.1	Starting Stopping and Monitoring OracleAS Cluster (Identity Management) .....	6-1
6.1.1.1	Starting OracleAS Cluster (Identity Management) .....	6-2
6.1.1.2	Stopping OracleAS Cluster (Identity Management) .....	6-3
6.1.1.3	Monitoring OracleAS Cluster (Identity Management) .....	6-4
6.1.2	Configuring OracleAS Cluster (Identity Management).....	6-6
6.1.2.1	Changing Configuration Files For OracleAS Cluster (Identity Management)....	6-6
6.1.2.2	Configuring A Load Balancer For OracleAS Cluster (Identity Management) ....	6-7
6.1.2.2.1	Configuring A Load Balancer For Distributed OracleAS Cluster (Identity Management).....	6-8
6.1.3	Failover For OracleAS Cluster (Identity Management).....	6-8
6.1.4	Backup and Recovery For OracleAS Cluster (Identity Management).....	6-9
6.1.5	Using Application Server Control With OracleAS Cluster (Identity Management) .....	6-10
6.1.6	Additional Considerations For OracleAS Cluster (Identity Management) .....	6-11
6.2	Managing Oracle Application Server Cold Failover Cluster (Infrastructure) .....	6-11
6.2.1	Starting Stopping and Monitoring OracleAS Cold Failover Cluster (Infrastructure) .....	6-11
6.2.1.1	Starting OracleAS Cold Failover Cluster (Infrastructure) .....	6-11
6.2.1.2	Stopping OracleAS Cold Failover Cluster (Infrastructure) .....	6-13
6.2.1.3	Monitoring OracleAS Cold Failover Cluster (Infrastructure) .....	6-14
6.2.2	Configuring OracleAS Cold Failover Cluster (Infrastructure) .....	6-15
6.2.2.1	Changing Configuration For OracleAS Cold Failover Cluster (Infrastructure).....	6-15

6.2.2.2	Configuring Virtual IPs For OracleAS Cold Failover Cluster (Infrastructure)	6-15
6.2.3	Failover For OracleAS Cold Failover Cluster (Infrastructure).....	6-16
6.2.3.1	Failover For OracleAS Cold Failover Cluster (Infrastructure) For Solaris Systems.....	6-16
6.2.3.2	Failover For OracleAS Cold Failover Cluster (Infrastructure) For Windows Systems .....	6-17
6.2.3.3	Failover For OracleAS Cold Failover Cluster (Infrastructure) For Linux Systems .....	6-19
6.2.4	Backing Up and Recovering OracleAS Cold Failover Cluster (Infrastructure).....	6-21
6.2.5	Using Application Server Control With OracleAS Cold Failover Cluster (Infrastructure) .....	6-21
6.3	Managing Oracle Application Server Cold Failover Cluster (Identity Management) ..	6-22
6.3.1	Starting Stopping and Monitoring OracleAS Cold Failover Cluster (Identity Management).....	6-23
6.3.1.1	Starting OracleAS Cold Failover Cluster (Identity Management) .....	6-23
6.3.1.2	Stopping OracleAS Cold Failover Cluster (Identity Management) .....	6-24
6.3.1.3	Monitoring OracleAS Cold Failover Cluster (Identity Management).....	6-25
6.3.2	Configuring OracleAS Cold Failover Cluster (Identity Management).....	6-26
6.3.2.1	Changing Configuration For OracleAS Cold Failover Cluster (Identity Management) .....	6-26
6.3.2.2	Configuring Virtual IPs For OracleAS Cold Failover Cluster (Identity Management) .....	6-27
6.3.3	Failover For OracleAS Cold Failover Cluster (Identity Management).....	6-27
6.3.3.1	Failover For OracleAS Cold Failover Cluster (Identity Management) On UNIX Systems.....	6-27
6.3.3.2	Failover For OracleAS Cold Failover Cluster (Identity Management) On Linux Systems.....	6-28
6.3.4	Backup and Recovery For OracleAS Cold Failover Cluster (Identity Management).....	6-29

## Part IV Disaster Recovery

### 7 Oracle Application Server Disaster Recovery

7.1	Oracle Application Server 10g Disaster Recovery Solution.....	7-2
7.1.1	Requirements.....	7-3
7.1.2	Topology .....	7-4
7.2	Preparing the OracleAS Disaster Recovery Environment .....	7-6
7.2.1	Planning and Assigning Hostnames.....	7-6
7.2.1.1	Physical Hostnames .....	7-8
7.2.1.2	Network Hostnames .....	7-8
7.2.1.3	Virtual Hostname .....	7-9
7.2.2	Configuring Hostname Resolution .....	7-9
7.2.2.1	Using Local Hostnaming File Resolution .....	7-10
7.2.2.2	Using DNS Resolution .....	7-11
7.2.2.2.1	Additional DNS Server Entries for Oracle Data Guard.....	7-13
7.3	Overview of Installing Oracle Application Server 10g Software .....	7-14
7.4	Overview of OracleAS Guard and asgctl .....	7-15
7.4.1	Overview of asgctl.....	7-15



7.4.2	OracleAS Guard Client .....	7-16
7.4.3	OracleAS Guard Server.....	7-17
7.4.4	asgctl Operations .....	7-17
7.4.5	OracleAS Guard Integration with OPMN.....	7-17
7.4.6	Supported OracleAS Disaster Recovery Configurations.....	7-18
7.4.7	Supported Oracle Application Server Releases and Operating Systems .....	7-18
7.5	OracleAS Guard Operations -- Standby Instantiation and Standby Synchronization ..	7-18
7.5.1	Configuring OracleAS Guard and Other Relevant Information .....	7-19
7.5.2	Standby Instantiation .....	7-20
7.5.3	Standby Synchronization.....	7-22
7.6	Runtime Operations -- OracleAS Guard Switchover and Failover Operations.....	7-23
7.6.1	Outages.....	7-23
7.6.1.1	Scheduled Outages .....	7-23
7.6.1.1.1	Site Switchover Operations.....	7-24
7.6.1.2	Unplanned Outages .....	7-26
7.6.1.2.1	Site Failover Operations .....	7-27
7.6.2	Monitoring OracleAS Guard Operations and Troubleshooting.....	7-30
7.6.2.1	Verifying the Farm .....	7-31
7.6.2.2	Displaying the Current Operation .....	7-32
7.6.2.3	Displaying a List of Recent Operations.....	7-32
7.6.2.4	Stopping an Operation.....	7-32
7.6.2.5	Tracing Tasks .....	7-33
7.6.2.6	Writing Information About the Farm to a File.....	7-33
7.6.2.7	Shutting Down a Farm with a Problem .....	7-33
7.6.2.8	Starting Up a Shutdown Farm.....	7-33
7.6.2.9	Error Messages.....	7-34
7.7	Wide Area DNS Operations .....	7-34
7.7.1	Using a Wide Area Load Balancer .....	7-34
7.7.2	Manually Changing DNS Names.....	7-34
7.8	Using OracleAS Guard Command-Line Utility (asgctl) .....	7-35
7.8.1	Typical OracleAS Guard Session Using asgctl .....	7-35
7.8.1.1	Invoking the OracleAS Guard Client and Connecting to the OracleAS Guard Server.....	7-36
7.8.1.2	Getting Help .....	7-36
7.8.1.3	Specifying the Primary Database .....	7-36
7.8.1.4	Verifying the Farm .....	7-37
7.8.1.5	Instantiating the Farm at the Secondary Site.....	7-37
7.8.1.6	Synchronizing the Secondary Site with the Primary Site .....	7-39
7.8.1.7	Disconnecting from the OracleAS Guard Server and Exiting the OracleAS Guard Client .....	7-40
7.8.1.8	Creating and Executing an asgctl Script .....	7-40
7.8.2	Periodic Scheduling of OracleAS Guard asgctl Scripts.....	7-40
7.8.3	Submitting OracleAS Guard Jobs to the Enterprise Manager Job System .....	7-40
7.8.4	Specifying Different Credentials for Two or More Nodes .....	7-41
7.8.4.1	Setting asgctl Credentials .....	7-41
7.8.4.2	Specifying the Primary Database .....	7-41
7.8.5	Reference Section: OracleAS Guard asgctl Command-line Commands .....	7-42

asgctl .....	7-44
connect asg .....	7-45
disconnect.....	7-46
dump farm .....	7-47
exit.....	7-49
failover.....	7-50
help.....	7-52
instantiate farm to .....	7-53
quit .....	7-55
set asg credentials .....	7-56
set echo .....	7-57
set new primary database.....	7-58
set primary database.....	7-59
set trace .....	7-60
show operation.....	7-61
shutdown farm .....	7-63
startup farm .....	7-64
stop operation.....	7-65
switchover farm to .....	7-66
sync farm to.....	7-69
verify farm.....	7-71

## Part V Appendices

### A Troubleshooting High Availability

A.1	Problems and Solutions .....	A-1
A.1.1	Cluster Configuration Assistant Fails During Installation .....	A-2
A.1.2	Oracle Ultra Search Configuration Assistant is Unable to Connect to Oracle Internet Directory During High Availability Infrastructure Installation .....	A-2
A.1.3	Unable to Perform Online Database Backup and Restore in OracleAS Cold Failover Cluster Environment .....	A-3
A.1.4	<b>odisrv</b> Process Does Not Failover .....	A-3
A.1.5	Oracle Ultra Search Web Crawler Does Not Failover .....	A-4
A.1.6	Unable to Restore OracleAS Metadata Repository to a Different Host .....	A-5
A.1.7	Cannot Connect to Database for Restoration (Windows) .....	A-5
A.1.8	Unpredictable Behavior from Oracle Application Server Cluster (Identity Management) Configuration.....	A-6
A.1.9	Wrong Name Specified for Load Balancer .....	A-6
A.1.10	OracleAS Disaster Recovery: Standby Site Not Synchronized .....	A-8
A.1.11	OracleAS Disaster Recovery: Failure to Bring Up Standby Instances After Failover or Switchover .....	A-8
A.1.12	OracleAS Disaster Recovery: Unable to Start Standalone OracleAS Web Cache Installations at the Standby Site .....	A-8
A.1.13	OracleAS Disaster Recovery: Standby Site Middle-tier Installation Uses Wrong Hostname .....	A-9

A.1.14	OracleAS Disaster Recovery: Failure of Farm Verification Operation with Standby Farm .....	A-9
A.1.15	OracleAS Disaster Recovery: Sync Farm Operation Returns Error Message .....	A-10
A.2	Need More Help? .....	A-11

## **B Manually Managed Oracle Application Server Cluster**

B.1	Overview of Manually Managed OracleAS Cluster .....	B-1
B.1.1	Oracle Application Server Manually Managed Clusters .....	B-2
B.1.2	What Are Manually Managed OracleAS Clusters? .....	B-2
B.1.3	When Do I Need To Use A Manually Managed OracleAS Cluster.....	B-3
B.1.3.1	No Database Requirement for Manually Managed OracleAS Cluster .....	B-3
B.1.3.2	Tiered Deployment Requirement for Manually Managed OracleAS Cluster ....	B-3
B.1.3.3	Tiered Deployment With Security Requirement .....	B-4
B.2	Configuring Manually Managed OracleAS Cluster .....	B-4
B.2.1	Associating Oracle Application Server Instances Together .....	B-5
B.2.2	Configuring OC4J Instances for State Replication .....	B-6
B.2.2.1	Configuring State Replication for Web Applications.....	B-6
B.2.2.2	Configuring State Replication for EJB Applications.....	B-6
B.2.3	Configuring the J2EE Application Properties.....	B-7
B.2.4	Configuring Oracle HTTP Server for Failover and Load Balancing .....	B-8
B.2.4.1	Understanding mod_oc4j Request Routing.....	B-8
B.2.4.2	Identifying the Instance Names.....	B-8
B.2.4.3	Configuring mod_oc4j Request Routing.....	B-9

## **C OracleAS Guard Error Messages**

C.1	DGA Error Messages .....	C-1
C.1.1	LRO Error Messages.....	C-2
C.1.2	Undo Error Messages .....	C-3
C.1.3	Create Template Error Messages.....	C-3
C.1.4	Switchover Physical Standby Error Messages.....	C-3
C.2	Duf Error Messages .....	C-4
C.2.1	Database Error Messages.....	C-9
C.2.2	Connection and Network Error Messages .....	C-13
C.2.3	SQL*Plus Error Messages .....	C-15
C.2.4	JDBC Error Messages .....	C-15
C.2.5	OPMN Error Messages .....	C-16
C.2.6	Net Services Error Messages .....	C-16
C.2.7	System Error Messages .....	C-18
C.2.8	Warning Error Messages .....	C-19
C.2.9	OracleAS Database Error Messages .....	C-19
C.2.10	OracleAS Farm Error Messages.....	C-20
C.2.11	OracleAS Backup and Restore Error Messages.....	C-21
C.2.12	OracleAS Guard Synchronize Error Messages.....	C-23
C.2.13	OracleAS Guard Instantiate Error Messages .....	C-23

## **D Manual Sync Operations**

D.1	Manually Synchronizing Baseline Installation with Standby Site Without Using OracleAS Guard asgctl Command-line Utility .....	D-1
D.1.1	Manually Backing Up the Production Site.....	D-2
D.1.1.1	Shipping OracleAS Infrastructure Database Archive Logs.....	D-3
D.1.1.2	Backing Up Configuration Files (OracleAS Infrastructure and Middle Tier) ....	D-3
D.1.2	Manually Restoring to Standby Site.....	D-4
D.1.2.1	Restoring Configuration Files (OracleAS Infrastructure and Middle Tier) .....	D-4
D.1.2.2	Restoring the OracleAS Infrastructure Database - Applying Log Files .....	D-5

## **E Setting Up a DNS Server**

## **F Secure Shell (SSH) Port Forwarding**

F.1	SSH Port Forwarding .....	F-1
-----	---------------------------	-----

## **Index**

---

---

# Send Us Your Comments

## **Oracle Application Server 10g High Availability Guide, 10g (10.1.2)**

**Part No. B14003-01**

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: [appserverdocs\\_us@oracle.com](mailto:appserverdocs_us@oracle.com)
- FAX: 650-506-7375 Attn: Oracle Application Server Documentation Manager
- Postal service:

Oracle Corporation  
Oracle Application Server Documentation  
500 Oracle Parkway, M/S 10p6  
Redwood Shores, CA 94065  
USA

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

If you have problems with the software, please contact your local Oracle Support Services.



---

---

# Preface

This preface contains these topics:

- Intended Audience
- Documentation Accessibility
- Organization
- Related Documents
- Conventions

## Intended Audience

*Oracle Application Server High Availability Guide* is intended for administrators, developers, and others whose role is to deploy and manage Oracle Application Server 10g with high availability requirements.

## Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at

<http://www.oracle.com/accessibility/>

### **Accessibility of Code Examples in Documentation**

JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

### **Accessibility of Links to External Web Sites in Documentation**

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

# Organization

The following chapters and appendices make up this guide:

## **Chapter 1, "Introduction"**

This chapter provides an introduction to high availability with Oracle Application Server 10g.

## **Chapter 2, "Oracle Application Server High Availability Framework"**

This chapter provides a conceptual explanation of high availability for each tier in Oracle Application Server, and how high availability can be achieved for the entire product as a result.

## **Chapter 3, "Middle-tier High Availability"**

This chapter provides a description of high availability in the Oracle Application Server 10g middle tier.

## **Chapter 4, "Managing and Operating Middle-tier High Availability"**

This chapter provides instructions to manage and operate the middle-tier high availability environment.

## **Chapter 5, "Oracle Application Server Infrastructure High Availability"**

This chapter describes the high availability solutions available for the Oracle Application Server Infrastructure.

## **Chapter 6, "Managing and Operating Infrastructure High Availability"**

This chapter provides instructions to set up and manage the Infrastructure high availability solutions.

## **Chapter 7, "Oracle Application Server Disaster Recovery"**

This chapter describes the disaster recovery solution for OracleAS. The solution covers both the middle and Infrastructure tiers.

## **Appendix A, "Troubleshooting High Availability"**

This appendix provides a list of possible high availability problems and their solutions.

## **Appendix B, "Manually Managed Oracle Application Server Cluster"**

This appendix provides instructions to configure and operate manually managed Oracle Application Server Cluster.

## **Appendix C, "OracleAS Guard Error Messages"**

This appendix contains a comprehensive list of Oracle Application Server Guard error messages and their descriptions.

## **Appendix D, "Manual Sync Operations"**

This appendix contains instructions for manually synchronizing the Oracle Application Server Disaster Recovery active and passive sites.



## Appendix E, "Setting Up a DNS Server"

This appendix provides instructions for setting up a DNS server relevant to the Oracle Application Server Disaster Recovery solution.

## Appendix F, "Secure Shell (SSH) Port Forwarding"

This appendix describes how secure shell (SSH) port forwarding may be used with Oracle Data Guard.

## Related Documents

For more information, see these Oracle resources:

- Oracle Application Server Documentation Library
- Oracle Application Server Platform-Specific Documentation on Oracle Application Server Disk 1

Printed documentation is available for sale in the Oracle Store at

<http://oraclestore.oracle.com/>

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

<http://otn.oracle.com/membership/>

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

<http://otn.oracle.com/documentation/>

## Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- Conventions in Text
- Conventions in Code Examples
- Conventions for Microsoft Windows Operating Systems

### Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
<b>Bold</b>	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an <b>index-organized table</b> .
<i>Italics</i>	Italic typeface indicates book titles or emphasis.	<i>Oracle9i Database Concepts</i> Ensure that the recovery catalog and target database do <i>not</i> reside on the same disk.

Convention	Meaning	Example
UPPERCASE monospace (fixed-width) font	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles.	You can specify this clause only for a NUMBER column.  You can back up the database by using the BACKUP command.  Query the TABLE_NAME column in the USER_TABLES data dictionary view.  Use the DBMS_STATS.GENERATE_STATS procedure.
lowercase monospace (fixed-width) font	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values.  <b>Note:</b> Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter sqlplus to open SQL*Plus.  The password is specified in the orapwd file.  Back up the datafiles and control files in the /disk1/oracle/dbs directory.  The department_id, department_name, and location_id columns are in the hr.departments table.  Set the QUERY_REWRITE_ENABLED initialization parameter to true.  Connect as oe user.  The JRepUtil class implements these methods.
lowercase italic monospace (fixed-width) font	Lowercase italic monospace font represents placeholders or variables.	You can specify the <i>parallel_clause</i> .  Run <i>Uold_release</i> .SQL where <i>old_release</i> refers to the release you installed prior to upgrading.

## Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL\*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[ ]	Brackets enclose one or more optional items. Do not enter the brackets.	DECIMAL ( <i>digits</i> [ , <i>precision</i> ])
{ }	Braces enclose two or more items, one of which is required. Do not enter the braces.	{ENABLE   DISABLE}
	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	{ENABLE   DISABLE} [COMPRESS   NOCOMPRESS]
...	Horizontal ellipsis points indicate either: <ul style="list-style-type: none"> <li>That we have omitted parts of the code that are not directly related to the example</li> <li>That you can repeat a portion of the code</li> </ul>	CREATE TABLE ... AS <i>subquery</i> ;  SELECT <i>col1</i> , <i>col2</i> , ... , <i>coln</i> FROM employees;

Convention	Meaning	Example
.	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	
Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	<pre>acctbal NUMBER(11,2); acct    CONSTANT NUMBER(4) := 3;</pre>
<i>Italics</i>	Italicized text indicates placeholders or variables for which you must supply particular values.	<pre>CONNECT SYSTEM/<i>system_password</i> DB_NAME = <i>database_name</i></pre>
UPPERCASE	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.	<pre>SELECT last_name, employee_id FROM employees; SELECT * FROM USER_TABLES; DROP TABLE hr.employees;</pre>
lowercase	Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files.  <b>Note:</b> Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	<pre>SELECT last_name, employee_id FROM employees; sqlplus hr/hr CREATE USER mjones IDENTIFIED BY ty3MU9;</pre>

## Conventions for Microsoft Windows Operating Systems

The following table describes conventions for Microsoft Windows operating systems and provides examples of their use.

Convention	Meaning	Example
Choose <b>Start</b> > <i>menu item</i>	How to start a program.	To start the Database Configuration Assistant, choose <b>Start</b> > <b>Programs</b> > <b>Oracle - HOME_NAME</b> > <b>Configuration and Migration Tools</b> > <b>Database Configuration Assistant</b> .
File and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe ( ), and dash (-). The special character backslash (\) is treated as an element separator, even when it appears in quotes. If the filename begins with \\, then Windows assumes it uses the Universal Naming Convention.	c:\winnt\ "system32 is the same as C:\WINNT\SYSTEM32
C:\>	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the <i>command prompt</i> in this manual.	C:\oracle\oradata>

Convention	Meaning	Example
Special characters	The backslash (\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.	C:\> exp HR/HR TABLES=emp QUERY=\"WHERE job='REP'\"
HOME_NAME	Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.	C:\> net start OracleHOME_NAME_TNSListener
ORACLE_HOME and ORACLE_BASE	<p>In releases prior to Oracle8i release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level ORACLE_HOME directory. The default for Windows NT was C:\orant.</p> <p>This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level ORACLE_HOME directory. There is a top level directory called ORACLE_BASE that by default is C:\oracle\product\10.1.0. If you install the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is C:\oracle\product\10.1.0\db_n, where n is the latest Oracle home number. The Oracle home directory is located directly under ORACLE_BASE.</p> <p>All directory path examples in this guide follow OFA conventions.</p> <p>Refer to <i>Oracle Database Installation Guide for Windows</i> for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.</p>	Go to the ORACLE_BASE\ORACLE_HOME\rdbms\admin directory.

# Part I

---

## Overview

The chapters in this part provide an introduction to Oracle Application Server high availability. These chapters are:

- Chapter 1, "Introduction"
- Chapter 2, "Oracle Application Server High Availability Framework"



---

---

# Introduction

In this release of Oracle Application Server 10g (10.1.2), work has been done to improve and extend the high availability solutions for Oracle Application Server. New high availability solutions for Oracle Application Server have been tested and are described in this book. All of these solutions seek to ensure that applications that you deploy on Oracle Application Server meet the required availability to achieve your business goals. The solutions and procedures described in this book seek to eliminate single points of failure of any Oracle Application Server components with no or minimal outage in service.

This chapter explains high availability and its importance from the perspective of Oracle Application Server.

## 1.1 What is High Availability

This section provides an overview of high availability from a problem-solution perspective. It has the sections:

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

### 1.1.1 High Availability Problems

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

**Table 1-1** *Availability percentages and corresponding downtime values*

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

Table 1-2 depicts the various types of failures that are possible with a computer system.

**Table 1–2 System downtime and failure types**

Downtime Type	Failure Type
Unplanned downtime	System failure
	Data failure
	Disasters
	Human error
Planned downtime	System maintenance <sup>1</sup>
	Data maintenance

<sup>1</sup> Includes hardware and/or software changes (operating system, application server, configuration, application changes).

## 1.1.2 High Availability Solutions

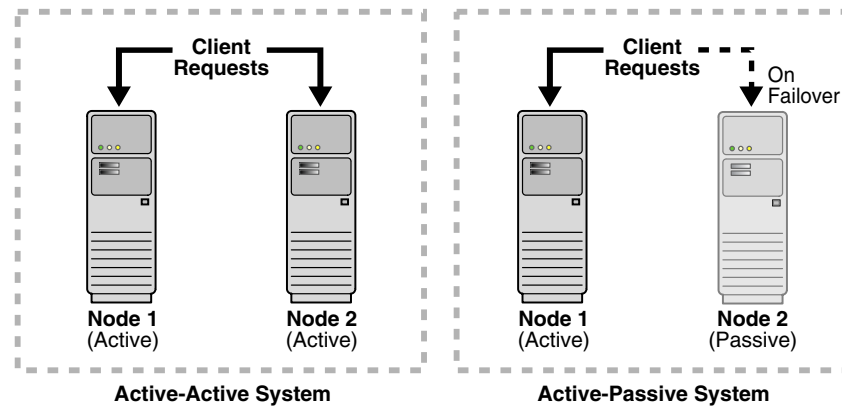
High availability solutions can be categorized into local high availability solutions that provide high availability in a single data center deployment, and disaster recovery solutions, which are usually geographically distributed deployments that protect your applications from disasters such as floods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

Amongst possible types of failures, process, node, and media failures as well as human errors can be protected by local high availability solutions. Local physical disasters can be protected by geographically distributed disaster recovery solutions.

To solve the high availability problem, a number of technologies and best practices are needed. The most important mechanism is redundancy. High availability comes from redundant systems and components. Local high availability solutions can be categorized, by their level of redundancy, into active-active solutions and active-passive solutions (see Figure 1–1). Active-active solutions deploy two or more active system instances and can be used to improve scalability as well as provide high availability. All instances handle requests concurrently.

Active-passive solutions deploy an active instance that handles requests and a passive instance that is on standby. In addition, a heartbeat mechanism is set up between these two instances. This mechanism is provided and managed through vendor-specific clusterware. Generally, vendor-specific cluster agents are also available to automatically monitor and failover between cluster nodes, so that when the active instance fails, an agent shuts down the active instance completely, brings up the passive instance, and application services can successfully resume processing. As a result, the active-passive roles are now switched. The same procedure can be done manually for planned or unplanned down time. Active-passive solutions are also generally referred to as cold failover clusters. For a list of available vendor-specific cluster agents for Oracle Application Server active-passive solutions, see *Oracle Technology Network*.



**Figure 1–1 Active-active and active-passive high availability solutions**

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

- **Process death detection and automatic restart**

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

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

- **Clustering**

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

- **Configuration management**

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

- **State replication and routing**

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

- **Server load balancing and failover**

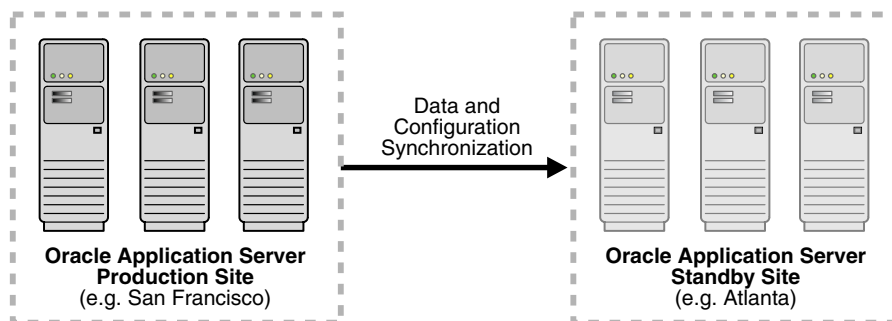
When multiple instances of identical server components are available, client requests to these components can be load balanced to ensure that the instances have roughly the same workload. With a load balancing mechanism in place, the instances are redundant. If any of the instances fail, requests to the failed instance can be sent to the surviving instances.

- **Backup and recovery**

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

Disaster recovery solutions typically set up two homogeneous sites, one active and one passive. Each site is a self-contained system. The active site is generally called the production site, and the passive site is called the standby site. During normal operation, the production site services requests; in the event of a site failover or switchover, the standby site takes over the production role and all requests are routed to that site. To maintain the standby site for failover, not only must the standby site contain homogeneous installations and applications, data and configurations must also be synchronized constantly from the production site to the standby site.

**Figure 1–2 Geographically distributed disaster recovery**



## 1.2 Oracle Application Server High Availability Concepts

An overview of high availability for Oracle Application Server is presented in the following sections:

- Section 1.2.1, "Terminology"
- Section 1.2.2, "Oracle Application Server Base Architecture"
- Section 1.2.3, "Oracle Application Server High Availability Architectures"
- Section 1.2.4, "Choosing the Best High Availability Architecture"

### 1.2.1 Terminology

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

- **active-active:** In a high availability system, the equivalent members of that system can be servicing requests concurrently. Under normal operation where none of the members have failed, all equivalent members are active and none are on standby. This is called an active-active system.
- **active-passive:** In a high availability system, some members of the system can be actively servicing requests and performing work, while other members can be inactive. These inactive members are known to be passive. They are not activated until one or more of the active nodes have failed. Consumers of services provided by the system may or may not notice the failure. An active-active system generally provides more transparency and options for scalability to consumers than an active-passive system.

- **failover:** When a member of a highly available system fails unexpectedly (unplanned downtime), in order to continue offering services to its consumers, the system undergoes a failover operation. If the system is an active-passive system, the passive member is activated during the failover operation and consumers are directed to it instead of the failed member. The failover process can be performed manually, or it can be automated by setting up hardware cluster services to detect failures and move cluster resources from the failed node to the standby node. If the system is an active-active system, the failover is performed by the load balancer entity serving requests to the active members. If an active member fails, the load balancer detects the failure and automatically redirects requests for the failed member to the surviving active members.
- **failback:** After a system undergoes a successful failover operation, the original failed member can be repaired over time and be re-introduced into the system as a standby member. If desired, a failback process can be initiated to activate this member and deactivate the other. This process reverts the system back to its pre-failure configuration.
- **hardware cluster:** A hardware cluster is a collection of computers that provides a single view of network services (for example: an IP address) or application services (for example: databases, Web servers) to clients of these services. Each node in a hardware cluster is a standalone server that runs its own processes. These processes can communicate with one another to form what looks like a single system that cooperatively provides applications, system resources, and data to users.

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

- **cluster agent:** The software that runs on a node member of a hardware cluster that coordinates availability and performance operations with other nodes. Clusterware provides resource grouping, monitoring, and the ability to move services. A cluster agent can automate the service failover.
- **clusterware:** A software that manages the operations of the members of a cluster as a system. It allows one to define a set of resources and services to monitor via a heartbeat mechanism between cluster members and to move these resources and services to a different member in the cluster as efficiently and transparently as possible.
- **shared storage:** Even though each hardware cluster node is a standalone server that runs its own set of processes, the storage subsystem required for any cluster-aware purpose is usually shared. Shared storage refers to the ability of the cluster to be able to access the same storage, usually disks, from both the nodes. While the nodes have equal access to the storage, only one node, the primary node, has active access to the storage at any given time. The hardware cluster's software grants the secondary node access to this storage if the primary node fails. For the OracleAS Infrastructure in the OracleAS Cold Failover Cluster environment, its ORACLE\_HOME is on such a shared storage file system. This file system is mounted by the primary node; if that node fails, the secondary node

takes over and mounts the file system. In some cases, the primary node may relinquish control of the shared storage, such as when the hardware cluster's software deems the Infrastructure as unusable from the primary node and decides to move it to the secondary.

- **primary node:** The node that is actively executing one or more Infrastructure installations at any given time. If this node fails, the Infrastructure is failed over to the secondary node. Since the primary node runs the active Infrastructure installation(s), it is considered the "hot" node. See the definition for "secondary node" in this section.
- **secondary node:** This is the node that takes over the execution of the Infrastructure if the primary node fails. Since the secondary node does not originally run the Infrastructure, it is considered the "cold" node. And, because the application fails from a hot node (primary) to a cold node (secondary), this type of failover is called cold failover. See the definition for "primary node" in this section.
- **network hostname:** Network hostname is a name assigned to an IP address either through the `/etc/hosts` file (in UNIX), `C:\WINDOWS\system32\drivers\etc\hosts` file (in Windows), or through DNS resolution. This name is visible in the network that the machine to which it refers to is connected. Often, the network hostname and physical hostname are identical. However, each machine has only one physical hostname but may have multiple network hostnames. Thus, a machine's network hostname may not always be its physical hostname.
- **Oracle Application Server Cluster:** An Oracle Application Server Cluster (OracleAS Cluster) is a collection of application server instances that are configured to serve the same application workload. With appropriate front-end load balancing, any instance in an Oracle Application Server Cluster can serve client requests.
- **Oracle Application Server Farm:** A Farm is a collection of application server instances that share the same Oracle Application Server Infrastructure (database-based repository) or that use the same application server instance for their file-based repository host. An Oracle Application Server Farm can include application server instances that are in an OracleAS Cluster, as well as those that are not.

**See Also:** *Oracle Application Server Concepts*

- **Oracle Application Server instance:** An Oracle Application Server instance (also called an application server instance and OracleAS instance) is the set of processes required to run the configured components within an application server installation. There can be only one application server instance per application server installation. The terms installation and instance are sometimes used interchangeably; however, note that an installation is the set of files installed into an Oracle home, while an instance is a set of processes associated with those files.
- **component instance:** A component instance is a set of binaries, configuration files, and an associated name identifier of those files. At runtime, a process is typically created for each component instance. Component instances include Oracle HTTP Server and multiple Oracle Application Server Containers for J2EE (OC4J) instances (see Chapter 3 for more details).
- **physical hostname:** For the purpose of discussion in this book, a differentiation is made between the terms physical hostname and network hostname. Physical hostname is used to refer to the "internal name" of the current machine. In UNIX, this is the name returned by the command `hostname`.

Physical hostname is used by Oracle Application Server middle-tier installation types to reference the local host. During installation, the installer automatically retrieves the physical hostname from the current machine and stores it in the Oracle Application Server configuration metadata on disk.

- **switchover:** During normal operation, active members of a system may require maintenance or upgrading. A switchover process can be initiated to allow a substitute member to take over the workload performed by the member that requires maintenance or upgrading, which undergoes planned downtime. The switchover operation ensures continued service to consumers of the system.
- **switchback:** When a switchover operation is performed, a member of the system is deactivated for maintenance or upgrading. When the maintenance or upgrading is completed, the system can undergo a switchback operation to activate the upgraded member and bring the system back to the pre-switchover configuration.
- **virtual hostname:** Virtual hostname is a network addressable hostname that maps to one or more physical machines via a load balancer or a hardware cluster. For load balancers, the name "virtual server name" is used interchangeably with virtual hostname in this book. A load balancer can hold a virtual hostname on behalf of a set of servers, and clients communicate indirectly with the machines using the virtual hostname. A virtual hostname in a hardware cluster is a network hostname assigned to a cluster virtual IP. Because the cluster virtual IP is not permanently attached to any particular node of a cluster, the virtual hostname is not permanently attached to any particular node either.

You can specify a virtual hostname for Oracle Application Server Infrastructure during installation in the Specify Virtual Hostname screen of the Oracle Application Server installer. This OracleAS Infrastructure virtual hostname can be managed by a hardware cluster or a load balancer and is used by the middle-tier and OracleAS Infrastructure components to access the OracleAS Infrastructure. This is regardless of whether the OracleAS Infrastructure is in a single node installation, in the OracleAS Cold Failover Cluster solution, or in the OracleAS Cluster solution.

The virtual hostname is the hostname associated with the virtual IP. This is the name that is chosen to give the OracleAS middle-tier a single system view of the Infrastructure with the help of a hardware cluster or a server load balancer. This name-IP entry must be added to the DNS that the site uses, so that the middle-tier nodes can associate with the Infrastructure without having to add this entry into their local `/etc/hosts` (or equivalent) file. For example, if the two physical hostnames of the hardware cluster are `node1.mycompany.com` and `node2.mycompany.com`, the single view of this cluster can be provided by the name `selfservice.mycompany.com`. In the DNS, `selfservice` maps to the virtual IP address of the Infrastructure, which either floats between `node1` and `node2` via a hardware cluster or maps to `node1` and `node2` by a server load balancer, all without the middle tier knowing which physical node is active and actually servicing a particular request.

**See Also:** Section 1.2.2, "Oracle Application Server Base Architecture" on page 1-8

You cannot specify a virtual hostname for OracleAS middle-tier installation, but you can still use a virtual hostname via a hardware cluster or a server load balancer by following the documented post-install configuration steps for cold failover cluster middle tiers. See the *Oracle Application Server Installation Guide*.

Thus, as far as installation is concerned, a virtual hostname applies only to the OracleAS Infrastructure host(s).

---

---

**Note:** Whenever the phrase "virtual hostname" is used in this document, it is assumed to be associated with a virtual IP address. In cases where just the IP address is needed or used, it will be explicitly stated.

---

---

- **virtual IP:** Also, cluster virtual IP and load balancer virtual IP. Generally, a virtual IP can be assigned to a hardware cluster or load balancer. To present a single system view of a cluster to network clients, a virtual IP serves as an entry point IP address to the group of servers which are members of the cluster. A virtual IP can be assigned to a server load balancer or a hardware cluster.

A hardware cluster uses a cluster virtual IP to present to the outside world the entry point into the cluster (it can also be set up on a standalone machine). The hardware cluster's software manages the movement of this IP address between the two physical nodes of the cluster while clients connect to this IP address without the need to know which physical node this IP address is currently active on. In a typical two-node hardware cluster configuration, each machine has its own physical IP address and physical hostname, while there could be several cluster IP addresses. These cluster IP addresses float or migrate between the two nodes. The node with current ownership of a cluster IP address is active for that address.

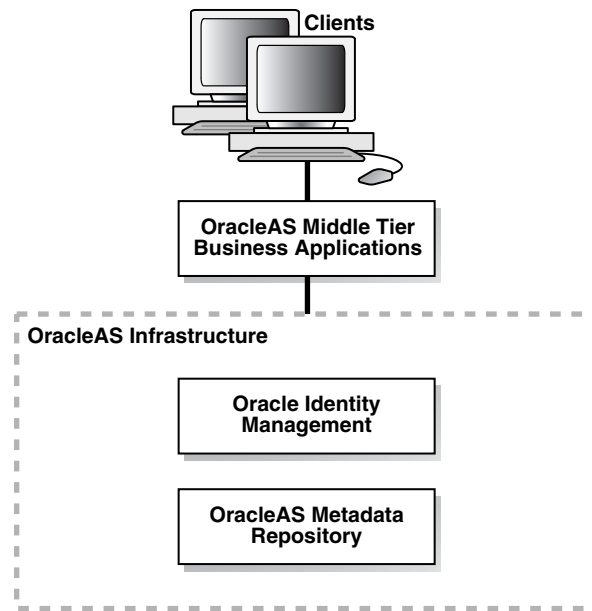
A load balancer also uses a virtual IP as the entry point to a set of servers. These servers tend to be active at the same time. This virtual IP address is not assigned to any individual server but to the load balancer which acts as a proxy between servers and their clients.

For a given Oracle Application Server Infrastructure installation, the virtual IP/virtual hostname associated with that installation is the IP address/hostname that is used by clients to connect to the OracleAS Infrastructure. The *Oracle Application Server Installation Guide* provides details on using virtual IPs and virtual hostnames for installations.

## 1.2.2 Oracle Application Server Base Architecture

The first thing to understand for high availability is the system's base architecture. Then, to make this system highly available, examine every component and connection path between components and make each one of them highly available. This produces a highly available architecture by essentially adding redundancy to the base architecture.

Figure 1–3 illustrates the base architecture of Oracle Application Server.

**Figure 1-3 Oracle Application Server base architecture**

At a high level, Oracle Application Server consists of the Oracle Application Server middle-tier business applications, Oracle Identity Management, and Oracle Application Server Metadata Repository. The latter two are part of the Oracle Application Server Infrastructure.

Oracle Identity Management software manages user authentication, authorization, and identity information. Functionally, its main components are:

- Oracle Application Server Single Sign-On
- Oracle Delegated Administration Services
- Oracle Internet Directory
- Oracle Directory Integration and Provisioning

Architecturally, Oracle Identity Management can be broken down into a Web server tier of Oracle HTTP Server, an OracleAS Single Sign-On/Oracle Delegated Administration Services middle-tier composed of an Oracle Application Server Containers for J2EE (OC4J) instance for these security applications, and an Oracle Internet Directory/Oracle Directory Integration and Provisioning tier at the back end. The OracleAS Metadata Repository is an Oracle database that manages configuration, management, and product metadata for components throughout the OracleAS Infrastructure and OracleAS middle-tier.

The middle tier hosts most of Oracle Application Server business applications, such as:

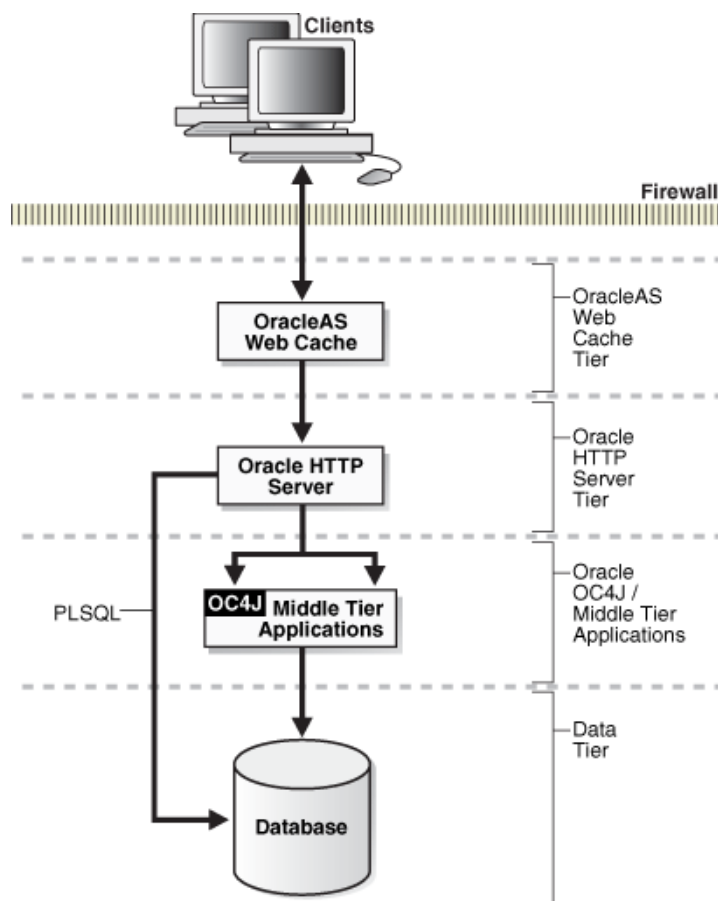
- Oracle Application Server Portal
- Oracle Application Server Wireless
- Oracle Application Server Integration

These applications rely on Oracle Identity Management and OracleAS Metadata Repository for security and metadata support. The middle tier also includes a Web caching sub-tier (Oracle Application Server Web Cache), a Web server sub-tier (Oracle HTTP Server), and OC4J instance(s). Behind the middle tier, the OracleAS Metadata Repository serves as the data tier. In actual deployments, other databases may also

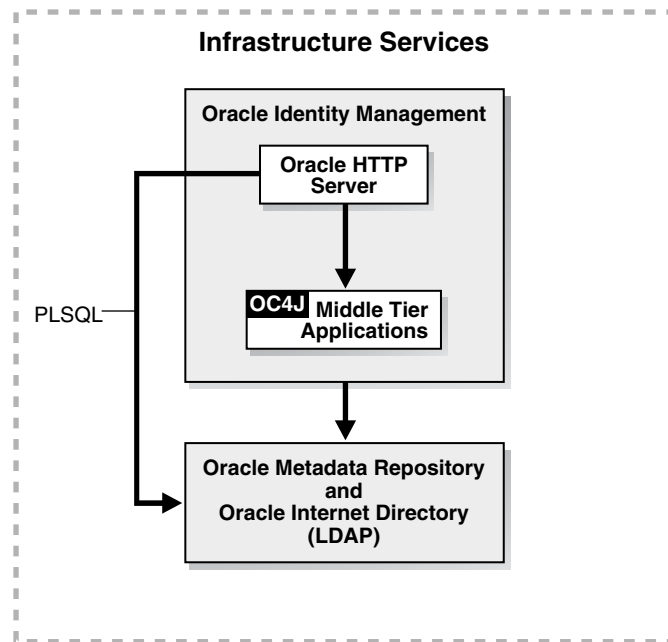
exist in the data tier (for example, a customer database for OC4J applications deployed on the middle tier).

Figure 1-4 shows the various sub-tiers that are traversed by client requests to the Oracle Application Server business applications and the Oracle Application Server Infrastructure services. An overall view of Infrastructure services is provided in Figure 1-5. These services include Oracle Identity Management, metadata repository, and LDAP services.

**Figure 1-4 Sub-tiers of the base architecture of Oracle Application Server**





**Figure 1–5 Overview of Infrastructure services**

**See Also:** *Oracle Application Server Concepts*

The base architecture supports many availability features, such as automatic process monitoring and restart, application server backup and recovery; however, it does not provide complete high availability. Several single points of failure exist. To eliminate them, redundancy has to be provided for each component. This can be achieved by extending the base architecture with additional high availability architectures.

### 1.2.3 Oracle Application Server High Availability Architectures

Oracle Application Server provides both local high availability and disaster recovery solutions for maximum protection against any kind of failure with flexible installation, deployment, and security options. The redundancy of Oracle Application Server local high availability and disaster recovery originates from its redundant high availability architectures.

At a high level, Oracle Application Server local high availability architectures include several active-active and active-passive architectures for the OracleAS middle-tier and the OracleAS Infrastructure. Although both types of solutions provide high availability, active-active solutions generally offer higher scalability and faster failover, although, they tend to be more expensive as well. With either the active-active or the active-passive category, multiple solutions exist that differ in ease of installation, cost, scalability, and security.

Building on top of the local high availability solutions is the Oracle Application Server Disaster Recovery solution, Oracle Application Server Guard. This unique solution combines the proven Oracle Data Guard technology in the Oracle Database with advanced disaster recovery technologies in the application realm to create a comprehensive disaster recovery solution for the entire application system. This solution requires homogenous production and standby sites, but other Oracle Application Server instances can be installed in either site as long as they do not interfere with the instances in the disaster recovery setup. Configurations and data must be synchronized regularly between the two sites to maintain homogeneity.

## 1.2.4 Choosing the Best High Availability Architecture

There is no single best high availability solution for all systems in the world, but there may be a best solution for your system. Perhaps the most important decision in designing a highly available system is choosing the most appropriate high availability architecture or type of redundancy based on service level requirements as needed by a business or application. Understanding the availability requirements of the business is critical since cost is also associated with the different levels of high availability.

Oracle Application Server offers many high availability solutions to meet service level requirements. The most comprehensive solution may not necessarily be the best for your application. To choose the correct high availability architecture, ensure you understand your business' service level requirements first.

The high level questions to determine your high availability architectures are:

1. Local high availability: does your production system need to be available 24 hours per day, 7 days per week, and 365 days per year?
2. Scalability: is the scalability of multiple active Oracle Application Server instances required?
3. Site-to-site disaster recovery: is this required?

Based on the answers to these questions, you need to make your selection in two dimensions:

1. Instance redundancy: base, active-active, or active-passive.
2. Site-to-site disaster recovery-enabled architecture: yes or no.

Table 1-3 shows the architecture choices based on business requirements.

**Table 1-3 Service level requirements and architecture choices**

Business Requirements			Architecture Choices	
Local High Availability	Scalability	Disaster Recovery	Instance Redundancy	Disaster Recovery
N	N	N	Base	N
Y	N	N	Active-passive	N
N	Y	N	Active-active	N
N	N	Y	Base	Y
Y	Y	N	Active-active	N
Y	N	Y	Active-passive	Y
N	Y	Y	Active-active (middle tier) Base (Infrastructure) <sup>1</sup>	Y
Y	Y	Y	Active-active (middle tier) Active-passive (Infrastructure) <sup>1</sup>	Y

<sup>1</sup> The Oracle Application Server Disaster Recovery solution in 10g (10.1.2) supports the base and active-passive Infrastructure architectures, but not an active-active architecture. For additional scalability in a base or active-passive architecture, extra computing power can be added to the infrastructure hardware (for example, high capacity CPUs, more memory).

Although you can choose different high availability architectures for your OracleAS middle-tier and OracleAS Infrastructure, their local high availability and disaster recovery requirements should be identical. Scalability requirements should be

evaluated separately for OracleAS middle-tier and OracleAS Infrastructure. The latter does not usually need to be as scalable as the middle tier because it handles fewer identity management requests.

Because of the differences in scalability requirements, deployment choices for the OracleAS middle-tier and the OracleAS Infrastructure may differ in architecture. For example, if your deployment requires local high availability, site-to-site disaster recovery, scalable middle tier but basic OracleAS Infrastructure scalability, you can choose an active-active middle tier, an active-passive OracleAS Infrastructure, and deploy a standby disaster recovery site that mirrors all middle-tier and OracleAS Infrastructure configuration in the production site.

## 1.3 Organization of this Guide

This guide has been organized into several chapters using the layers of the OracleAS middle-tier and OracleAS Infrastructure as a baseline. When the term "middle tier" is mentioned in this book, the reference is made generically to the Oracle Application Server middle-tier installation types.

Chapter 2 describes Oracle Application Server high availability functionalities that make all system components and applications highly available in Oracle Application Server. Understanding of this chapter is crucial to understanding the high availability of applications and services such as Oracle Application Server Portal, Oracle Identity Management, and even your own custom applications built on top of the Oracle Application Server platform.

Chapter 3 and Chapter 4 contain the description and configuration of the middle tier for high availability, respectively. In Chapter 3, we describe the specific application of high availability features introduced in Chapter 2, and then, focus on the supported redundant high availability architectures. This gives you a comparison to make the right architecture choices. In Chapter 4, we focus on managing and operating the Oracle Application Server middle-tier high availability solutions.

Chapter 5 and Chapter 6 have similar organization of information but for the Oracle Application Server Infrastructure. In Chapter 5, we describe the specific application of high availability features in the OracleAS Infrastructure, and then, focus on the supported OracleAS Infrastructure high availability architectures to help you make the right architecture choices. Chapter 6 documents how these OracleAS Infrastructure architectures are managed and operated.

Chapter 7 contains the setup and operational information for the site-to-site Oracle Application Server Disaster Recovery solution called Oracle Application Server Guard.

## 1.4 High Availability Information in Other Documentation

The following table provides a list of cross-references to high availability information in other documents in the Oracle library. This information mostly pertains to high availability of various Oracle Application Server components.

**Table 1–4 Cross-references to high availability information in Oracle documentation**

Component	Location of Information
Overall high availability concepts	In the high availability chapter of <i>Oracle Application Server Concepts</i> .
Oracle installer	In the chapter for installing in a high availability environment in <i>Oracle Application Server Installation Guide</i> .
Oracle Application Server Backup and Recovery Tool	In the backup and restore part of <i>Oracle Application Server Administrator's Guide</i> .

**Table 1–4 (Cont.) Cross-references to high availability information in Oracle documentation**

<b>Component</b>	<b>Location of Information</b>
Oracle Application Server Web Cache	<i>Oracle Application Server Web Cache Administrator's Guide</i>
Identity Management service replication	In "Advanced Configurations" chapter of <i>Oracle Application Server Single Sign-On Administrator's Guide</i> .
Identity Management high availability deployment	In "Directory Replication and High Availability" chapter of <i>Oracle Internet Directory Administrator's Guide</i> . In "Oracle Identity Management Deployment Planning" chapter of <i>Oracle Identity Management Concepts and Deployment Planning Guide</i> .
Database high availability	<i>Oracle High Availability Architecture and Best Practices</i>
Distributed Configuration Management commands	<i>Distributed Configuration Management Administrator's Guide</i>
Oracle Process Manager and Notification Server commands	<i>Oracle Process Manager and Notification Server Administrator's Guide</i>
OC4J high availability	<i>Oracle Application Server Containers for J2EE Services Guide</i> <i>Oracle Application Server Containers for J2EE User's Guide</i> <i>Oracle Application Server Containers for J2EE Enterprise JavaBeans Developer's Guide</i>
Java Object Cache	<i>Oracle Application Server Web Services Developer's Guide</i>
Load balancing to OC4J processes	<i>Oracle HTTP Server Administrator's Guide</i>
Oracle Application Server Wireless high availability	<i>Oracle Application Server Wireless Administrator's Guide</i>
Oracle Business Intelligence Discoverer high availability	<i>Oracle Business Intelligence Discoverer Configuration Guide</i>
Oracle Application Server Integration InterConnect ini file information	<i>Oracle Application Server Integration InterConnect User's Guide</i>

In addition, references to these and other documentation are noted in the text of this guide, where applicable.

---

# Oracle Application Server High Availability Framework

Whereas Chapter 1 provided an overview of high availability in general, this chapter introduces you to the specific sets of features, services, and environments that Oracle Application Server provides to ensure high availability for all its components and services. It contains the following sections:

- Section 2.1, "Redundant Architectures"
- Section 2.2, "High Availability Services in Oracle Application Server"

## 2.1 Redundant Architectures

Oracle Application Server provides redundancy by offering support for multiple instances supporting the same workload. These redundant configurations provide increased availability either through a distributed workload, through a failover setup, or both.

From the entry point to an Oracle Application Server system (content cache) to the back end layer (data sources), all the tiers that are crossed by a request can be configured in a redundant manner with Oracle Application Server. The configuration can be an active-active configuration using OracleAS Cluster or an active-passive configuration using OracleAS Cold Failover Cluster.

In the following sections, we describe the basics of these configurations:

- Section 2.1.1, "Oracle Application Server Active-Active Configurations: Oracle Application Server Clusters"
- Section 2.1.2, "Oracle Application Server Active-Passive Configurations: Oracle Application Server Cold Failover Clusters"

### 2.1.1 Oracle Application Server Active-Active Configurations: Oracle Application Server Clusters

Oracle Application Server provides an active-active redundant model for all its components with OracleAS Clusters. In an OracleAS Cluster, two or more Oracle Application Server instances are configured to serve the same application workload. These instances can reside on the same machine or on different machines.

The active instances may be front-ended by a load balancer router, which can redirect requests to any of the active instances, or by some other application-level configuration, such as address lists, to distribute the requests.

The most common properties of an OracleAS Cluster configuration include:

- Identical instance configuration  
The instances are meant to serve the same workload or application. Their configuration guarantees that they deliver the same exact reply to the same request. Some configuration properties may be identical and others may be instance-specific, such as local host name information.
- Managed collectively  
Changes made to one system will usually need to be propagated to the other systems in an active-active configuration.
- Operate independently  
In order to provide maximum availability, the loss of one Oracle Application Server instance in an active-active configuration should not affect the ability of the other instances to continue to serve requests.

The advantages of an OracleAS Cluster configuration include:

- Increased availability  
An active-active configuration is a redundant configuration. Loss of one instance can be tolerated because other instance can continue to serve the same requests.
- Increased Scalability and Performance  
Multiple identically-configured instances provide the capability to have a distributed workload shared among different machines and processes. If configured correctly, new instances can also be added as the demand of the application grows.

In general, the term OracleAS Cluster describes clustering at the Oracle Application Server instance level. However, if it is necessary to call out the specific type of instances being clustered, this document will use OracleAS Cluster (Type) to characterize the cluster solution. For example:

- two or more J2EE instances are known as Oracle Application Server Cluster (J2EE)
- two or more Oracle Application Server Portal instances are OracleAS Cluster or more specifically Oracle Application Server Cluster (Portal)
- two or more Oracle Identity Management instances are OracleAS Cluster or more specifically Oracle Application Server Cluster (Identity Management)

## 2.1.2 Oracle Application Server Active-Passive Configurations: Oracle Application Server Cold Failover Clusters

Oracle Application Server provides an active-passive model for all its components using Oracle Application Server Cold Failover Clusters. In an OracleAS Cold Failover Cluster configuration, two or more application server instances are configured to serve the same application workload but only one is active at any particular time. These instances can reside on the same machine or on different machines.

The most common properties of an OracleAS Cold Failover Cluster configuration include:

- Shared storage  
The passive Oracle Application Server instance in an active-passive configuration has access to the same Oracle binaries, configuration files, and data as the active instance.

- Failover procedure

An active-passive configuration also includes a set of scripts and procedures to detect failure of the Active instance and to failover to the Passive instance while minimizing downtime.

The advantages of an OracleAS Cold Failover Cluster configuration include:

- Increased availability

If the active instance fails for any reason or must be taken offline, an identically configured passive instance is prepared to take over at any time.

- Reduced operating costs

In an active-passive configuration only one set of processes is up and serving requests. Management of the active instance is generally less than managing an array of active instances.

- Application independence

Some applications may not be suited to an active-active configuration. This may include applications which rely heavily on application state or on information stored locally. An active-passive configuration has only one instance serving requests at any particular time.

In general, the term OracleAS Cold Failover Cluster describes clustering at the Oracle Application Server instance level. However, if it is necessary to call out the specific type of instances being clustered, this document will use OracleAS Cold Failover Cluster (Type) to characterize the cluster solution. For example

- Oracle Application Server Cold Failover Cluster (Identity Management)
- Oracle Application Server Cold Failover Cluster (Middle-Tier)

From the entry point of an Oracle Application Server system (content cache) to the back end layer (data sources), all the tiers that are crossed by a client request can be configured in a redundant manner either in an active-active configuration using Oracle Application Server Clusters or in an active-passive configuration using Oracle Application Server Cold Failover Clusters.

## 2.2 High Availability Services in Oracle Application Server

Oracle Application Server provides different features and topologies to support high availability across the its stack. This includes solutions that extend across both the OracleAS middle-tier and the OracleAS Infrastructure tier.

This section describes the following high availability services in Oracle Application Server:

- Section 2.2.1, "Process Death Detection and Automatic Restart"
- Section 2.2.2, "Configuration Management"
- Section 2.2.3, "State Replication"
- Section 2.2.4, "Server Load Balancing and Failover"
- Section 2.2.5, "Backup and recovery"
- Section 2.2.6, "Disaster Recovery"

## 2.2.1 Process Death Detection and Automatic Restart

An Oracle Application Server instance consists of many different running processes to serve client requests. Ensuring high availability means ensuring that all these processes run smoothly, fulfill requests, and do not experience any unexpected hangs or failures.

The interdependency of these processes must also be managed so that they are brought up in the proper sequence, with processes starting up only after the processes that they are dependent on have started successfully.

Oracle Application Server provides high availability and management services at the process level with Oracle Process Manager and Notification Server (OPMN)

### 2.2.1.1 Process Management with Oracle Process Manager and Notification Server

OPMN has the following capabilities:

- Provides automatic death detection of Oracle Application Server processes.
- Provides an integrated way to operate Oracle Application Server components.
- Provides automatic restart of Oracle Application Server processes when they become unresponsive, terminate unexpectedly, or become unreachable as determined by ping and notification operations.
- Channels all events from different Oracle Application Server component instances to all Oracle Application Server components that can utilize them.
- Enables gathering of host and Oracle Application Server process statistics and tasks.
- Does not depend on any other Oracle Application Server component being up and running before it can be started and used.

**See Also:** *Oracle Process Manager and Notification Server Administrator's Guide*

**2.2.1.1.1 Automated Process Management with OPMN** OPMN can be used to explicitly manage the following Oracle Application Server processes:

- Oracle HTTP Server
- Oracle Application Server Containers for J2EE
- Distributed Configuration Management daemon
- OracleAS Log Loader
- OracleAS Guard (for disaster recovery)
- Oracle Internet Directory
- OracleAS Port Tunnel
- OracleAS Web Cache
- Oracle Business Intelligence Discoverer
- OracleAS Wireless

In addition, OPMN implicitly manages any applications that rely on the above components. For example, any J2EE applications that run under OC4J are managed by OPMN.



OPMN is also extensible, providing the capability to add information about custom processes including load environment information, stopping procedures, and methods for death detection and restart.

**2.2.1.1.2 Distributed Process Control with OPMN** Although OPMN can manage processes on a local Oracle Application Server instance, OPMN daemons running on different instances can also work together to provide distributed process management and control.

For example, a command issued on one machine can be used to start all processes or a specific process type across all local and remote Oracle Application Server instances.

OPMN consists of two major components:

- Oracle Notification Server (ONS)

The ONS is the transport mechanism for failure, recovery, startup, and other related notifications between components in Oracle Application Server. It operates according to a publish-subscribe model: an Oracle Application Server component receives a notification of a certain type per its subscription to ONS. When such a notification is published, ONS sends it to the appropriate subscribers.

- Oracle Process Manager (PM)

The PM is the centralized process management mechanism in Oracle Application Server and is used to manage Oracle Application Server processes. It is responsible for starting, restarting, stopping, and monitoring every process it manages. The PM handles all requests sent to OPMN associated with controlling a process or obtaining status about a process. The PM is also responsible for performing death-detection and automatic restart of the processes it manages. The Oracle Application Server processes that the PM is configured to manage are specified in a file named `opmn.xml`. The PM waits for a user command to start specific or all processes. When a specific process or all processes are to be stopped, the PM receives a request as specified by the request parameters.

## 2.2.2 Configuration Management

Managing and ensuring component high availability involves not only managing processes but also the configuration information for those processes both locally and across a set of Oracle Application Server instances.

### 2.2.2.1 Configuration Management with Distributed Configuration Management

Distributed Configuration Management (DCM) is a management framework that enables you to create and manage multiple OracleAS instances as one. Multiple OracleAS instances enable the application server to handle large volumes of traffic reliably since the workload is distributed among multiple OracleAS instances.

DCM enables you to:

- keep a configuration synchronized across multiple OracleAS instances
- archive and restore versions of configurations
- export and import configurations between OracleAS instances

DCM enables you to archive, import and export, and synchronize the configurations of multiple OracleAS instances as if they were a single OracleAS instance. To provide this management functionality, DCM keeps information about an OracleAS instance's configuration in either a file-based or an Oracle database-based repository known as the OracleAS Metadata Repository.

The OracleAS Metadata Repository contains:

- configuration files for Oracle HTTP Server, OC4J, OPMN, and OracleAS JAAS Provider components
- deployed J2EE applications
- information about the OPMN instance or OracleAS Cluster

**2.2.2.1.1 Configuration Synchronization and Management with DCM** With DCM, you can manage configuration information for the following Oracle Application Server components and applications:

- Oracle HTTP Server
- Oracle Application Server Containers for J2EE
- Oracle Process Manager and Notification Server
- Oracle Application Server Java Authentication and Authorization Service (JAAS) Provider
- J2EE applications

The configuration information for each of these components is stored in the metadata repository for each OracleAS instance. Once an OracleAS instance is managed by DCM, configuration information can then be:

- archived for future use
- restored locally from a previous archive
- replicated to another OracleAS instance to provide configuration synchronization across a cluster of OracleAS instances

**2.2.2.1.2 Distributed Application Deployment with DCM** Oracle's Distributed Configuration Management tool, `dcmctl`, enables synchronization of configuration information across a cluster of OracleAS instances. This includes the ability to deploy new J2EE applications on one instance of the cluster and then have the same application automatically deployed by each member of the cluster.

Once an application has been deployed in this way, any instance in the cluster can then receive and serve requests for that application.

## 2.2.3 State Replication

One of the advantages of a distributed application is the ability to set up multiple redundant processes that can all serve the same requests. In the event that one of these application processes becomes unavailable, another application process can service the request.

Some applications may require Oracle Application Server to maintain stateful information across consecutive requests. In order to provide transparent failover of these requests, it is necessary to recreate this application state across multiple processes. Oracle Application Server enables the replication of state in J2EE applications through Oracle Application Server Cluster (OC4J). In an OracleAS Cluster (OC4J), several processes work together to deliver the same J2EE application and replicate the state created by it. This enables the transparent failover of requests between the participants in the cluster. Two different types of state are typically maintained in a J2EE application: HTTP session state (updated by servlets and JSPs) and stateful session EJB state (updated by stateful session EJB instances). OracleAS Cluster (OC4J) enables the replication of both.

**See Also:** The OC4J Clustering chapter in the *Oracle Application Server Containers for J2EE User's Guide*.

## 2.2.4 Server Load Balancing and Failover

Load balancing involves the ability to distribute requests among two or more processes.

Features of a software or hardware load balancer includes:

- load balancing algorithm
 

A rule or set of rules for how to allocate requests across the different instances. The most common load balancing algorithms include simple round-robin or assignment based on some weighted property of the instance such as the response time or capacity of that instance relative to other instances.
- death detection
 

The ability to recognize failed requests to one or more instances, and additionally, the ability to mark those instances as inactive so that no further requests will be forwarded to them.

### 2.2.4.1 Internal Load Balancing Mechanism Provided in Oracle Application Server

Different load balancing mechanisms are provided to communicate the components in an Oracle Application Server system. Load balancing takes place:

- from Oracle Application Server Web Cache to Oracle HTTP Servers
- from Oracle HTTP Servers to OC4J processes for J2EE applications
- from Oracle HTTP Servers to the database for PLSQL applications
- intra OC4J processes from the presentation layer components (servlets and JSPs) to the business layer components (EJBs)
- from OC4J processes to databases

All sub-tiers in Oracle Application Server are enabled to manage failures in the connections that they establish with the next tier as follows:

- Connections established from OracleAS Web Cache to Oracle HTTP Servers: OracleAS Web Cache detects failures in the replies returned by Oracle HTTP Servers and routes the new requests to the available Oracle HTTP Servers.
- Connections established from Oracle HTTP Servers to OC4J processes: Oracle HTTP Server maintains a routing table of available OC4J processes and routes new requests only to those OC4J processes that are up and running.
- Connections established from Oracle HTTP Servers to databases: `mod_plsql` detects failures in the database and routes requests to the available database nodes.
- Connections established between OC4J processes: OC4J detects failures in the RMI invocations to the EJB tier and fails communication over to available EJB nodes.
- Connections established between OC4J processes and databases: OC4J drivers are enabled to detect failures of database nodes and re-route requests to available nodes.

### 2.2.4.2 External Load Balancers

In order to load balance requests among many Oracle Application Server instances in an active-active configuration, Oracle recommends the use of an external load balancer.

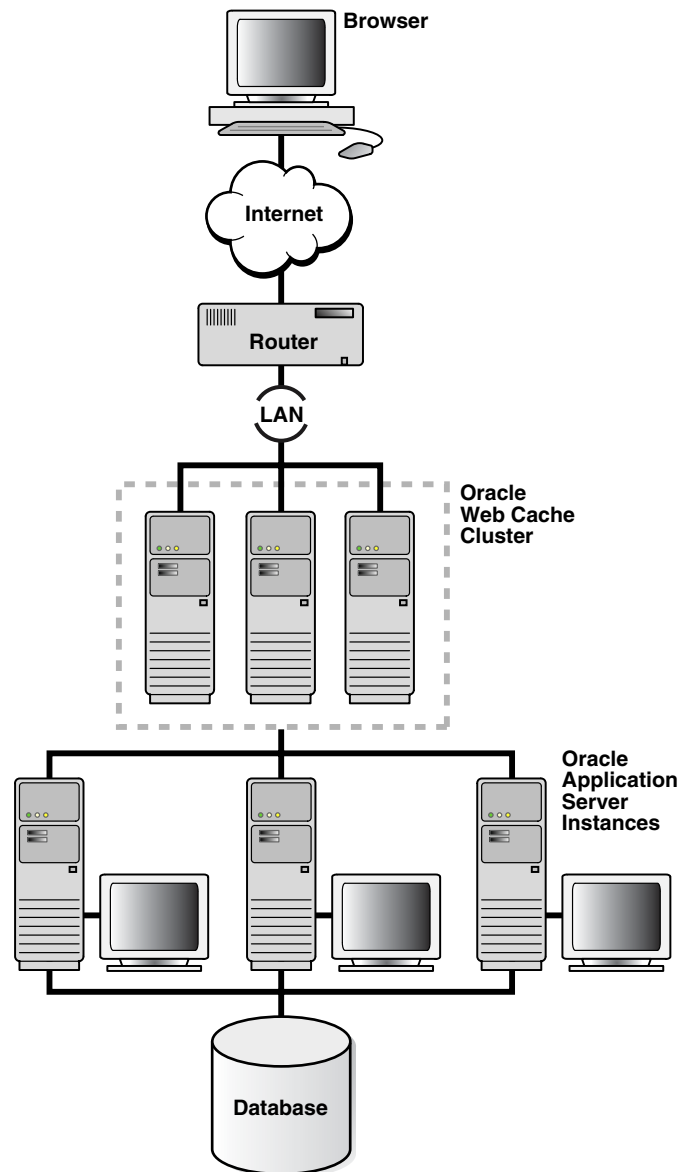
When several Oracle Application Server instances are grouped to work together, they present themselves as a single virtual entry point to the system, which hides the multiple instance configuration. External load balancers can send requests to any application server instance in a cluster, as any instance can service any request. An administrator can raise the capacity of the system by introducing additional application server instances. These instances can be installed on multiple nodes to allow for redundancy in case of node failure.

There are two types of external load balancers you can use with Oracle Application Server instances: hardware load balancers and network load balancers. Table 2-1 summarizes these.

**Table 2-1** *Types of external load balancers*

Load Balancer Type	Description
Hardware Load Balancer	Hardware load balancing involves placing a hardware load balancer, such as Big-IP or Alteon, in front of a group of Oracle Application Server instances or OracleAS Web Cache. The hardware load balancer routes requests to the Oracle HTTP Server or OracleAS Web Cache instances in a client-transparent fashion.
Windows Network Load Balancer (applicable to Windows version of Oracle Application Server)	With some Windows operating systems, you can use the operating system to perform network load balancing. For example, with Microsoft Advanced Server, the NLB functionality allows you to send requests to different machines that share the same virtual IP or MAC address. The servers themselves do not need to be clustered at the operating system level.

Figure 2-1 depicts an example deployment of a hardware load balancing router with Oracle Application Server.

**Figure 2–1 Example load balancing router deployment with Oracle Application Server**

Load balancing improves scalability by providing an access point through which requests are routed to one of many available instances. Instances can be added to the group that the load balancer serves to accommodate additional users.

Load balancing improves availability by routing requests to the most available instances. If one instance goes down, or is particularly busy, the load balancer can send requests to another active instance.

**See Also:**

- Section 4.10, "Using OracleAS Single Sign-On With OracleAS Cluster (Middle-Tier)" on page 4-37
- Section 6.1.2.2, "Configuring A Load Balancer For OracleAS Cluster (Identity Management)" on page 6-7

## 2.2.5 Backup and recovery

Protecting against the loss of any system components is critical to maintaining a highly available environment. Regular, complete backups of all Oracle Application Server environment is recommended.

A complete Oracle Application Server environment backup includes:

- A full backup of all files in the middle-tier Oracle homes (this includes Oracle software files and configuration files).
- A full backup of all files in the OracleAS Infrastructure Oracle home (this includes Oracle software files and configuration files).
- A complete cold backup of the OracleAS Metadata Repository.
- A full backup of the Oracle system files on each host in your environment.

### 2.2.5.1 Oracle Application Server Backup and Recovery Tool

The most frequently changing critical files in an Oracle installation are configuration files and data files. Oracle provides the Oracle Application Server Backup and Recovery Tool (OracleAS Backup and Recovery Tool) to backup these configuration and data files.

The OracleAS Backup and Recovery Tool is a Perl script and associated configuration files. You can use this tool to backup and recover the following types of files:

- configuration files in the middle-tier and OracleAS Infrastructure Oracle homes
- OracleAS Metadata Repository files

The OracleAS Backup and Recovery Tool is installed by default whenever you install Oracle Application Server. The tool is installed in the `$ORACLE_HOME/backup_restore` directory.

The OracleAS Backup and Recovery Tool supports the following installation types:

- J2EE and Web Cache
- Portal and Wireless
- OracleAS Infrastructure (Identity Management and Metadata Repository)
- OracleAS Infrastructure (Identity Management only)
- OracleAS Infrastructure (Metadata Repository only)
- OracleAS TopLink (standalone or installed into a OracleAS middle-tier Oracle home)
- Oracle Application Server Integration Business Activity Monitoring
- Oracle Content Management Software Development Kit

**See Also:** *Oracle Application Server Administrator's Guide*

## 2.2.6 Disaster Recovery

Disaster recovery refers to how a system can be recovered from catastrophic site failures caused by natural or unnatural disasters. Additionally, disaster recovery can also refer to how a system is managed for planned outages. For most disaster recovery situations, the solution involves replicating an entire site, not just pieces of hardware or subcomponents. This also applies to the Oracle Application Server Disaster Recovery (OracleAS Disaster Recovery) solution.

In the most common configuration, a standby site is created to mirror the production site. Under normal operation, the production site actively services client requests. The standby site is maintained to mirror the applications and content hosted by the production site.

### **2.2.6.1 Oracle Application Server Guard**

Oracle Application Server Guard (OracleAS Guard) automates the restoration of a production site on its corresponding standby site. To protect a complete Oracle Application Server environment from disasters, OracleAS Guard performs the following operations:

- Instantiates the standby site: instantiates an Oracle Application Server standby farm that mirrors a primary farm.
- Verifies configuration: verifies that a farm meets the requirements to be used as a standby farm for the corresponding primary farm.
- Site synchronization: synchronizes the production and the standby sites.

**See Also:** Section 7.4, "Overview of OracleAS Guard and asgctl" on page 7-15





# Part II

---

## Middle-tier High Availability

This part contains chapters that discuss high availability for the middle tier. These chapters are:

- Chapter 3, "Middle-tier High Availability"
- Chapter 4, "Managing and Operating Middle-tier High Availability"



---

---

## Middle-tier High Availability

This chapter describes solutions that are available to protect the Oracle Application Server middle-tier from failures. It is organized into the following main sections:

- Section 3.1, "Redundancy"
- Section 3.2, "Highly Available Middle-tier Configuration Management Concepts"
- Section 3.3, "Middle-tier Backup and Recovery Considerations"
- Section 3.4, "Middle-tier Applications"

### 3.1 Redundancy

The OracleAS middle-tier can be configured to provide two types of redundancy:

- Active-Active
- Active-Passive

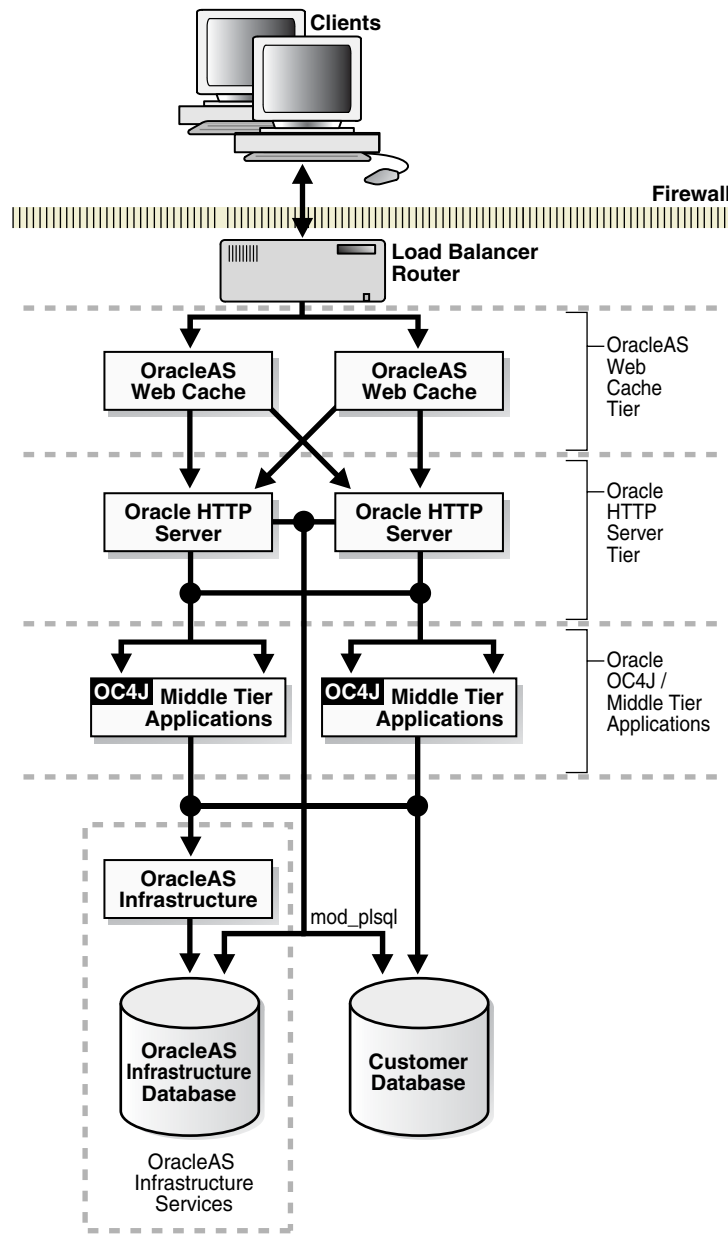
The ensuing sections provide details on how the middle tier achieves these redundancy types.

#### 3.1.1 Active-Active

An Oracle Application Server middle-tier can be made redundant in an active-active configuration with Oracle Application Server Cluster (Middle-Tier). An OracleAS Cluster (Middle-Tier) is a set of application server middle-tier instances configured to act in active-active configuration to deliver greater scalability and availability than a single instance. Using middle-tier OracleAS Clusters removes the single point of failure that a single instance poses. While a single application server instance leverages the operating resources of a single host, a cluster can span multiple hosts, distributing application execution over a greater number of CPUs. A single application server instance is vulnerable to the failure of its host and operating system, but a cluster continues to function despite the loss of an operating system or a host, hiding any such failure from clients

Figure 3–1 presents the various sub-tiers of the OracleAS middle-tier in a redundant active-active configuration. Each sub-tier is configured with redundant processes so that the failure of any of these processes is handled by the sub-tier above the processes such that the failure has no effect in the incoming requests from clients.

**Figure 3-1 Overall active-active architecture of OracleAS middle-tier**



The following sub-sections describe features that characterize each sub-tier's active-active configuration:

- Section 3.1.1.1, "Oracle Application Server Web Cache Sub-Tier"
- Section 3.1.1.2, "Oracle HTTP Server Sub-Tier"
- Section 3.1.1.3, "Oracle Application Server Containers for J2EE Sub-Tier"

### 3.1.1.1 Oracle Application Server Web Cache Sub-Tier

Two or more OracleAS Web Cache instances can be clustered together to create a single logical cache. Physically, the cache can be distributed amongst several nodes. If one node fails, a remaining node in the same cluster can fulfill the requests serviced by the failed node. The failure is detected by the remaining nodes in the cluster who take over ownership of the cacheable content of the failed member. The load balancing

mechanism in front of the OracleAS Web Cache cluster, for example, a hardware load balancing appliance, redirects the requests to the live OracleAS Web Cache nodes.

OracleAS Web Cache clusters also add to the availability of OracleAS instances. By caching static and dynamic content in front of the OracleAS instances, requests can be serviced by OracleAS Web Cache reducing the need for the requests to be fulfilled by OracleAS instances, particularly for Oracle HTTP Servers. The load and stress on OracleAS instances is reduced, thereby increasing availability of the components in the instances.

Oracle Application Server Web Cache can also perform a stateless or stateful load balancing role for Oracle HTTP Servers. Load balancing is done based on the percentage of the available capacity of each Oracle HTTP Server, or, in other words, the weighted available capacity of each Oracle HTTP Server. If the weighted available capacity is equal for several Oracle HTTP Servers, OracleAS Web Cache uses round robin to distribute the load. Refer to *Oracle Application Server Web Cache Administrator's Guide* for the formula to calculate weighted available capacity.

Table 3–1 provides a summary of the high availability characteristics of OracleAS Web Cache.

**Table 3–1 OracleAS Web Cache high availability characteristics**

Component	Protection from Node Failure	Protection from Service Failure	Protection from Process Failure	Automatic Re-routing	State Replication	Configuration Cloning
OracleAS Web Cache	OracleAS Web Cache cluster protects from single point of failure. An external load balancer should be deployed in front of this cluster to route requests to live OracleAS Web Cache nodes.	In an OracleAS Web Cache cluster, pings are made to a specific URL in each cluster member to ensure that the URL is still serviceable.	OPMN monitors OracleAS Web Cache processes and restarts them upon process failure.	OracleAS Web Cache members in a cluster ping each other to verify that peer members are alive or have failed. External load balancers provide failover capabilities for requests routed to OracleAS Web Cache components.	OracleAS Web Cache clustering manages cached contents that need to be transferred between OracleAS Web Cache nodes.	OracleAS Web Cache cluster maintains uniform configuration across cluster.

In the case of failure of a Oracle HTTP Server, OracleAS Web Cache redistributes the load to the remaining Oracle HTTP Servers and polls the failed server intermittently until it comes back online. Thereafter, OracleAS Web Cache recalculates the load distribution with the revived Oracle HTTP Server in scope.

**See Also:**

- Section 3.1.1.2, "Oracle HTTP Server Sub-Tier" on page 3-3
- *Oracle Application Server Web Cache Administrator's Guide*

### 3.1.1.2 Oracle HTTP Server Sub-Tier

Oracle HTTP Server and Oracle Application Server Web Cache provide HTTP and HTTPS request handling for Oracle Application Server requests. Each HTTP request is

met by a response from Oracle HTTP Server or from OracleAS Web Cache, if the content requested is cached.

Oracle HTTP Server routes a request to different plug-in modules depending on the type of request received. These modules in turn delegate the request to different types of processes. The most common modules are `mod_oc4j` for J2EE applications and `mod_plsql` for PLSQL applications. `mod_oc4j` delegates requests to OC4J processes. `mod_plsql` delegates requests to database processes. For all these types of requests, no state is required to be maintained in the Oracle HTTP Server processes.

This section covers the following topics:

- Section 3.1.1.2.1, "Oracle HTTP Server High Availability Summary"
- Section 3.1.1.2.2, "OC4J Load Balancing Using `mod_oc4j`"
- Section 3.1.1.2.3, "Database Load Balancing with `mod_plsql`"

**3.1.1.2.1 Oracle HTTP Server High Availability Summary** Table 3–2 summarizes some of the Oracle Application Server high availability features for Oracle HTTP Server.

**Table 3–2 Oracle HTTP Server high availability characteristics**

Component	Protection from Node Failure	Protection from Service Failure	Protection from Process Failure	Automatic Re-routing	State Replication	Configuration Cloning
Oracle HTTP Server	OracleAS Cluster protects from single point of failure. A load balancer should be deployed in front of Oracle HTTP Server instances. This can be an external load balancer or OracleAS Web Cache.	Load balancer in front of Oracle HTTP Server sends request to another Oracle HTTP Server if first one doesn't respond or is deemed failed through URL pings. Load balancer can be either OracleAS Web Cache or hardware appliance.	OPMN monitors Oracle HTTP Server processes and restarts them upon process failure. Each Oracle HTTP Server is also notified by OPMN when another Oracle HTTP Server process in the OracleAS Cluster fails.	Load balancer in front of Oracle HTTP Server auto re-routes to another Oracle HTTP Server if first does not respond.	None.	OracleAS Cluster allows configuration to be replicated across to other Oracle HTTP Servers in the cluster through DCM.

**See Also:**

- Section 3.1.1.1, "Oracle Application Server Web Cache Sub-Tier" on page 3-2
- Section 2.2.1, "Process Death Detection and Automatic Restart" on page 2-4
- Section 2.2.4.2, "External Load Balancers" on page 2-8

**3.1.1.2.2 OC4J Load Balancing Using `mod_oc4j`** The Oracle HTTP Server module, `mod_oc4j` provides routing for HTTP requests that are handled by OC4J. Whenever a request is received for a URL that matches one of the mount points specified in `mod_oc4j.conf`, the request is routed to one of the available destinations specified for that URL. A destination can be a single OC4J process, or a set of OC4J instances. If an OC4J

process fails, OPMN detects the failure and `mod_oc4j` does not send requests to the failed OC4J process until the OC4J process is restarted.

Using `mod_oc4j` configuration options, you can specify different load balancing routing algorithms depending on the type and complexity of routing you need. Stateless requests are routed to any destination available based on the algorithm specified in `mod_oc4j.conf`. Stateful HTTP requests are forwarded to the OC4J process that served the previous request using session identifiers, unless `mod_oc4j` determines through communication with OPMN that the process is not available. In this case, `mod_oc4j` forwards the request to an available OC4J process following the specified load balancing protocol.

Table 3–3 summarizes the routing styles that `mod_oc4j` provides. For each routing style, Table 3–3 lists the different algorithms that you can configure to modify the routing behavior. These `mod_oc4j` configuration options determine the OC4J process where `mod_oc4j` sends incoming HTTP requests to be handled.

**See Also:**

- Section 4.2.5.1, "Using and Configuring `mod_oc4j` Load Balancing" on page 4-10
- *Oracle HTTP Server Administrator's Guide* for information on using weighted routing and selecting local affinity with `mod_oc4j` load balancing options.

**Table 3–3 `mod_oc4j` Routing Algorithms Summary**

Routing Method	Description
Round Robin	Using the simple round robin configuration, all OC4J processes, remote and local to the application server instance running the Oracle HTTP Server, are placed in an ordered list. Oracle HTTP Server then chooses an OC4J process at random for the first request. For each subsequent request, Oracle HTTP Server forwards requests to another OC4J process in round robin style.  The round robin configuration supports local affinity and weighted routing options.
Random	Using the simple random configuration, all OC4J processes, remote and local to the application server instance running the Oracle HTTP Server, are placed in an ordered list. For every request, Oracle HTTP Server chooses an OC4J process at random and forwards the request to that instance.  The random configuration supports local affinity and weighted routing options.
Metric-Based	Using the metric-based configuration OC4J processes, remote and local to the application server instance running the Oracle HTTP Server, are placed into an ordered list. OC4J processes then regularly communicate to Oracle HTTP Server how busy they are and Oracle HTTP Server uses this information to send requests to the OC4J processes that are less busy. The overhead in each OC4J node is measured using the runtime performance metrics of OC4J processes. When there are no local OC4J processes available, <code>mod_oc4j</code> routes requests to each OC4J process on different hosts as per their performance metrics only.  The metric-based configuration supports a local affinity option.

**OC4J Load Balancing Using Local Affinity and Weighted Routing Options**

Using `mod_oc4j` options, you can select a routing method for routing OC4J requests. If you select either round robin or random routing, you can also use local affinity or weighted routing options. If you select metric-based routing, you can also use the local affinity option.

Using the weighted routing option, a weight is associated with OC4J processes on a node, as configured in `mod_oc4j`, on a node by node basis. During request routing, `mod_oc4j` then uses the routing weight to calculate which OC4J process to assign

requests to. Thus, OC4J processes running on different nodes can be assigned different weights.

Using the local affinity option, `mod_oc4j` keeps two lists of available OC4J processes to handle requests, a local list and a remote list. If processes are available from the local list then requests are assigned locally using the random routing method or, for metric-based routing using metric-based routing. If no processes are available in the local list, then `mod_oc4j` selects processes randomly from the remote list when random method, using a round robin method for the round robin method, or using metric-based routing with the metric-based method.

### Choosing a `mod_oc4j` Routing Algorithm

Table 3–3 summarizes the available routing options. To select a routing algorithm to configure with `mod_oc4j`, you need to consider the type of environment where Oracle HTTP Server runs. Use the following guidelines to help determine which configuration options to use with `mod_oc4j`:

- For a Oracle Application Server cluster setup, with multiple identical machines running Oracle HTTP Server and OC4J in the same node, the round robin with local affinity algorithm is preferred. Using this configuration, an external router distributes requests to multiple machines running Oracle HTTP Server and OC4J. In this case Oracle HTTP Server gains little by using `mod_oc4j` to route requests to other machines, except in the extreme case that all OC4J processes on the same machine are not available.
- For a tiered deployment, where one tier of machines contains Oracle HTTP Server and another contains OC4J instances that handle requests, the preferred algorithms are simple round robin and simple metric-based. To determine which of these two is best in a specific setup, you may need to experiment with each and compare the results. This is required because the results are dependent on system behavior and incoming request distribution.
- For a heterogeneous deployment, where the different application server instances run on nodes that have different characteristics, using the weighted round robin algorithm is preferred. Tune the number of OC4J processes running on each application server instance may allow you to achieve the maximum benefit. For example, a machine with a weight of 4 gets 4 times as many requests as a machine with a weight of 1, but if the system with a weight of 4 may not be running 4 times as many OC4J processes.
- Metric-based load balancing is useful when there are only a few metrics that dominate the performance of an application. For example, CPU or number of database connections.

#### See Also:

- Section 4.2.5.1, "Using and Configuring `mod_oc4j` Load Balancing" on page 4-10
- *Oracle HTTP Server Administrator's Guide* for information on using weighted routing and selecting local affinity with `mod_oc4j` load balancing options.

**3.1.1.2.3 Database Load Balancing with `mod_plsql`** `mod_plsql` maintains a pool of connections to the database and reuses established database connections for subsequent requests. If there is no response from a database connection in a connection pool, `mod_plsql` detects this, discards the dead connection, and creates a fresh database connection for subsequent requests.



The dead database connection detection feature of `mod_plsql` eliminates the occurrence of errors when a database node or instance goes down. This feature is also extremely useful in high availability configurations like Real Application Clusters. If a node in a Real Application Clusters database fails, `mod_plsql` detects this and immediately starts servicing requests using the other Real Application Clusters nodes. `mod_plsql` provides different configuration options to satisfy maximum protection or maximum performance needs. By default, `mod_plsql` tests all pooled database connections which were created prior to the detection of a failure, but it also allows constant validation of all pooled database connections prior to issuing a request.

**See Also:** *Oracle Application Server mod\_plsql User's Guide*

### 3.1.1.3 Oracle Application Server Containers for J2EE Sub-Tier

The Oracle Application Server Containers for J2EE tier consists of the Oracle Application Server implementation of the J2EE container. This section discusses how the various OC4J components can be made highly available and consists of the following topics:

- Section 3.1.1.3.1, "OracleAS Cluster (OC4J)"
- Section 3.1.1.3.2, "OC4J Distributed Caching Using Java Object Cache"
- Section 3.1.1.3.3, "JMS High Availability"

**3.1.1.3.1 OracleAS Cluster (OC4J)** Oracle Application Server provides several strategies for ensuring high availability with OC4J instances, both within an application server instance and across a cluster that includes multiple application server instances.

Besides the high availability features described in this section, other Oracle Application Server features enable OC4J processes to be highly available, including the load balancing feature in Oracle HTTP Server and the Oracle Process Manager and Notification Server system that automatically monitors and restarts processes.

**See Also:**

- Section 3.1.1.2, "Oracle HTTP Server Sub-Tier" on page 3-3
- Section 2.2.1, "Process Death Detection and Automatic Restart" on page 2-4

The following sections explain the strategies for ensuring high availability for stateful applications in OC4J instances. Overall, there are two strategies:

- Web Application Session State Replication with OracleAS Cluster (OC4J)
- Stateful Session EJB State Replication with OracleAS Cluster (OC4J)

#### **Web Application Session State Replication with OracleAS Cluster (OC4J)**

When a stateful Web application is deployed to OC4J, multiple HTTP requests from the same client may need to access the application. However, if the application running on the OC4J server experiences a problem where the OC4J process fails, the state associated with a client request may be lost. Using Oracle Application Server, there are two ways to guard against such failures:

- State safe applications save their state in a database or other persistent storage system, avoiding the loss of state when the server goes down. Obviously, there is a performance cost for continually writing the application state to persistent storage.

---

---

**Note:** In this release of Oracle Application Server, 10g (10.1.2), saving application state to persistent storage is the application developer's responsibility.

---

---

- Stateful applications can use OC4J session state replication, with OracleAS Cluster (OC4J), to automatically replicate the session state across multiple processes in an application server instance, and in a cluster, across multiple application instances which may run on different nodes.

An OC4J instance is the entity to which J2EE applications are deployed and configured. An OC4J instance is characterized by a specific set of binaries and configuration files. Several OC4J processes can be started for each OC4J instance. The OC4J process is what executes the J2EE applications for the OC4J instance. Within the application server instance, you can configure multiple OC4J instances, each with its own number of OC4J processes. The advantage of this is for configuration management and application deployment for separate OC4J processes in a cluster.

OC4J processes can be grouped into OracleAS Cluster (OC4J) to support session state replication for the high availability of Web applications. Using an OracleAS Cluster (OC4J) together with `mod_oc4j` request routing provides stateful failover in the event of a software or hardware problem. For example, if an OC4J process that is part of an OracleAS Cluster (OC4J) fails, `mod_oc4j` is notified of the failure by OPMN and routes requests to another OC4J process in the same cluster.

Each OC4J instance in a cluster has the following features:

- The configuration of the OC4J instance is valid for one or more OC4J executable processes. This way, you can duplicate the configuration for multiple OC4J processes by managing these processes in the OC4J instance construct. When you modify the cluster-wide configuration within the OC4J instance, the modifications are valid for all OC4J processes.
- Each OC4J instance can be configured with one or more OC4J processes.
- When you deploy an application to an OC4J instance, all OC4J processes share the same application properties and configuration defined in the OC4J instance. The OC4J instance is also responsible for replicating the state of its applications.
- The number of OC4J processes is specific to each OC4J instance. This must be configured for each application server instance in the cluster. The OC4J process configuration provides flexibility to tune according to the specific hardware capabilities of the host. By default, each OC4J instance is instantiated with a single OC4J process.

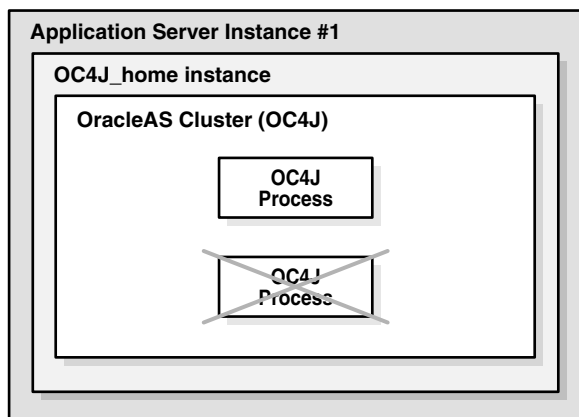
#### **Web Application Session State Replication Protecting Against Software Problems**

To guard against software problems, such as OC4J process failure or hang, you can configure an OC4J instance to run multiple OC4J processes in the same OracleAS Cluster (OC4J). The processes in the OracleAS Cluster (OC4J) communicate their session state between each other. Using this configuration provides failover and high availability by replicating state across multiple OC4J processes running on an application server instance.

In the event of a failure, Oracle HTTP Server forwards requests to active (alive) OC4J process within the OracleAS Cluster (OC4J). In this case, the Web application state for the client is preserved and the client does not notice any loss of service.

Figure 3–2 shows this type of software failure within an application server instance and the failover to the surviving process.

**Figure 3–2 Web Application Session State Failover Within an OracleAS Cluster (OC4J) in an OC4J instance**



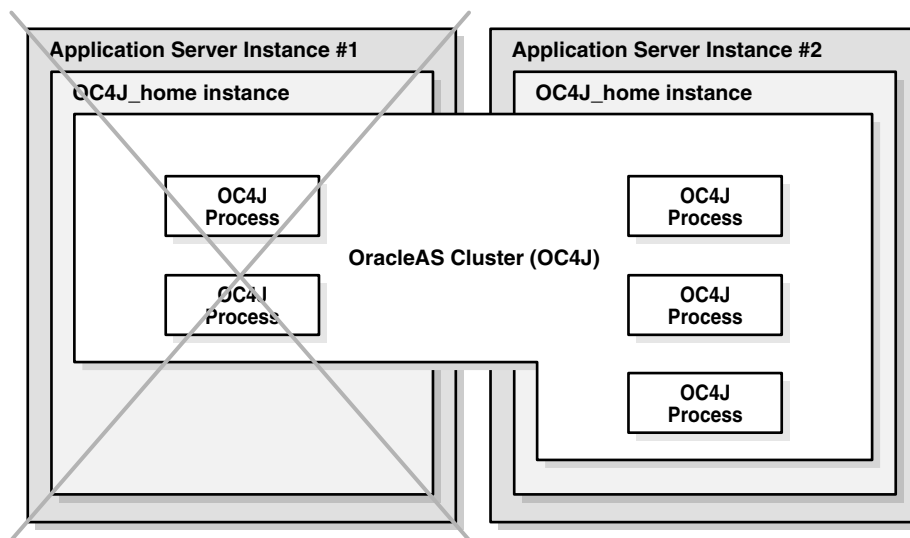
### Web Application Session State Replication Protecting Against Hardware Problems

To guard against hardware problems, such as the failure of the node where an application server instance runs, you can configure OracleAS Cluster (OC4J) across application server instances that are in more than one node in an OracleAS Cluster. By configuring an OracleAS Cluster (OC4J) that uses the same name across multiple application server instances, the OC4J processes can share session state information across the OracleAS Cluster (OC4J). When an application server instance fails or is not available, for example, when the node it runs on goes down, Oracle HTTP Server forwards requests to an OC4J process in an application server instance that is available. Thus, Oracle HTTP Server forwards requests only to active (alive) OC4J processes within the cluster.

In this case, the Web application state for the client is preserved and the client does not notice any irregularity.

Figure 3–3 depicts an OracleAS Cluster (OC4J) configured across two Oracle Application Server instances. This configuration allows for web application session state replication failover within an OracleAS Cluster (OC4J).

**Figure 3–3 Web Application Session State Failover Within an OracleAS Cluster (OC4J)**



**Configuring OracleAS Cluster (OC4J) for Web Application Session State**

**Replication** To protect against software or hardware failure while maintaining state with the least number of OC4J processes, you need to configure at least two OC4J processes in the same cluster. For example, if you have two application server instances, instance 1 and instance 2, you can configure two OC4J processes in the `default_island` on each application server instance. With this configuration, stateful session applications are protected against hardware and software failures, and the client maintains state if either of the following types of failures occurs:

- If one of the OC4J processes fails, then the client request is redirected to the other OC4J process in the `default_island` on the same application server instance. State is preserved and the client does not notice any irregularity.
- If application server instance 1 terminates abnormally, then the client is redirected to the OC4J process in the `default_island` on application server instance 2. The state is preserved and the client does not notice any irregularity.

**See Also:** Section 4.4.3.1, "Configuring OC4J Islands and OC4J Processes" on page 4-28

**Stateful Session EJB State Replication with OracleAS Cluster (OC4J)**

Using OC4J, stateful session EJBs can be configured to provide state replication across OC4J processes associated to an application server instance or across an OracleAS Cluster. This EJB replication configuration provides high availability for stateful session EJBs by using multiple OC4J processes to run instances of the same stateful session EJB.

---

---

**Note:** Use of EJB replication OracleAS Cluster (OC4J-EJB) for high availability is independent of middle-tier OracleAS Clusters and can involve multiple application server instances installed across nodes that are or are not part of middle-tier OracleAS Clusters.

---

---

OracleAS Cluster (OC4J-EJB)s provide high availability for stateful session EJBs. They allow for failover of these EJBs across multiple OC4J processes that communicate over the same multicast address. Thus, when stateful session EJBs use replication, this can protect against process and node failures and can provide for high availability of stateful session EJBs running on Oracle Application Server.

**See Also:**

- Section 4.4.2.4, "Configuring EJB Application State Replication With OracleAS Cluster (OC4J-EJB)" on page 4-26
- *Oracle Application Server Containers for J2EE User's Guide*
- *Oracle Application Server Containers for J2EE Enterprise JavaBeans Developer's Guide*

**JNDI Namespace Replication** When EJB clustering is enabled, JNDI namespace replication is also enabled between the OC4J instances in a middle-tier OracleAS Cluster. New bindings to the JNDI namespace in one OC4J instance are propagated to other OC4J instances in the middle-tier OracleAS Cluster. Re-bindings and unbindings are not replicated.

The replication is done outside the scope of each OracleAS Cluster (OC4J). In other words, multiple OracleAS Clusters (OC4J) in an OC4J instance have visibility into the same replicated JNDI namespace.

**See Also:** *Oracle Application Server Containers for J2EE Services Guide*

**EJB Client Routing** In EJB client routing, EJB classes take on the routing functionality that `mod_oc4j` provides between Oracle HTTP Server and servlets/JSPs. Clients invoke EJBs using the Remote Method Invocation (RMI) protocol. The RMI protocol listener is set up by in the RMI configuration file, `rmi.xml`, for each OC4J instance. It is separate from the Web site configuration. EJB clients and the OC4J tools access the OC4J server through a configured RMI port. OPMN designates a range of ports that the RMI listener could be using.

When you use the "`opmn:ormi://`" prefix string in the EJB look up, the client retrieves the assigned RMI port automatically. The load balancing and client request routing is provided by OPMN selecting the different OC4J processes available. The algorithm used for this load balancing is the random algorithm. Multiple OPMN URLs separated by commas can be used for higher availability.

**See Also:** The EJB primer section in *Oracle Application Server Containers for J2EE User's Guide*.

**3.1.1.3.2 OC4J Distributed Caching Using Java Object Cache** Oracle Application Server Java Object Cache provides a distributed cache that can serve as a high availability solution for applications deployed to OC4J. The Java Object Cache is an in-process cache of Java objects that can be used on any Java platform by any Java application. It allows applications to share objects across requests and across users, and coordinates the life cycle of the objects across processes.

Java Object Cache enables data replication among OC4J processes even if they do not belong to the same OracleAS Cluster (OC4J), application server instance, or overall Oracle Application Server Cluster.

By using Java Object Cache, performance can be improved since shared Java objects are cached locally, regardless of which application produces the objects. This also improves availability; in the event that the source for an object becomes unavailable, the locally cached version will still be available.

**See Also:** The Java Object Cache chapter in the *Oracle Application Server Web Services Developer's Guide* for complete information on using Java Object Cache

**3.1.1.3.3 JMS High Availability** Two JMS providers are available with Oracle Application Server. Due to differing implementations, each provider achieves high availability in different ways. As such, they are discussed in two sections:

- Oracle Application Server JMS High Availability
- Oracle JMS High Availability

Oracle Application Server JMS (OracleAS JMS) is implemented in OC4J. Hence, it utilizes OPMN for process monitoring and restart.

Oracle JMS (OJMS) is implemented through Oracle Streams Advanced Queuing (AQ). It requires the Oracle database and can have active-active availability through the database Real Application Clusters and Transparent Application Failover (TAF) features.

Table 3–4 provides an overview of high availability and configuration characteristics of the two JMS providers. The sections following the table discuss each provider in more detail.

**Table 3–4 Summary of high availability and configuration characteristics of OJMS and OracleAS JMS**

Characteristic	OJMS	OracleAS JMS
Process-Level High Availability	OPMN (JMS application)	OPMN
Node-Level High Availability	Real Application Clusters (AQ, TAF)	OracleAS Cold Failover Cluster (Middle-Tier)
Configuration	Real Application Clusters configuration, resource provider configuration	dedicated JMS server, <code>jmx.xml</code> configuration, <code>opmn.xml</code> configuration
Message Store	in Real Application Clusters database	in dedicated JMS server/persistence files
Failover	same or different machine (depending on Real Application Clusters setup)	same or different machine only in active-passive configuration with OracleAS Cold Failover Cluster (Middle-Tier)  (see Section 3.1.2.1, "Oracle Application Server Cold Failover Cluster (Middle-Tier)" on page 3-15)

---

**Note:** The *Oracle Application Server Containers for J2EE Services Guide* provides detailed information and instructions on setting up OracleAS JMS and OJMS to be highly available. High availability for third party JMS providers is not discussed as it is provider-specific.

---

The following sections provide details on how each JMS provider achieves high availability.

### Oracle Application Server JMS High Availability

High availability for OracleAS JMS can be achieved through grouping multiple instances of OC4J together in one cluster. This cluster is called Oracle Application Server Cluster (OC4J-JMS). OPMN can be used to monitor and restart OC4J processes in the event of process failure.

Oracle Application Server Cluster (OC4J-JMS) provides an environment wherein JMS applications deployed in this environment can load balance JMS requests across multiple OC4J instances or processes. Redundancy is also achieved as the failure of an OC4J instance with a JMS server does not impact the availability of the JMS service since at least one other OC4J instance is available with a JMS server.

Both the JMS client and the JMS server contain state about each other, which includes information about connections, sessions, and durable subscriptions. Application developers can configure their environment and use a few simple techniques when writing their applications to make them cluster-friendly.

OracleAS Cluster (OC4J-JMS) allows for two configurations:

- OracleAS JMS Server Distributed Destinations

This configuration requires multiple OC4J instances. Each instance contains a JMS server, queue destination, and application. There is no inter-process communication between JMS servers, queues, and applications in other OC4J instances. The sender and receiver of each application must be deployed together in an OC4J instance. A message enqueued to the JMS server in one OC4J process can be dequeued only from that OC4J process.

This configuration has the following advantages:

- High throughput is achieved since applications and JMS servers are executing within the same JVMs and no inter-process communication is required.
- There is no single point of failure. As long as one OC4J process is running, requests can be processed.
- Destination objects can be persistent or in-memory. Persistence is file-based.

The disadvantage of this configuration is that there is no failover from one JMS server to another.

- **Dedicated OracleAS JMS Server**

This configuration defines that only one OC4J instance has the dedicated JMS server in an OracleAS Cluster (OC4J-JMS). The OC4J instance with the JMS server handles all messages. Message ordering is always maintained due to the single JMS server. All JMS applications use this dedicated server to host their connection factories and destination, and to service their enqueue and dequeue requests.

Only one OC4J JVM acts as the dedicated JMS server for all JMS applications within the Oracle Application Server Cluster (OC4J-JMS). The single JVM ensures that other JVMs do not attempt to use the same set of persistent files. Other OC4J execute only applications. The single JMS server can be configured by limiting the JMS port range in the `opmn.xml` file to only one port for the dedicated OC4J instance. The single port value ensures that OPMN always assigns the same port value to the dedicated JMS server. This port value is used to define the connection factory in the `jms.xml` file that other OC4J instances in the OracleAS Cluster (OC4J-JMS) use to connect to the dedicated JMS server.

Refer to the JMS chapter in the *Oracle Application Server Containers for J2EE Services Guide* for more information on how to modify the `opmn.xml` file for this dedicated JMS server configuration.

**See Also:** The section "Abnormal Termination" in the Java Message Service chapter of the *Oracle Application Server Containers for J2EE Services Guide*. This section describes how to manage persistence files when an unexpected failure occurs.

### **Oracle JMS High Availability**

High availability for Oracle JMS (OJMS) can be achieved using a Real Application Clusters database. AQ queues and topics should be available in the Real Application Clusters environment.

Each JMS application in Oracle Application Server uses OC4J resource providers to point to the backend Real Application Clusters database. JMS operations invoked on objects derived from these resources providers are directed to the database.

An OJMS application that uses a Real Application Clusters database must be able to handle database failover scenarios. Two failover scenarios are possible:

- **Real Application Clusters Network Failover**

In the event of the failure of a database instance, a standalone OJMS application running against a Real Application Clusters database must have code to obtain the connection again and to determine if the connection object is invalid or not. The code must re-establish the connection if necessary. Use the API method `com.evermind.sql.DbUtil.oracleFatalError()` to determine if a connection object is invalid. If invalid, a good strategy is to aggressively roll back transactions and recreate the JMS state, such as connections, session, and

messages, that were lost. Refer to the JMS chapter in *Oracle Application Server Containers for J2EE Services Guide* for a code example.

- **Transparent Application Failover**

For most cases when Transparent Application Failover (TAF) is configured, an OJMS application will not be aware of a failed database instance that it is connected to. Hence, the application code need not perform any tasks to handle the failure.

However, in some cases, OC4J may throw an ORA error when a failure occurs. OJMS passes these errors to the application as a `JMSException` with a linked SQL exception. To handle these exceptions, the following can be done:

- As in the previous point, "Real Application Clusters Network Failover", provide code to use the method `com.evermind.sql.DbUtil.oracleFatalError()` to if the error is a fatal error. If it is, follow the approach outlined in the previous point. If it is not, the client can recover by sleeping for a short period of time and then wake up and retry the last operation.
- Failback and transient errors caused by incomplete failover can be recovered from by attempting to use the JMS connection after a short time. Pausing allows the database failover to recover from the failure and reinstate itself.
- In the case of transaction exceptions, such as "Transaction must roll back" (ORA-25402) or "Transaction status unknown (ORA-25405), the current operation must be rolled back and all operations past the last commit must be retried. The connection is not usable until the cause of the exception is dealt with. If the retry fails, close and re-create all connections and retry all uncommitted operations.

### **Clustering Best Practises**

The following are best practise guidelines for working with clustered JMS servers:

- **Minimize JMS client-side state.**
  - Perform work in transacted sessions.
  - Save/checkpoint intermediate program state in JMS queues/topics for full recoverability.
  - Do not depend on J2EE application state to be serializable or recoverable across JVM boundaries. Always use transient member variables for JMS objects, and write `passivate/activate` and `serialize/deserialize` functions that save and recover JMS state appropriately.
- **Do not use nondurable subscriptions on topics.**
  - Nondurable topic subscriptions duplicate messages per active subscriber. Clustering and load balancing creates multiple application instances. If the application creates a nondurable subscriber, it causes the duplication of each message published to the topic. This is either inefficient or semantically invalid.
  - Use only durable subscriptions for topics. Use queues whenever possible.
- **Do not keep durable subscriptions alive for extended periods of time.**
  - Only one instance of a durable subscription can be active at any given time. Clustering and load-balancing creates multiple application instances. If the



application creates a durable subscription, only one instance of the application in the cluster succeeds. All other instances fail with a `JMSEException`.

- Create, use, and close a durable subscription in small time/code windows, minimizing the duration when the subscription is active.
- Write application code that accommodates failure to create durable subscription due to clustering (when some other instance of the application running in a cluster is currently in the same block of code) and program appropriate back-off strategies. Do not always treat the failure to create a durable subscription as a fatal error.

## 3.1.2 Active-Passive

Active-passive high availability for the middle tier is achieved using a cold failover cluster. This is discussed in the following section.

### 3.1.2.1 Oracle Application Server Cold Failover Cluster (Middle-Tier)

A two-node OracleAS Cold Failover Cluster (Middle-Tier) can be used to achieve active-passive availability for Oracle Application Server middle-tier components. In a OracleAS Cold Failover Cluster (Middle-Tier), one node is active while the other is passive, on standby. In the event that the active node fails, the standby node is activated, and the middle-tier components continue servicing clients from that node. All middle-tier components are failed over to the new active node. No middle-tier components run on the failed node after the failover.

In the OracleAS Cold Failover Cluster (Middle-Tier) solution, a virtual hostname and a virtual IP are shared between the two nodes (the virtual hostname maps to the virtual IP in their subnet). However, only one node, the active node, can use these virtual settings at any one time. When the active node fails and the standby node is made active, the virtual IP is moved to the new active node. All requests to the virtual IP are now serviced by the new active node.

The OracleAS Cold Failover Cluster (Middle-Tier) can use the same machines as the OracleAS Cold Failover Cluster (Infrastructure) solution. In this scenario, two pairs of virtual hostnames and virtual IPs are used, one pair for the middle-tier cold failover cluster and one pair for the OracleAS Cold Failover Cluster (Infrastructure) solution. Figure 5–6 illustrates such a scenario. In this set up, the middle-tier components can failover independently from the OracleAS Infrastructure.

No shared storage is required for the middle-tier cold failover cluster except in the case where Oracle Application Server JMS file-based persistence is used (a shared disk for storing the persistence files should be set up).

Each node must have an identical mount point for the middle-tier software. One installation for the middle-tier software must be done on each node, and both installations must have the same local Oracle home path.

---

**Note:** For instructions on installing the OracleAS Cold Failover Cluster (Middle-Tier), see the *Oracle Application Server Installation Guide*. For instructions on managing it, see Section 4.7, "Managing OracleAS Cold Failover Cluster (Middle-Tier)" on page 4-31.

---

#### 3.1.2.1.1 Managing Failover

The typical deployment expected for the solution is a two-node hardware cluster with one node running the OracleAS Infrastructure and the other running the Oracle

Application Server middle-tier. If either node needs to be brought down for hardware or software maintenance or crashes, the surviving node can be brought online, and the OracleAS Infrastructure or Oracle Application Server middle-tier service can be started on this node.

However, since a typical middle-tier cold failover deployment does not require any shared storage (except for when OracleAS JMS file persistence is used), alternate deployments may include two standalone machines on a subnet, each with a local installation of the middle-tier software and a virtual IP which can failover between them.

The overall steps for failing over to the standby node are as follows:

1. Stop the middle-tier service on current primary node (if node is still available).
2. Fail over the virtual IP to the new primary node.
3. If OracleAS JMS file based persistence is using a shared disk for the messages, the shared disk is failed over to the new primary node.
4. Start the middle-tier service on the new primary node.

For failover management, two approaches can be employed:

- Automated failover using a cluster manager facility

The cluster manager offers services, which allows development of packages to monitor the state of a service. If the service or the node is found to be down, it automatically fails over the service from one node to the other node. The package can be developed to try restarting the service on a given node before failing over.

- Manual failover

For this approach, the failover steps outlined above are executed manually. Since both the detection of the failure and the failover itself is manual, this method may result in a longer period of service unavailability.

#### 3.1.2.1.2 OracleAS JMS in an OracleAS Cold Failover Cluster (Middle-Tier) Environment

OracleAS JMS can be deployed in an active-passive configuration by leveraging the two-node OracleAS Cold Failover Cluster (Middle-Tier) environment. In such an environment, the OC4J instances in the active node provide OracleAS JMS services, while OC4J instances in the passive node are inactive. OracleAS JMS file-based persistence data is stored in a shared disk.

Upon the failure of the active node, the entire middle-tier environment is failed over to the passive node, including the OracleAS JMS services and the shared disk used to persist messages. The OC4J instances in the passive node are started up together with other processes for the middle-tier environment to run. This node is now the new active node. OracleAS JMS requests are then serviced by this node from thereon.

**See Also:** Section 3.1.2.1, "Oracle Application Server Cold Failover Cluster (Middle-Tier)"

## 3.2 Highly Available Middle-tier Configuration Management Concepts

This section describes how configuration management can improve high availability for the middle tier. It covers the following:

- Section 3.2.1, "Oracle Application Server Clusters Managed Using DCM"
- Section 3.2.2, "Manually Managed Oracle Application Server Clusters"

### 3.2.1 Oracle Application Server Clusters Managed Using DCM

Distributed Configuration Management (DCM) is a management framework that enables you to manage the configurations of multiple Oracle Application Server instances. When administering an OracleAS Cluster that is managed using DCM, an administrator uses either Application Server Control Console or `dcmctl` commands to manage and configure common configuration information on one Oracle Application Server instance. DCM replicates the common configuration information across all Oracle Application Server instances within the OracleAS Cluster. The common configuration information for the cluster is called the cluster-wide configuration.

---

**Note:** There is configuration information that can be configured individually, per Oracle Application Server instance within a cluster (these configuration options are also called instance-specific parameters).

---

This section covers the following:

- Section 3.2.1.1, "What is a DCM-Managed OracleAS Cluster?"
- Section 3.2.1.2, "Oracle Application Server DCM Configuration Repository Types"

#### 3.2.1.1 What is a DCM-Managed OracleAS Cluster?

A DCM-Managed OracleAS Cluster provides distributed configuration information and lets multiple Oracle Application Server instances be configured together.

The features of a DCM-Managed OracleAS Cluster include:

- Synchronization of configuration across instances in the DCM-Managed OracleAS Cluster.
- OC4J distributed application deployment – deploying to one OC4J triggers deployment to all OC4Js.
- Distributed diagnostic logging – all members of a DCM-Managed OracleAS Cluster log to same the same log file repository when the log loader is enabled.
- A shared OC4J island is setup by default. (Replication is not enabled automatically for the applications deployed in the cluster, each application needs to be marked as "distributable" in its `web.xml` file, and multicast replication needs to be enabled in the replication properties for the OC4J instance).
- Load-balancing – Oracle HTTP Server is automatically configured to share load among all DCM-Managed OracleAS Cluster members.
- Distributed process control – DCM-Managed OracleAS Cluster membership enables the `opmnctl` DCM-Managed OracleAS Cluster scope start, stop, and restart commands.

Each application server instance in an DCM-Managed OracleAS Cluster has the same base configuration. The base configuration contains the cluster-wide configuration and excludes instance-specific parameters.

**See Also:**

- Section 4.2, "Using DCM-Managed OracleAS Clusters" on page 4-2
- Section 4.2.6, "Understanding DCM-Managed OracleAS Cluster Membership" on page 4-13

For Oracle Application Server high availability, when a system in an Oracle Application Server cluster is down, there is no single point of failure for DCM. DCM remains available on all the available nodes in the cluster.

Using DCM helps reduce deployment and configuration errors in a cluster; these errors could, without using DCM, be a significant cause of system downtime.

Enterprise Manager uses DCM commands to perform application server configuration and deployment. You can also issue DCM commands manually using the `dcmtcl` command.

DCM enables the following configuration commands:

- Create or remove a cluster
- Add or remove application server instances to or from a cluster
- Synchronize configuration changes across application server instances

Note the following when making configuration changes to a cluster or deploying applications to a cluster:

- If Enterprise Manager is up and managing the cluster, you can invoke the DCM command-line tool from any host where a clustered application server instance is running. The DCM daemon must be running on each node in the cluster.
- If Enterprise Manager is not up and managing the cluster, if you want configuration changes to be applied dynamically across the cluster, the DCM daemon must be running on each node in the cluster. To start the DCM daemon, run the DCM command-line tool, `dcmtcl`, on each application server instance in the cluster.

**See Also:** *Distributed Configuration Management Administrator's Guide*

### 3.2.1.2 Oracle Application Server DCM Configuration Repository Types

Oracle Application Server supports two types of DCM configuration repositories: Database-based and File-based DCM configuration repositories. The DCM configuration repository stores configuration information and metadata related to the instances in an OracleAS Farm, and when the OracleAS Farm contains DCM-Managed OracleAS Clusters, stores both cluster-wide configuration information and instance-specific parameters for instances in DCM-Managed OracleAS Clusters.

- An OracleAS Database-based Farm stores repository information and protects configuration information using an Oracle database.
- An OracleAS File-based Farm stores repository information in the file system, when the Farm contains a DCM-Managed OracleAS Cluster, the DCM configuration repository stores both cluster-wide configuration information and instance-specific parameters. Using an OracleAS File-based Farm, cluster-wide configuration information and related metadata is stored on the file system of an Oracle Application Server instance that is the **repository host** (host). Oracle Application Server instances that are part of a OracleAS File-based Farm depend on the repository host to store cluster-wide configuration information.

**See Also:** *Distributed Configuration Management Administrator's Guide*

### 3.2.2 Manually Managed Oracle Application Server Clusters

Using a Manually Managed OracleAS Cluster, it is the administrator's responsibility to synchronize the configuration of Oracle Application Server instances within the OracleAS Cluster. A full discussion of Manually Managed OracleAS Clusters can be found in Appendix B, "Manually Managed Oracle Application Server Cluster".

### 3.3 Middle-tier Backup and Recovery Considerations

Once a failure has occurred in your system, it is important to recover from that failure as quickly as possible. Depending on the type of failure, recovery of a middle-tier installation involves one or both of the following tasks:

- restart processes
- restore middle-tier files, which are:
  - Oracle system files
  - Oracle software files
  - configuration files

---

---

**Note:** The *Oracle Application Server Administrator's Guide* contains all required backup and recovery strategies and procedures.

---

---

The restoration of middle-tier files can be done from backups made using procedures described in the "Backup Strategy and Procedures" chapter of the *Oracle Application Server Administrator's Guide*. The backups encompass both the middle-tier and Infrastructure installations and are performed Oracle home by Oracle home. Thus, the restoration of the middle tier is also performed Oracle home by Oracle home. Each restoration can be done on the same node that the backup was taken from or on a new node. The "Recovery Strategies and Procedures" chapter in the *Oracle Application Server Administrator's Guide* provide details on using backups for recovery.

Restoration of a middle-tier installation on the same node restores the Oracle home, Oracle Application Server configuration files, and DCM repository. The backup of the Oracle home and configuration files is done when performing a complete Oracle Application Server environment backup, which is a backup of the entire Oracle Application Server system. Additionally, any time stamped backups of the configuration files should be restored, if required.

Restoration of a middle-tier installation on a new node requires the restoration of Oracle system files, the middle-tier Oracle home, and configuration files. Because the host is new, the DCM-managed and non DCM-managed components have to be updated with the host information.

**See Also:**

- *Distributed Configuration Management Administrator's Guide*
- Section 4.3.2.7, "Best Practices for Repository Backups" on page 4-21

## 3.4 Middle-tier Applications

This section provides high availability information on middle-tier application components, which are:

- Section 3.4.1, "Oracle Application Server Portal"
- Section 3.4.2, "Oracle Application Server Wireless"
- Section 3.4.3, "OracleAS Integration B2B"
- Section 3.4.4, "Oracle Application Server Integration InterConnect"
- Section 3.4.5, "Oracle Business Intelligence Discoverer"

### 3.4.1 Oracle Application Server Portal

An OracleAS Portal request's lifecycle is serviced by a number of OracleAS components. These are:

- OracleAS Web Cache
- Oracle HTTP Server and the following modules:
  - `mod_oc4j` (on middle and Infrastructure tiers)
  - `mod_osso` (on Infrastructure tier to access OracleAS Single Sign-On)
  - `mod_plsql` (on middle tier with OracleAS Portal DAD and Infrastructure tier with ORASSO DAD)
  - `mod_oradav` (on middle tier)
- OC4J (the Portal Page Engine runs as a stateless servlet)
- OracleAS Portal repository (contains OracleAS Portal schemas and also caches group memberships of users after their retrieval from Oracle Internet Directory)
- OracleAS Single Sign-On
- Oracle Internet Directory (including Oracle Delegated Administration Services and Oracle Directory Integration and Provisioning)
- Various web and database portlet providers

In order for OracleAS Portal to be highly available, all these components must be highly available individually. Of particular importance is the availability of Oracle Identity Management because OracleAS Portal uses it for portlet security and management functions.

Reference the following table to find out where you can find high availability information for each of the components mentioned above.

**Table 3-5 High availability information for components involved in an OracleAS Portal request**

Component	Where to find information
OracleAS Web Cache	See Section 3.1.1.1, "Oracle Application Server Web Cache Sub-Tier" on page 3-2.
Oracle HTTP Server	See: Section 3.2, "Highly Available Middle-tier Configuration Management Concepts" on page 3-16 Section 3.1.1.2, "Oracle HTTP Server Sub-Tier" on page 3-3

**Table 3–5 (Cont.) High availability information for components involved in an OracleAS Portal request**

Component	Where to find information
OC4J	See: Section 3.1.1.3, "Oracle Application Server Containers for J2EE Sub-Tier" on page 3-7  Note: The Portal Page Engine is stateless.
OracleAS Portal repository	See: Chapter 5, "Oracle Application Server Infrastructure High Availability" and Chapter 6, "Managing and Operating Infrastructure High Availability" of this book.  <i>Oracle Application Server Portal Configuration Guide</i>
OracleAS Single Sign-On	See: Chapter 5, "Oracle Application Server Infrastructure High Availability" and Chapter 6, "Managing and Operating Infrastructure High Availability" of this book.  <i>Oracle Identity Management Concepts and Deployment Planning Guide</i>
Oracle Internet Directory	See: Chapter 5, "Oracle Application Server Infrastructure High Availability" and Chapter 6, "Managing and Operating Infrastructure High Availability" of this book.  <i>Oracle Internet Directory Administrator's Guide</i>  <i>Oracle Identity Management Concepts and Deployment Planning Guide</i>
Web Provider	See: Section 3.2, "Highly Available Middle-tier Configuration Management Concepts" on page 3-16 Section 3.1.1.2, "Oracle HTTP Server Sub-Tier" on page 3-3 Section 3.1.1.3.1, "OracleAS Cluster (OC4J)" on page 3-7 Section 3.1.1.1, "Oracle Application Server Web Cache Sub-Tier" on page 3-2 (OracleAS Web Cache could be providing access to the provider)
Database Providers	For providers using <code>mod_plsql</code> , Oracle HTTP Server high availability is relevant. See Section 3.1.1.2, "Oracle HTTP Server Sub-Tier" on page 3-3 and Section 3.2, "Highly Available Middle-tier Configuration Management Concepts" on page 3-16.  For database high availability, see Chapter 5.  See also: Section 3.1.1.1, "Oracle Application Server Web Cache Sub-Tier" on page 3-2 (OracleAS Web Cache could be providing access to the provider)

**See Also:** *Oracle Application Server Portal Configuration Guide*

The following are several considerations when deploying OracleAS Portal for high availability:

- Section 3.4.1.1, "Enabling Redundancy for OracleAS Portal"
- Section 3.4.1.2, "Configuring Load Balancer Routers for OracleAS Portal"
- Section 3.4.1.3, "Session Binding for Web Clipping Portlet"

- Section 3.4.1.4, "OracleAS Portal and OracleAS Web Cache"

### 3.4.1.1 Enabling Redundancy for OracleAS Portal

For redundancy, you can set up OracleAS Portal in a multiple middle-tier environment, front-ended by a load balancing router (LBR) to access the same OracleAS Metadata Repository.

The purpose of a Load Balancing Router (LBR) is to provide a single published address to the client tier and to front-end a farm of servers that actually service the requests, based on the distribution of the requests done by the LBR. The LBR itself is a very fast network device that can distribute Web requests to a large number of physical servers.

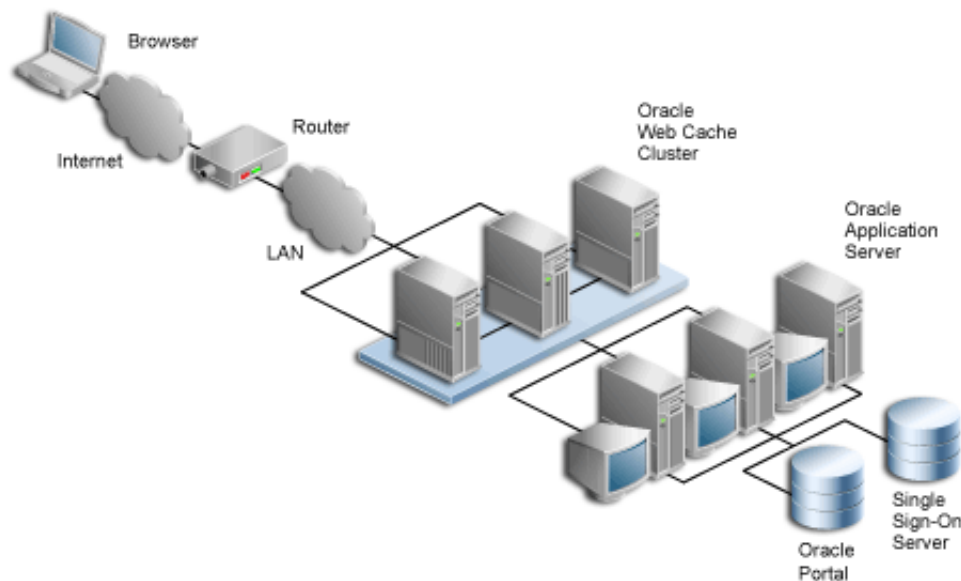
For full details on how you can configure multiple middle tiers using a LBR, see the section "Configuring Multiple Middle Tiers with a Load Balancing Router" in the *Oracle Application Server Portal Configuration Guide*.

### 3.4.1.2 Configuring Load Balancer Routers for OracleAS Portal

The purpose of a Load Balancing Router (LBR) is to provide a single published address to the client browsers, and provide a "farm" of Web servers which actually service the requests, based on the distribution of the requests done by the LBR. The LBR itself is a very fast network device which can distribute Web requests to a large number of physical servers.

If you want to install multiple OracleAS middle-tier servers to handle a large load, you could configure OracleAS Portal as illustrated in Figure 3–4.

**Figure 3–4 Redundant configuration for OracleAS Portal**



This example shows that the LBR balances the load between all three instances of the OracleAS Web Cache cluster. Each one of the OracleAS Web Cache clusters can in turn load balance any of the middle-tier servers which communicate with OracleAS Single Sign-On Server and OracleAS Portal.

In this example, assume that the three OracleAS Web Cache instances are *wc1*, *wc2*, and *wc3*, and the OracleAS middle-tier servers are *svr1*, *svr2*, and *svr3*. Hence, in the



above example, wc1 can load balance svr1, svr2, as well svr3. wc2 and wc3 can also do the same.

All the OracleAS middle-tier servers must have Database Access Descriptor (DAD) entries for each of the databases. A good way to accomplish this is to have the middle-tier servers share a file system that contains the configuration information for the DADs, so that the OracleAS Portal instances can share cache files.

The important points to consider with this configuration include:

- The Internet DNS maps the name `www.myportal.com` to the external IP address on the LBR.
- The LBR performs load balancing of requests to `www.myportal.com` to `svr1.company.com`, `svr2.company.com`, and `svr3.company.com`, addressing the request to their IP addresses, but still containing `www.myportal.com` in the `Host:` field of the HTTP request.
- Each of the middle-tier hosts accepts requests to `www.myportal.com`, and their `httpd.conf` files assert that name as the `ServerName`. Hence, the names `svr1`, `svr2`, and so on are not used.
- Unless your LBR does port mapping, you should configure the internal servers to use the same ports as the LBR.
- Optimal cache utilization can be realized by mounting a shared file system on which to write the cache files. If you decide not to have the middle-tier servers share a cache directory, caching will still work, but with a lower hit ratio

**See Also:** *Oracle Application Server Enterprise Deployment Guide*

### 3.4.1.3 Session Binding for Web Clipping Portlet

The session binding feature in OracleAS Web Cache is used to bind user sessions to a given origin server to maintain state for a period of time. Although almost all components running in a default OracleAS Portal middle-tier are stateless, session binding is required for two reasons:

- The Web Clipping Studio, used by both the OracleAS Web Clipping Portlet and the Web Page Data Source of OmniPortlet, uses HTTP session to maintain state, for which session binding must be enabled.
- Enabling session binding forces all the user requests to go to a given OracleAS Portal middle-tier, resulting in a better cache hit ratio for the OracleAS Portal cache.

**See Also:** *Oracle Application Server Portal Configuration Guide*

### 3.4.1.4 OracleAS Portal and OracleAS Web Cache

An OracleAS Portal installation includes an installation of OracleAS Web Cache. When deploying multiple OracleAS Portal middle-tiers, the OracleAS Web Cache installations should be configured as a Oracle Application Server Cluster (Web Cache).

This allows Invalidation Requests from an OracleAS Portal instance to be automatically sent to all OracleAS Web Cache instances in the OracleAS Cluster (Web Cache).

**See Also:** *Oracle Application Server Web Cache Administrator's Guide* for information on configuring OracleAS Cluster (Web Cache).

## 3.4.2 Oracle Application Server Wireless

OracleAS Wireless runs as a native OC4J application. This means that availability and session replication are managed by OPMN and OracleAS Cluster (OC4J) state replication respectively. Additionally, the messaging servers run as standalone Java applications, which are managed by Application Server Control Console.

### 3.4.2.1 OracleAS Wireless Clustering Architecture

Each OracleAS Wireless server process which runs on a single Java Virtual Machine (JVM) is referred to as a node. Nodes within an OracleAS Cluster (OC4J) are capable of serving the same wireless applications, because the session for each client is replicated among all the nodes within an OracleAS Cluster (OC4J) in preparation for failover.

By default, the requests from the same client are always redirected to the same Wireless server process. If one process goes down, then the fault tolerance feature is supported for both stateful and stateless requests as follows:

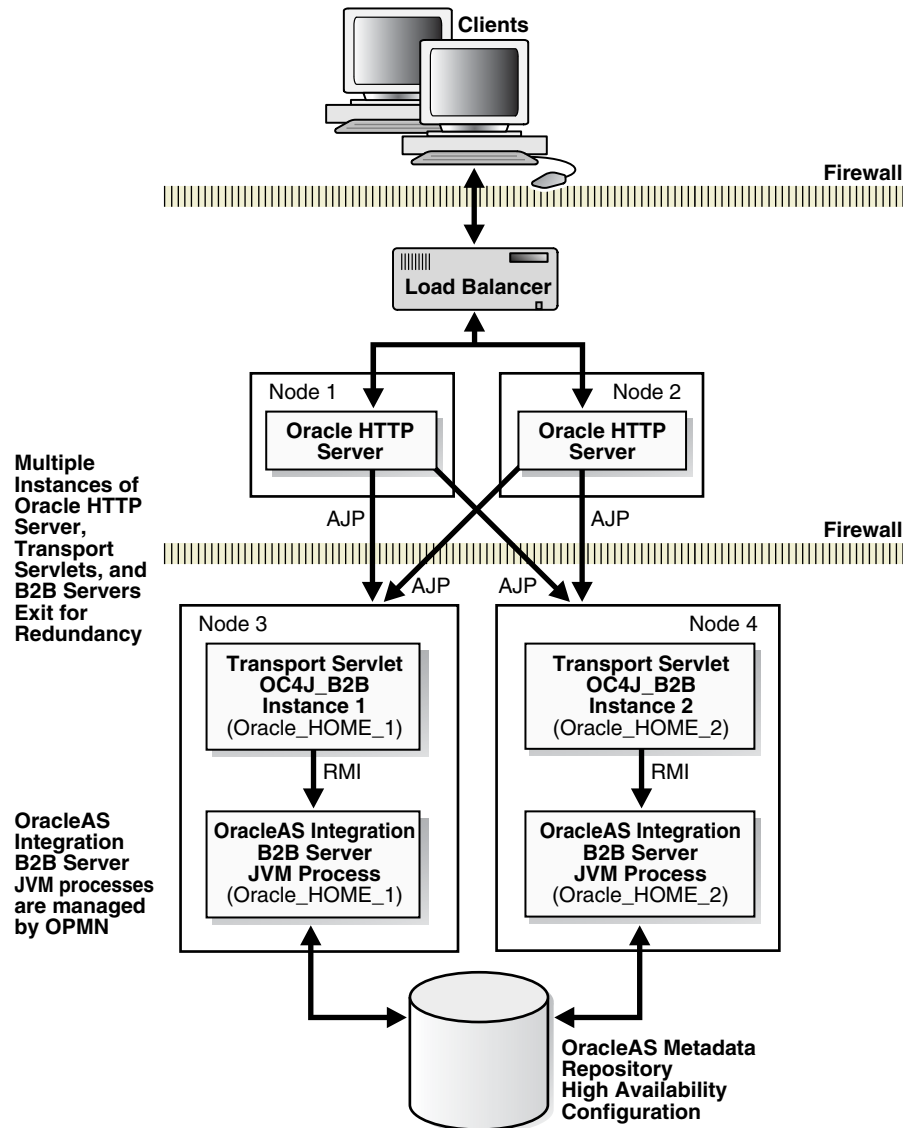
- Stateless Requests - Fault tolerance is achieved by redirecting the client to another working process.
- Stateful Requests - The session state is propagated to the processes within the same OracleAS Cluster (OC4J), which enables another process in that same OracleAS Cluster (OC4J) to pick up the request from a given client if a failover occurs.

For detailed configuration steps, refer to the *Oracle Application Server Wireless Administrator's Guide*.

## 3.4.3 OracleAS Integration B2B

OracleAS Integration B2B employs several components from the Oracle Application Server stack at runtime. These include Oracle HTTP Server, Oracle Application Server Containers for J2EE, and OracleAS Metadata Repository. The OracleAS Integration B2B high availability configuration is depicted in Figure 3-5.

**Figure 3-5 High availability configuration of OracleAS Integration B2B**



In order for OracleAS Integration B2B services to be highly available, the following components must be highly available:

- Oracle HTTP Server
- OC4J transport servlet
- OracleAS Integration B2B server JVM
- OracleAS Infrastructure

For discussion purposes, the runtime architecture can be segmented into the following tiers:

- **Web Server and OC4J Tier**
- **OracleAS Integration B2B Tier**
- **OracleAS Infrastructure Tier**

If each of these tiers has active-active availability, the OracleAS Integration B2B service has active-active availability. Otherwise, if one of the tiers is active-passive, the

OracleAS Integration B2B service is active-passive. For example, if the OracleAS Infrastructure tier uses the OracleAS Cold Failover Cluster (Infrastructure) configuration, the OracleAS Integration B2B service has active-passive availability.

### **Web Server and OC4J Tier**

This tier consists of Oracle HTTP Server and the OC4J transport servlet instances. The servlets are deployed in OC4J containers and can utilize the high availability properties of the containers. They can be grouped together into OracleAS Clusters (OC4J) and be synchronized by DCM for consistent configuration. The OC4J instances are load balanced by mod\_oc4j.

For active-active availability, the web server and OC4J tier is front-ended by a load balancer router appliance and/or OracleAS Web Cache. If OracleAS Web Cache is used, it should be configured into an OracleAS Cluster (Web Cache). Monitoring and automatic restart of OracleAS Web Cache, Oracle HTTP Server, and OC4J processes are performed by OPMN.

The transport servlets perform the tasks of forwarding requests and receiving responses from the OracleAS Integration B2B instances. The servlets do not maintain state for each request handled. They communicate with the OracleAS Integration B2B instances through Java RMI. Each instance of OracleAS Integration B2B is registered in the `web.xml` file of each of the OC4J containers hosting the transport servlets. The servlets forward requests to the OracleAS Integration B2B instances using the round-robin model. If any of the OracleAS Integration B2B instances fail, the servlets re-route requests to the next instance in the round-robin queue after a specified timeout period.

### **OracleAS Integration B2B Tier**

The OracleAS Integration B2B tier consists of the OracleAS Integration B2B server runtime. This is a Java application, but its instances do not run in OC4J containers. They run in their own standalone JVM processes.

The OracleAS Integration B2B server has the following characteristics:

- Its runtime is stateless for each request it processes. If a runtime process fails and a request message is not completely processed, the client is expected to retry the request. If the failure occurs after the initial message has been completely processed, all subsequent incomplete processing results are stored in the database, and any other runtime instances can resume processing. Each processing step is atomic.
- It uses JDBC to access the OracleAS Metadata Repository to make changes to the OracleAS Integration B2B metadata schemas. High availability for JDBC connections is achieved by Oracle Net.
- Only one runtime is instantiated from each OracleAS Integration B2B Oracle home.
- Only one runtime instance exists for each OracleAS instance.

To ensure that the server has active-active availability, multiple instances of its runtime should be instantiated. Ideally, these instances should be deployed in more than one node to protect from node failure. For each instance, OPMN ensures that failure detection and automatic restart of each instance is managed.

Inbound communication to the OracleAS Integration B2B instances is received by the load balancer fronting the Oracle HTTP Servers. The load balancer distributes requests to the Oracle HTTP Server instances, which forwards the requests to the transport

servlets via `mod_oc4j` load balancing. The transport servlets communicate the requests to the OracleAS Integration B2B instances using the RMI protocol.

Outbound communication from the OracleAS Integration B2B instances occurs as follows. The instances send responses to the Oracle HTTP Servers, which are configured as proxy servers. This configuration can be accomplished by specifying the proxy host and port properties in the `tip.properties` file.

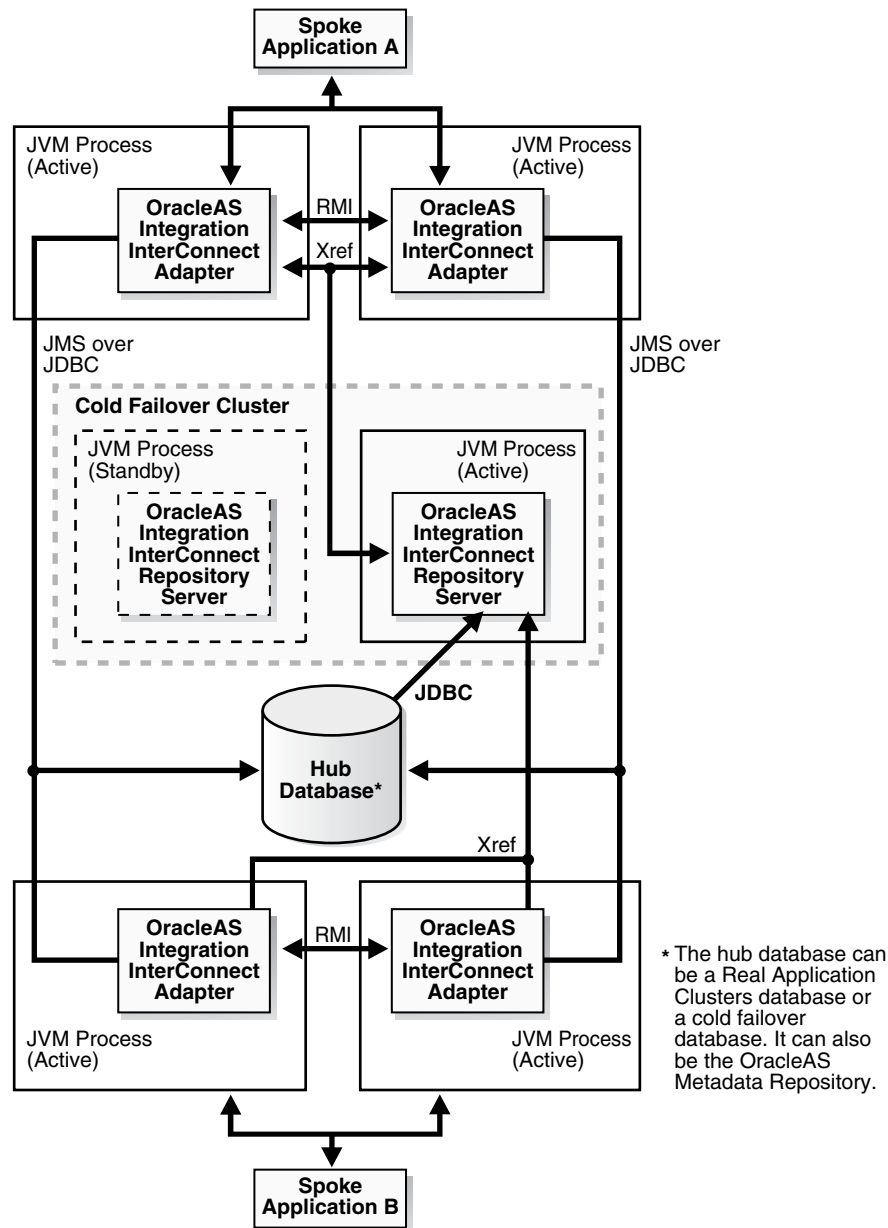
#### **OracleAS Infrastructure Tier**

High availability in the Infrastructure tier can be enabled by any of the high availability configurations for the OracleAS Infrastructure explained in Chapter 5, "Oracle Application Server Infrastructure High Availability". These configurations ensure that the OracleAS Metadata Repository and Oracle Identity Management components are highly available for the web server and OC4J, and OracleAS Integration B2B tiers. For active-active availability, one of the configurations described in Section 5.3.1, "Active-Active High Availability Solutions", should be used. This allows the entire OracleAS Integration B2B service stack to have active-active availability.

#### **3.4.4 Oracle Application Server Integration InterConnect**

OracleAS Integration InterConnect has a hub and spoke architecture. Figure 3–6 provides an overview of the OracleAS Integration InterConnect components integrating two spoke applications as an example.

**Figure 3-6 OracleAS Integration InterConnect runtime components with two spoke application as an example**



The OracleAS Integration InterConnect components are:

- OracleAS Integration InterConnect Adapters
- OracleAS Integration InterConnect Repository Server
- OracleAS Integration InterConnect Hub database

For OracleAS Integration InterConnect to have high availability, all its components must be highly available. One additional requirement is for the data or message sources that provide information to the adapters to be highly available. These are the spoke applications. Because these applications are customer-dependent and not part of

the Oracle Application Server product, their high availability discussion is outside the scope of this book.

For the purpose of high availability discussion, the OracleAS Integration InterConnect components can be segmented into the following tiers:

- Adapter Tier
- Repository Server Tier
- Hub Database Tier

The following sections provide details on how high availability can be achieved for each tier.

**See Also:** OracleAS Integration InterConnect documentation for detailed information about OracleAS Integration InterConnect components.

### Adapter Tier

Except for the HTTP adapter, each adapter runs in a standalone JVM process (not OC4J) and is stateless. This JVM process can be configured as a custom OPMN application to achieve process failure detection and automatic restart. The custom application can be configured in the `opmn.xml` file. Refer to the *Oracle Process Manager and Notification Server Administrator's Guide* for instructions on how to do this. After the configuration, the adapter processes should be started using OPMN (`opmnctl` command).

OPMN only monitors and restarts individual processes. In order for the adapter tier to be fully redundant, multiple adapter processes are required. The adapters can be set up using an active-active or active-passive approach:

- Active-Active

Multiple active adapter processes can be deployed either on the same machine or separate machines. The adapter processes process incoming messages from the spoke application and deliver messages to the hub database concurrently. In the event that one adapter process fails, messages are delivered to the surviving process or processes. The adapters coordinate with each other to balance their workload from the spoke application.

- Active-Passive

Two adapter processes can be deployed in a cold failover cluster configuration to achieve active-passive availability.

In a cold failover cluster, two machines can be clustered together using clusterware such as HP MC/Service Guard or Sun Cluster. This type of clustering is a commonly used solution for making adapters highly available. One node of the cluster is "cold", passively waiting to take over in the event of a failure, while the other is "hot", or actively running the adapter software. When the "hot" or "active" node fails, the clusterware restarts the software on the cold node to bring the adapter back online. Figure 3–6 shows a cold failover cluster for the adapters.

If the hub database is a Real Application Clusters database, the adapters are enabled to work with the multiple database instances in the Real Application Clusters. Real Application Clusters technology provides consistent and uninterrupted service without having to restart the adapters if a database instance fails. The adapters connect to the first of the listed available nodes in the `adapter.ini` or `hub.ini` files. If one of the Real Application Clusters nodes fails, the database connection is established with the next available node in the `adapter.ini` or `hub.ini` file recursively until a

successful connection. Failover is transparent to the spoke application. Refer to the "Hub Database Tier" section below for more information on how the adapter process can be made aware of Real Application Clusters hub database instances.

**See Also:** OracleAS Integration InterConnect adapters installation documentation for details on `adapter.ini` and `hub.ini` files associated with specific adapters.

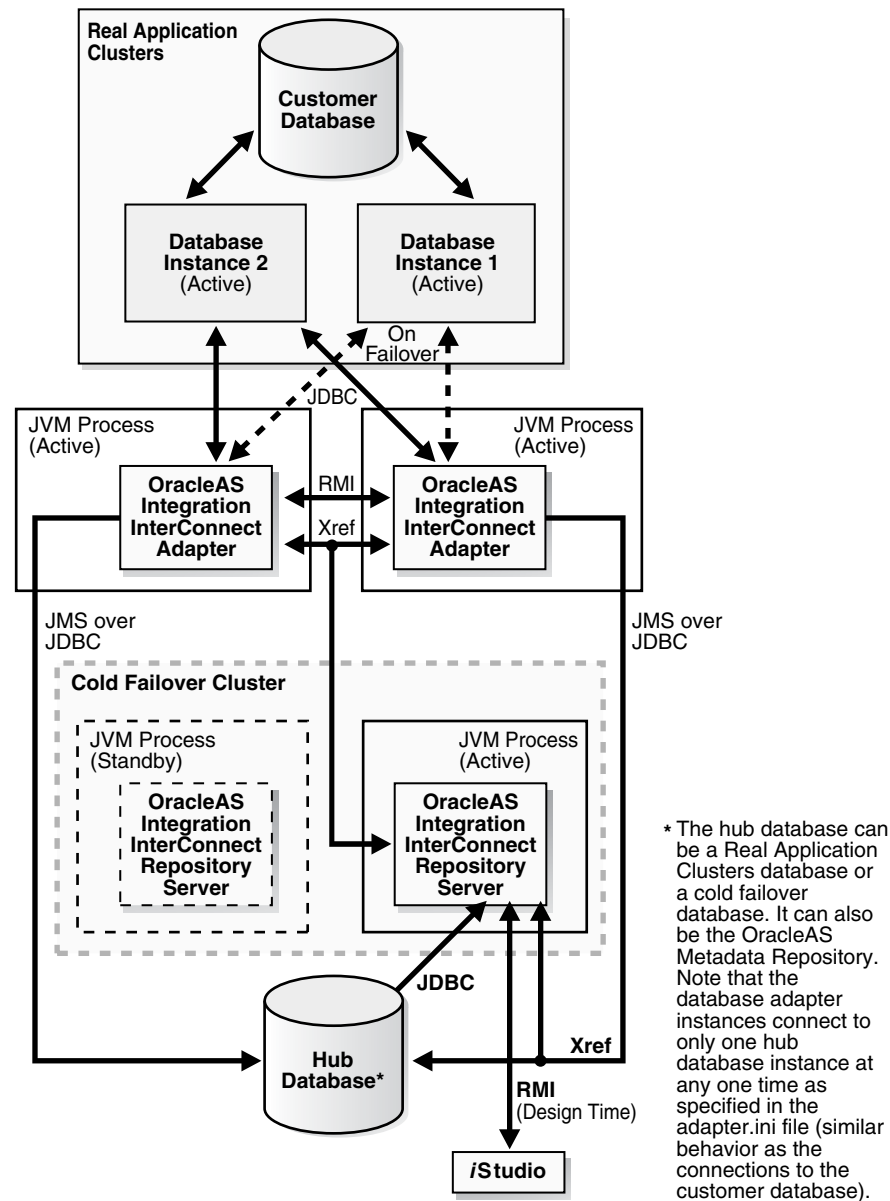
High availability specific to each adapter type can be achieved as follows:

- **Database Adapter**

You can deploy multiple database adapter instances serving the same application (Figure 3-7 shows an example). Since the adapters are stateless, they share tasks coming from their spoke application, the customer database. Connection failure handling to a database is done by rolling back unfinished transactions and retrying other data sources. The same connection failure handling mechanism is also used for JMS communication (JMS over JDBC) to the Advanced Queue in the hub database.



Figure 3-7 Example of multiple database adapter instances (showing only single spoke)



After an application is designed, it can be deployed to the database adapters when the adapters are first started. Upon initial startup, the adapters fetch metadata from the hub database through the OracleAS Integration InterConnect Repository Server, similar to the way OracleAS Integration InterConnect iStudio accesses the data at design time. Once these metadata are retrieved by the adapters, they are cached locally on a file-based cache. Thus, subsequent adapter startup does not need to access the Repository Server.

At run time, the database adapters access the customer database through JDBC. There can be multiple JDBC data sources, which the adapters iterate through should a connection fail. Tasks from the customer database are processed by all database adapter instances in coordination so that a task is only processed once. In order to have readily available data from the customer database, the database should be Real Application Clusters-enabled or be in a cold failover configuration

(applies to the hub database as well). The adapters communicate with the Repository Server at run time to implement the Xref feature.

- **HTTP adapter:**

The HTTP adapter consists of a standalone Java application, an OC4J transport servlet, and Oracle HTTP Server. The Java application implements the HTTP adapter logic and communicates with a servlet in OC4J using the RMI protocol. Each Java application process communicates with only one OC4J process. The Oracle HTTP Server is required to communicate with spoke applications over HTTP.

For high availability, more than one set of Oracle HTTP Server, OC4J, and Java application process should be deployed on redundant nodes. `mod_oc4j` load balancing can be used to distribute requests between Oracle HTTP Server and OC4J instances across nodes. But communication between OC4J instances and the Java application processes is one-to-one. A load balancer router can be deployed in front of the Oracle HTTP Server instances to distribute requests to these instances.

The HTTP adapter Java application also communicates with the hub database and Repository Server. The Java application works with these components the same way as the database adapter as described above. The hub database should be made highly available through using a Real Application Clusters database or cold failover cluster configuration. The Repository Server can be made highly available using a cold failover cluster configuration.

In the event a HTTP adapter process fails, an inbound message to the adapter process (servlet to adapter) can be lost if the message is still in the transport or RMI layer. But once the message arrives at the adapter agent layer, the message is persisted and can be picked up later when the adapter is restarted. Also, the transport servlet will not be able to enqueue any other messages to the adapter process as the RMI server fails with the adapter process. These messages in the OC4J process will not be processed as the transport servlet's `doPost()` method will respond with an error message stating that the RMI server is unavailable.

- **FTP/SMTP adapter:**

To achieve high availability for the FTP/SMTP adapter, multiple adapter instances can be deployed on separate machines with a load balancer routing requests to them. Since adapters process messages atomically and are stateless, if any one of the adapter instances fail, the redundant deployment allows the failure to be transparent to senders and recipients of messages.

- **MQ/AQ adapter:**

High availability specifics of this adapter are similar to those of the database adapter. This is because access to the MQ/AQ database is also through JDBC (JMS). Refer to the database adapter description above.

- **File Adapter:**

For the file adapter to achieve high availability, multiple adapter instances are required to access a network file system. If one adapter instance fails, another instance can process the requests for the failed instance.

- **OEM Adapters:**

The OEM adapter model is similar to that of the HTTP adapter, that is, it has an OC4J transport servlet, Oracle HTTP Server, and a standalone Java application. The Java application implements the adapter logic and communicates with a servlet in OC4J using the RMI protocol. Each Java application process

communicates with only one OC4J process. The Oracle HTTP Server is required to communicate with spoke applications over HTTP.

### Repository Server Tier

This tier consists of the Repository Server instance. Only a single instance can be actively running at any one time. Hence, the Repository Server can be deployed in a two-node cold failover cluster configuration with the nodes using shared storage. This configuration provides for node-level failover.

For Repository Server process high availability, the process can be configured as a custom application for OPMN in the `opmn.xml` file. This allows OPMN to monitor and automatically restart the Repository Server process if it fails. After the modification, the Repository Server process should be started using OPMN (`opmnctl` command).

The repository server is only used at run time for the Xref feature. Otherwise, it is only needed during design time and deployment time, when adapters are first started and fetch application metadata from the hub database).

### Hub Database Tier

The hub database can be any database, including the OracleAS Metadata Repository database. It stores OracleAS Integration InterConnect metadata such as application view and common view formats. iStudio accesses the hub database at design time through RMI via the Repository Server, which is a JVM process. The Repository Server can communicate with multiple hub database instances as multiple JDBC data sources. Internally, the Repository Server iteratively retries each data source with timeouts.

The OracleAS Integration InterConnect hub database can be made highly available by using Real Application Clusters. The following are some guidelines:

- Enable the Repository Server process to be aware of the Real Application Clusters hub database instances.

The Repository Server process can be made aware of the Real Application Clusters database instances by specifying the list of available nodes hosting the database instances. Specifically, enter the host, port, and instance information of all the nodes in the `repository.ini` or `hub.ini` file. If a Real Application Clusters node connected to the Repository Server process fails, then the next node entry in the `repository.ini` and `hub.ini` file is used.

- Enable the adapter processes to be aware of the Real Application Clusters hub database instances.

The adapter processes can be made aware of the Real Application Clusters database instances by specifying the list of available nodes hosting the database instances. Specifically, enter the host, port, and instance information of all the nodes in the `adapter.ini` file. If a node connected to an adapter process fails, the next node entry in the `adapter.ini` file is used.

The hub connections of all the OracleAS Integration InterConnect adapters and the spoke connections of the database and AQ adapters support Real Application Clusters.

**See Also:** OracleAS Integration InterConnect adapters installation documentation for details on `adapter.ini` and `hub.ini` files associated with specific adapters.

### 3.4.5 Oracle Business Intelligence Discoverer

Web connections to OracleBI Discoverer Server are managed through the Discoverer servlet. The servlet is responsible for brokering between the client and a Discoverer session component that then manages the actual transactions. Discoverer session components are initiated and managed by the OAD (Object Activation Daemon). Each machine has an OAD that manages its own Discoverer sessions. The OAD and session component are both monitored and managed by OPMN.

Oracle Business Intelligence Discoverer can be configured for high availability in the following ways:

- *Process monitoring and restart*

OPMN is configured to monitor and restart Oracle Business Intelligence Discoverer processes on each middle-tier node. See Chapter 4 of *Oracle Business Intelligence Discoverer Configuration Guide*.

- *Load balancing*

OracleAS Web Cache can be set up to perform as a load balancer for Oracle Business Intelligence Discoverer requests. See Chapter 5 of *Oracle Business Intelligence Discoverer Configuration Guide*.

**See Also:** The chapter on installing in a multi machine environment in the *Oracle Business Intelligence Discoverer Configuration Guide* for multi machine considerations and pre-requisites for providing load balancing for OracleBI Discoverer.

#### 3.4.5.1 Oracle Business Intelligence Discoverer Preferences Server

The OracleBI Discoverer Preference Server stores individual user preferences across sessions. It is managed, like the session server, by the OAD. In a multiple machine environment, distributed session servers can be configured to access one centrally located OracleBI Discoverer Preferences Server. The latter is monitored and managed by OPMN.

The OracleBI Discoverer Preferences Server can be made highly available by deploying multiple instances that are fronted and serviced by a load balancer router and/or OracleAS Web Cache. Several considerations should be noted for managing session information for this scenario.

When deploying multiple OracleBI Discoverer middle-tiers behind a load balancer, there are two options for configuring the OracleBI Discoverer Preferences Server such that user preferences are consistent across a session:

1. Enable session binding either in the load balancer router or in the OracleAS Web Cache tier. This will ensure that a particular user will always be directed to the machine where their local preferences are stored. For details on configuring session binding for your load balancer router, refer to instructions from your particular load balancer hardware vendor. For configuring OracleAS Web Cache session binding, see the *Oracle Application Server Web Cache Administrator's Guide*.
2. Configure all the OracleBI Discoverer Servers to share a single preference server. This ensures that all user preferences are centralized, although, all preference information is now dependent on the availability of one machine.

---

---

**Note:** For instructions on how to configure a centralized OracleBI Discoverer Preferences Server, see the chapter on installing in a multi machine environment in the *Oracle Business Intelligence Discoverer Configuration Guide*.

---

---

### **Protecting the OracleBI Discoverer Preferences Server**

Loss of either the machine which hosts the OracleBI Discoverer Preference Server or the information stored on that server will not impact availability of OracleBI Discoverer. It will, however, mean that users will lose their stored preferences information.

To limit the loss of preferences information, the data on the OracleBI Discoverer Preferences Server should be backed up regularly, in particular, the file <ORACLE\_HOME>/discoverer/util/pref.txt. This file holds the actual preferences information.



---

# Managing and Operating Middle-tier High Availability

This chapter describes how to perform configuration changes and on-going maintenance for the Oracle Application Server middle-tier.

This chapter covers the following topics:

- Section 4.1, "Middle-tier High Availability Configuration Overview"
- Section 4.2, "Using DCM-Managed OracleAS Clusters"
- Section 4.3, "Availability Considerations for the DCM Configuration Repository"
- Section 4.4, "Using Oracle Application Server Clusters (OC4J)"
- Section 4.5, "Using Oracle Application Server Clusters (Portal)"
- Section 4.6, "Using Oracle Application Server Clusters (Web Cache)"
- Section 4.7, "Managing OracleAS Cold Failover Cluster (Middle-Tier)"
- Section 4.8, "Managing Custom Processes With OPMN"
- Section 4.9, "Managing Oracle Application Server Middle-tier Upgrades"
- Section 4.10, "Using OracleAS Single Sign-On With OracleAS Cluster (Middle-Tier)"

## 4.1 Middle-tier High Availability Configuration Overview

Oracle Application Server provides different configuration options to support high availability for the Oracle Application Server middle-tier.

This section covers the following topics:

- Section 4.1.1, "DCM-Managed Oracle Application Server Clusters"
- Section 4.1.2, "Manually Managed Oracle Application Server Clusters"

### 4.1.1 DCM-Managed Oracle Application Server Clusters

When administering a DCM-Managed OracleAS Cluster, an administrator uses either Application Server Control Console or `dcmctl` commands to manage and configure common configuration information on one Oracle Application Server instance. DCM then propagates and replicates the common configuration information across all Oracle Application Server instances within the DCM-Managed OracleAS Cluster. The common configuration information for the cluster is called the cluster-wide configuration.

---

---

**Note:** There is configuration information that can be configured individually, per Oracle Application Server instance within a cluster (these configuration options are also called **instance-specific parameters**).

---

---

Each application server instance in an DCM-Managed OracleAS Cluster has the same base configuration. The base configuration contains the cluster-wide configuration and excludes instance-specific parameters.

**See Also:**

- Section 3.2.1, "Oracle Application Server Clusters Managed Using DCM" on page 3-17.
- Section 4.2.6, "Understanding DCM-Managed OracleAS Cluster Membership" on page 4-13

## 4.1.2 Manually Managed Oracle Application Server Clusters

Using a Manually Managed OracleAS Cluster, it is the administrator's responsibility to synchronize the configuration of Oracle Application Server instances within the OracleAS Cluster.

**See Also:** Appendix B, "Manually Managed Oracle Application Server Cluster"

## 4.2 Using DCM-Managed OracleAS Clusters

This section describes how to create and use a DCM-Managed OracleAS Cluster and covers the following topics:

- Section 4.2.1, "Creating DCM-Managed OracleAS Clusters"
- Section 4.2.2, "Adding Instances To DCM-Managed OracleAS Clusters"
- Section 4.2.3, "Removing Instances from DCM-Managed OracleAS Clusters"
- Section 4.2.4, "Starting Stopping and Deleting DCM-Managed OracleAS Clusters"
- Section 4.2.5, "Configuring Oracle HTTP Server Options for DCM-Managed OracleAS Clusters"
- Section 4.2.6, "Understanding DCM-Managed OracleAS Cluster Membership"

**See Also:** *Distributed Configuration Management Administrator's Guide* for information on `dcmctl` commands

### 4.2.1 Creating DCM-Managed OracleAS Clusters

An OracleAS Farm contains a collection of Oracle Application Server instances. In an OracleAS Farm, you can view a list of all application server instances when you start Application Server Control Console. The application server instances shown in the Standalone Instances area on the Application Server Control Console Farm Home Page are available to be added to DCM-Managed OracleAS Clusters.

Each Oracle Application Server Farm has the characteristic that it uses either a File Based Repository or a Database-Based Repository. The steps for associating an application server instance with an OracleAS Farm differ depending on the type of the respiratory.



This section covers the following:

- Section 4.2.1.1, "Associating An Instance With An OracleAS Database-based Farm"
- Section 4.2.1.2, "Associating An Instance With An OracleAS File-based Farm"
- Section 4.2.1.3, "Using the Application Server Control Console Create Cluster Page"

---



---

**Note:** This section covers procedures for clusterable middle-tier instances that are part of an OracleAS Farm. For purposes of this section, a clusterable instance is a middle-tier instance, where the `dcmctl isclusterable` command returns the value `true`.

---



---

#### 4.2.1.1 Associating An Instance With An OracleAS Database-based Farm

If you have not already done so during the Oracle Application Server installation process, you can associate an application server instance with an OracleAS Database-based Farm:

For an OracleAS Database-based Farm, do the following to add an application server instance to the OracleAS Farm:

1. Navigate to the Application Server Control Console Instance Home Page.
2. In the **Home** area, select the **Infrastructure** link and follow the instructions for associating an application server instance with an Oracle Application Server Infrastructure.

**See Also:** *Oracle Application Server Administrator's Guide*

#### 4.2.1.2 Associating An Instance With An OracleAS File-based Farm

This section covers the following topics:

- Section 4.2.1.2.1, "Creating An OracleAS File-based Farm Repository Host"
- Section 4.2.1.2.2, "Adding Instances To An OracleAS File-based Farm"

##### 4.2.1.2.1 Creating An OracleAS File-based Farm Repository Host

You can instruct the Oracle Application Server installer to create an OracleAS File-based Farm when you install Oracle Application Server. If you did not create an OracleAS File-based Farm during installation, then you can create the OracleAS File-based Farm with the following steps.

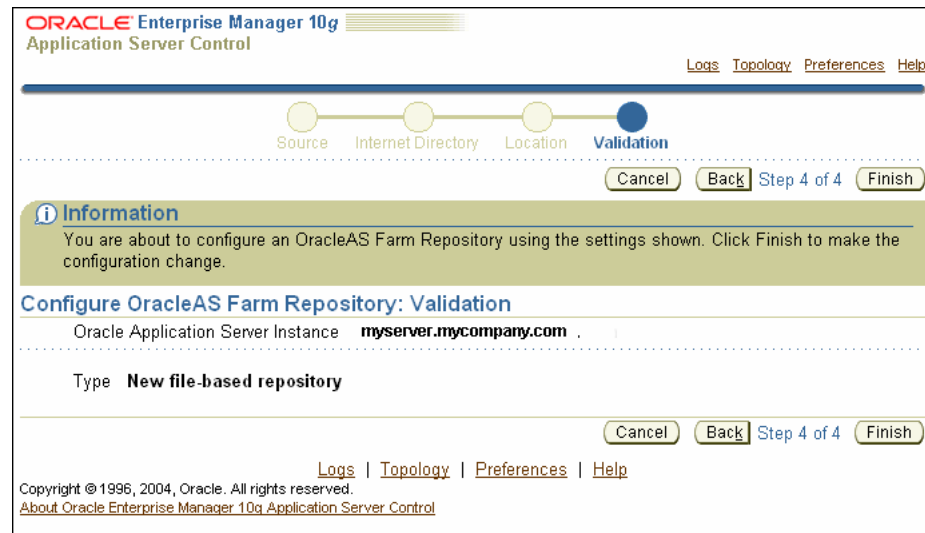
1. Using the Application Server Control Console for the instance that you want to use as the repository host, select the Infrastructure link to navigate to the Infrastructure page. If a repository is not configured, then the Farm Repository field shows "Not Configured", as shown in Figure 4-1.

**Figure 4–1 Application Server Control Console Farm Repository Management**

- On the Infrastructure page, in the OracleAS Farm Repository Management area, select the **Configure** button to start the Configure OracleAS Farm Repository wizard. The repository creation wizard appears. The appropriate host name appears under Configure Oracle Farm Repository Source. Select the **New file-based repository** button and select **Next**, as shown in Figure 4–2.

**Figure 4–2 Application Server Control Console Create Repository Wizard Step 1**

- The wizard jumps to Step 4 of 4, Validation, as shown in Figure 4–3.

**Figure 4–3 Application Server Control Console Create Repository Wizard Step 4**

4. Select **Finish**, and Oracle Application Server creates the OracleAS File-based Farm.
5. When the wizard completes, note the Repository ID shown in the OracleAS Farm Repository Management area on the Infrastructure page. You need to use the Repository ID to add instances to the OracleAS File-based Farm.

When you go to the Application Server Control Console Home page, notice that the home page shows the OC4J instance and the Oracle HTTP Server are stopped, and the page now includes a Farm link in the General area.

#### 4.2.1.2.2 Adding Instances To An OracleAS File-based Farm

To add standalone application server instances to an OracleAS File-based Farm, perform the following steps:

1. Obtain the Repository ID for the OracleAS File-based Farm that you want to join. To find the Repository ID, on any Oracle Application Server instance that uses the OracleAS File-based Farm, select the Infrastructure link, and check the value of the File-based Repository ID field in the OracleAS Farm Repository Management area.
2. Switch to the Application Server Control Console for the standalone instance that you want to add to the OracleAS File-based Farm and select the Infrastructure link. If a repository is not configured, then the Farm Repository field shows "Not Configured", as shown in Figure 4–1.
3. Select the **Configure** button to start the Configure OracleAS Farm Repository wizard. The repository creation wizard appears, as shown in Figure 4–2. The appropriate host name appears in the OracleAS Instance field under the Configure Oracle Farm Repository Source area.
4. Select the **Existing file-based repository** button and select **Next**. The repository creation wizard then brings up the Location page, Step 3 of 4, as shown in Figure 4–4.

**Figure 4–4 Application Server Control Console Add Instance to Farm**

ORACLE Enterprise Manager 10g  
Application Server Control

Logs Topology Preferences Help

Source Internet Directory **Location** Validation

Cancel Back Step 3 of 4 Next

**Configure OracleAS Farm Repository: Location**

OracleAS Instance **myserver.mycompany.com**

To add this instance to an existing farm, enter the file-based repository ID for the farm. The ID can be found on the Infrastructure page of an instance already in the farm.

\* File-based Repository ID

Cancel Back Step 3 of 4 Next

Logs | Topology | Preferences | Help

Copyright © 1996, 2004, Oracle. All rights reserved.  
[About Oracle Enterprise Manager 10g Application Server Control](#)

5. Enter the repository ID for the Repository Host and select **Next**.
6. This shows wizard Step 4 of 4 page, Configure OracleAS Farm Repository Validation. Select **Finish**. When the wizard completes, the standalone instance joins the OracleAS File-based Farm.
7. After the wizard completes, you return to the Application Server Control Console Infrastructure page.

#### 4.2.1.3 Using the Application Server Control Console Create Cluster Page

Using the Application Server Control Console Farm Home Page, you can create a new DCM-Managed OracleAS Cluster.

From the Farm Home page, create a new DCM-Managed OracleAS Cluster as follows:

1. Select the Farm link to navigate to the Farm Home Page.

---

**Note:** Application Server Control Console shows the Farm Home Page when an Oracle Application Server instance is part of a farm.

---

2. Select the **Create Cluster** button. Application Server Control Console displays the Create Cluster page as shown in Figure 4–5.

Figure 4–5 Create Cluster Page

ORACLE Enterprise Manager 10g  
Application Server Control

Topology Preferences Help

Farm >  
Create Cluster

Cancel Create

Enter the name of the cluster you wish to create.

\* Cluster Name

Cancel Create

Topology | Preferences | Help

Copyright © 1996, 2004, Oracle. All rights reserved.  
About Oracle Enterprise Manager 10g Application Server Control

3. Enter a name for the new cluster and click **Create**. Each new cluster name within the farm must be unique.

A confirmation page appears.

4. Click **OK** to return to the Farm Home Page.

After creating a new cluster, the Farm Home page shows the cluster in the Clusters area. After creating a new cluster, the cluster is empty and does not include any application server instances. Use the **Join Cluster** button on the Farm Home page to add application server instances to the cluster.

**See Also:** Section 4.2.2, "Adding Instances To DCM-Managed OracleAS Clusters" on page 4-7

## 4.2.2 Adding Instances To DCM-Managed OracleAS Clusters

To add application server instances to a DCM-Managed OracleAS Cluster, do the following:

1. Navigate to the Farm Home Page. To navigate to the Farm Home page from an Oracle Application Server instance Home page, select the link next to the Farm field in the General area on the Home page.

---

**Note:** If the Farm field is not shown, then the instance is not part of a Farm and you will need to associate the standalone instance with a Farm.

---

2. Select the radio button for the application server instance that you want to add to a cluster from the Standalone Instances section.
3. Click **Join Cluster**.

Figure 4–6 shows the Join Cluster page.

**Figure 4–6 Join Cluster Page**

Clusters

Select the cluster for the instance to join.

Select Name	Status	Instances
<input checked="" type="radio"/> cluster1	↑	1
<input type="radio"/> cluster2	↓	0

Cancel Join

[Topology](#) | [Preferences](#) | [Help](#)

Copyright © 1996, 2004, Oracle. All rights reserved.  
[About Oracle Enterprise Manager 10g Application Server Control](#)

4. Select the radio button of the cluster that you want the application server instance to join.
5. Click **Join**. OracleAS adds the application server instance to the selected cluster and then displays a confirmation page.
6. Click **OK** to return to the Farm Home Page.

Repeat these steps for each additional standalone application server instance you want to join the cluster.

Note the following when adding application server instances to a DCM-Managed OracleAS Cluster:

1. When adding application server instances to a DCM-Managed OracleAS Cluster, the order that you add instances is significant. The first application server instance that joins the DCM-Managed OracleAS Cluster is used as the base configuration for all additional application server instances that join the cluster. The base configuration includes all cluster-wide configuration information. It does not include instance-specific parameters.
2. After the first application server instance joins the DCM-Managed OracleAS Cluster, the base configuration overwrites existing cluster-wide configuration information for subsequent application server instances that join the cluster. Each additional application server instance, after the first, that joins the cluster inherits the base configuration specified for the first application server instance that joins the cluster.
3. Before an application server instance joins a DCM-Managed OracleAS Cluster, Application Server Control Console stops the instance. You can restart the application server instance by selecting the cluster link, selecting the appropriate instance from within the cluster, and then selecting the **Start** button.
4. An application server instance is removed from the Standalone Instances area when the instance joins a DCM-Managed OracleAS Cluster.
5. To add multiple standalone application server instances to a DCM-Managed OracleAS Cluster in a single operation, use the `dcmctl joinCluster` command.
6. When an application server instance contains certain Oracle Application Server components, it is not clusterable. Use the `dcmctl isClusterable` command to test if an application server instance is clusterable. If the application server instance is not clusterable, then Application Server Control Console returns an error when you attempt to add the instance to a DCM-Managed OracleAS Cluster.
7. To be clusterable, all application server instances that are to be members of a DCM-Managed OracleAS Cluster must be installed on the same flavor operating system. For example, different variants of UNIX are clusterable together, but they are not clusterable with Windows systems.

---

---

**Note:** For adding instances to a OracleAS File-based Farm, where the instances will be added to an DCM-Managed OracleAS Cluster, there is no known fixed upper limit on the number of instances; a DCM-Managed OracleAS Cluster of 12 instances has been tested successfully.

---

---

### 4.2.3 Removing Instances from DCM-Managed OracleAS Clusters

To remove an application server instance from a cluster, do the following:

1. Select the cluster in which you are interested on the Farm Home Page. This brings you to the cluster page.
2. Select the radio button of the application server instance to remove from the cluster and click **Remove**.

To remove multiple standalone application server instances, you need to repeat these steps multiple times.

Note the following when removing application server instances from an DCM-Managed OracleAS Cluster:

- Before an application server instance leaves a cluster, Application Server Control Console stops the instance. After the operation completes, you restart the application server instance from the Standalone Instances area of the Farm Home Page.
- The `dcmctl leaveCluster` command removes one application server instance from the cluster at each invocation.
- When the last application server instance leaves a cluster, cluster-wide configuration information associated with the cluster is removed. The cluster is now empty and the base configuration is not set. Subsequently, Oracle Application Server uses the first application server instance that joins the cluster as the base configuration for all additional application server instances that join the cluster.
- You can remove an application server instance from the cluster at any time. The first instance to join a cluster does not have special properties. The base configuration is created from the first instance to join the cluster, but this instance can be removed from the cluster in the same manner as the other instances.

### 4.2.4 Starting Stopping and Deleting DCM-Managed OracleAS Clusters

Figure 4–7 shows the Application Server Control Console Farm Home Page, including two clusters, cluster1 and cluster2.

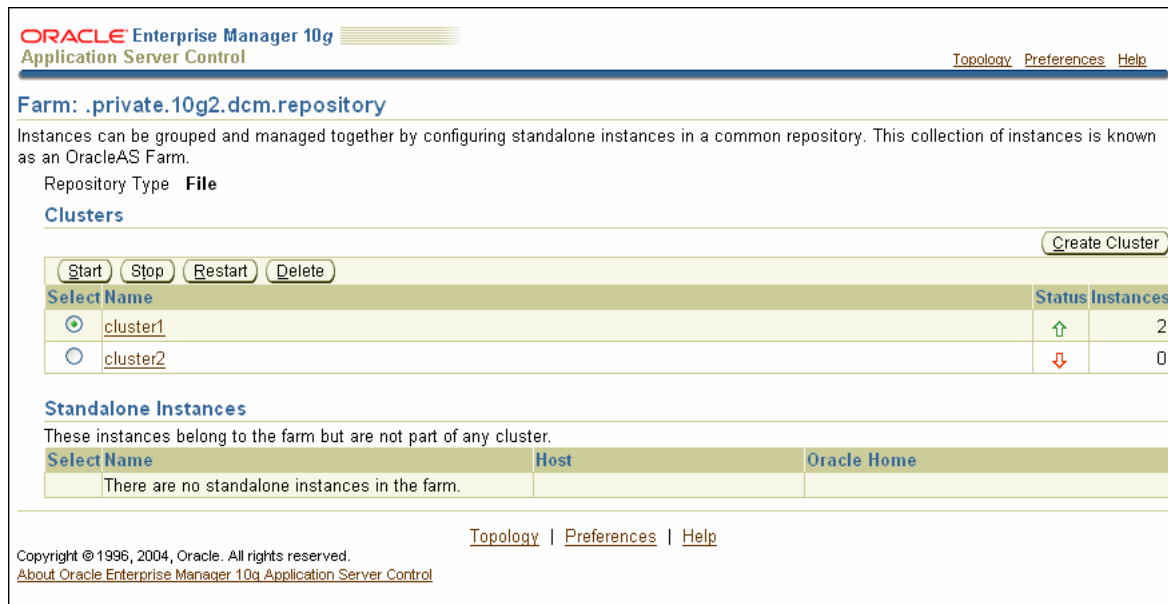
**Figure 4–7 Oracle Application Server 10g Farm Page**

Table 4–1 lists the cluster control options available on the Farm Home Page.

**Table 4–1 Oracle Application Server Farm Page Options**

If you want to...	Then...
Start all application server instances in a DCM-Managed OracleAS Cluster	Select the radio button next to the cluster and click <b>Start</b>
Restart all application server instances in an DCM-Managed OracleAS Cluster	Select the radio button next to the cluster and click <b>Restart</b>
Stop all application server instances in an DCM-Managed OracleAS Cluster	Select the radio button next to the cluster and click <b>Stop</b>
Delete a DCM-Managed OracleAS Cluster, including any application server instances still included in the cluster (the instances are removed from the cluster and become standalone instances in the Farm).	Select the radio button next to the cluster and click <b>Delete</b>

## 4.2.5 Configuring Oracle HTTP Server Options for DCM-Managed OracleAS Clusters

This section describes Oracle HTTP Server options for DCM-Managed Oracle Application Server Clusters.

This section covers the following:

- Section 4.2.5.1, "Using and Configuring mod\_oc4j Load Balancing"
- Section 4.2.5.2, "Configuring Oracle HTTP Server Instance-Specific Parameters"
- Section 4.2.5.3, "Configuring mod\_plsql With Real Application Clusters"

### 4.2.5.1 Using and Configuring mod\_oc4j Load Balancing

Using DCM-Managed OracleAS Clusters the Oracle HTTP Server module mod\_oc4j load balances requests to OC4J processes. The Oracle HTTP Server, using mod\_oc4j configuration options, supports different load balancing policies. By specifying load



balancing policies DCM-Managed OracleAS Clusters provide performance benefits along with failover and high availability, depending on the network topology and host machine capabilities.

By default, `mod_oc4j` uses weights to select a node to forward a request to. Each node uses a default weight of 1. A node's weight is taken as a ratio compared to the weights of the other available nodes to define the number of requests the node should service compared to the other nodes in the DCM-Managed OracleAS Cluster. Once a node is selected to service a particular request, by default, `mod_oc4j` uses the `roundrobin` policy to select OC4J processes on the node. If an incoming request belongs to an established session, the request is forwarded to the same node and the same OC4J process that started the session.

The `mod_oc4j` load balancing policies do not take into account the number of OC4J processes running on a node when calculating which node to send a request to. Node selection is based on the configured weight for the node, and its availability.

To modify the `mod_oc4j` load balancing policy, Administrators use the `Oc4jSelectMethod` and `Oc4jRoutingWeight` configuration directives in the `mod_oc4j.conf` file.

Using Application Server Control Console, configure the `mod_oc4j.conf` file as follows:

1. Select the HTTP\_Server component from the System Components area of an instance home page.
2. Select the Administration link on the HTTP\_Server page.
3. Select the Advanced Server Properties link on the HTTP\_Server page Administration page.
4. On the Advanced Server Properties page, select the `mod_oc4j.conf` link from the Configuration Files area.
5. On the Edit `mod_oc4j.conf` page, within the `<IfModule mod_oc4j.c>` section, and specify the directives `Oc4jSelectMethod` and `Oc4jRoutingWeight` to select the desired load balancing option.

---

**Note:** If you do not use Application Server Control Console, you can edit `mod_oc4j.conf` and use the `dcmctl updateConfig` command to propagate changes to other `mod_oc4j.conf` files across a DCM-Managed OracleAS Cluster as follows:

```
% dcmctl updateconfig -ct ohs
% opmnctl @cluster:<cluster_name> restartproc ias-component=HTTP_
Server process-type=HTTP_Server
```

Where `cluster_name` is the name of the cluster.

The `opmnctl restartproc` command is required for the changes to take effect across all the instances in the cluster.

---

**See Also:**

- Section 4.4.3, "Configuring OC4J Instance-Specific Parameters" on page 4-28
- *Oracle HTTP Server Administrator's Guide* for information on using `mod_oc4j` load balancing directives
- *Oracle Application Server Performance Guide*

**4.2.5.2 Configuring Oracle HTTP Server Instance-Specific Parameters**

You can modify the Oracle HTTP Server ports and listening addresses on the Server Properties Page, which can be accessed from the Oracle HTTP Server Home Page. You can modify the virtual host information by selecting a virtual host from the Virtual Hosts section on the Oracle HTTP Server Home Page.

Table 4–3 shows the Oracle HTTP Server instance-specific parameters.

**4.2.5.3 Configuring mod\_plsql With Real Application Clusters**

This section covers the following:

- Section 4.2.5.3.1, "Configuring Detection and Cleanup of Dead Connections"
- Section 4.2.5.3.2, "Using Oracle Directory for Lookups"

**4.2.5.3.1 Configuring Detection and Cleanup of Dead Connections**

Using Oracle HTTP Server with the `mod_plsql` module, if a database becomes unavailable, the connections to the database need to be detected and cleaned up. This section explains how to configure `mod_plsql` to detect and cleanup dead connections.

The `mod_plsql` module maintains a pool of connections to the database and reuses established connections for subsequent requests. If there is no response from a database connection, `mod_plsql` detects this case, discards the dead connection, and creates a new database connection for subsequent requests.

By default, when a Real Application Clusters node or a database instance goes down and `mod_plsql` previously pooled connections to the node or instance, the first `mod_plsql` request which uses a dead connection in its pool results in a failure response of HTTP-503 that is sent to the end-user. The `mod_plsql` module processes this failure and uses it to trigger detection and removal of all dead connections in the connection pool. The `mod_plsql` module pings all connection pools that were created before the failure response. This ping operation is performed at the time of processing for the next request that uses a pooled connection. If the ping operation fails, the database connection is discarded, and a new connection is created and processed.

Setting the `PlsqlConnectionValidation` parameter to `Automatic` causes the `mod_plsql` module to test all pooled database connections that were created before a failed request. This is the default configuration.

Setting the `PlsqlConnectionValidation` parameter to `AlwaysValidate` causes `mod_plsql` to test all pooled database connections before issuing any request. Although the `AlwaysValidate` configuration option ensures greater availability, it also introduces additional performance overhead.

You can specify the timeout period for `mod_plsql` to test a bad database connection in a connection pool. The `PlsqlConnectionTimeout` parameter, which specifies the maximum time `mod_plsql` should wait for the test request to complete before it assumes that a connection is not usable.

**See Also:** *Oracle Application Server mod\_plsql User's Guide*

#### 4.2.5.3.2 Using Oracle Directory for Lookups

Oracle Net clients can use a Directory Server to lookup connect descriptors. At the beginning of a request, the client uses a connect identifier to the Directory Server where it is then resolved into a connect descriptor.

The advantage of using a Directory Server is that the connection information for a server can be centralized. If the connection information needs to be changed, either because of a port change or a host change, the new connection information only needs to be updated once, in the Directory Server, and all Oracle Net clients using this connection method will be able to connect to the new host.

**See Also:** *Oracle Database Net Services Administrator's Guide* for instructions on configuring Directory Naming.

## 4.2.6 Understanding DCM-Managed OracleAS Cluster Membership

After an DCM-Managed OracleAS Cluster is created, you can add Oracle Application Server Instances to it. This section describes DCM-Managed OracleAS Cluster configuration and the characteristics of clusterable Oracle Application Server Instances.

This section covers the following topics:

- Section 4.2.6.1, "How the Common Configuration is Established"
- Section 4.2.6.2, "Parameters Excluded from the Common Configuration: Instance-Specific Parameters"

### 4.2.6.1 How the Common Configuration is Established

The order in which the Oracle Application Server Instances are added to the DCM-Managed OracleAS Cluster is significant. The common configuration that will be replicated across the DCM-Managed OracleAS Cluster is established by the first Oracle Application Server Instance added to the cluster. The configuration of the first Oracle Application Server Instance added is inherited by all Oracle Application Server Instances that subsequently join the DCM-Managed OracleAS Cluster.

The common configuration includes all cluster-wide configuration information—namely, DCM-Managed OracleAS Cluster and Oracle Application Server Instance attributes, such as components configured. For example, if the first Oracle Application Server Instance to join the cluster has four OC4J instances, then the common configuration includes those four OC4J instances and the applications deployed on the OC4J instances. Instances that subsequently join the DCM-Managed OracleAS Cluster replicate the OC4J instances and their deployed applications. (In addition, when the Oracle Application Server Instance joins the DCM-Managed OracleAS Cluster, DCM removes any OC4J components that do not match the common configuration). Furthermore, changes to one Oracle Application Server Instance in the DCM-Managed OracleAS Cluster, such as adding new OC4J instances or removing OC4J instances, are replicated across the DCM-Managed OracleAS Cluster; the components configured are part of the replicated cluster-wide, common configuration.

When the last Oracle Application Server Instance leaves a DCM-Managed OracleAS Cluster, the DCM-Managed OracleAS Cluster becomes an empty DCM-Managed OracleAS Cluster, and the next Oracle Application Server Instance to joins the DCM-Managed OracleAS Cluster provides the new common configuration for the DCM-Managed OracleAS Cluster.

#### 4.2.6.2 Parameters Excluded from the Common Configuration: Instance-Specific Parameters

Some parameters only apply to a given Oracle Application Server Instance or computer; these parameters are instance-specific parameters. DCM does not propagate instance-specific parameters to the Oracle Application Server Instances in a DCM-Managed OracleAS Cluster. When you change an instance-specific parameter, if you want the change to apply across the DCM-Managed OracleAS Cluster, you must apply the change individually to each appropriate Oracle Application Server Instance.

**Table 4–2 OC4J Instance-specific Parameters**

Parameter	Description
island definitions	Specific to an Oracle Application Server Instance. Stateful OC4J applications that need to replicate state require that all of the islands in each OC4J instance across the DCM-Managed OracleAS Cluster have the same name.
number of processes	Specific to a computer. You may want to tune this parameter according to the computer's capabilities.
command-line options	Specific to a computer.
port numbers for RMI, JMS and AJP communication	Specific to a computer.

**Table 4–3 Oracle HTTP Server Instance-Specific Parameters**

Parameter	Description
ApacheVirtualHost	Specific to a computer.
Listen	Specific to a computer. This directive binds the server to specific addresses or ports.
OpmnHostPort	Specific to a computer.
Port	Specific to a computer. This directive specifies the port to which the standalone server listens.
User	Specific to a computer.
Group	Specific to a computer.
NameVirtualHost	Specific to a computer. This directive specifies the IP address on which the server receives requests for a name-based virtual host. This directive can also specify a port.
ServerName	Specific to a computer. This directive specifies the host name that the server should return when creating redirection URLs. This directive is used if <code>gethostbyname</code> does not work on the local host. You can also use it if you want the server to return a DNS alias as a host name (for example, <code>www.abccompany.com</code> ).
PerlBlob	Specific to a computer.

**Table 4–4** *OPMN Instance-Specific Parameters*

Parameter in opmn.xml file	Description
All configuration for the notification server: opmn/notification-server	Specific to a computer.
process-manager elements: log-file process-module	Specific to an Oracle Application Server Instance.
ias_instance attributes: id ORACLE_HOME ORACLE_CONFIG_HOME	Specific to an Oracle Application Server Instance.
The following elements and attributes of opmn/process-manager/ias-instance /ias-component/process-type port.range start stop ping restart process-set	Specific to an Oracle Application Server Instance. Although instance-specific, these elements and attributes have default configurations (the configurations are not propagated, but retrieved from repository).  The <code>MissingLocalValuePolicy</code> flag indicates that element or attribute has a default value:  <code>MissingLocalValuePolicy="UseRepositoryValue"</code>
The following opmn/process-manager/ias-instance attributes and elements:  module-data/category/data[id='config-file'] ias-component/module-data/category/data[id=' config-file'] ias-component/process-type/ module-data/category/data[id='config-file'] ias-component/process-type/ process-set/module-data/category/data[id='c onfig-file'] environment ias-component/environment ias-component/process-type/ environment  ias-component/process-type/ process-set/environment	Most of the <code>HTTP_Server</code> and <code>OC4J</code> parameters are cluster-wide; only those shown here are instance-specific.  In general: <ul style="list-style-type: none"> <li>■ Any data element whose <code>id</code> is <code>config-file</code> is instance-specific.</li> <li>■ Any environment element is instance-specific.</li> </ul> <b>See Also:</b> Appendix B, "Troubleshooting DCM", in the <i>Distributed Configuration Management Administrator's Guide</i> for a complete <code>opmn.xml</code> file that shows the hierarchy of these elements and attributes.
All other components whose <code>id</code> is not <code>HTTP_Server</code> or <code>OC4J</code> in opmn/process-manager/ias-instance/ ias-component:  [id='dcm-daemon'] [id='WebCache'] [id='OID'] [id='IASPT'] [id='wireless'] [id='Discoverer'] [id='LogLoader'] [id='Custom']	

## 4.3 Availability Considerations for the DCM Configuration Repository

This section covers availability considerations for the DCM configuration repository, and covers the following topics:

- Section 4.3.1, "Availability Considerations for DCM-Managed OracleAS Cluster (Database)"
- Section 4.3.2, "Availability Considerations for DCM-Managed OracleAS Cluster (File-based)"

---

---

**Note:** The availability of the configuration repository only affects the Oracle Application Server configuration and administration services. It does not affect the availability of the system for handling requests, or availability of the applications running in a DCM-Managed OracleAS Cluster.

---

---

### 4.3.1 Availability Considerations for DCM-Managed OracleAS Cluster (Database)

This section covers availability considerations for the DCM configuration repository when using DCM-Managed OracleAS Clusters with an OracleAS Database-based Farm.

Using an OracleAS Database-based Farm, with a Database that uses Oracle Real Application Clusters (RAC) or other Database level high availability solution protects the system by providing high availability, scalability, and redundancy during failures of DCM configuration repository Database.

**See Also:** The *Oracle Database High Availability Architecture and Best Practices* guide for a description of Oracle Database high availability solutions.

### 4.3.2 Availability Considerations for DCM-Managed OracleAS Cluster (File-based)

Using an OracleAS File-based Farm, the DCM configuration repository resides on one Oracle Application Server instance at any time. A failure of the host that contains the DCM configuration repository requires manual failover (by migrating the repository host to another host).

This section covers availability considerations for the DCM configuration repository when using DCM-Managed OracleAS Clusters with an OracleAS File-based Farm.

- Section 4.3.2.1, "Selecting the Instance to Use for a OracleAS File-based Farm Repository Host"
- Section 4.3.2.2, "Protecting Against The Loss of a Repository Host"
- Section 4.3.2.3, "Impact of Repository Host Unavailability"
- Section 4.3.2.4, "Impact of Non-Repository Host Unavailability"
- Section 4.3.2.5, "Updating and Checking the State of Local Configuration"
- Section 4.3.2.6, "Performing Administration on a DCM-Managed OracleAS Cluster"
- Section 4.3.2.7, "Best Practices for Repository Backups"
- Section 4.3.2.8, "Best Practices for Managing Instances In OracleAS File-based Farms"

---

---

**Note:** The information in this section does not apply to a DCM-Managed Oracle Application Server Cluster that uses a OracleAS Database-based Farm (with the repository type, database).

---

---

#### 4.3.2.1 Selecting the Instance to Use for a OracleAS File-based Farm Repository Host

An important consideration for using DCM-Managed OracleAS Clusters with a OracleAS File-based Farm is determining which Oracle Application Server instance is the repository host.

Consider the following when selecting the repository host for an OracleAS File-based Farm:

- When the repository host instance is temporarily unavailable, a DCM-Managed OracleAS Cluster that uses a OracleAS File-based Farm is still able to run normally, but it cannot update any configuration information.
- Since the Oracle Application Server instance that is the repository host instance stores and manages the DCM-Managed OracleAS Cluster related configuration information in its file system, the repository host instance should use mirrored or RAID disks. If the repository host instance uses disk mirroring, this improves the availability of the DCM-Managed OracleAS Cluster.
- When the repository host instance is not available, read-only configuration operations are not affected on any Oracle Application Server instances that are running (the OracleAS Farm cluster-wide configuration information is distributed and managed through local Java Object Cache).
- When the repository host instance is not available, operations that attempt to change configuration information in the file-based repository will generate an error. These operations must be delayed until the repository host instance is available, or until the repository host instance is relocated to another application server instance within the OracleAS File-based Farm.

#### 4.3.2.2 Protecting Against The Loss of a Repository Host

Using a OracleAS File-based Farm, one instance in the farm is designated as the repository host. The repository host holds configuration information for all instances in the OracleAS File-based Farm. Access to the repository host is required for all configuration changes, write operations, for instances in the OracleAS File-based Farm. However, instances have local configuration caches to perform read operations, where the configuration is not changing.

In the event of the loss of the repository host, any other instance in the OracleAS File-based Farm can take over as the new repository host if an exported copy of the old repository hosts is available. We recommend that you make regular backups of the repository host, and save the backups on a separate system.

**See Also:** *Distributed Configuration Management Administrator's Guide*

#### 4.3.2.3 Impact of Repository Host Unavailability

When the repository host is unavailable, only read-only operations are allowed. No configuration *changes* are allowed. If an operation is attempted that requires updates to the repository host, such as use of the `updateConfig` command, `dcmctl` reports a message, for example,

ADMN-100205

Base Exception:

The DCM repository is not currently available. The OracleAS 10g instance, "myserver.mycompany.com", is using a cached copy of the repository information. This operation will update the repository, therefore the repository must be available.

If the repository host is permanently down, or unavailable for the long-term, then the repository host should be relocated. If the restored repository is not recent, local instance archives can be applied to bring each instance up to a newer state.

**See Also:** Section 4.3.2.6, "Performing Administration on a DCM-Managed OracleAS Cluster" on page 4-19

#### 4.3.2.4 Impact of Non-Repository Host Unavailability

When the instances in a DCM-Managed OracleAS Cluster, other than the repository host instance are down, all other instances can function properly. If an instance is experiencing a short-term outage, the instance automatically updates its configuration information when it becomes available again.

If an instance is permanently lost, this will have no affect on other instances in the OracleAS File-based Farm. However, to maintain consistency, it will be necessary to delete all records pertaining to the lost instance.

To delete configuration information for a lost instance, use the following command:

```
dcmctl destroyInstance
```

#### 4.3.2.5 Updating and Checking the State of Local Configuration

It is important that all configuration changes complete successfully, and that all instances in a cluster are "In Sync". The local configuration information must match the information stored in the repository. DCM does not know about manual changes to configuration files, and such changes could make the instances in a cluster have an In Sync status of false.

Use the following `dcmctl` command to return a list of all managed components with their In Sync status:

```
dcmctl getState -cl cluster_name
```

The In Sync status of true implies that the local configuration information for a component is the same as the information that is stored in the repository.

If you need to update the file-based repository with changed, local information, use the `dcmctl` command `updateConfig`, as follows,

```
dcmctl updateconfig  
dcmctl getstate
```

Use the command `resyncInstance` to update local information with information from the repository. For example,

```
dcmctl resyncinstance
```

By default this command only updates configuration information for components whose In Sync status is false. Use the `-force` option to update all components, regardless of their In Sync status.



#### 4.3.2.6 Performing Administration on a DCM-Managed OracleAS Cluster

During planned administrative downtimes, with a DCM-Managed OracleAS Cluster using an OracleAS File-based Farm that runs on multiple hosts with sufficient resources, you can perform administrative tasks while continuing to handle requests. This section describes how to relocate the repository host in a DCM-Managed OracleAS Cluster, while continuing to handle requests.

These procedures are useful for performing administrative tasks on a DCM-Managed OracleAS Cluster, such as the following:

- Relocating the repository for repository host node decommission.
- Applying required patches to the DCM-Managed OracleAS Cluster.
- Applying system upgrades, changes, or patches that require a system restart for a host in the DCM-Managed OracleAS Cluster.

---



---

**Note:** Using the procedures outlined in this section, only administration capabilities are lost during a planned downtime.

---



---

Use the following steps to relocate the repository host in a DCM-Managed OracleAS Cluster.

1. Issue the following DCM command, on UNIX systems:

```
cd $ORACLE_HOME/dcm/bin
dcmctl exportRepository -f file
```

On Windows systems:

```
cd %ORACLE_HOME%\dcm\bin
dcmctl exportRepository -f file
```

---



---

**Note:** After this step, do not perform configuration or administration commands that would change the configuration. Otherwise those changes will not be copied when the repository file is imported to the new repository host.

---



---

2. Stop the administrative system, including Enterprise Manager and the DCM daemon in each instance of the OracleAS File-based Farm, except for the instance that is going to be the new repository host.

On UNIX systems use the following commands on each instance in the cluster:

```
$ORACLE_HOME/bin/emctl stop iasconsole
$ORACLE_HOME/opmn/bin/opmnctl stopproc ias-component=dcm-daemon
```

On Windows systems use the following commands on each instance in the cluster:

```
%ORACLE_HOME%\bin\emctl stop iasconsole
%ORACLE_HOME%\opmn\bin\opmnctl stopproc ias-component=dcm-daemon
```

At this point, the DCM-Managed OracleAS Cluster can still handle requests.

3. Import the saved repository on the host that is to be the repository host instance.

On UNIX systems, use the following commands:

```
cd $ORACLE_HOME/dcm/bin/  
dcmctl importRepository -file file name
```

On Windows systems, use the following commands:

```
cd %ORACLE_HOME%\dcm\bin\  
dcmctl importRepository -file file name
```

where *file name* is the name of the file you specified in the `exportRepository` command.

While `importRepository` is active, the DCM-Managed OracleAS Cluster can still handle requests.

---

---

**Note:** The `importRepository` command issues a prompt that specifies that the system that is the currently hosting the repository must be shutdown. However, only the `dcm-daemon` on the system that is currently hosting the repository must be shutdown, and not the entire system.

---

---

4. Use the following command to start all components on the new repository host. Do not perform administrative functions at this time.

On UNIX systems:

```
$ORACLE_HOME/opmn/bin/opmnctl startall
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl startall
```

5. On the system that was the repository host, indicate that the instance is no longer the host by issuing the following command,

```
dcmctl repositoryRelocated
```

6. Start Application Server Control Console on the new repository host instance. The repository has now been relocated, and the new repository instance now handles requests.

On UNIX systems use the following commands on each instance in the cluster:

```
$ORACLE_HOME/bin/emctl start iasconsole
```

On Windows systems use the following commands on each instance in the cluster:

```
%ORACLE_HOME%\bin\emctl start iasconsole
```

7. Shut down the Oracle Application Server instance associated with the old repository host, using the following commands:

On UNIX systems:

```
$ORACLE_HOME/opmn/bin/opmnctl stopall
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl startall
```

You can now perform the required administrative tasks on the old repository host system, such as the following.

- Applying required patches to the repository host system in the DCM-Managed OracleAS Cluster.
- Decommission the node.
- Applying system upgrades, changes, or patches that require a system restart for the DCM-Managed OracleAS Cluster.

After completing the administrative tasks on the system that was the repository host, if you want to switchback the repository host, you need to perform these steps again.

#### 4.3.2.7 Best Practices for Repository Backups

When you export repository files and archives, keep the files in known locations and backup the exports and archives regularly. It is also recommended that exported repositories be available to non-repository instances, not only as a backup means but also for availability. If the repository instance becomes unavailable, a new instance can become the new repository host but only if an exported repository file is available.

Oracle Application Server does not provide an automated repository backup procedure. However, to assure that you can recover from loss of configuration data, you need to put a repository backup plan in place. Perform repository backups on a regular basis, frequently, and perform a repository backup after any configuration changes or topology changes where instances are added or removed.

Send repository backups to different nodes that are available to other nodes in the OracleAS File-based Farm.

**See Also:** *Distributed Configuration Management Administrator's Guide*

#### 4.3.2.8 Best Practices for Managing Instances In OracleAS File-based Farms

When joining or leaving an OracleAS File-based Farm, all the managed processes are shutdown on the instance that is joining or leaving the OracleAS File-based Farm. If you want the instance to be available, then after performing the leave or join farm operation, restart the instance.

It is recommended that you make a backup of the local configuration before either leaving or joining an OracleAS File-based Farm. For example, use the following command to create an archive:

```
dcmctl createarchive -arch myarchive -comment "Archive before leaving xyz farm"
dcmctl exportarchive -arch myarchive -f /archives/myarchive
```

Archives are portable across OracleAS File-based Farm. When an instance joins a new farm it can apply archives created on a previous farm.

**See Also:** *Distributed Configuration Management Administrator's Guide*

## 4.4 Using Oracle Application Server Clusters (OC4J)

This section describes Oracle Application Server Cluster (OC4J) configuration and the use of Oracle Application Server Cluster (OC4J) with DCM-Managed OracleAS Clusters.

Using Oracle Application Server Cluster (OC4J) allows Web applications to replicate state and provides for high availability and failover for applications that run under OC4J. You can configure this feature without using a DCM-Managed OracleAS

Cluster. However, when you use both of these Oracle Application Server features together, this simplifies and improves manageability and high availability. This section assumes that you are using both Oracle Application Server Cluster (OC4J) and DCM-Managed OracleAS Cluster.

This section covers the following:

- Overview of OracleAS Cluster (OC4J) Configuration
- Cluster-Wide Configuration Changes and Modifying OC4J Instances
- Configuring OC4J Instance-Specific Parameters

**See Also:** Oracle Application Server Containers for J2EE User's Guide for detailed information on configuring OC4J Instances

#### 4.4.1 Overview of OracleAS Cluster (OC4J) Configuration

Configuring Oracle Application Server Cluster (OC4J) allows Web applications to replicate state, and provides for high availability and failover for applications that run on OC4J. After application server instances join a DCM-Managed OracleAS Cluster, the application server instances, and the OC4J instances have the following properties:

- Each application server instance has the same cluster-wide configuration. When you use Application Server Control Console or `dcmctl` to modify any cluster-wide OC4J parameters, the modifications are propagated to all application server instances in the cluster. To make cluster-wide OC4J configuration changes you need to change the configuration parameters on a single application server instance; Oracle Application Server then propagates the modifications to all the other application server instances within the cluster.
- When you modify any instance-specific parameters on an OC4J instance that is part of a DCM-Managed OracleAS Cluster, the change is not propagated across the DCM-Managed OracleAS Cluster. Changes to instance-specific parameters are only applicable to the specific application server instance where the change is made. Since different hosts running application server instances could each have different capabilities, such as total system memory, it may be appropriate for the OC4J processes within an OC4J instance to run with different configuration options.

Table 4–5 provides a summary of the OC4J instance-specific parameters. Other OC4J parameters are cluster-wide parameters and are replicated across DCM-Managed OracleAS Clusters.

**Table 4–5 OC4J Instance-Specific Parameters Summary for DCM-Managed OracleAS Cluster**

OC4J Parameter	Description
islands definitions	<p>While you want to keep the names of islands consistent across the application server instances, the definition of the islands and the number of OC4J processes associated with each island is configured on each instance, and the Oracle Application Server configuration management system does not replicate the configuration across the DCM-Managed OracleAS Cluster.</p> <p><b>Note:</b> state is replicated in OC4J islands with the same name across application boundaries and across the cluster. So to assure high availability, with stateful applications, the OC4J island names must be the same in each OC4J instance across the cluster.</p>
number of OC4J processes	<p>While you want to keep the names of islands consistent across the application server instances, the definition of the islands and the number of OC4J processes associated with each island is configured on each instance, and DCM does not replicate the configuration across the DCM-Managed OracleAS Cluster.</p> <p>On different hosts you can tune the number of OC4J processes specified to run per island to match the host capabilities.</p>
port numbers	The RMI, JMS, and AJP port numbers can be different for each host.
command line options	The command line options you use can be different for each host.

## 4.4.2 Cluster-Wide Configuration Changes and Modifying OC4J Instances

This section covers the following topics:

- Section 4.4.2.1, "Creating or Deleting OC4J Instances In An OracleAS Cluster (OC4J)"
- Section 4.4.2.2, "Deploying Applications On An OracleAS Cluster (OC4J)"
- Section 4.4.2.3, "Configuring Web Application State Replication With OracleAS Cluster (OC4J)"
- Section 4.4.2.4, "Configuring EJB Application State Replication With OracleAS Cluster (OC4J-EJB)"
- Section 4.4.2.5, "Configuring Stateful Session Bean Replication for OracleAS Cluster (OC4J-EJB)s"

**See Also:** *Oracle Application Server Containers for J2EE User's Guide* for complete information OC4J configuration and application deployment

### 4.4.2.1 Creating or Deleting OC4J Instances In An OracleAS Cluster (OC4J)

You can create a new OC4J instance on any application server instance within a DCM-Managed OracleAS Cluster and the OC4J instance will be propagated to all application server instances across the cluster.

To create an OC4J instance, do the following:

1. Navigate to any application server instance within the DCM-Managed Oracle Application Server Cluster.
2. Select **Create OC4J Instance** under the System Components area. This brings up the Create OC4J instance page.
3. Enter a name in the OC4J Instance name field.

4. Select **Create**.
5. The Oracle Application Server creates the instances and then DCM propagates the new OC4J instance across the DCM-Managed OracleAS Cluster.

A new OC4J instance is created with the name you provided. This OC4J instance shows up on each application server instance across the cluster, in the System Components section.

To delete an OC4J instance, select the checkbox next to the OC4J instance you wish to delete, then select **Delete OC4J Instance**. The Oracle Application Server DCM system propagates the OC4J removal across the cluster.

#### 4.4.2.2 Deploying Applications On An OracleAS Cluster (OC4J)

Using DCM-Managed OracleAS Cluster, when you deploy an application to one application server instance, the application is propagated to all application server instances across the cluster.

To deploy an application across a cluster, do the following:

1. Select the cluster you want to deploy the application to.
2. Select any application server instance from within the cluster.
3. Select an OC4J instance on the application server instance where you want to deploy the application.
4. Deploy the application to the OC4J instance using either Application Server Control Console or `dcmctl` commands.
5. The Distributed Configuration Management system then propagates the application across the DCM-Managed Oracle Application Server Cluster.

**See Also:** *Oracle Application Server Containers for J2EE User's Guide* for complete information on deploying applications to an OC4J instance.


#### 4.4.2.3 Configuring Web Application State Replication With OracleAS Cluster (OC4J)

To assure that Oracle Application Server maintains, across DCM-Managed OracleAS Cluster, the state of stateful Web applications you need to configure state replication for the Web applications.

To configure state replication for stateful Web applications, do the following:

1. Select the Administration link on the OC4J Home Page.
2. Select the Replication Properties link in the Instance Properties area.
3. Scroll down to the Web Applications section. Figure 4-8 shows this section.

**Figure 4–8 Web State Replication Configuration**

**Replication Properties** Page Refreshed Aug 26, 2004 2:02:00 PM 

**TIP** Changes here affect all OC4J instances in cluster "cluster1".

**Web Applications**

**TIP** Setting session state replication here will enable session state replication for all web applications. The load-on-startup property will be automatically set to true for all web modules.

Replicate session state

Multicast Host (IP)

Multicast Port

4. Select the **Replicate session state** checkbox.

Optionally, you can provide the multicast host IP address and port number. If you do not provide the host and port for the multicast address, it defaults to host IP address 230.0.0.1 and port number 9127. The host IP address must be between 224.0.0.2 through 239.255.255.255. Do not use the same multicast address for both HTTP and EJB multicast addresses.

---

**Note:** When choosing a multicast address, ensure that the address does not collide with the addresses listed in

<http://www.iana.org/assignments/multicast-addresses>

Also, if the low order 23 bits of an address is the same as the local network control block, 224.0.0.0 – 224.0.0.255, then a collision may occur. To avoid this problem, provide an address that does not have the same bits in the lower 23 bits of the address as the addresses in this range.

---

5. Add the `<distributable/>` tag to all `web.xml` files in all Web applications. If the Web application is serializable, you must add this tag to the `web.xml` file.

The following shows an example of this tag added to `web.xml`:

```
<web-app>
  <distributable/>
  <servlet>
    ...
  </servlet>
</web-app>
```

---

**Note:** In order for sessions to be replicated to a just-started instance that joins a running cluster, for example where sessions are already being replicated between instances, the web module in the application maintaining the session has to be configured with the load-on-startup flag set to true. This is a cluster wide configuration parameter. See Figure 4–9 for details on setting this flag.

---

**Figure 4–9 Application Server Control Console Website Properties Page For Setting Load On Startup**

ORACLE Enterprise Manager 10g  
Application Server Control

Farm > Cluster: cluster1 > Application Server MyCompany.com > OC4J:OC4J\_ha >

Website Properties

TIP Changes here affect all OC4J instances in cluster 'cluster1'. Page Refreshed Dec 12, 2004 5:07:07 PM

**Default Web Module**

Name: defaultWebApp  
 Application: default  
 Load on startup: true

**URL Mappings for Web Modules**

Name	Application	URL Mapping	Load on startup
dms	default	/cmsoc4j	<input checked="" type="checkbox"/>
hacemoweb	ha	/fa	<input checked="" type="checkbox"/>
rolling	secondha	/rolirg33	<input checked="" type="checkbox"/>

Revert Apply

Copyright © 1996, 2004, Oracle. All rights reserved.  
About Oracle Enterprise Manager 10g Application Server Control

**See Also:** *Oracle Application Server Containers for J2EE User's Guide*

#### 4.4.2.4 Configuring EJB Application State Replication With OracleAS Cluster (OC4J-EJB)

To create an EJB cluster also known as OracleAS Cluster (OC4J-EJB), you specify the OC4J instances that are to be involved in the cluster, configure each of them with the same multicast address, username, and password, and deploy the EJB, which is to be clustered, to each of the nodes in the cluster.

EJBs involved in a OracleAS Cluster (OC4J-EJB) cannot be sub-grouped in an island. Instead, all EJBs within the cluster are in one group. Also, only session beans are clustered.

The state of all beans is replicated at the end of every method call to all nodes in the cluster using a multicast topic. Each node included in the OracleAS Cluster (OC4J-EJB) is configured to use the same multicast address.

The concepts for understanding how EJB object state is replicated within a cluster are described in the *Oracle Application Server Containers for J2EE Enterprise JavaBeans Developer's Guide*.

To configure EJB replication, do the following:

1. Select the Administration link on the OC4J Home Page.
2. Select the Replication Properties link in the Instance Properties area.
3. In the EJB Applications section, select the **Replicate State** checkbox.

Figure 4–10 shows this section.



**Figure 4–10 EJB State Replication Configuration**

EJB Applications	
<input checked="" type="checkbox"/> <b>TIP</b> EJB applications replicate state between all OC4J processes in the OC4J instance.	
<input type="checkbox"/> Replicate State	
Multicast Host (IP)	<input type="text"/>
Multicast Port	<input type="text"/>
Username	<input type="text"/>
Password	<input type="text"/>
RMI Server Host	[ALL]
<small>This is usually the name of the machine where the OC4J instance is running.</small>	

- Provide the username and password, which is used to authenticate itself to other hosts in the OracleAS Cluster (OC4J-EJB). If the username and password are different for other hosts in the cluster, they will fail to communicate. You can have multiple username and password combinations within a multicast address. Those with the same username/password combinations will be considered a unique cluster.

Optionally, you can provide the multicast host IP address and port number. If you do not provide the host and port for the multicast address, it defaults to host IP address 230.0.0.1 and port number 9127. The host IP address must be between 224.0.0.2 through 239.255.255.255. Do not use the same multicast address for both Web Application and EJB multicast addresses.

---

**Note:** When choosing a multicast address, ensure that the address does not collide with the addresses listed in

<http://www.iana.org/assignments/multicast-addresses>

Also, if the low order 23 bits of an address is the same as the local network control block, 224.0.0.0 – 224.0.0.255, then a collision may occur. To avoid this provide an address that does not have the same bits in the lower 23 bits of the address as the addresses in this range.

---

- Configure the type of EJB replication within the `orion-ejb-jar.xml` file within the JAR file. See "Configuring Stateful Session Bean Replication for OracleAS Cluster (OC4J-EJB)s" on page 4-27 for full details. You can configure these within the `orion-ejb-jar.xml` file before deployment or add this through the Application Server Control Console screens after deployment. To add this after deployment, drill down to the JAR file from the application page.

#### 4.4.2.5 Configuring Stateful Session Bean Replication for OracleAS Cluster (OC4J-EJB)s

For stateful session beans, you may have you modify the `orion-ejb-jar.xml` file to add the state replication configuration. Since you configure the replication type for the stateful session bean within the bean deployment descriptor, each bean can use a different type of replication.

Stateful session beans require state to be replicated among nodes. In fact, stateful session beans must send all their state between the nodes, which can have a noticeable effect on performance. Thus, the following replication modes are available to you to decide on how to manage the performance cost of replication:

**4.4.2.5.1 End of Call Replication** The state of the stateful session bean is replicated to all nodes in the cluster, with the same multicast address, at the end of each EJB method call. If a node loses power, then the state has already been replicated.

To use end of call replication, set the `replication` attribute of the `<session-deployment>` tag in the `orion-ejb-jar.xml` file to "endOfCall".

For example,

```
<session-deployment replication="EndOfCall" .../>
```

**4.4.2.5.2 JVM Termination Replication** The state of the stateful session bean is replicated to only one other node in the cluster, with the same multicast address, when the JVM is terminating. This is the most performant option, because the state is replicated only once. However, it is not very reliable for the following reasons:

- The state is not replicated if the power is shut off unexpectedly. The JVM termination replication mode does not guarantee state replication in the case of lost power.
- The state of the bean exists only on a single node at any time; the depth of failure is equal to one node.

To use JVM termination replication, set the `replication` attribute of the `<session-deployment>` tag in the `orion-ejb-jar.xml` file to "VMTermination".

For example,

```
<session-deployment replication="VMTermination" .../>
```

### 4.4.3 Configuring OC4J Instance-Specific Parameters

This section covers the instance-specific parameters that are not replicated across DCM-Managed OracleAS Clusters.

This section covers the following:

- Section 4.4.3.1, "Configuring OC4J Islands and OC4J Processes"
- Section 4.4.3.2, "Configuring Port Numbers and Command Line Options"

**See Also:** *Oracle Application Server Containers for J2EE User's Guide* for complete information OC4J configuration and application deployment

#### 4.4.3.1 Configuring OC4J Islands and OC4J Processes

To provide a redundant environment and to support high availability using DCM-Managed OracleAS Clusters, you need to configure multiple OC4J processes within each OC4J instance.

Using DCM-Managed OracleAS Cluster, state is replicated in OC4J islands with the same name within OC4J instances and across instances in the DCM-Managed OracleAS Cluster. To assure high availability, with stateful applications, OC4J island names within an OC4J instance must be the same in corresponding OC4J instances across the DCM-Managed OracleAS Cluster. It is the administrator's responsibility to make sure that island names match where session state replication is needed in a DCM-Managed OracleAS Cluster.

The number of OC4J processes on an OC4J instance within a DCM-Managed OracleAS Cluster is an instance-specific parameter since different hosts running application

server instances in the DCM-Managed OracleAS Cluster could each have different capabilities, such as total system memory. Thus, it could be appropriate for a DCM-Managed OracleAS Cluster to contain application server instances that each run different numbers of OC4J processes within an OC4J instance.

To modify OC4J islands and the number of processes each OC4J island contains, do the following:

1. Select the Administration link on the OC4J Home Page of the application server instance of interest in the DCM-Managed OracleAS Cluster.
2. Select **Server Properties** in the Instance Properties area.
3. Scroll down to the Multiple VM Configuration section. This section defines the islands and the number of OC4J processes that should be started on this application server instance in each island.

Figure 4–11 displays the Multiple VM Configuration Islands section.

**Figure 4–11 OC4J instance Island and Number of Processes Configuration**

Select	Island ID	Number of Processes
<input type="radio"/>	default_island	2
<input type="radio"/>	island2	3

4. Create any islands for this OC4J instance within the cluster by clicking **Add Another Row**. You can supply a name for each island within the Island ID field. You can designate how many OC4J processes should be started within each island by the number configured in the Number of Processes field.

#### 4.4.3.2 Configuring Port Numbers and Command Line Options

Figure 4–12 shows the section where you can modify these ports and set command line options.

To modify OC4J ports or the command line options, do the following:

1. Select the Administration link on the OC4J Home Page of the application server instance of interest in the cluster.
2. Select **Server Properties** in the Instance Properties area.
3. Scroll down to the Multiple VM Configuration section. This section defines the ports and the command line options for OC4J and for the JVM that runs OC4J processes.

Figure 4–12 shows the Ports and Command line options areas on the Server Properties page.

**Figure 4–12 OC4J Ports and Command Line Options Configuration**

**Multiple VM Configuration**

☑ **TIP** If OC4J is running, newly added islands and associated processes will be automatically started.

**Islands**

Island ID	Number of Processes	Related Links
default_island	1	<a href="#">Virtual Machine Metrics</a>

[Add Another Row](#)

**Ports**

RMI Ports:

JMS Ports:

AJP Ports:

**RMI-IIOP Ports**

IIOP Ports:

IIOP SSL (Server only):

IIOP SSL (Server and Client):

**Command Line Options**

Java Executable:

OC4J Options:

Java Options:

## 4.5 Using Oracle Application Server Clusters (Portal)

Details on configuring OracleAS Portal for high availability cover a number of Oracle Application Server components, including the following:

- OracleAS Web Cache
- Oracle HTTP Server
- OC4J (the Portal Page Engine runs as a stateless servlet)
- OracleAS Portal repository (contains OracleAS Portal schemas and also caches group memberships of users after their retrieval from Oracle Internet Directory)
- OracleAS Single Sign-On
- Oracle Internet Directory (including Oracle Delegated Administration Services and Oracle Directory Integration and Provisioning)

**See Also:**

- Section 3.4.1, "Oracle Application Server Portal" on page 3-20.
- *Oracle Application Server Enterprise Deployment Guide*

## 4.6 Using Oracle Application Server Clusters (Web Cache)

You can configure multiple instances of Oracle Application Server Web Cache to run as independent caches, with no interaction with one another. However, to increase the availability and scalability of your Web cache, you can configure multiple instances of OracleAS Web Cache to run as members of a cache cluster, also called OracleAS Cluster (Web Cache). A cache cluster is a loosely coupled collection of cooperating OracleAS Web Cache instances working together to provide a single logical cache.

**See Also:** *Oracle Application Server Web Cache Administrator's Guide*

## 4.7 Managing OracleAS Cold Failover Cluster (Middle-Tier)

This section provides instructions for managing Oracle Application Server Cold Failover Cluster (Middle-Tier). Using OracleAS Cold Failover Cluster (Middle-Tier) provides cost reductions for a highly available system, as compared to a fully available active-active middle-tier system. In addition, some applications may not function properly in an active-active OracleAS Cluster environment (for example, an applications that relies queuing or other synchronous methods). In this case, using an OracleAS Cold Failover Cluster (Middle-Tier) provides for high availability using the existing application without modifications.

This section covers the following topics:

- Section 4.7.1, "Managing Configuration and Deployment for OracleAS Cold Failover Cluster (Middle-Tier)"
- Section 4.7.2, "Managing Failover for OracleAS Cold Failover Cluster (Middle-Tier)"

### 4.7.1 Managing Configuration and Deployment for OracleAS Cold Failover Cluster (Middle-Tier)

Any application deployment or configuration change needs to be applied to both nodes of the OracleAS Cold Failover Cluster (Middle-Tier). This is an administrator responsibility for the Administrator managing the OracleAS Cold Failover Cluster (Middle-Tier) environment.

This section covers the following:

- Section 4.7.1.1, "Configuration and Deployment Changes for OracleAS Cold Failover Cluster (Middle-Tier)"
- Section 4.7.1.2, "Backup and Recovery for OracleAS Cold Failover Cluster (Middle-Tier)"
- Section 4.7.1.3, "Using Application Server Control Console for OracleAS Cold Failover Cluster (Middle-Tier)"

#### 4.7.1.1 Configuration and Deployment Changes for OracleAS Cold Failover Cluster (Middle-Tier)

Any applications deployed or any configuration changes made to the middle-tier installation should be made on both nodes of the cold failover cluster. This needs to be ensured by the administrator managing the environment.

Using a OracleAS Cold Failover Cluster (Middle-Tier), application deployment is applied, as with any other middle-tier environment. To deploy applications on the passive node, bring up the node and then deploy the application. For the J2EE installation of OC4J and Oracle HTTP Server, the application deployment is like any other multiple middle-tier environment. The passive node can be brought up during the deployment phase and the application deployment can be done on this node. Similarly, applications can be deployed on the active node.

---

---

**Note:** Using OracleAS Cold Failover Cluster (Middle-Tier) with OracleBI Discoverer, the active instance of OracleBI Discoverer is also the preference server. User preferences that users create while logged on to the active instance need to be synced up to the values on the passive instance. This allows the values to be available when the passive instance becomes the active instance during a failover. Thus, you need to periodically copy or update the following files to the passive instance so that they are current during a failover.

```
$ORACLE_HOME/discoverer/.reg_key.dc  
$ORACLE_HOME/discoverer/.reg_key.dc.bak  
$ORACLE_HME/discoverer/util/pref.txt
```

---

---

#### 4.7.1.2 Backup and Recovery for OracleAS Cold Failover Cluster (Middle-Tier)

Both nodes of the OracleAS Cold Failover Cluster (Middle-Tier) should be backed up. The procedure for this remains the same as for any other middle tier and is documented in the *Oracle Application Server Administrator's Guide*. Each installation needs to be backed up. During restoration, each backup can only be restored to the host it was backed up from. It should not be restored to the other node.

#### 4.7.1.3 Using Application Server Control Console for OracleAS Cold Failover Cluster (Middle-Tier)

To monitor or manage a node using Application Server Control Console, login to the Console using the physical hostname of the current active node. The Application Server Control Console processes can be up and running on both nodes of the cluster simultaneously. When changes to the environment are made, including configuration changes or application deployments, perform the changes on both nodes of the OracleAS Cold Failover Cluster (Middle-Tier).

### 4.7.2 Managing Failover for OracleAS Cold Failover Cluster (Middle-Tier)

Using a OracleAS Cold Failover Cluster (Middle-Tier), a failure in the active node, or a decision to stop the active node and failover to the passive node requires that you make the formerly passive node active (perform a failover operation).

The failover management itself can be performed using either of the following failover processes:

- Automated using a cluster manager facility. The cluster manager offers services, which uses packages to monitor the state of a service. If the service or the node is found to be down, it automatically fails over the service from one node to the other node.
- Manual failover. In this case, perform the manual failover steps as outlined in this section. Since both the detection of the failure and the failover itself is performed manually, the system may be unavailable for a longer period using manual failover.

This section covers the following topics:

- Section 4.7.2.1, "Manual Failover for OracleAS Cold Failover Cluster (Middle-Tier)"
- Section 4.7.2.2, "Manual Failover for the Virtual IP in OracleAS Cold Failover Cluster (Middle-Tier)"

- Section 4.7.2.3, "Manual Failover of Components for OracleAS Cold Failover Cluster (Middle-Tier)"
- Section 4.7.2.4, "Manual Failover of OracleAS Cluster (OC4J-JMS)"

#### 4.7.2.1 Manual Failover for OracleAS Cold Failover Cluster (Middle-Tier)

The failover process to make the formerly passive node the new active node includes the following steps:

1. Stop all middle-tier services on current active node (if the node is still available).
2. Failover the virtual IP to the new active node.
3. Failover the components to the new active node.
4. Start the middle-tier service on the new active node.

---

**Note:** The failover process requires that you previously performed the post-installation steps that set up and configured the OracleAS Cold Failover Cluster (Middle-Tier), as outlined in the Oracle Application Server Installation Guide for your platform.

---

#### 4.7.2.2 Manual Failover for the Virtual IP in OracleAS Cold Failover Cluster (Middle-Tier)

Perform the following steps to failover the Virtual IP in an OracleAS Cold Failover Cluster (Middle-Tier):

1. Stop all Oracle Application Server processes on the failed node, if possible, using the following command, on UNIX systems:

```
$ORACLE_HOME/opmn/bin/opmnctl stopall
```

On Windows systems,

```
%ORACLE_HOME%\opmn\bin\opmnctl stopall
```

2. Stop Oracle Application Server Administration processes on the failed node, if possible, using the following commands, on UNIX systems:

```
$ORACLE_HOME/bin/emctl stop iasconsole
```

```
$ORACLE_HOME/bin/emctl stop agent
```

On Windows systems,

```
%ORACLE_HOME%\bin\emctl stop iasconsole
```

```
%ORACLE_HOME%\bin\opmnctl stop agent
```

3. Perform a failover of the virtual IP from the failed node to the new active node.

**On Sun SPARC Solaris systems:**

- a. If the failed node is usable, login as root and execute the following command (on the failed node):

```
> ifconfig <interface_name> removeif <virtual_IP>
```

- b. Login as root on the new active node and execute the command:

```
> ifconfig <interface_name> addif <virtual_IP> up
```

**On Linux systems:**

- a. If the failed node is usable, login as root on the failed node and execute the following command:

```
> /sbin/ifconfig <interface_name> down
```

- b. Login as root on the new active node and execute the command:

```
> ifconfig <interface_name> netmask <netmask> <virtual_IP> up
```

**On Windows systems:**

On the failed node, move the group that was created using Oracle Fail Safe as follows:

- a. Start up Oracle Fail Safe Manager.
- b. Right-click the group that was created during the OracleAS middle-tier installation and select "Move to different node".

---



---

**Note:** If OracleAS JMS is using file-persistence, fail over the shared disk as well.

---



---

### 4.7.2.3 Manual Failover of Components for OracleAS Cold Failover Cluster (Middle-Tier)

After performing the failover steps for the virtual IP on the new active node, perform the following steps to failover on the OracleAS Cold Failover Cluster (Middle-Tier) system. Perform the following steps to stop and start Oracle Application Server processes:

1. Stop Oracle Application Server processes on the new active node and start OPMN only.

Execute the following commands on UNIX systems:

```
> $ORACLE_HOME/opmn/bin/opmnctl stopall
> $ORACLE_HOME/opmn/bin/opmnctl start
```

Execute the following commands on Windows systems:

```
> %ORACLE_HOME%\opmn\bin\opmnctl stopall
> %ORACLE_HOME%\opmn\bin\opmnctl start
```

2. Stop Oracle Application Server Administration processes on the new active node, using the following commands

On UNIX systems:

```
$ORACLE_HOME/bin/emctl stop iasconsole
$ORACLE_HOME/bin/emctl stop agent
```

On Windows systems,

```
%ORACLE_HOME%\bin\emctl stop iasconsole
%ORACLE_HOME%\bin\opmnctl stop agent
```

3. On the current active node, execute the following commands.

On UNIX systems:

```
> $ORACLE_HOME/opmn/bin/opmnctl stopall
> $ORACLE_HOME/opmn/bin/opmnctl startall
```



On Windows systems:

```
> %ORACLE_HOME%\opmn\bin\opmnctl stopall
> %ORACLE_HOME%\opmn\bin\opmnctl startall
```

4. If you use Application Server Control Console, start Oracle Application Server Administration processes on the current active node using the following commands.

On UNIX systems:

```
$ORACLE_HOME/bin/emctl start agent
$ORACLE_HOME/bin/emctl start iasconsole
```

On Windows systems,

```
%ORACLE_HOME%\bin\emctl start agent
%ORACLE_HOME%\bin\opmnctl start iasconsole
```

#### 4.7.2.4 Manual Failover of OracleAS Cluster (OC4J-JMS)

If you are using OracleAS Cluster (OC4J-JMS), and the system fails abnormally, you may need to perform additional failover steps such as removing lock files for OracleAS JMS file based persistence.

**See Also:** *Abnormal Termination in the Oracle Application Server Containers for J2EE Services Guide* section, "Oracle Application Server JMS".

## 4.8 Managing Custom Processes With OPMN

Oracle Process Manager and Notification Server (OPMN) supports management of custom processes, processes that you create and configure, as part of an Oracle Application Server installation. The features that OPMN provides for custom process management include the following:

- Provides a command-line interface for process control and monitoring for the process on an Oracle Application Server instance or across the OracleAS Cluster
- Channels all events from different Oracle Application Server component instances to all Oracle Application Server components that can utilize them
- Solves interdependency issues between Oracle Application Server components by enabling you to start and stop components and custom processes in order
- Provides automatic restart of custom processes when they become unresponsive, terminate unexpectedly, or become unreachable as determined by ping and notification operations
- Provides automatic death detection of custom processes

---

---

**Note:** If you add a custom process to OPMN that monitors and manages other processes, then OPMN only determines the status of the process it is monitoring.

For example, when OPMN is managing and monitoring Oracle Internet Directory processes, OPMN only manages the OIDMON process. The OIDMON process is in turn responsible for monitoring other Oracle Internet Directory processes. Thus when OPMN reports that Oracle Internet Directory is UP, OIDMON is running but, depending on the Oracle Internet Directory configuration, other processes such as OIDLDPD may still be in the process of starting or may even have failed in their initialization.

---

---

**See Also:** *Oracle Process Manager and Notification Server Administrator's Guide*

## 4.9 Managing Oracle Application Server Middle-tier Upgrades

When performing upgrades using systems in a high availability environment, the ultimate goal should be to upgrade all Oracle Application Server instances to the same version—in this case, Oracle Application Server 10g (10.1.2). Running all of the Oracle Application Server instances at the same version level is not mandatory; however, doing so makes it easier to manage, troubleshoot, and maintain J2EE applications and the Oracle Application Server components.

If you choose to maintain previous versions of the Oracle Application Server, you must consider which combinations of versions are supported.

This section covers the following topics:

- Section 4.9.1, "Upgrading Oracle Application Server Instances"
- Section 4.9.2, "Upgrading DCM-Managed OracleAS Clusters"
- Section 4.9.3, "Upgrading Stateful OC4J Applications"

### 4.9.1 Upgrading Oracle Application Server Instances

When you upgrade an OracleAS Cluster, Oracle Application Server instances need to be upgraded in a specific order to avoid unsupported or unstable configurations.

The general procedure for upgrading Oracle Application Server instances is:

1. Upgrade each of the middle-tier instances first.

For example, both Oracle Application Server 10g (9.0.4) and Oracle Application Server 10g (10.1.2) Application Server instances function using an Oracle Application Server 10g (9.0.4) Infrastructure.

2. Upgrade the Metadata Repository.
3. Upgrade the Identity Management system.

**See Also:**

- For information on upgrading Windows systems, Oracle Application Server 10g Upgrade and Compatibility Guide 10g Release 2 (10.1.2) for Windows
- For information on Upgrading UNIX systems, Oracle Application Server 10g Upgrade and Compatibility Guide 10g Release 2 (10.1.2) for UNIX

## 4.9.2 Upgrading DCM-Managed OracleAS Clusters

When using DCM-Managed OracleAS Clusters, each instance joined to the cluster must use the same Oracle Application Server version. Thus, before you upgrade instances in a DCM-Managed OracleAS Cluster, you need to do the following:

1. Leave the DCM-Managed Oracle Application Server Cluster using either Application Server Control Console, or the DCM `leavecluster` command.
2. Ensure that any old instance archives that need to be retained are exported to the file system using the DCM `exportarchive` command. The upgrade procedure does not upgrade archives. These archives can then be re-imported after the upgrade process.
3. After completing the upgrades for all the Oracle Application Server instances that are part of the DCM-Managed OracleAS Cluster, you can then join the instances into a new DCM-Managed OracleAS Cluster.

**See Also:** Section 4.3.2.6, "Performing Administration on a DCM-Managed OracleAS Cluster" on page 4-19 for information on how to minimize downtime while upgrading instances in a DCM-Managed OracleAS Cluster that uses a OracleAS File-based Farm.

## 4.9.3 Upgrading Stateful OC4J Applications

Beginning with Oracle Application Server 10g (10.1.2), OC4J applications running in a Oracle Application Server Cluster (OC4J) that use `HTTPSession` to store state can be upgraded with no session loss.

**See Also:**

- For information on upgrading Windows systems, Oracle Application Server 10g Upgrade and Compatibility Guide 10g Release 2 (10.1.2) for Windows
- For information on Upgrading UNIX systems, Oracle Application Server 10g Upgrade and Compatibility Guide 10g Release 2 (10.1.2) for UNIX

## 4.10 Using OracleAS Single Sign-On With OracleAS Cluster (Middle-Tier)

To enable Oracle Application Server Single Sign-On with OracleAS Cluster, the OracleAS Single Sign-On server needs to be aware of the entry point into the OracleAS Cluster, which is commonly the load balancing mechanism in front of the Oracle HTTP Servers. Usually, this is either Oracle Application Server Web Cache, a network load balancer appliance, or an Oracle HTTP Server installation.

In order to register an OracleAS Cluster's entry point with the OracleAS Single Sign-On server, use the `ssoreg.sh` script.

In order to use OracleAS Single Sign-On functionality, all Oracle HTTP Server instances in a OracleAS Cluster must have an identical OracleAS Single Sign-On registration.

- Each Oracle HTTP Server is registered with the same OracleAS Single Sign-On server.
- Each Oracle HTTP Server redirects a success, logout, cancel, or home message to the public network load balancer. In an OracleAS Cluster, each Oracle HTTP Server should redirect message URLs to the network load balancer. Since the client cannot access an Oracle HTTP Server directly, the client interacts with the network load balancer.

If you do not use a network load balancer, then the OracleAS Single Sign-On configuration must originate with whatever you use as the incoming load balancer—Oracle Application Server Web Cache, Oracle HTTP Server, and so on.

To configure a DCM-Managed OracleAS Cluster for single sign-on, execute the `ssoreg.sh` script against one of the application server instances in the DCM-Managed OracleAS Cluster. This tool registers the OracleAS Single Sign-On server and the redirect URLs with all Oracle HTTP Servers in the OracleAS Cluster:

1. On one of the application server instances, define the configuration with the `ssoreg.sh` script. Run `ssoreg.sh` with the options specified as follows, substituting your information for the italicized portions of the parameter values. See Table 4–6 for a full description of these values.

```

$ORACLE_HOME/sso/bin/ssoreg.sh
-oracle_home_path <orcl_home_path>
-site_name <site_name>
-config_mod_osso TRUE
-mod_osso_url <URL>
-u <userid>
[-virtualhost <virtual_host_name>]
[-update_mode CREATE | DELETE | MODIFY]
[-config_file <config_file_path>]
[-admin_info <admin_info>]
[-admin_id <adminid>]

```

- Specify the host, port, and SID of the database used by the Single Sign-On server.
  - Specify the host and port of the front-end load balancer in `mod_osso_url` parameter. This should be a HTTP or HTTPS URL depending on the site security policy regarding SSL access to OracleAS Single Sign-On protected resources.
  - Specify the root user of the host that you are executing this tool on in the `-u` option.
2. DCM propagates the configuration to all other Oracle HTTP Servers in the DCM-Managed OracleAS Cluster.

**Table 4–6** *ssoreg.sh Parameter Values*

Parameter	Value
oracle_home_path <orcl_home_path>	Absolute path to the Oracle home of the application server instance, where you are invoking this tool.
site_name <site_name>	Name of the site, typically, the effective host name and port of the partner application. For example, application.mydomain.com.
config_mod_osso TRUE	If set to TRUE, this parameter indicates that the application being registered is mod_osso. You must include config_mod_osso for osso.conf to be generated.
mod_osso_url <URL>	The effective URL of the partner application. This is the URL that is used to access the partner application. The value should be specified in this URL format:  http://oracle_http_host.domain:port
u <userid>	The user name that will start the Oracle HTTP Server. In UNIX, this name is usually "root." On Windows systems it is SYSTEM. The parameter u is mandatory.
virtualhost <virtual_ host_name>	Optional. Use this parameter only if registering an Oracle HTTP virtual host with the OracleAS Single Sign-On server.  If you create a virtual host, be sure, in the httpd.conf file, to fill in the following directive for each protected URL:  <VirtualHost host_name> OssConfigFile \$ORACLE_ HOME/Apache/Apache/conf/osso/host_name/osso.conf OssIpCheck off #<Location /your_protected_url> # AuthType basic # Require valid-user #</Location> #Other configuration information for the virtual host </VirtualHost>  The commented lines must be uncommented before the application is deployed.
update_mode CREATE   DELETE   MODIFY	Optional. Creates, deletes, or modifies the partner registration record. CREATE, the default, generates a new record. DELETE removes the existing record. MODIFY deletes the existing record and then creates a new one.
config_file <config_ file_path>	Optional. Location of the osso.conf file for the virtual host if one is being configured. It may, for example, be \$ORACLE_ HOME/Apache/Apache/conf/osso/virtual_host_name/osso.conf.  Note that the osso.conf for the non-virtual host is located at \$ORACLE_ HOME/Apache/Apache/conf/osso.
admin_id <name>	(Optional) User name of the mod_osso administrator. This shows up in the Single Sign-On tool as contact information.
admin_info <text>	(Optional) Additional information about the mod_osso administrator, such as e-mail address. This shows up in the OracleAS Single Sign-On tool as contact information.

The `ssoreg.sh` script establishes all information necessary to facilitate secure communication between the Oracle HTTP Servers in the OracleAS Cluster and the Single Sign-On server.

---

---

**Note:** When using OracleAS Single Sign-On with the Oracle HTTP Servers in the OracleAS Cluster, set the `KeepAlive` directive to `OFF`. When the Oracle HTTP Servers are behind a network load balancer; if the `KeepAlive` directive is set to `ON`, then the network load balancer maintains state with the Oracle HTTP Server for the same connection, which results in an HTTP 503 error. Modify the `KeepAlive` directive in the Oracle HTTP Server configuration. This directive is located in the `httpd.conf` file of the Oracle HTTP Server.

---

---

**See Also:** *Oracle Application Server Single Sign-On Administrator's Guide*

# Part III

---

---

## OracleAS Infrastructure High Availability

The chapters in this part discuss high availability for the OracleAS Infrastructure. These chapters are:

- Chapter 5, "Oracle Application Server Infrastructure High Availability"
- Chapter 6, "Managing and Operating Infrastructure High Availability"





---

---

# Oracle Application Server Infrastructure High Availability

This chapter focuses on the high availability aspects of the Oracle Application Server Infrastructure. It discusses the features and architectural solutions for high availability of the OracleAS Infrastructure and is divided into the following sections:

- Section 5.1, "Oracle Application Server Infrastructure Overview"
- Section 5.2, "Oracle Application Server Infrastructure Components"
- Section 5.3, "High Availability Configurations for Infrastructure"
- Section 5.4, "OracleAS Infrastructure Backup and Recovery Considerations"

## 5.1 Oracle Application Server Infrastructure Overview

Oracle Application Server provides a completely integrated infrastructure and framework for development and deployment of enterprise applications. An Oracle Application Server Infrastructure installation type provides centralized product metadata, security and management services, and configuration information and data repositories for the Oracle Application Server middle-tier. By integrating the Infrastructure services required by the middle tier, time and effort required to develop enterprise applications are reduced. In turn, the total cost of developing and deploying these applications is reduced, and the deployed applications are more reliable.

## 5.2 Oracle Application Server Infrastructure Components

The OracleAS Infrastructure consists of several components that contribute to its role and function. These components work with each other to provide the Infrastructure's product metadata, security, and management services. This section describes these components, which are:

- Section 5.2.1, "Oracle Application Server Metadata Repository"
- Section 5.2.2, "Oracle Identity Management"
- Section 5.2.3, "Oracle HTTP Server"
- Section 5.2.4, "Oracle Application Server Containers for J2EE (OC4J)"
- Section 5.2.5, "Oracle Enterprise Manager 10g Application Server Control Console"

## 5.2.1 Oracle Application Server Metadata Repository

The Oracle Application Server Metadata Repository (OracleAS Metadata Repository) stores component-specific information that is accessed by the Oracle Application Server middle-tier or Infrastructure components as part of their application deployment. The end user or the client application does not access this data directly. It stores three main types of metadata corresponding to the three main Infrastructure services described in Section 5.1, "Oracle Application Server Infrastructure Overview". These types of metadata are:

- product metadata
- identity management metadata
- management metadata

Table 5–1 shows the Oracle Application Server components that store and use these types of metadata during application deployment.

**Table 5–1 Metadata and Infrastructure Components**

Type of Metadata	Infrastructure Components Involved
Product metadata (includes demo data)	Oracle Application Server Metadata Repository
Identity Management metadata	OracleAS Single Sign-On, Oracle Internet Directory, Oracle Application Server Certificate Authority
Management metadata	Distributed Configuration Management,

Oracle Application Server metadata and customer or application data can co-exist in the same database as the Oracle Application Server Metadata Repository; the difference is in which applications that are allowed to access them. Customer applications will not be able to access product or management metadata.

### 5.2.1.1 When to Use Oracle Application Server Metadata Repository

OracleAS Metadata Repository is needed for all application deployments except for those using the J2EE and Web Cache installation type. Oracle Application Server provides two middle-tier installation options:

- **J2EE and Web Cache:** Installs Oracle HTTP Server, Oracle Application Server Containers for J2EE (OC4J), Oracle Application Server Web Cache (OracleAS Web Cache), Web Services, ADF Business Components, and Application Server Control Console.
- **Portal and Wireless:** Installs all components of J2EE and OracleAS Web Cache, plus UDDI, Oracle Application Server Portal (OracleAS Portal), Oracle Ultra Search, and Oracle Application Server Wireless (OracleAS Wireless).

The Distributed Configuration Management (DCM) component enables middle-tier management, and stores its metadata in the OracleAS Metadata Repository for the Portal and Wireless installation. For the J2EE and Web Cache installation type, by default, DCM uses a file-based repository. If you choose to associate the J2EE and Web Cache installation type with an OracleAS Infrastructure, the file-based repository is moved into the OracleAS Metadata Repository.

---

---

**Note:** The OracleAS Metadata Repository can be installed in an existing Real Application Clusters database or in a two-node cold failover cluster database. In a cold failover cluster, the database operates in an active-passive configuration using a hardware cluster to fail over database processes from the active to passive nodes. This is a configuration for the database that is agnostic to the Oracle Application Server Infrastructure services. The *Oracle Application Server Installation Guide* provides information on these installation scenarios.

---

---

## 5.2.2 Oracle Identity Management

The Oracle Identity Management framework in the Infrastructure includes the following components:

- Section 5.2.2.1, "Oracle Internet Directory"
- Section 5.2.2.2, "Oracle Application Server Single Sign-On"
- Section 5.2.2.3, "Oracle Delegated Administration Services"
- Section 5.2.2.4, "Oracle Directory Integration and Provisioning"
- Section 5.2.2.5, "Oracle Application Server Certificate Authority"

**See Also:** *Oracle Identity Management Concepts and Deployment Planning Guide*

### 5.2.2.1 Oracle Internet Directory

Oracle Internet Directory is Oracle's implementation of a directory service using the Lightweight Directory Access Protocol (LDAP) version 3. It provides a centralized repository for creating and managing users for the rest of the Oracle Application Server components such as OC4J, Oracle Application Server Portal, or Oracle Application Server Wireless. Central management of user authorization and authentication enables users to be defined centrally in Oracle Internet Directory and shared across all Oracle Application Server components.

Oracle Internet Directory is provided with a Java-based management tool (Oracle Directory Manager), a Web-based administration tool (Oracle Delegated Administration Services) for trusted proxy-based administration, and several command-line tools. Oracle Delegated Administration Services provide a means of provisioning end users in the Oracle Application Server environment by delegated administrators who are not the Oracle Internet Directory administrator. It also allows end users to modify their own attributes.

Oracle Internet Directory also enables Oracle Application Server components to synchronize data about users and group events, so that those components can update any user information stored in their local application instances.

**See Also:** *Oracle Internet Directory Administrator's Guide*

### 5.2.2.2 Oracle Application Server Single Sign-On

OracleAS Single Sign-On is a multi-part environment which is made up of both middle-tier and database functions allowing for a single user authentication across partner applications. An application can become a partner application that uses single sign-on either by using the SSOSDK or via the Apache `mod_ossso` module. This module allows Apache (and subsequently URLs) to be made partner applications.

OracleAS Single Sign-On is fully integrated with Oracle Internet Directory, which stores user information. It supports LDAP-based user and password management through Oracle Internet Directory.

OracleAS Single Sign-On supports Public Key Infrastructure (PKI) client authentication, which enables PKI authentication to a wide range of Web applications. Additionally, it supports the use of X.509 digital client certificates and Kerberos Security Tickets for user authentication.

By means of an API, OracleAS Single Sign-On can integrate with third-party authentication mechanisms such as Netegrity Site Minder.

**See Also:** *Oracle Application Server Single Sign-On Administrator's Guide*. (This guide also includes Identity Management replication instructions.)

### 5.2.2.3 Oracle Delegated Administration Services

Oracle Delegated Administration Services provide trusted proxy-based administration of directory information by users and application administrators. They employ a delegated administration model and application that enables the administrator of the Oracle Identity Management system to selectively delegate access rights to an administrator of an individual application, or directly to a user. Certain aspects of application administration can be delegated to users, which reduces the deployment cost of the applications.

With delegated administration, the management of the data inside the identity management system can be distributed to many different administrators depending upon their security requirements. This combination of centralized repository and delegated privileges results in secure and scalable administration in the identity management infrastructure.

Identity management privileges are created and granted to the bootstrap user who can further delegate these authorizations through the Oracle Delegated Administration Services self-service console. Some of these privileges include:

- Common identity management operational privileges, such as user creation, user profile modification, and group creation
- Privileges to install new Oracle applications using the identity management infrastructure
- Privileges to administer Oracle Delegated Administration Services

**See Also:** *Oracle Identity Management Concepts and Deployment Planning Guide*

### 5.2.2.4 Oracle Directory Integration and Provisioning

Oracle Directory Integration and Provisioning consists of a set of services and interfaces built into Oracle Internet Directory that facilitate the development of synchronization and provisioning solutions between Oracle Internet Directory and other repositories, such as third-party directories (for example, SunONE Directory and Microsoft Active Directory Services), application user repositories (as might be stored in a flat file, for example), or database tables containing HR information.

Oracle Directory Integration and Provisioning includes a documented API and incorporates available industry standards where they exist, making it possible for Oracle Corporation, customers, and third parties to develop and deploy customized synchronization and provisioning solutions. It also facilitates interoperability between Oracle Internet Directory and third-party meta directory and provisioning solutions.

Oracle Directory Integration and Provisioning enables you to:

- Synchronize data between Oracle Identity Management and other connected directories.
- Send notifications to target applications to reflect changes to a user's status or information.
- Develop and deploy your own connectivity agents.

Using Oracle Directory Integration and Provisioning, Oracle Identity Management leverages current investment in planning and deployment of a third-party enterprise directory. This provides the means to map and inherit major considerations such as directory naming, directory tree structure, schema extensions, access control, and security policies. Established procedures in an existing framework for user enrollment, identity, and account provisioning can be seamlessly incorporated into the corresponding operations of Oracle Identity Management.

**See Also:** *Oracle Identity Management Concepts and Deployment Planning Guide*

### 5.2.2.5 Oracle Application Server Certificate Authority

Oracle Application Server Certificate Authority (OCA) issues, revokes, renews, and publishes X.509 v3 certificates to support PKI-based strong authentication methods. Usually, OCA is deployed on standalone mode.

---



---

**Note:** OCA cannot be installed as part of an Oracle Application Server Cluster (Identity Management). It has to be installed separately.

---



---

**See Also:** *Oracle Application Server Certificate Authority Administrator's Guide*

## 5.2.3 Oracle HTTP Server

The Infrastructure installation type installs Oracle HTTP Server for the Infrastructure. This is used to service requests from other distributed components of the Infrastructure and middle-tier instances. In the Infrastructure, Oracle HTTP Server services requests for OracleAS Single Sign-On and Oracle Delegated Administration Services. The latter is implemented as a servlet in an OC4J process in the Infrastructure.

**See Also:** *Oracle HTTP Server Administrator's Guide*

## 5.2.4 Oracle Application Server Containers for J2EE (OC4J)

In the Infrastructure, OC4J is installed in the Infrastructure to run Oracle Delegated Administration Services and OracleAS Single Sign-On. The former runs as a servlet in OC4J.

Oracle Delegated Administration Services provide a self-service console (for end users and application administrators) that can be customized to support third-party applications. In addition, it provides a number of services for building customized administration interfaces that manipulate Oracle Internet Directory data. Oracle Delegated Administration Services are a component of Oracle Identity Management.

**See Also:** *Oracle Internet Directory Administrator's Guide* for more information about Oracle Delegated Administration Services.

## 5.2.5 Oracle Enterprise Manager 10g Application Server Control Console

Oracle Enterprise Manager 10g Application Server Control Console (Application Server Control Console) provides a Web-based interface for managing Oracle Application Server components and applications. Using the Oracle Application Server Console, you can do the following:

- monitor Oracle Application Server components, Oracle Application Server middle-tier and Infrastructure instances, OracleAS Clusters, and deployed J2EE applications and their components
- configure Oracle Application Server components, instances, OracleAS Clusters, and deployed applications
- operate Oracle Application Server components, instances, OracleAS Clusters, and deployed applications
- manage security for Oracle Application Server components and deployed applications

For more information on Oracle Enterprise Manager and its two frameworks, see *Oracle Enterprise Manager Concepts*.

**See Also:** *Oracle Application Server Administrator's Guide* - provides descriptions on Application Server Control Console and instructions on how to use it.

## 5.3 High Availability Configurations for Infrastructure

As described earlier the OracleAS Infrastructure provides the following services

- product metadata
- security service
- management service

From an availability standpoint, these services are provided by the following components, which must all be available to guarantee availability of the Infrastructure:

- OracleAS Metadata Repository
- Oracle Net listener
- Oracle HTTP Server
- For Oracle Identity Management:
  - Oracle Internet Directory and Oracle Internet Directory monitor
  - OC4J Oracle Delegated Administration Services instance
  - OracleAS Single Sign-On
  - Oracle Directory Integration and Provisioning
- For Oracle Management Services:
  - Distributed Configuration Management

For the Infrastructure to provide all essential services, all of the above components must be available. On UNIX platforms, this means that the processes associated with these components must be up and active. Any high availability solution must be able to detect and recover from any type of software failures of any of the processes associated with the Infrastructure components. It must also be able to detect and

recover from any type of hardware failures on the hosts that are running the Infrastructure.

In Oracle Application Server, all of the Infrastructure processes, except the database, its listener, and Application Server Control Console, are started, managed, and restarted by the Oracle Process Management and Notification (OPMN) framework. This means any failure of an OPMN-managed process is handled internally by OPMN. OPMN is automatically installed and configured at installation time. However, any database process failure or database listener failure is not handled by OPMN. Also, failure of any OPMN processes leaves the Infrastructure in a non-resilient mode if the failure is not detected and appropriate recovery steps are not taken. Refer to Section 2.2.1, "Process Death Detection and Automatic Restart" on page 2-4 for further discussion on process management and monitoring.

This release of Oracle Application Server provides intra-site high availability solutions for the OracleAS Infrastructure, which can be generally categorized into the following:

- Section 5.3.1, "Active-Active High Availability Solutions"
- Section 5.3.2, "Active-Passive High Availability Solutions"

Table 5–2 summarizes each of the above. In general, there are three solutions. One is active-active, the other two are active-passive. For each of these solutions, you can choose to have co-located Oracle Identity Management components or the OracleAS Single Sign-On and Oracle Delegated Administration Services components can be distributed out into their own nodes separate from Oracle Internet Directory node. For the OracleAS Cluster (Identity Management) and OracleAS Cold Failover Cluster (Identity Management), which are focused on Oracle Identity Management, the OracleAS Metadata Repository can be installed in its own or existing cold failover database or an existing Real Application Clusters database.

**Table 5–2 Summary of intra-site high availability solutions**

Solutions	Description
Active-Active	<p>The primary characteristics of this solution are that multiple active instances of each of the OracleAS Infrastructure services are available during normal operations, and all of the instances are servicing requests concurrently. If an instance fails, the remaining active instances take over the work load of the failed instance.</p> <p>Active-active solutions are:</p> <ul style="list-style-type: none"> <li>■ OracleAS Cluster (Identity Management)</li> <li>■ Distributed OracleAS Cluster (Identity Management)</li> </ul>
Active-Passive	<p>In this configuration, only one of the OracleAS Infrastructure instances is active at any time. When the active instance fails, the passive instance becomes active and takes over the entire workload of the failed instance.</p> <p>Active-passive solutions are:</p> <ul style="list-style-type: none"> <li>■ OracleAS Cold Failover Cluster (Infrastructure)</li> <li>■ Distributed OracleAS Cold Failover Cluster (Infrastructure)</li> <li>■ OracleAS Cold Failover Cluster (Identity Management)</li> <li>■ Distributed OracleAS Cold Failover Cluster (Identity Management)</li> </ul>

The above high availability solutions provide protection from local hardware and software failures that cannot be detected and recovered by OPMN. Examples of such

failures are a system panic or node crash. These solutions, however, cannot protect the Infrastructure from site failures or media failures, which result in damage to or loss of data.

Oracle Application Server provides a site-level active-passive disaster recovery solution to protect against disasters and site failures. This solution is described in Chapter 7, "Oracle Application Server Disaster Recovery".

A site failure or disaster will most likely affect all the systems including middle tiers, OracleAS Infrastructure, and backend databases. Hence, the disaster recovery solution also provides mechanisms to protect all components in the middle tier and OracleAS Infrastructure including any databases used by these components.

In short, the intra-site high availability solutions provide resilience for only the Infrastructure from local hardware and software failures. The middle tier can continue to function with a resilient Infrastructure. The Oracle Application Server Disaster Recovery solution, on the other hand, deals with a complete site failure, which requires failing over not only the Infrastructure but also the middle tier. The intra-site high availability solutions for the Infrastructure are discussed in the following sections.

In the case of media failures, the Oracle Application Server Backup and Recovery Tool provides a way for Oracle Application Server metadata, in both the OracleAS Metadata Repository database and file system, to be backed up for recovery if a storage media fails or if the metadata is somehow corrupted. For more details about this tool see *Oracle Application Server Administrator's Guide*.

### 5.3.1 Active-Active High Availability Solutions

The active-active solutions specifically provide active-active high availability for the Oracle Identity Management components. They are also known as **Oracle Application Server Cluster (Identity Management)** solutions.

Two OracleAS Cluster (Identity Management) solutions exist: non distributed and distributed Oracle Identity Management components. For both, the Oracle Identity Management components need not be installed on machines that are part of a hardware cluster.

**Table 5–3 OracleAS Cluster (Identity Management) solutions (active-active)**

Solution	Description
OracleAS Cluster (Identity Management)	In the non distributed solution, all the Oracle Identity Management components are installed on each of two or more nodes. These nodes are deployed behind a load balancer, which directs requests to them. If a node fails, the load balancer directs requests to the remaining node(s).
Distributed OracleAS Cluster (Identity Management)	In the distributed solution, OracleAS Single Sign-On and Oracle Delegated Administration Services components are deployed on separate machines from Oracle Internet Directory and Oracle Directory Integration and Provisioning. These machines are fronted by a load balancer, which gives them active-active availability. Separating out OracleAS Single Sign-On and Oracle Delegated Administration Services components enable the other Oracle Identity Management components to be secured behind a firewall.



### Oracle Application Server Metadata Repository High Availability Options

For both of the above solutions, the Oracle Identity Management components on all nodes are connected to the same directory store database. High availability for this database, which is also used by the OracleAS Metadata Repository, is achieved by using Oracle Application Server Metadata Repository Creation Assistant to install the directory store and metadata repository into an existing database. This database is already installed in one of the following high availability configurations:

- Real Application Clusters
- two-node cold failover cluster database

Active-active solutions are presented in the following sections:

- Section 5.3.1.1, "OracleAS Cluster (Identity Management)"
- Section 5.3.1.2, "Distributed OracleAS Cluster (Identity Management)"

**See Also:** *Oracle Internet Directory Administrator's Guide*

#### 5.3.1.1 OracleAS Cluster (Identity Management)

In this configuration, Oracle Identity Management components (Oracle Internet Directory, OracleAS Single Sign-On, Oracle Delegated Administration Services, and Oracle Directory Integration and Provisioning) are deployed together in two or more nodes. Each node runs all of the Oracle Identity Management components mentioned above. Traffic to these nodes is load balanced by a redundant load balancer. Figure 5-1 shows the layout of this configuration.

---

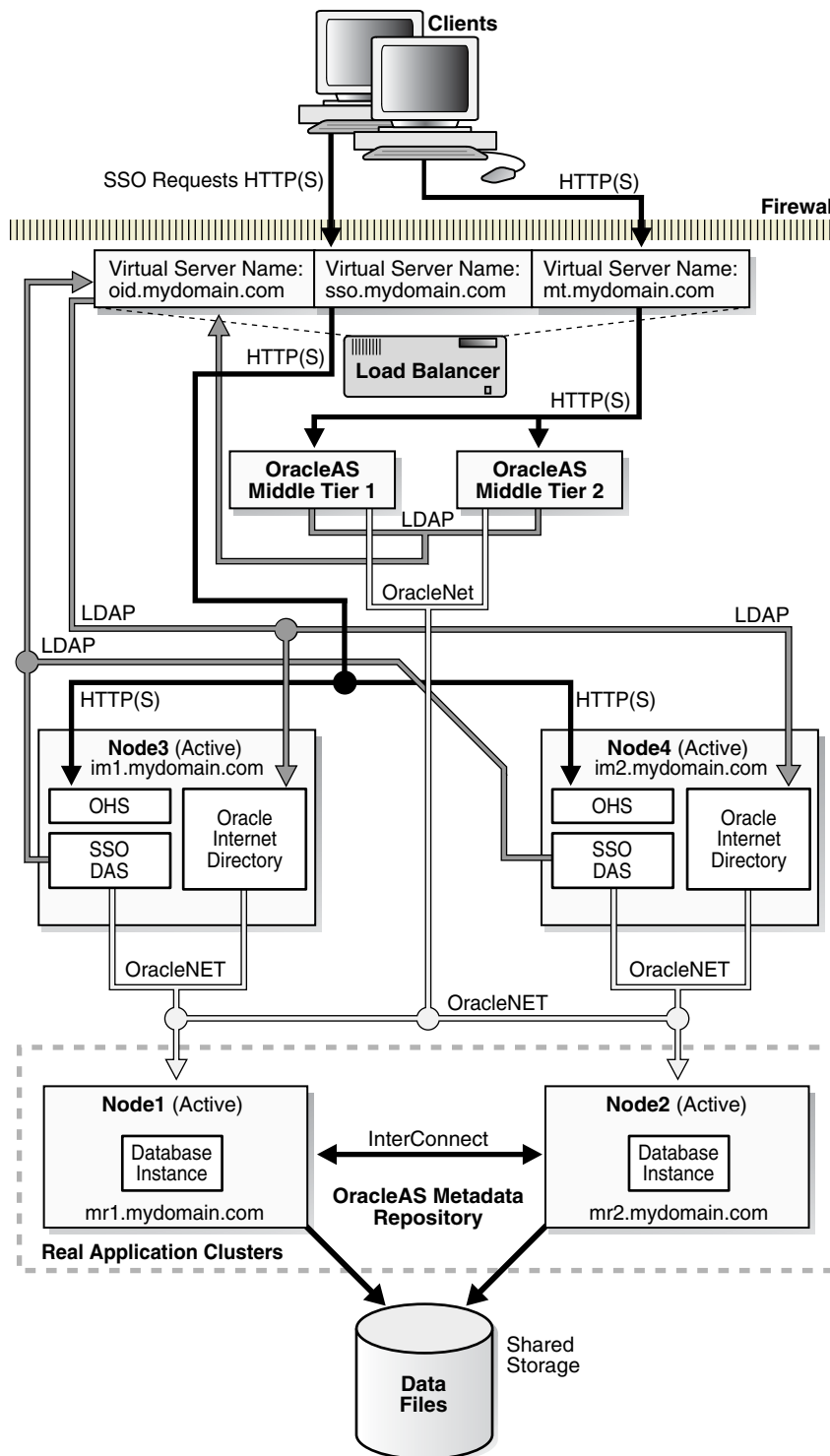
---

**Note:** Oracle Application Server Certificate Authority cannot be installed as part of an Oracle Application Server Cluster (Identity Management). It has to be installed separately.

---

---

**Figure 5–1 OracleAS Cluster (Identity Management) configuration**



The nodes hosting the Oracle Identity Management components should be functionally equivalent. These nodes provide active-active availability for Oracle Identity Management services. OracleAS Single Sign-On and Oracle Delegated Administration Services are deployed in a single OC4J\_SECURITY instance in each Oracle home. An Oracle Internet Directory process also runs in each node.

The load balancer can be configured with three virtual server names: one for single sign-on, one for Oracle Internet Directory, and one for the middle tier (as shown in Figure 5-1 ). The single sign-on virtual server name is used by requests from clients to access OracleAS Single Sign-On. The Oracle Internet Directory virtual server name is used by LDAP and JNDI requests from the middle tier and OracleAS Single Sign-On to access Oracle Internet Directory. And, clients use the middle-tier virtual server name to access the middle tier.

OracleAS Single Sign-On/Oracle Delegated Administration Services and Oracle Internet Directory access the database instances through Oracle Net load balancing. Note that OracleAS Single Sign-On establishes connection pools to access the database. A connection in the pool can be to any of the database instances in the Real Application Clusters cluster.

OPMN on each node provides process management, monitoring, and notification services for the OC4J\_SECURITY instances and Oracle HTTP Server processes. When a process for one of these instances fails, OPMN detects the failure and attempts to restart it. If the restart is unsuccessful, the load balancer detects the failure (usually through a non response timeout) and directs requests to another active OC4J\_SECURITY instance in another node of the Oracle Application Server Identity Management Cluster.

oidmon monitors the Oracle Internet Directory processes oidldapd, oidrepld and odisrv while OPMN monitors oidmon. If oidldapd, oidrepld, or odisrv fails, oidmon will attempt to restart it locally. Similarly, if oidmon fails, OPMN will try to restart it locally. Only one odisrv process and one oidrepld process can be active at any time in an OracleAS Cluster (Identity Management) while multiple oidldapd processes can run in the same cluster. Refer to *Oracle Internet Directory Administrator's Guide* for more details.

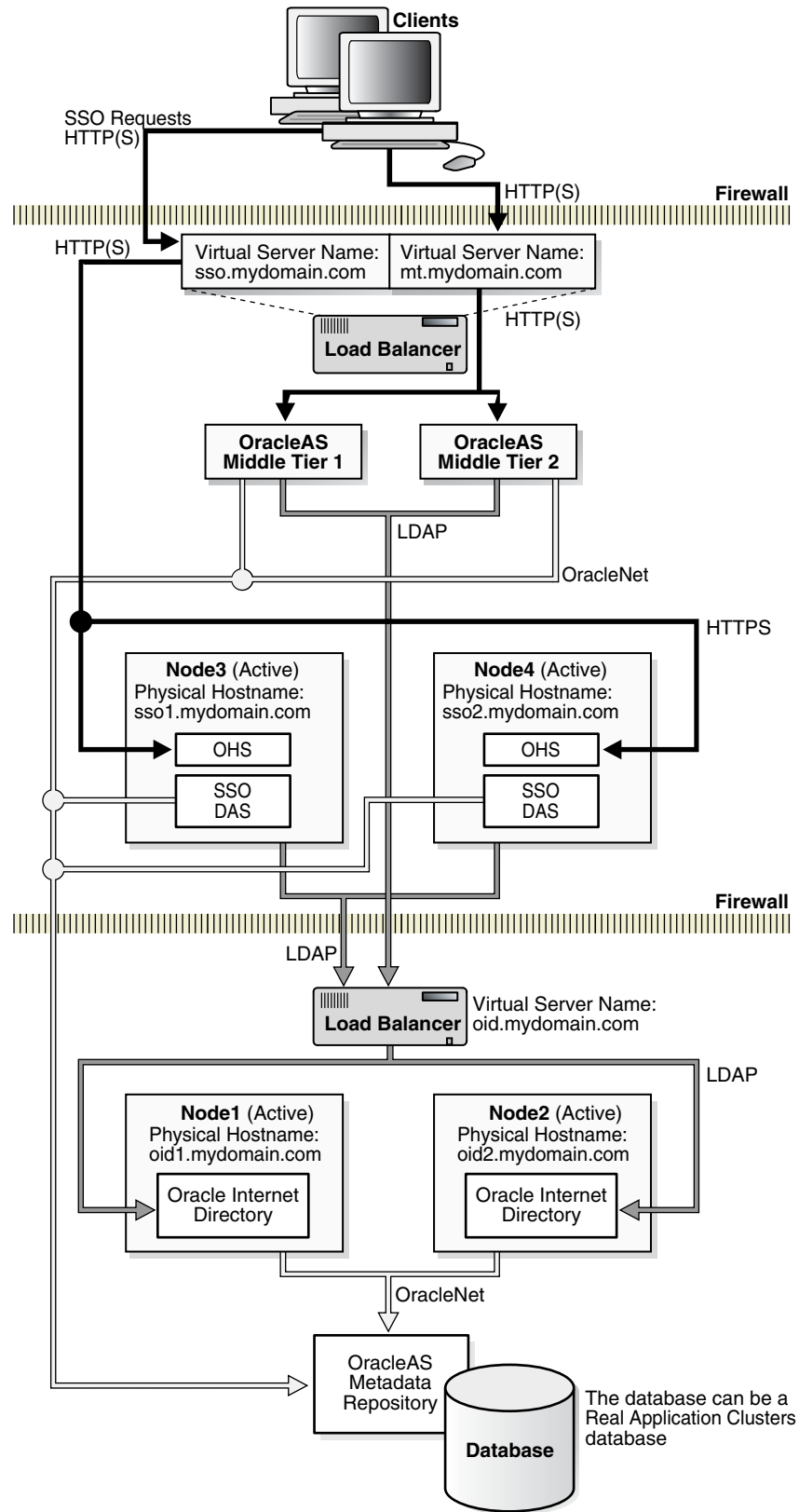
In the event an Oracle Identity Management node fails, the load balancer detects the failure and redirects requests to a remaining active node. Because each node provides identical services as the others, all requests will be fulfilled by the remaining nodes. For more information on Oracle Internet Directory in OracleAS Cluster (Identity Management) and how directory replication can provide high availability, see the OracleAS Cluster (Identity Management) chapter in the *Oracle Internet Directory Administrator's Guide*.

For the database tier, node failures are managed by Oracle Net and Real Application Clusters if the OracleAS Metadata Repository is installed in an existing Real Application Clusters database. Oracle Net redirects requests to remaining active database instances if any of the other database instances fail. If the OracleAS Metadata Repository is installed in a cold failover cluster database, node failure is performed by switching the virtual hostname and IP to the standby node and starting the relevant processes on that node. Section 6.2, "Managing Oracle Application Server Cold Failover Cluster (Infrastructure)" provides instructions on how to accomplish these tasks.

### 5.3.1.2 Distributed OracleAS Cluster (Identity Management)

This configuration is a variation of that described in Section 5.3.1.1. Instead of co-locating all Oracle Identity Management components on each of the OracleAS Cluster (Identity Management) nodes, the OracleAS Single Sign-On and Oracle Delegated Administration Services components are distributed out to another set of OracleAS Cluster (Identity Management) nodes. Hence, this configuration, as shown in Figure 5-2, has three tiers: database (OracleAS Metadata Repository) tier, Oracle Internet Directory tier, and OracleAS Single Sign-On/Oracle Delegated Administration Services tier.

**Figure 5–2 Distributed OracleAS Cluster (Identity Management) Configuration**



The OracleAS Single Sign-On/Oracle Delegated Administration Services tier is fronted by a redundant load balancer that distributes requests to the nodes in this tier. The Oracle Application Server middle-tier and clients access OracleAS Single Sign-On and Oracle Delegated Administration Services using the virtual server name of this load balancer. (Note that the middle tier could also be installed in the same nodes as the OracleAS Single Sign-On and Oracle Delegated Administration Services nodes.)

The advantage of this configuration is that the nodes hosting the OracleAS Single Sign-On and Oracle Delegated Administration Services components can be deployed in the DMZ. That allows the Oracle Internet Directory to be deployed inside an intranet, protected by the firewalls (as depicted in Figure 5–2).

---

**Note:** Oracle Application Server Certificate Authority cannot be installed as part of an Distributed OracleAS Cluster (Identity Management). It has to be installed separately.

---

### 5.3.2 Active-Passive High Availability Solutions

The Oracle Application Server active-passive solutions use a cold failover cluster configuration on a hardware cluster. These solutions are explained in Table 5–4. The variations in the solutions are based on the way OracleAS Infrastructure components are set up and distributed.

**Table 5–4 Overview of OracleAS Cold Failover Cluster solutions**

Solution	Description
OracleAS Cold Failover Cluster (Infrastructure)	<p>This a two-node, active-passive configuration on a hardware cluster. The two nodes are connected to shared storage. OracleAS Metadata Repository and Oracle Identity Management components are installed together into the same Oracle home, which resides in the shared storage. A new database is installed at the same time for the OracleAS Metadata Repository.</p> <p>Hence, OracleAS Metadata Repository and Oracle Identity Management are active together on one node and both are passive on the other node. This solution is the easiest to install and configure out-of-box.</p> <p>Details can be found in Section 5.3.2.1.</p>
Distributed OracleAS Cold Failover Cluster (Infrastructure)	<p>In this configuration, the OracleAS Single Sign-On and Oracle Delegated Administration Services components are installed in separate machines from the Oracle Internet Directory and Oracle Directory Integration and Provisioning components.</p> <p>OracleAS Single Sign-On and Oracle Delegated Administration Services are installed in two or more machines that are load balanced by a load balancer router. OracleAS Single Sign-On and Oracle Delegated Administration Services are active-active. However, Oracle Internet Directory and Oracle Directory Integration and Provisioning are installed in an OracleAS Cold Failover Cluster with the OracleAS Metadata Repository. A new database is installed for the OracleAS Metadata Repository. The OracleAS Cold Failover Cluster enables active-passive availability.</p> <p>Note that for this configuration, OracleAS Single Sign-On/Oracle Delegated Administration Services are active-active, and Oracle Internet Directory and OracleAS Metadata Repository are active-passive.</p> <p>Details can be found in Section 5.3.2.2.</p>

**Table 5–4 (Cont.) Overview of OracleAS Cold Failover Cluster solutions**

Solution	Description
OracleAS Cold Failover Cluster (Identity Management)	<p>This configuration requires the installation of Oracle Identity Management components in a hardware cluster. The OracleAS Metadata Repository is installed separately and can be installed in an existing database using OracleAS Metadata Repository Creation Assistant. Hence, Oracle Identity Management has a different Oracle home from OracleAS Metadata Repository. Failover of Oracle Identity Management can be performed independently of OracleAS Metadata Repository and vice versa. Details can be found in Section 5.3.2.3.</p>
Distributed OracleAS Cold Failover Cluster (Identity Management)	<p>This configuration differs from the overall OracleAS Cold Failover Cluster (Identity Management) (Section 5.3.2.3) solution in that the OracleAS Single Sign-On and Oracle Delegated Administration Services components are installed in separate machines from the Oracle Internet Directory and Oracle Directory Integration and Provisioning components.</p> <p>OracleAS Single Sign-On and Oracle Delegated Administration Services are deployed on separate machines from other Oracle Identity Management components. They can be installed on the OracleAS middle-tier machines and can have active-active availability as they are fronted by a load balancer. Separating out OracleAS Single Sign-On and Oracle Delegated Administration Services components enable the other Oracle Identity Management components to be secured behind a firewall. The other Oracle Identity Management components, Oracle Internet Directory and Oracle Directory Integration and Provisioning are installed on a two node active-passive cold failover hardware cluster. OracleAS Metadata Repository is installed in an existing database using OracleAS Metadata Repository Creation Assistant. This database can use a cold failover cluster, Real Application Clusters, or other database-certified configurations to provide high availability.</p> <p>This configuration is very similar to that in Section 5.3.2.2, "Distributed OracleAS Cold Failover Cluster (Infrastructure)". The difference is that, in the hardware cluster where Oracle Internet Directory and OracleAS Metadata Repository are installed, each has a separate Oracle home from the other.</p> <p>Note that for this configuration, OracleAS Single Sign-On/Oracle Delegated Administration Services are active-active, and Oracle Internet Directory server is active-passive. OracleAS Metadata Repository availability is dependent on the high availability configuration used by the database.</p> <p>Details of this solution can be found in Section 5.3.2.4.</p>

### Oracle Application Server Metadata Repository High Availability Options

For the solutions in this section, various high availability options are available for the OracleAS Metadata Repository.

For the OracleAS Cold Failover Cluster (Infrastructure) solution and its distributed variant, a new database is installed for the OracleAS Metadata Repository. Hence, the OracleAS Metadata Repository is in a cold failover cluster database.

For the OracleAS Cold Failover Cluster (Identity Management) and its distributed variant, the OracleAS Metadata Repository is installed in an existing database. This database can be in one of the following:

- Real Application Clusters

- cold failover cluster database

This section is divided into the following:

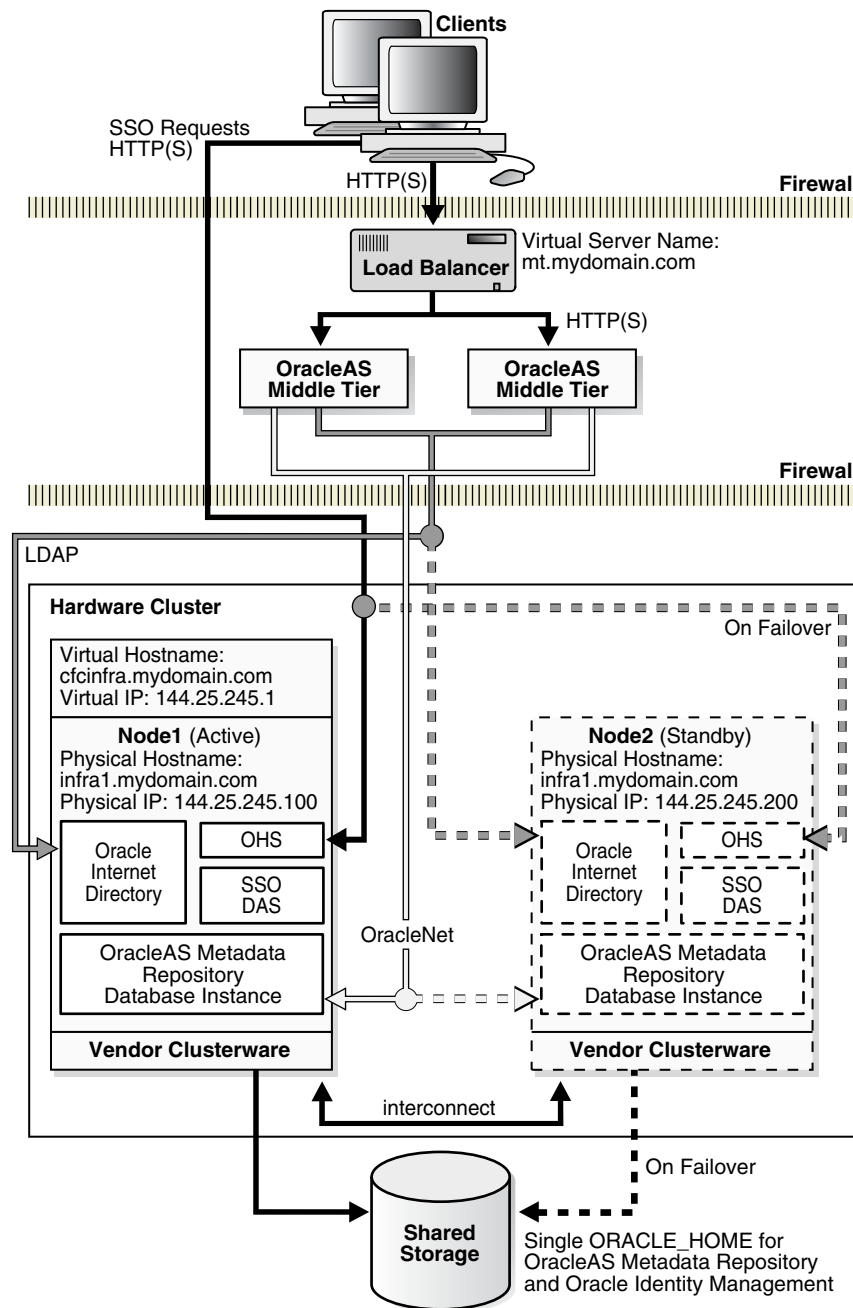
- Section 5.3.2.1, "OracleAS Cold Failover Cluster (Infrastructure)"
- Section 5.3.2.2, "Distributed OracleAS Cold Failover Cluster (Infrastructure)"
- Section 5.3.2.3, "OracleAS Cold Failover Cluster (Identity Management)"
- Section 5.3.2.4, "Distributed OracleAS Cold Failover Cluster (Identity Management)"

### **5.3.2.1 OracleAS Cold Failover Cluster (Infrastructure)**

Figure 5–3 shows the architecture of an OracleAS Cold Failover Cluster high availability solution with co-located Oracle Identity Management and OracleAS Metadata Repository. Co-location implies that Oracle Identity Management and OracleAS Metadata Repository are installed in the same Oracle home.

The architecture consists of a two-node cluster, which are attached to shared storage. The shared storage contains the single Oracle home shared by both nodes. Only one node, the active node, is mounted to the shared storage at any time.

**Figure 5-3 Normal operation of OracleAS Cold Failover Cluster (Infrastructure)**



### 5.3.2.1.1 Installation

Installation of the co-located OracleAS Cold Failover Cluster solution involves running the installer on a single node. A single ORACLE\_HOME is then installed on the shared storage for both nodes. Only one node is active at any time and this active node mounts the shared storage and provides access to the ORACLE\_HOME.

During the installation process, when asked to select the Infrastructure installation type, select "Identity Management and OracleAS Metadata Repository". The installer installs both OracleAS Metadata Repository and Oracle Identity Management components (Oracle Internet Directory server, Oracle Directory Integration and Provisioning, OracleAS Single Sign-On, and Oracle Delegated Administration



Services) into an `ORACLE_HOME` in the shared storage. At the same time, a new database is installed for the OracleAS Metadata Repository.

Also during installation, a high availability addressing screen appears where you can specify the virtual hostname for the hardware cluster. This name is associated with the virtual IP of the hardware cluster and is used to access the active node of the OracleAS Cold Failover Cluster.

**See Also:** *Oracle Application Server Installation Guide*

#### 5.3.2.1.2 Runtime

For illustration purposes as shown in Figure 5–3, assume a virtual/logical IP address of 144.25.245.1 is active on physical Node 1. Hence, Node 1 is the primary or active node. The virtual hostname, `cfcinfra.mydomain.com`, is mapped to this virtual IP address, and the middle tier associates the Infrastructure with `cfcinfra.mydomain.com`.

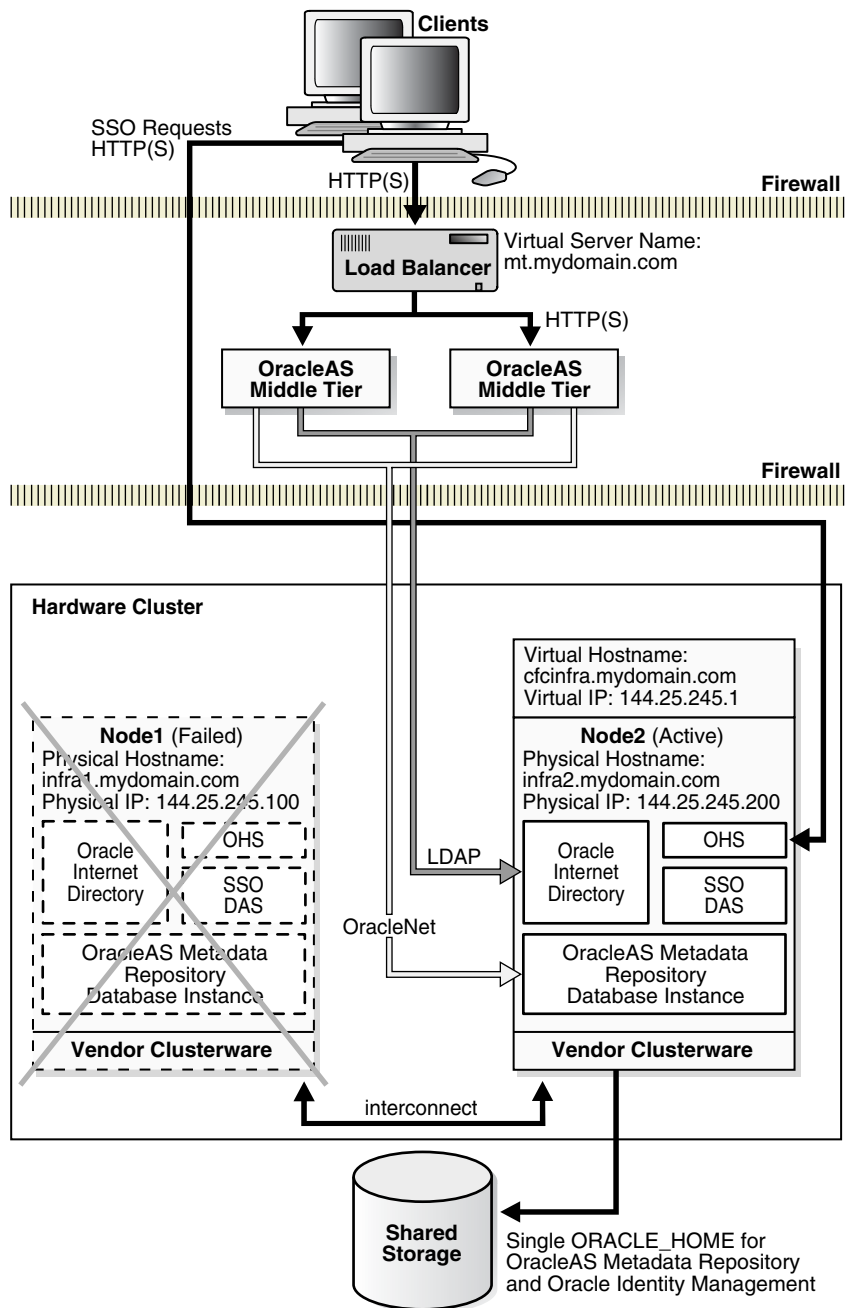
In normal operating mode, the hardware cluster's software enables the virtual IP 144.25.245.1 on physical Node 1 and starts all Infrastructure processes (database, database listener, Oracle Enterprise Manager 10g process, and OPMN) on that node. OPMN then starts, monitors, and restarts, if necessary, any of the following failed Infrastructure processes: Oracle Internet Directory, OC4J instances, and Oracle HTTP Server.

#### 5.3.2.1.3 Failover

If the primary node fails, the virtual IP address 144.25.245.1 is manually enabled on the secondary node (Figure 5–4). All the Infrastructure processes are then started on the secondary node. The middle-tier processes accessing the Infrastructure will see a temporary loss of service as the virtual IP and the shared storage are moved over and the database, database listener, and all other Infrastructure processes are started. Once the processes are up, middle-tier processes that were retrying during this time are reconnected. New connections are not aware that a failover has occurred.

If you plan to use the Automatic Storage Management (ASM) feature of Oracle Database 10g for the OracleAS Metadata Repository, the CSS daemon must be configured on the standby node. The CSS daemon synchronizes ASM instances with the database instances that use the ASM instances for database file storage. Specific instructions are provided in the chapter on installing OracleAS Cold Failover Cluster in the *Oracle Application Server Installation Guide*.

**Figure 5-4 OracleAS Cold Failover Cluster (Infrastructure) after primary node failover**



While the hardware cluster framework can start, monitor, detect, restart, or failover Infrastructure processes, these actions are not automatic and involve some scripting or simple programming. Some of these details are discussed in Chapter 6.

For information on setting up and operating the OracleAS Cold Failover Cluster solution for the Infrastructure, see *Oracle Application Server Installation Guide*. This guide covers pre-installation, installation, and post-installation tasks, if any.

**5.3.2.1.4 Summary of High Availability Capabilities** The co-located OracleAS Cold Failover Cluster high availability configuration provides the following capabilities:

- Node failure protection - hardware cluster protects from planned or unplanned node outage.
- Process failure detection and restart performed by hardware cluster system and OPMN.

**5.3.2.1.5 OracleAS Cold Failover Cluster (Infrastructure) Solution for Windows Using Oracle Fail Safe** The OracleAS Cold Failover Cluster solution consists of a two-node cluster accessing a shared disk (see Figure 5-5) that contains the Infrastructure's data files. At any point in time, only one node is active. During normal operation, the second node is on standby. OracleAS middle-tier components access the cluster through a virtual hostname that is mapped to a virtual IP in the subnet. In the example in Figure 5-5, the virtual hostname `cfcinfra.mydomain.com` and virtual IP 144.25.245.1 are used. When a failover occurs from node 1 to node 2, these virtual hostname and IP are moved to the standby node, which now becomes the active node. The failure of the active node is transparent to the OracleAS middle-tier components.

---

---

**Note:** Only static IP addresses can be used in the OracleAS Cold Failover Cluster (Infrastructure) solution for Windows.

---

---



- Application Server Control Console

Central to the Windows OracleAS Cold Failover Cluster solution is the concept of resource groups. A group is a collection of resources defined through Oracle Fail Safe. During failover from the active node to the standby node, the group, and hence, the resources in it, failover as a unit. During installation and configuration of the OracleAS Cold Failover Cluster, a single group is defined for the solution. This group consists of the following:

- virtual IP for the cluster
- virtual hostname for the cluster
- shared disk
- Infrastructure database
- TNS listener for the database
- OPMN
- Application Server Control Console

The integration of Oracle Fail Safe and Microsoft Cluster Server provide an easy to manage environment and automatic failover functionality in the OracleAS Cold Failover Cluster solution. The Infrastructure database, its TNS listener, and OPMN are installed as Windows services and are monitored by Oracle Fail Safe and Microsoft Cluster Server. Upon failure of any of these Windows services, Microsoft Cluster Server will try to restart the service three times (default setting) before failing the group to the standby. Additionally, OPMN monitors, starts, and restarts the Oracle Internet Directory, OC4J, and Oracle HTTP Server processes.

---



---

**See Also:** *Oracle Application Server Installation Guide* for details on the installation process and requirements for installation.

---

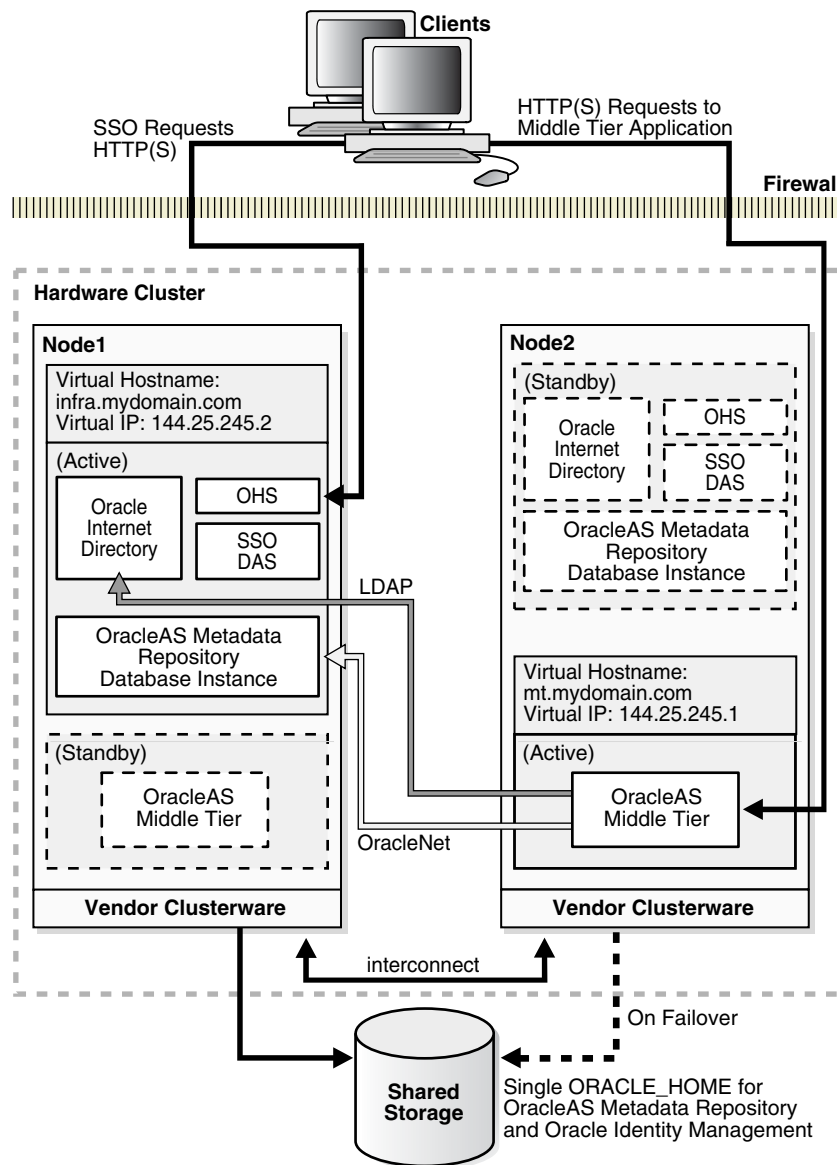


---

**5.3.2.1.6 OracleAS Cold Failover Cluster (Infrastructure) with Co-located OracleAS Cold Failover Cluster (Middle-Tier)** This is a solution where the middle tier, in a cold failover configuration, is co-located on the same nodes as the OracleAS Cold Failover Cluster (Infrastructure). The middle tier is in an OracleAS Cold Failover Cluster (Middle-Tier). The OracleAS Cold Failover Cluster (Infrastructure) and OracleAS Cold Failover Cluster (Middle-Tier) have separate virtual hostnames mapping to separate virtual IPs. This allows the Infrastructure to failover independently from the middle tier and vice versa.

In Figure 5–6, the Infrastructure is active on Node 1 while the middle tier is active on Node 2. If Node 2 fails, the middle tier can failover to Node 1. If Node 1 fails, the Infrastructure can failover to Node 2. By having the Infrastructure active on one node and the middle tier active on the other node during normal operation, resources are efficiently utilized as both nodes are performing work and no node is idle. This configuration also provides high performance isolation since the middle tier and the OracleAS Infrastructure services run in separate environments. See *Oracle Application Server Installation Guide* for instructions on installing such a configuration.

**Figure 5–6 OracleAS Cold Failover Cluster (Middle-Tier) on the same nodes as OracleAS Cold Failover Cluster (Infrastructure)**



\* OracleAS middle tier Oracle homes are installed on local storage on nodes 1 and 2

### 5.3.2.2 Distributed OracleAS Cold Failover Cluster (Infrastructure)

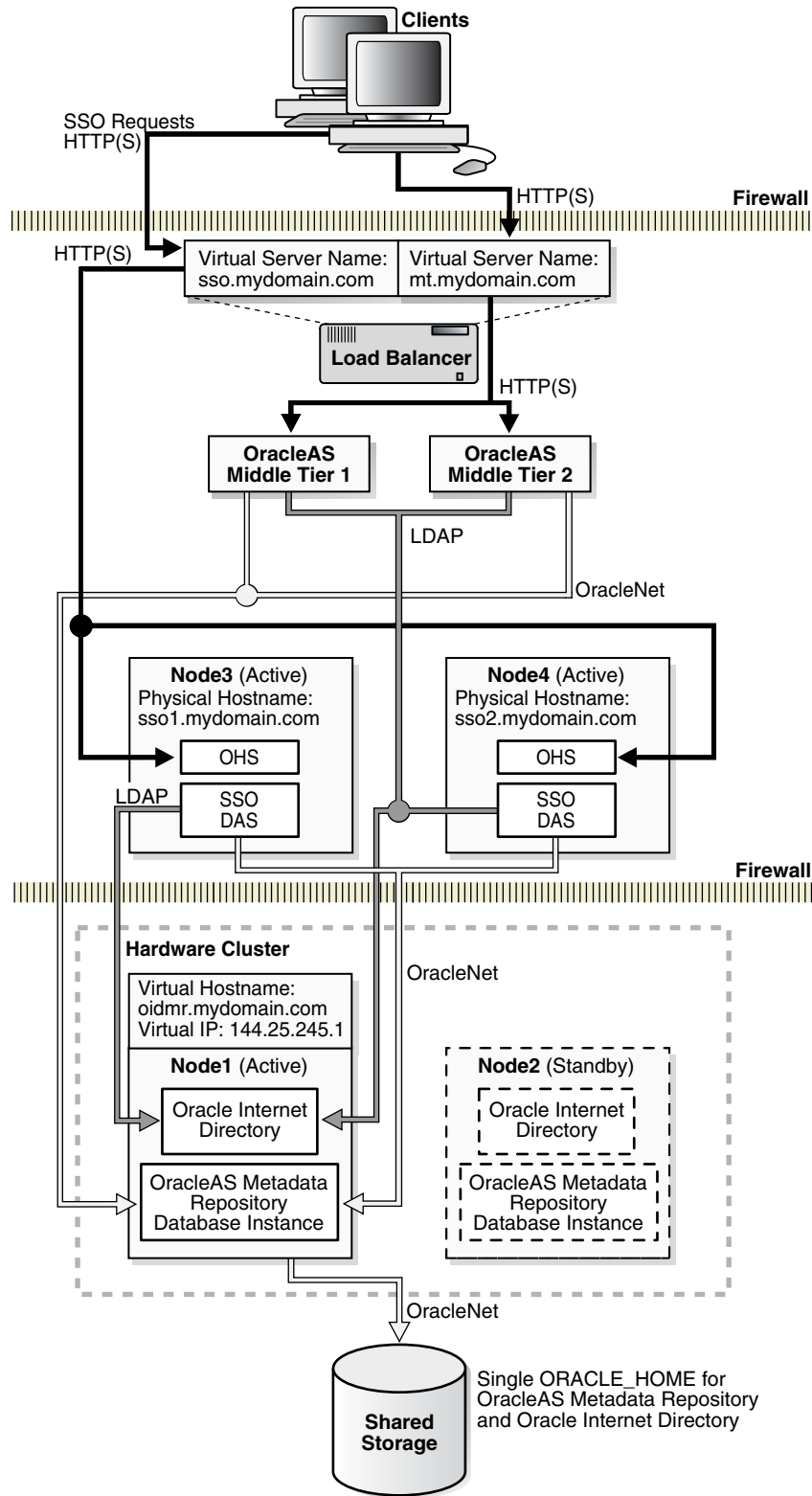
In the previous sections describing OracleAS Cold Failover Cluster (Infrastructure), Oracle Identity Management components are deployed together in an OracleAS Cold Failover Cluster with OracleAS Metadata Repository.

To provide more robust security, the OracleAS Single Sign-On and Oracle Delegated Administration Services components of Oracle Identity Management can be deployed separately from the Oracle Internet Directory and Oracle Directory Integration and Provisioning components (in a DMZ, for example). In such a set up, OracleAS Single Sign-On and Oracle Delegated Administration Services components are installed in two nodes in an active-active configuration. They are installed on their own nodes or are co-located with other Oracle Application Server middle-tier components. These

nodes are front-ended by a hardware load balancer to enable load balancing and failover between the two nodes. Clients connect to OracleAS Single Sign-On and Oracle Delegated Administration Services using a virtual server name configured in the load balancer (`sso.mydomain.com` in Figure 5-7).

Figure 5-7 provides a diagrammatic example of this configuration.

**Figure 5–7 Distributed OracleAS Cold Failover Cluster (Infrastructure) deployment**



In the example presented in Figure 5–7, the OracleAS Single Sign-On and Oracle Delegated Administration Services components are deployed between two firewalls. The figure shows the OracleAS middle-tier installed on the same nodes as OracleAS Single Sign-On and Oracle Delegated Administration Services.



Clients from outside the first firewall can access single sign-on and self-service directory administration services; the second firewall prevents clients from accessing the identity and metadata repositories directly.

The above configuration also allows straightforward active-active installation and deployment of OracleAS Single Sign-On and Oracle Delegated Administration Services. Failover and load balancing for these components are performed by a redundant load balancer. If there is no firewall separating the middle tier and OracleAS Single Sign-On and Oracle Delegated Administration Services components, the same load balancer can be configured to load balance the middle tier also.

Other Oracle Identity Management components, the Oracle Internet Directory and Oracle Directory Integration and Provisioning, are deployed in the same OracleAS Cold Failover Cluster as the OracleAS Metadata Repository. Failover from the active node to the passive node occurs at the node level. Hence, Oracle Internet Directory, Oracle Directory Integration and Provisioning, and OracleAS Metadata Repository fail over together.

During runtime, the following applies:

- Only one node of the hardware cluster is active. The virtual hostname (`oidmr.mydomain.com`) is occupied by the active node. The shared disk is mounted only on the active node.
- Oracle Internet Directory accesses its repository through the active database instance.
- OracleAS Single Sign-On accesses its schema in the database using connection pools over JDBC (and hence, Oracle Net). These connection pools are established to the active node in the OracleAS Cold Failover Cluster.
- OracleAS Single Sign-On and Oracle Delegated Administration Services use the OracleAS Cold Failover Cluster (Infrastructure)'s virtual hostname (`oidmr.mydomain.com` in the above example) to access the Oracle Internet Directory process in the active node.
- OracleAS Single Sign-On and Oracle Delegated Administration Services applications are deployed in the single `OC4J_SECURITY` instance on their respective nodes.
- OPMN performs process management tasks in the OracleAS Single Sign-On and Oracle Delegated Administration Services nodes.

---

**See Also:** *Oracle Application Server Installation Guide*. for installation details of this configuration

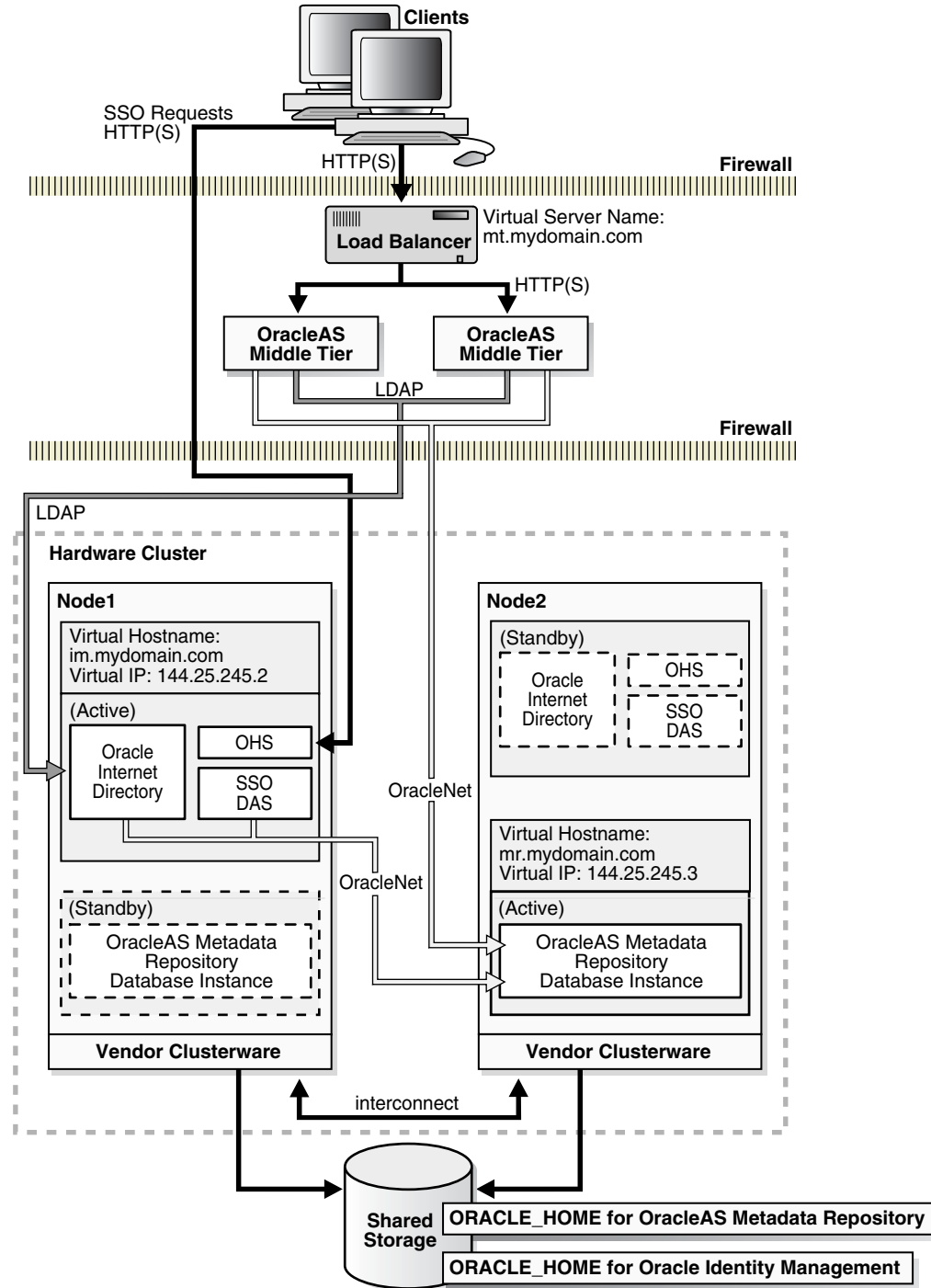
---

### 5.3.2.3 OracleAS Cold Failover Cluster (Identity Management)

For this configuration, Oracle Identity Management is installed in a OracleAS Cold Failover Cluster. Two installations are performed on the same hardware cluster, one for Oracle Identity Management and one for OracleAS Metadata Repository. This allows Oracle Identity Management components to be installed in a cold failover configuration and the OracleAS Metadata Repository in an existing database. This database can be a Real Application Clusters database that is already installed in the hardware cluster (shown in Figure 5–8). Alternatively, the database can be in a cold failover cluster configuration. The cold failover configuration provides the database with active-passive availability while the Real Application Clusters database has active-active availability. The installation of OracleAS Metadata Repository is performed using OracleAS Metadata Repository Creation Assistant.

Each installation has its unique Oracle home that is a different path from the other. Figure 5–8 shows two shared disks with an Oracle home in each. Both Oracle homes may also be installed in one shared disk as long as their paths are unique.

**Figure 5–8 OracleAS Cold Failover Cluster (Identity Management) (with OracleAS Metadata Repository installed in a cold failover database shown)**



\* Oracle Homes above have separate paths  
 \*\* Shared storage can be the same disk but must have two mount points, one for each node in the hardware cluster

A virtual IP address is used to represent the active node for Oracle Identity Management components. A virtual hostname is mapped to this address. This virtual hostname is used by the middle tier and clients to access the Oracle Identity Management components.

Middle-tier components and applications access Oracle Identity Management services by making LDAP requests to Oracle Internet Directory, HTTP/HTTPS requests to OracleAS Single Sign-On or Oracle Delegated Administration Services. Clients can perform single sign-on by making direct HTTP/HTTPS requests to OracleAS Single Sign-On server via the single sign-on URL. OracleAS Single Sign-On establishes connection pools to access the OracleAS Metadata Repository database. A connection in the pool uses Oracle Net to communicate with the active database instance(s). Oracle Net is also used by middle-tier components and Oracle Internet Directory servers to connect to the database.

Both Oracle Identity Management and OracleAS Metadata Repository are active in Node 1. In Node 2, all components are passive, on standby, unless the database that contains the OracleAS Metadata Repository is a Real Application Clusters database. In this case, the database instance is active on Node 2.

#### **5.3.2.4 Distributed OracleAS Cold Failover Cluster (Identity Management)**

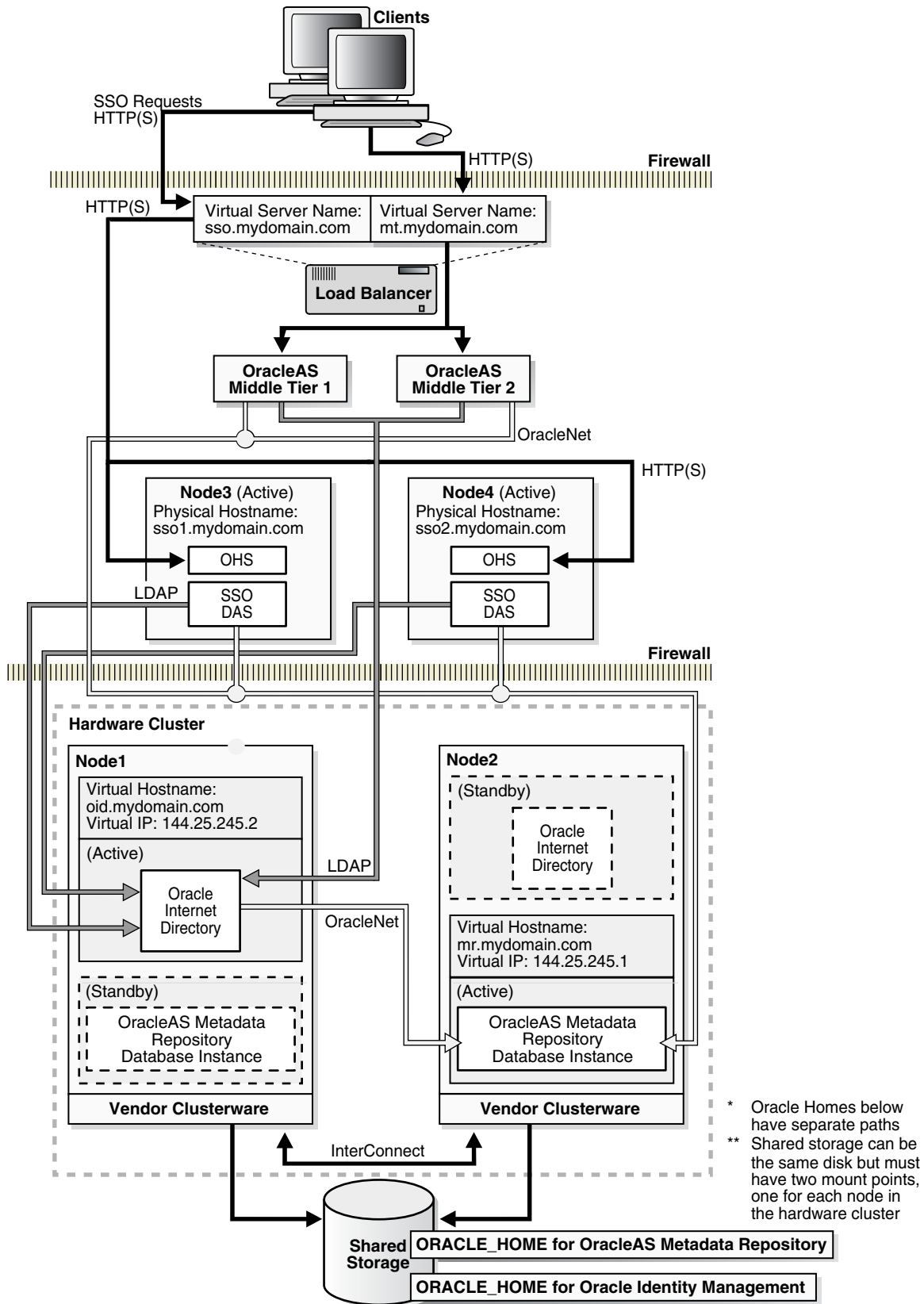
This configuration is similar to that described in Section 5.3.2.3, "OracleAS Cold Failover Cluster (Identity Management)" except for the deployment of OracleAS Single Sign-On and Oracle Delegated Administration Services components in their own nodes. Requests to these nodes are load balanced by a load balancer router, thus, enabling OracleAS Single Sign-On and Oracle Delegated Administration Services to have active-active availability.

Oracle Internet Directory is installed on its own two-node OracleAS Cold Failover Cluster with shared storage. In this configuration, Oracle Internet Directory has active-passive availability.

For the OracleAS Metadata Repository, it can be installed, using OracleAS Metadata Repository Creation Assistant, in an existing Real Application Clusters database for active-active availability. Otherwise, it can be installed in a cold failover cluster database for active-passive availability. If a cold failover database is used, the OracleAS Metadata Repository has its own virtual hostname and virtual IP that is different from those of Oracle Internet Directory.

Figure 5–9 shows the distributed OracleAS Cold Failover Cluster (Identity Management) configuration with the OracleAS Metadata Repository in a cold failover cluster database.

**Figure 5–9 Distributed OracleAS Cold Failover Cluster (Identity Management) with OracleAS Metadata Repository in an existing cold failover cluster database**



## 5.4 OracleAS Infrastructure Backup and Recovery Considerations

This section contains considerations for backup and recovery for the Infrastructure. It has the following sections:

- Section 5.4.1, "OracleAS Cold Failover Cluster"
- Section 5.4.2, "Oracle Identity Management"

---



---

**Note:** Complete procedures for backup and recovery of the OracleAS Infrastructure are documented in the *Oracle Application Server Administrator's Guide*.

---



---

### 5.4.1 OracleAS Cold Failover Cluster

When performing backup and recovery operations for a OracleAS Cold Failover Cluster, take note of the following:

- backup considerations for OracleAS Cold Failover Cluster:
  - Oracle recommends that you locate archive logs for the OracleAS Metadata Repository on the shared disk. This ensures that when failing over from one cluster node to another in the case of media recovery, the archive logs are also failed over and available.
  - You can generate archive logs to a local file system, however, make this destination accessible to both nodes so that no matter which node is active, the database instance will always output archive logs to the same location. Otherwise, the backup operation will not be able to see all archive log files.
  - Proper capacity planning is required in order to ensure adequate space is available to store the desired number of archive logs.

- Recovery considerations for OracleAS Cold Failover Cluster

There are no special considerations for recovering OracleAS Cold Failover Cluster. As mentioned in the backup considerations above, if archive logs are stored on a local file system, in the case of media recovery, all archive logs must be made available to the application server instance performing the recovery. Recovery can be performed on either node of the cluster.

Before taking a cold backup or restoring the metadata repository database, the OracleAS Backup and Recovery Tool shuts down the database first. In the Windows OracleAS Cold Failover Cluster environment, the Oracle Fail Safe Manager performs database polling and restarts the database if it is down. Hence, every time before you perform 'backup\_cold' or 'restore\_repos' with the OracleAS Backup and Recovery Tool on the primary (active) node, you must disable database polling in the Oracle Fail Safe Manager and re-enable it after the backup/restore operation.

### 5.4.2 Oracle Identity Management

In an OracleAS Cluster (Identity Management) or OracleAS Cold Failover Cluster (Identity Management) environment (or their distributed equivalents), each Oracle Identity Management installation needs to be backed up and restored individually. The backup performed on each Oracle Identity Management installation can be restored only on the respective instance in case of any failure.

If the DCM repository is located in the Infrastructure database, the OracleAS Backup and Recovery Tool requires at least one Oracle Internet Directory process to be up during backup and restore operations. So, in case of a failure on all the Oracle Identity

Management nodes, you need to first perform the restore operation on one of the Oracle Identity Management nodes and bring up the Oracle Internet Directory process on that node. Subsequently, you can then perform the restore operation on other Oracle Identity Management nodes.

If you lose an Oracle Identity Management node completely and need to restore it to a new node, please refer to the "Restoring an Identity Management Instance to a New Host" procedure in the *Oracle Application Server Administrator's Guide*.

---

---

**Note:** To determine if the DCM repository is in a database, execute the `dcmctl whichfarm` command and check for the "Repository Type: Database" or "Repository Type: Database (host)" line in the output.

---

---

---

---

# Managing and Operating Infrastructure High Availability

This chapter provides instructions to manage Oracle Application Server Infrastructure high availability configurations and covers operations such as starting, stopping, and recovering from scheduled or unplanned outages. The instructions provided apply to the configurations described in Chapter 5 and are organized as follows:

- Section 6.1, "Managing Oracle Application Server Cluster (Identity Management)"
- Section 6.2, "Managing Oracle Application Server Cold Failover Cluster (Infrastructure)"
- Section 6.3, "Managing Oracle Application Server Cold Failover Cluster (Identity Management)"

---

---

**Note:** For details on installing for high availability, refer to the *Oracle Application Server Installation Guide* for your platform.

---

---

## 6.1 Managing Oracle Application Server Cluster (Identity Management)

This section covers the following topics:

- Section 6.1.1, "Starting Stopping and Monitoring OracleAS Cluster (Identity Management)"
- Section 6.1.2, "Configuring OracleAS Cluster (Identity Management)"
- Section 6.1.3, "Failover For OracleAS Cluster (Identity Management)"
- Section 6.1.4, "Backup and Recovery For OracleAS Cluster (Identity Management)"
- Section 6.1.5, "Using Application Server Control With OracleAS Cluster (Identity Management)"
- Section 6.1.6, "Additional Considerations For OracleAS Cluster (Identity Management)"

### 6.1.1 Starting Stopping and Monitoring OracleAS Cluster (Identity Management)

This section covers the following topics:

- Starting OracleAS Cluster (Identity Management)
- Stopping OracleAS Cluster (Identity Management)
- Monitoring OracleAS Cluster (Identity Management)

### 6.1.1.1 Starting OracleAS Cluster (Identity Management)

Use the following steps on each node of the OracleAS Cluster (Identity Management) to start the OracleAS Infrastructure:

1. Set the `ORACLE_HOME` environment variable to the Infrastructure Oracle home.
2. Set the `ORACLE_SID` environment variable to the OracleAS Metadata Repository system identifier (the default is `orcl`).

---

---

**Note:** On Windows systems if you are using Oracle Fail Safe with a cluster group, you can bring up the cluster group and skip steps 3, 4, and 5.

---

---

3. Start the OracleAS Metadata Repository Net listener:

On UNIX systems:

```
$ORACLE_HOME/bin/lsnrctl start
```

On Windows systems:

```
%ORACLE_HOME%\bin\lsnrctl start
```

4. Start the OracleAS Metadata Repository:

On UNIX systems:

```
$ORACLE_HOME/bin/sqlplus /nolog
```

On Windows systems:

```
%ORACLE_HOME%\bin\sqlplus /nolog
```

At the SQL prompt, enter the following:

```
SQL> connect SYS as SYSDBA
SQL> startup
SQL> quit
```

5. Start OPMN and all OPMN-managed processes on each node:

On UNIX systems:

```
$ORACLE_HOME/opmn/bin/opmnctl startall
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl startall
```

6. Start the Application Server Control Console:

On UNIX systems:

```
$ORACLE_HOME/bin/emctl start iasconsole
```

On Windows systems:

```
%ORACLE_HOME%\bin\emctl start iasconsole
```

7. Start the middle-tier instances that use the Infrastructure.



---



---

**Note:** If you use a Distributed OracleAS Cluster (Identity Management) with different hosts for the OracleAS Infrastructure and the Oracle Internet Directory, OracleAS Single Sign-On, and Delegated Administration Services, then perform steps 1-4 on the OracleAS Infrastructure system and perform steps 1, 2, 5, and 6 in the Oracle Internet Directory, OracleAS Single Sign-On, or Delegated Administration Services Oracle homes.

---



---

**See Also:**

- Section 5.3.1.1, "OracleAS Cluster (Identity Management)" on page 5-9
- Section 5.3.1.2, "Distributed OracleAS Cluster (Identity Management)" on page 5-11
- <http://www.oracle.com/technology/docs/tech/windows/failsafe/index.html> for documentation on Oracle Fail Safe
- *Oracle Application Server Administrator's Guide* for information on starting and stopping middle-tier instances

### 6.1.1.2 Stopping OracleAS Cluster (Identity Management)

Use the following steps on each node of the OracleAS Cluster (Identity Management) to stop the OracleAS Infrastructure:

1. Stop middle-tier instances that use the Infrastructure.
2. Set the `ORACLE_HOME` environment variable to the OracleAS Infrastructure's Oracle home.
3. Set the `ORACLE_SID` environment variable to the OracleAS Metadata Repository database system identifier (default is `orcl`).
4. Stop the Application Server Control Console with the following command.

On UNIX systems:

```
$ORACLE_HOME/bin/emctl stop iasconsole
```

On Windows systems:

```
%ORACLE_HOME%\bin\emctl stop iasconsole
```

5. Stop OPMN and all OPMN-managed processes for each OracleAS instance locally.

On UNIX systems, use the following command:

```
$ORACLE_HOME/opmn/bin/opmnctl stopall
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl stopall
```

6. Stop the OracleAS Infrastructure database as follows:

On UNIX systems, use the following command:

```
$ORACLE_HOME/bin/sqlplus /nolog
```

On Windows systems, use the following command:

```
%ORACLE_HOME%\bin\sqlplus /nolog
```

At the SQL prompt, enter the following:

```
SQL> connect SYS as SYSDBA
SQL> shutdown
SQL> quit
```

**7. Stop the OracleAS Infrastructure Net listener.**

On UNIX systems, use the following command:

```
$ORACLE_HOME/bin/lsnrctl stop
```

On Windows systems, use the following command:

```
%ORACLE_HOME%\bin\lsnrctl stop
```

---



---

**Note:** If you run in an environment that uses different hosts for the OracleAS Infrastructure and the OracleAS Single Sign-On server, then perform step 1. Then, if the OracleAS Single Sign-On or Delegated Administration Services is in a separate Oracle home, perform steps 2, 4, and 5 in that Oracle home. If Oracle Internet Directory is in a separate Oracle home then perform steps 2, 4, and 5 in that Oracle home.

Finally, perform steps 2, 3, 6, and 7 in the OracleAS Infrastructure Oracle home.

---



---

**See Also:**

- Section 5.3.1.1, "OracleAS Cluster (Identity Management)" on page 5-9
- Section 5.3.1.2, "Distributed OracleAS Cluster (Identity Management)" on page 5-11
- *Oracle Application Server Administrator's Guide* for information on starting and stopping middle-tier instances
- *Oracle Internet Directory Administrator's Guide* for information on process control of Oracle Internet Directory components.

**6.1.1.3 Monitoring OracleAS Cluster (Identity Management)**

Use the steps in this section to monitor status across OracleAS Cluster (Identity Management).

1. In the OracleAS Metadata Repository Oracle home, check the status of the OracleAS Cluster (Identity Management) as follows:

Try connecting and checking the state of the OracleAS Metadata Repository:

On UNIX systems:

```
$ORACLE_HOME/bin/sqlplus /nolog
```

On Windows systems:

```
%ORACLE_HOME%\bin\sqlplus /nolog
```

At the SQL prompt, enter the following:

```
SQL> connect SYS as SYSDBA
SQL> select status from v$instance
SQL> quit
```

Try the following command to check the status of the Net listener:

On UNIX systems:

```
$(ORACLE_HOME)/bin/lsnrctl status
```

On Windows systems:

```
%ORACLE_HOME%\bin\lsnrctl status
```

2. In all of the Oracle Internet Directory, OracleAS Single Sign-On, and Delegated Administration Services Oracle homes, check the status of OPMN and OPMN-managed processes:

On UNIX systems:

```
$(ORACLE_HOME)/opmn/bin/opmnctl status
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl status
```

3. Check the status of Application Server Control.

On UNIX systems:

```
$(ORACLE_HOME)/bin/emctl status iasconsole
```

On Windows systems:

```
%ORACLE_HOME%\bin\emctl status iasconsole
```

4. On the Oracle Internet Directory, OracleAS Single Sign-On, Delegated Administration Services, or Oracle Internet Directory Oracle home, check the status of Oracle Internet Directory:

On UNIX systems:

```
$(ORACLE_HOME)/ldap/bin/ldapcheck
```

On Windows systems:

```
%ORACLE_HOME%\ldap\bin\ldapcheck
```

Verify that you can log in to Oracle Internet Directory:

On UNIX systems:

```
$(ORACLE_HOME)/bin/oidadmin
```

On Windows systems:

```
%ORACLE_HOME%\bin\oidadmin
```

Use the following login and password:

Login: orcladmin

Passwd: <orcladmin\_password>

---

---

**Note:** After installation, the *orcladmin\_password* is the same as the *ias\_admin* password.

---

---

5. In the Oracle Internet Directory or OracleAS Single Sign-On Oracle home, verify you can log in to OracleAS Single Sign-On in the two OracleAS Single Sign-On Oracle homes:

`http://<host>:<HTTP_port>/pls/orasso`

Login: orcladmin

Passwd: <orcladmin\_password>

---

---

**Note:** After installation the *orcladmin\_password* is the same as the *ias\_admin* password.

---

---

6. In the Oracle Internet Directory or Delegated Administration Services Oracle home, verify you can log in to Delegated Administration Services in the two Delegated Administration Services Oracle homes:

`http://<host>:<HTTP_port>/oiddas`

Login: oracleadmin

Passwd: <orcladmin\_password>

---

---

**Note:** After installation the *orcladmin\_password* is the same as the *ias\_admin* password.

---

---

## 6.1.2 Configuring OracleAS Cluster (Identity Management)

This section covers the following topics:

- Changing Configuration Files For OracleAS Cluster (Identity Management)
- Configuring A Load Balancer For OracleAS Cluster (Identity Management)

### 6.1.2.1 Changing Configuration Files For OracleAS Cluster (Identity Management)

The OracleAS Cluster (Identity Management) instances need to contain common configuration files. If you need to change the configuration for one instance, you also need to update the configuration in other instances in the OracleAS Cluster (Identity Management).

To assure that configuration files stay the same across the Oracle Application Server Cluster (Identity Management), do the following:

- Use the following command to save configuration changes related to OPMN, Oracle HTTP Server, or OC4J\_SECURITY on one Oracle Application Server Cluster (Identity Management) node:

On UNIX systems:

```
$ORACLE_HOME/dcm/bin/dcmctl updateConfig
```

On Windows systems:

```
%ORACLE_HOME%\dcm\bin\dcmctl updateConfig
```

The `dcmctl updateConfig` command propagates configuration file changes across the Oracle Application Server Cluster (Identity Management) nodes.

- Oracle Internet Directory is not automatically managed for configuration changes across the Oracle Application Server Cluster (Identity Management). Changes that you need to make to configuration files, primarily the wallet files, should be made to all nodes in the Oracle Application Server Cluster (Identity Management).

**See Also:** "Using Application Server Control With OracleAS Cluster (Identity Management)" on page 6-10

### 6.1.2.2 Configuring A Load Balancer For OracleAS Cluster (Identity Management)

A load balancer should be configured to detect service down on a node and automatically stop traffic to that node. Also, the load balancer is recommended to be in a fault tolerant mode. This section provides instructions for configuring a load balancer for OracleAS Cluster (Identity Management).

To configure a load balancer for OracleAS Cluster (Identity Management), perform the following steps:

1. Verify that the load balancer virtual server name you select does not contain the physical hostnames of the nodes in the OracleAS Cluster (Identity Management).

When the installer copies files to different nodes in the OracleAS Cluster (Identity Management), it replaces the current hostname in the files with the hostname of the target node. Ensure that the load balancer's virtual server name does not contain the hostnames of the nodes in the cluster, or the installer might change the virtual server name of the load balancer as well.

For example, if you are installing on nodes named `rac-1` and `rac-2`, be sure that the load balancer virtual server name does not contain "rac-1" or "rac-2". When the installer is installing files to `rac-2`, it searches for the string "rac-1" in the files and replaces it with "rac-2". If the load balancer's virtual server name happens to be `LB-rac-1x`, the installer sees the string "rac-1" in the name and replaces it with "rac-2", thus mangling the virtual server name to `LB-rac-2x`.

2. The load balancer should be notified of a failure. It should stop directing non Oracle Net traffic to the node with the failed instance (if the load balancer has ability to automatically detect this case, it is highly recommend to use this ability).
3. To configure the load balancer for automatic monitoring of the Oracle Internet Directory and OracleAS Single Sign-On, Oracle Delegated Administration Services, set up monitors for the following:

LDAP port

LDAP SSL port

HTTP or HTTPS listen port (depending on the deployment type)

It is recommended that these monitors use the respective protocols to monitor the services. That is LDAP for the LDAP port, LDAP over SSL for the LDAP SSL port, and HTTP/HTTPS for the web server port. If the load balancer does not offer one

or all of these monitors, consult the load balancer documentation for details on the best method to set up the load balancer.

#### 6.1.2.2.1 Configuring A Load Balancer For Distributed OracleAS Cluster (Identity Management)

To configure the load balancer for distributed OracleAS Cluster (Identity Management):

1. Configure virtual server names and ports for the load balancer.

Configure the load balancer with two virtual server names and associated ports:

- Configure a virtual server name for LDAP connections. For this virtual server, configure two ports: one for SSL and one for non-SSL connections.
- Configure a virtual server name for HTTP connections. For this virtual server, configure two ports: one for SSL and one for non-SSL connections.

2. Verify the following:

- Check that the virtual server names are associated with IP addresses and they are part of your DNS. The nodes that will be running Oracle Application Server must be able to access these virtual server names.
- Check that the load balancer is configured to load-balance traffic across the appropriate nodes.

3. Set up cookie persistence on the load balancer.

On your load balancer, set up cookie persistence for HTTP traffic. Specifically, set up cookie persistence for URIs starting with `/oiddas/`. This is the URI for Delegated Administration Services. If your load balancer does not allow you to set cookie persistence at the URI level, then set the cookie persistence for all HTTP traffic. In either case, set the cookie to expire when the browser session expires. Refer to your load balancer documentation for details.

Keep the following in mind when setting the persistence or stickiness for a load balancer for Distributed OracleAS Cluster (Identity Management):

- For Oracle Internet Directory, do not set a persistence setting for the load balancer.
- For OracleAS Single Sign-On, a persistence setting is not required. However, setting a Persistence or stickiness compatible with Oracle HTTP Server is ok.
- For Delegated Administration Services, some kind of load balancer persistence is required. Cookie based persistence is highly recommended.

### 6.1.3 Failover For OracleAS Cluster (Identity Management)

Using OracleAS Cluster (Identity Management), if a node within the cluster fails, the cluster includes other nodes which can take over for the failed node. In an OracleAS Cluster (Identity Management), OPMN manages Oracle Application Server processes and when possible, OPMN restarts crashed processes. If Oracle Internet Directory is stopped on one node, the load balancer directs Oracle Internet Directory traffic to another Oracle Internet Directory on another node in the OracleAS Cluster (Identity Management).

---



---

**Note:** Only one `odisrv` and one `oidrepld` can be active at a time in an OracleAS Cluster (Identity Management), while multiple `oidldapd` processes can be running in the same cluster.

---



---

On Windows systems, if one of the cluster nodes goes down, Oracle Failsafe detects the failure and initiates a failover of the managed Oracle services immediately.

If OC4J\_SECURITY is down on a node, the active Oracle HTTP Servers direct traffic to the surviving OC4J\_SECURITY instance (this is by virtue of the fact that they are clustered). If Oracle HTTP Server is down on a node then the surviving Oracle HTTP Server on the other node services the request. When the Oracle HTTP Server services the request, Oracle Internet Directory Monitor polls the Oracle Database Server to verify that all other Oracle Internet Directory nodes are running. If, after five minutes an Oracle Internet Directory Monitor on one of the nodes has not reported, then the other Oracle Internet Directory nodes regard it as having failed. At this point, the following occurs on one of the other nodes that are still running:

1. The Oracle Internet Directory Monitor on that node brings up the processes that were running on the failed node.
2. The Oracle Internet Directory on that node continues processing the operations that were previously underway on the failed node.
3. The Oracle Internet Directory Monitor on that node logs that it has brought up the processes that were previously running on the failed node.

---

**Note 1:** When a node goes down or the processes on a node are brought down due to planned maintenance operations, the load balancer should be reconfigured to not send traffic to this node

---



---

**Note 2:** If the primary node running either the directory replication server (`oidrep1d`), or the Oracle Directory Integration and Provisioning server (`odisrv`), or both fails, then the Oracle Internet Directory Monitor on the secondary node starts these processes on the secondary node after five minutes.

Normal shutdown is not treated as a failover - that is, after a normal shutdown, the Oracle Internet Directory Monitor on the secondary node does not start these processes on the secondary node after five minutes.

---

**See Also:** Section 5.3.2.1, "OracleAS Cold Failover Cluster (Infrastructure)"

### 6.1.4 Backup and Recovery For OracleAS Cluster (Identity Management)

For backing up OracleAS Cluster (Identity Management) environments and recovering these backups during failures, use the general backup and recovery procedures provided in the *Oracle Application Server Administrator's Guide*.

The backup performed on each OracleAS Cluster (Identity Management) instance can be restored only on the respective instance, in case of any failure. If the DCM repository is located in the infrastructure database, the backup and recovery tool requires at least one Oracle Internet Directory process to be up during backup and restore operations. Thus, in case of a failure on all the OracleAS Cluster (Identity Management) nodes, you need to first perform a restore operation on one of the OracleAS Identity Management nodes and bring up the Oracle Internet Directory process on that node; subsequently you can perform the restore operation on other OracleAS Identity Management nodes. If you lose an OracleAS Identity Management

node completely and need to restore it to a new node, refer to the "Restoring an Identity Management Instance to a New Host" procedure.

To determine if the DCM repository is a database, use either the Oracle Enterprise Manager 10g Application Server Control Console, or use the following `dcmctl` command.

On UNIX systems:

```
$ORACLE_HOME/dcm/bin/dcmctl whichfarm
```

On Windows systems:

```
%ORACLE_HOME%\dcm\bin\dcmctl whichfarm
```

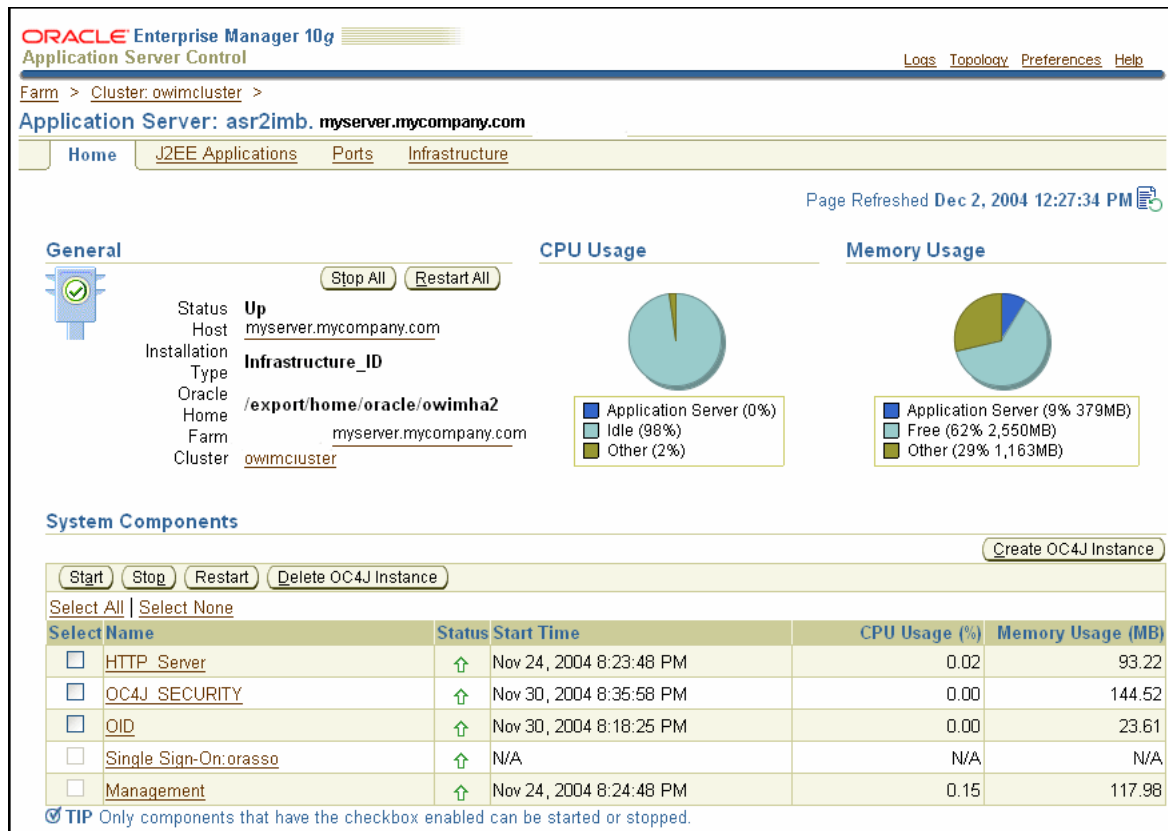
The command output shows either "Repository Type: Database" or "Repository Type: Database (host)" for a database repository.

**See Also:** *Oracle Application Server Administrator's Guide*

### 6.1.5 Using Application Server Control With OracleAS Cluster (Identity Management)

You can use Application Server Control Console to manage a OracleAS Cluster (Identity Management). Figure 6–1 shows a sample Application Server Control Console page.

**Figure 6–1 Application Server Control Console With OracleAS Cluster (Identity Management)**





## 6.1.6 Additional Considerations For OracleAS Cluster (Identity Management)

The following list includes important guidelines for managing an Oracle AS Cluster (Identity Management) environment:

- The port number used by the directory servers must be the same on all the nodes. Use of `staticports` feature to enforce this is strongly recommended. Even the LDAP ports configured in the LDAP virtual server on the load balancer should be the same as the LDAP ports configured on all the physical Oracle Internet Directory nodes.
- The time value on all nodes should be synchronized using Greenwich Mean Time so that there is a discrepancy of no more than 250 seconds between them.
- If you change the password to the Oracle Internet Directory-designated database, then you must update each of the other nodes in the Oracle AS Cluster (Identity Management) environment.
- In a Windows environment, make sure that Microsoft Cluster service is running.

**See Also:** *Oracle Internet Directory Administrator's Guide*

## 6.2 Managing Oracle Application Server Cold Failover Cluster (Infrastructure)

This section covers the following topics:

- Section 6.2.1, "Starting Stopping and Monitoring OracleAS Cold Failover Cluster (Infrastructure)"
- Section 6.2.2, "Configuring OracleAS Cold Failover Cluster (Infrastructure)"
- Section 6.2.3, "Failover For OracleAS Cold Failover Cluster (Infrastructure)"
- Section 6.2.4, "Backing Up and Recovering OracleAS Cold Failover Cluster (Infrastructure)"
- Section 6.2.5, "Using Application Server Control With OracleAS Cold Failover Cluster (Infrastructure)"

### 6.2.1 Starting Stopping and Monitoring OracleAS Cold Failover Cluster (Infrastructure)

This section covers the following topics:

- Starting OracleAS Cold Failover Cluster (Infrastructure)
- Stopping OracleAS Cold Failover Cluster (Infrastructure)
- Monitoring OracleAS Cold Failover Cluster (Infrastructure)

#### 6.2.1.1 Starting OracleAS Cold Failover Cluster (Infrastructure)

Use the following steps to start the Infrastructure in an OracleAS Cold Failover Cluster:

1. Set the `ORACLE_HOME` environment variable to the Infrastructure's Oracle home.
2. Set the `ORACLE_SID` environment variable to the metadata repository's system identifier.
3. Set the `PATH` environment variable to include the Infrastructure's `$ORACLE_HOME/bin` directory.

On Windows, use the following command to set the PATH:

```
set PATH=%ORACLE_HOME%\bin;%PATH%
```

**Important:** Specify the path of the working Oracle home as the first entry in the PATH environment variable if there are several Oracle homes installed on the machine. Also, ensure that the full paths of the executables you use are specified.

4. Login as root. Enable volume management software and mount the file system (if necessary).
5. Enable the virtual IP address on the current node.
6. Start the OracleAS Infrastructure database listener.

On UNIX systems:

```
$(ORACLE_HOME)/bin/lsnrctl start
```

On Windows systems:

```
%ORACLE_HOME%\bin\lsnrctl start
```

7. Start the OracleAS Infrastructure database as follows:

On UNIX systems:

```
$(ORACLE_HOME)/bin/sqlplus /nolog
```

On Windows systems:

```
%ORACLE_HOME%\bin\sqlplus /nolog
```

At the SQL prompt, enter the following:

```
SQL> connect SYS as SYSDBA  
SQL> startup
```

8. Start OPMN and all OPMN-managed processes for each OracleAS instance locally.

On UNIX systems:

```
$(ORACLE_HOME)/opmn/bin/opmnctl startall
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl startall
```

9. Start the Application Server Control Console:

On UNIX systems, use one of the following commands:

```
$(ORACLE_HOME)/bin/emctl start iasconsole
```

On Windows systems, use one of the following commands:

```
%ORACLE_HOME%\bin\emctl start iasconsole
```

### 6.2.1.2 Stopping OracleAS Cold Failover Cluster (Infrastructure)

Use the following steps to stop the OracleAS Infrastructure in an OracleAS Cold Failover Cluster:

1. Set the ORACLE\_HOME environment variable to the Infrastructure's Oracle home.
2. Set the ORACLE\_SID environment variable to the metadata repository's system identifier.
3. Stop the Application Server Control Console.
 

On UNIX systems:

```
$ORACLE_HOME/bin/emctl stop iasconsole
```

On Windows systems:

```
%ORACLE_HOME%\bin\emctl stop iasconsole
```
4. Stop OPMN and all OPMN-managed processes for each OracleAS instance locally.
 

To shutdown the OPMN daemon and all OPMN-managed processes:

On UNIX systems:

```
$ORACLE_HOME/opmn/bin/opmnctl stopall
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl stopall
```
5. Stop the OracleAS Infrastructure database as follows:
 

On UNIX systems, use the following command:

```
$ORACLE_HOME/bin/sqlplus /nolog
```

On Windows systems, use the following command:

```
%ORACLE_HOME%\bin\sqlplus /nolog
```

At the SQL prompt, enter the following:

```
SQL> connect SYS as SYSDBA
SQL> shutdown
```
6. Stop the OracleAS Infrastructure database listener.
 

On UNIX systems:

```
$ORACLE_HOME/bin/lsnrctl stop
```

On Windows systems:

```
%ORACLE_HOME%\bin\lsnrctl stop
```
7. Login as root. Disable volume management software and unmount the file system (if necessary).
8. Disable the virtual IP address from the current node.

Last two steps, step 7 and step 8, are only required if you are stopping on the current node so that you failover to the other node. Other wise it is not a mandatory part of the stop process.

### 6.2.1.3 Monitoring OracleAS Cold Failover Cluster (Infrastructure)

Use the steps in this section to monitor status across OracleAS Cold Failover Cluster (Infrastructure).

1. Check the status of the OracleAS Cold Failover Cluster (Infrastructure):

Try connecting and checking the state of the OracleAS Metadata Repository:

```
sqlplus /nolog
SQL> connect SYS as SYSDBA
SQL> select status from v$instance
```

Try the following command to check the status of the Net listener:

On UNIX systems:

```
$ORACLE_HOME/bin/lsnrctl status
```

On Windows systems:

```
%ORACLE_HOME%\bin\lsnrctl status
```

2. Check the status of OPMN and OPMN-managed processes:

On UNIX systems:

```
$ORACLE_HOME/opmn/bin/opmnctl status
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl status
```

3. Check the status of Application Server Control.

On UNIX systems:

```
$ORACLE_HOME/bin/emctl status iasconsole
```

On Windows systems:

```
%ORACLE_HOME%\bin\emctl status iasconsole
```

4. Check the status of Oracle Internet Directory:

On UNIX systems:

```
$ORACLE_HOME/ldap/bin/ldapcheck
```

On Windows systems:

```
%ORACLE_HOME%\ldap\bin\ldapcheck
```

Verify that you can log in to Oracle Internet Directory:

On UNIX systems:

```
$ORACLE_HOME/bin/oidadmin
```

On Windows systems:

```
ORACLE_HOME\bin\oidadmin
```

Use the following login and password:

Login: orcladmin

Passwd: <orcladmin\_password>

---



---

**Note:** After installation, the `orcladmin_password` is the same as the `ias_admin` password.

---



---

5. Verify you can log in to OracleAS Single Sign-On:

`http://<host>:<HTTP_port>/pls/orasso`

Login: orcladmin

Passwd: <orcladmin\_password>

6. Verify you can log in to Delegated Administration Services:

`http://<host>:<HTTP_port>/oiddas`

Login: oracleadmin

Passwd: <orcladmin\_password>

## 6.2.2 Configuring OracleAS Cold Failover Cluster (Infrastructure)

This section covers the following topics:

- Changing Configuration For OracleAS Cold Failover Cluster (Infrastructure)
- Configuring Virtual IPs For OracleAS Cold Failover Cluster (Infrastructure)

### 6.2.2.1 Changing Configuration For OracleAS Cold Failover Cluster (Infrastructure)

OracleAS Cold Failover Cluster (Infrastructure) is a two-node, active-passive configuration on a hardware cluster. The two nodes are connected to shared storage. The OracleAS Metadata Repository and the OracleAS Identity Management components are installed together into the same Oracle home, which resides in the shared storage. Thus, if you need to change the configuration for one instance you can use the standard OracleAS Infrastructure administration techniques.

**See Also:**

- "Using Application Server Control With OracleAS Cold Failover Cluster (Infrastructure)" on page 6-21
- *Oracle Application Server Administrator's Guide*

### 6.2.2.2 Configuring Virtual IPs For OracleAS Cold Failover Cluster (Infrastructure)

The Oracle Application Server Installation Guides for your platform cover the instructions for configuring the virtual IPs for a OracleAS Cold Failover Cluster (Infrastructure).

**See Also:**

- Section, 11.2.2 Map the Virtual Hostname and Virtual IP Address in the *Oracle Application Server Installation Guide 10g Release 2 (10.1.2) for Solaris Operating System (SPARC)*
- Section 11.3.2 Get a Virtual Address for the Cluster in the *Oracle Application Server Installation Guide 10g Release 2 (10.1.2) for Microsoft Windows*

## 6.2.3 Failover For OracleAS Cold Failover Cluster (Infrastructure)

This section covers the following topics:

- Failover For OracleAS Cold Failover Cluster (Infrastructure) For Solaris Systems
- Failover For OracleAS Cold Failover Cluster (Infrastructure) For Windows Systems
- Failover For OracleAS Cold Failover Cluster (Infrastructure) For Linux Systems

### 6.2.3.1 Failover For OracleAS Cold Failover Cluster (Infrastructure) For Solaris Systems

The following shows the steps to failover from the active node to the standby node, for Solaris systems with a Veritas Volume Manager.

On the failed node:

1. Stop and if necessary kill all processes belonging to the OracleAS Cold Failover Cluster (Infrastructure) instance on this node.
2. Login as root.
3. Stop the Oracle Cluster Synchronization Services (CSS) daemon, `ocssd`, if it is running. Use the following command:
 

```
> /etc/init.d/init.cssd stop
```
4. Before executing this step, ensure that the file system is not busy. If it is busy, check which processes are using the file system and stop them if required. Unmount the file system using the following command:
 

```
> umount <mount_point>
```
5. Login as root and deport the disk group. For example, if you are using Sun Cluster with Veritas Volume Manager, deport the disk group using the following commands:
 

```
> su - root
> vxdg deport <disk_group_name>
```
6. If the failed node is usable, execute this command to release the virtual IP:
 

```
> ifconfig <interface_name> removeif <virtual_IP>
```

On the new active node:

1. Login as root.
2. Execute the following command to assign the virtual IP to this node:
 

```
> ifconfig <interface_name> addif <virtual_IP> up
```
3. Import the disk group. For example, if you are using Sun Cluster with Veritas Volume Manager, use the following commands:
 

```
> vxdg import <disk_group_name>
> vxvol -g <disk_group_name> startall
```
4. Mount the file system using the following command:
 

```
> mount /dev/vx/dsk/<disk_group_name>/<volume_name> <mount_point>
```

5. If the Oracle Cluster Synchronization Services (CSS) daemon, `ocssd`, is required, run the following command as the user which installed the Oracle home:

```
> /etc/init.d/init.cssd start
```

6. Start all OracleAS Infrastructure processes on this new active node with the following command:

```
> $ORACLE_HOME/opmn/bin/opmnctl startall
```

### 6.2.3.2 Failover For OracleAS Cold Failover Cluster (Infrastructure) For Windows Systems

The figures, Figure 6–2, Figure 6–3, and Figure 6–4 show the Oracle Fail Safe Manager screens for a sample failover from the active node to the standby node on Windows.

**Figure 6–2** Screen 1 Performing Failover With OracleAS Cold Failover Cluster (Infrastructure)

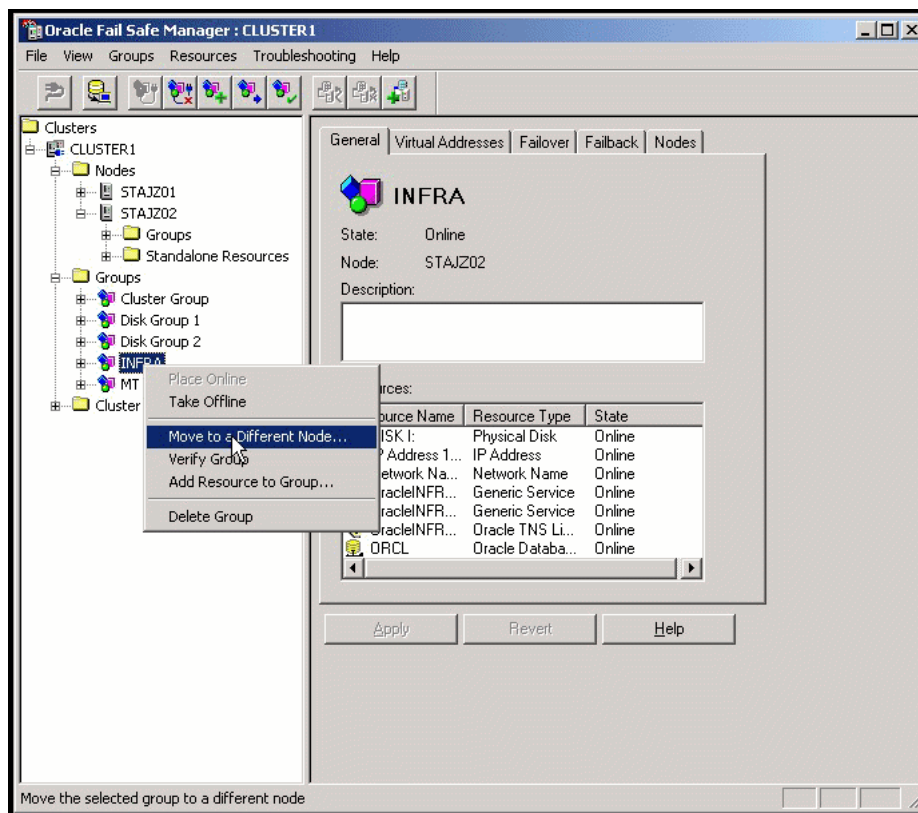
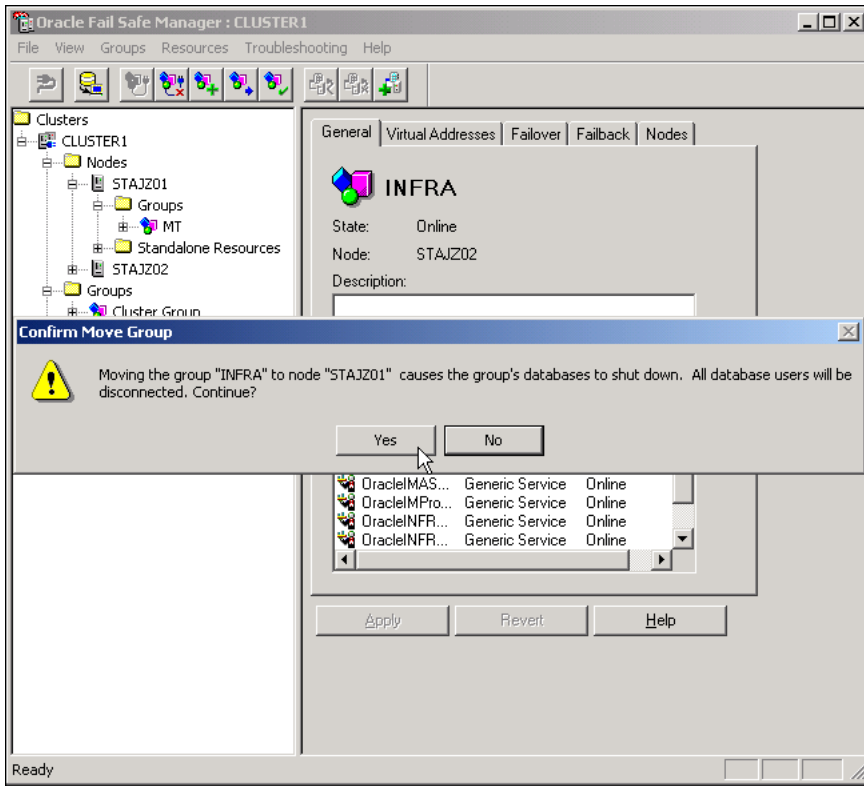
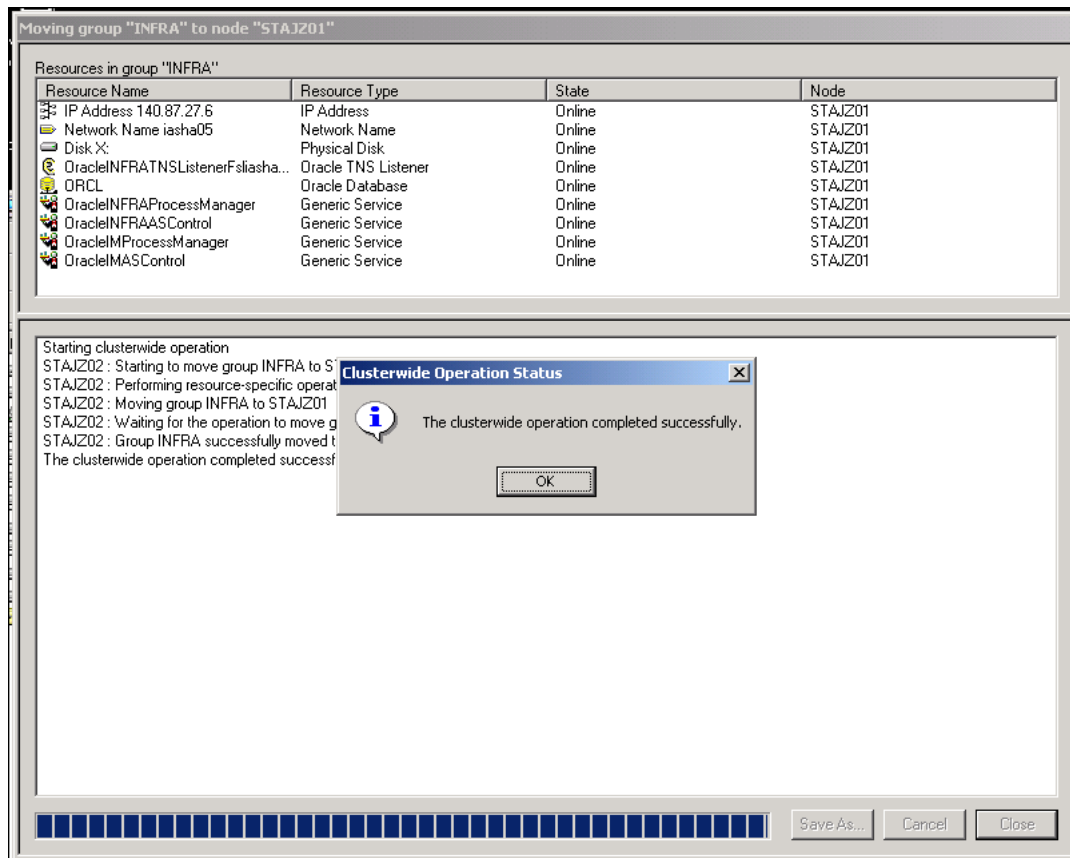


Figure 6-3 Screen 2 Performing Failover With OracleAS Cold Failover Cluster (Infrastructure)





**Figure 6–4 Screen 3 Performing Failover With OracleAS Cold Failover Cluster (Infrastructure)**

### 6.2.3.3 Failover For OracleAS Cold Failover Cluster (Infrastructure) For Linux Systems

The following shows the steps to failover from the active node to the standby node, on Linux Systems. Starting on the failed node:

1. Make sure all processes belonging to the OracleAS Cold Failover Cluster (Infrastructure) instance on the failed node are down.
2. Login as root.
3. Use the following command to stop the Oracle Cluster Synchronization Services (CSS) daemon, `ocssd`, if it is running:

```
> /etc/init.d/init.cssd stop
```

4. Unmount the file system using the following command:

```
> umount <mount_point>
```

If the file system is busy, check which processes are using the file system with the following command:

```
> fuser -muv <Shared Storage Partition>
```

Stop the processes, if required, using the following command:

```
> fuser -k <Shared Storage Partition>
```

5. If the failed node is usable, execute the following command to release the virtual IP:

```
> ifconfig <interface_name> down
```

For example,

```
> ifconfig eth1:1 down
```

On the new active node:

1. Login as root.
2. Execute the following command to assign the virtual IP to this node (the new active node):

```
> ifconfig <interface_name> netmask <subnet mask> up
```

For example,

```
> ifconfig 144.88.27.125 netmask 255.255.252.0 up
```

3. Verify that the virtual IP is up and working using `telnet` from a different host (subnet/domain).
4. Mount the file system using the following command:

```
> mount <Shared Storage Partition> <mount_point>
```

For example:

```
> mount /dev/sdc1 /oracle
```

5. If the Oracle Cluster Synchronization Services (CSS) daemon, `ocssd`, is required, run the following command as the user that installed the Oracle home:

```
> /etc/init.d/init.cssd start
```

6. Start all OracleAS Infrastructure processes on this new active node with the following commands:
  - a. Set the `ORACLE_HOME` environment variable to the Infrastructure's Oracle home.
  - b. Set the `ORACLE_SID` environment variable to the metadata repository's system identifier.
  - c. Start the OracleAS Infrastructure database as follows:

```
$ORACLE_HOME/bin/sqlplus /nolog
```

At the SQL prompt, enter the following:

```
SQL> connect SYS as SYSDBA
SQL> startup
```

- d. Start the OracleAS Infrastructure database listener.

```
$ORACLE_HOME/bin/lsnrctl start
```

- e. Start OPMN and all OPMN-managed processes using the following command:

```
$ORACLE_HOME/opmn/bin/opmnctl startall
```

- f. Start the Application Server Control Console:

```
§ORACLE_HOME/bin/emctl start iasconsole
```

## 6.2.4 Backing Up and Recovering OracleAS Cold Failover Cluster (Infrastructure)

For backing up OracleAS Cold Failover Cluster environments and recovering these backups during failures, use the general backup and recovery procedures provided in the *Oracle Application Server Administrator's Guide*.

Additionally, the following considerations should be noted:

Backup considerations:

- Oracle recommends that you locate archive logs for the OracleAS Metadata Repository on the shared disk. This ensures that, when failing over from one cluster node to another in the case of media recovery, the archive logs are also failed over and available.
- You can generate archive logs to a local file system; however, the same path must be available during runtime on whichever node is hosting the OracleAS Infrastructure instance.
- Proper capacity planning is required in order to ensure adequate space is available to store the desired number of archive logs.

Recovery considerations:

- If archive logs are stored on a local file system, in the case of media recovery, all archive logs must be made available to the application server instance performing the recovery. Recovery can be performed on either node of the cluster.

**See Also:** *Oracle Application Server Administrator's Guide*

## 6.2.5 Using Application Server Control With OracleAS Cold Failover Cluster (Infrastructure)

You can use Application Server Control Console to manage a OracleAS Cold Failover Cluster (Infrastructure). Figure 6–5 shows a sample OracleAS Cold Failover Cluster (Infrastructure) screen.

Figure 6–5 Application Server Control Console With OracleAS Cold Failover Cluster (Infrastructure)

The screenshot displays the Oracle Enterprise Manager 10g Application Server Control interface. At the top, it shows the Oracle logo and 'Enterprise Manager 10g Application Server Control'. The main content area is divided into several sections:

- General:** Shows the server status as 'Up' with a green checkmark icon. It includes buttons for 'Stop All' and 'Restart All'. Details include Host: myserver.mycompany.com, Installation Type: Infrastructure, Oracle Home: /ias1012c1/conf1infra, and Farm: asdb1.myserver.mycompany.com.
- CPU Usage:** A pie chart showing 0% for Application Server, 95% for Idle, and 5% for Other.
- Memory Usage:** A pie chart showing 18% (372MB) for Application Server, 32% (661MB) for Free, and 50% (1,016MB) for Other.
- System Components:** A table listing various components with checkboxes for selection, status indicators, start times, CPU usage percentages, and memory usage in MB. Buttons for 'Start', 'Stop', 'Restart', and 'Delete OC4J Instance' are present above the table. A 'TIP' note at the bottom states: 'This table contains only the enabled components of the application server. Only components that have the checkbox enabled can be started or stopped.'

Select	Name	Status	Start Time	CPU Usage (%)	Memory Usage (MB)
<input type="checkbox"/>	HTTP_Server	↑	Dec 1, 2004 4:25:22 PM	0.10	50.47
<input type="checkbox"/>	OC4J_SECURITY	↑	Dec 1, 2004 4:25:43 PM	0.04	108.48
<input type="checkbox"/>	oca	↑	Dec 1, 2004 4:25:43 PM	0.06	111.19
<input type="checkbox"/>	OID	↑	Dec 1, 2004 4:25:27 PM	0.00	18.07
<input type="checkbox"/>	Single Sign-On:orasso	↑	N/A	N/A	N/A
<input type="checkbox"/>	Management	↑	Dec 1, 2004 4:37:36 PM	0.20	83.82

## 6.3 Managing Oracle Application Server Cold Failover Cluster (Identity Management)

This section covers the following topics:

- Section 6.3.1, "Starting Stopping and Monitoring OracleAS Cold Failover Cluster (Identity Management)"
- Section 6.3.2, "Configuring OracleAS Cold Failover Cluster (Identity Management)"
- Section 6.3.3, "Failover For OracleAS Cold Failover Cluster (Identity Management)"
- Section 6.3.4, "Backup and Recovery For OracleAS Cold Failover Cluster (Identity Management)"

## 6.3.1 Starting Stopping and Monitoring OracleAS Cold Failover Cluster (Identity Management)

This section covers the following topics:

- Starting OracleAS Cold Failover Cluster (Identity Management)
- Stopping OracleAS Cold Failover Cluster (Identity Management)
- Monitoring OracleAS Cold Failover Cluster (Identity Management)

### 6.3.1.1 Starting OracleAS Cold Failover Cluster (Identity Management)

This section describes how to start all processes in an OracleAS Cold Failover Cluster (Identity Management). Follow this procedure after you have restarted your host, or any other time you would like to start up your OracleAS Cold Failover Cluster (Identity Management).

1. Set the `ORACLE_HOME` environment variable to the Infrastructure Oracle home.
2. Set the `ORACLE_SID` environment variable to the OracleAS Metadata Repository system identifier (the default is `orcl`).
3. Start the OracleAS Metadata Repository Net listener:

On UNIX systems:

```
$ORACLE_HOME/bin/lsnrctl start
```

On Windows systems:

```
%ORACLE_HOME%\bin\lsnrctl start
```

4. Start the OracleAS Metadata Repository:

On UNIX systems:

```
$ORACLE_HOME/bin/sqlplus /nolog
```

On Windows systems:

```
%ORACLE_HOME%\bin\sqlplus /nolog
```

At the SQL prompt, enter the following:

```
SQL> connect SYS as SYSDBA
SQL> startup
SQL> quit
```

5. Start OPMN and all OPMN-managed processes:

On UNIX systems:

```
$ORACLE_HOME/opmn/bin/opmnctl startall
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl startall
```

6. Start the Application Server Control Console:

On UNIX systems:

```
$ORACLE_HOME/bin/emctl start iasconsole
```

On Windows systems:

```
%ORACLE_HOME%\bin\emctl start iasconsole
```

7. Start the middle-tier instances that use the Infrastructure.

**See Also:**

- *Oracle Application Server Administrator's Guide* for information on starting and stopping middle-tier instances.
- *Oracle Internet Directory Administrator's Guide* for information on process control of Oracle Internet Directory components.

### 6.3.1.2 Stopping OracleAS Cold Failover Cluster (Identity Management)

This section describes how to stop all processes in a OracleAS Cold Failover Cluster (Identity Management). Follow this procedure when you are preparing to shut down your host, or any other time you would like to stop your entire OracleAS Cold Failover Cluster (Identity Management).

1. Stop middle-tier instances that use the Infrastructure.
2. Set the ORACLE\_HOME environment variable to the OracleAS Infrastructure's Oracle home.
3. Set the ORACLE\_SID environment variable to the OracleAS Metadata Repository database system identifier (default is orcl).
4. Stop the Application Server Control Console with the following command.

On UNIX systems:

```
$ORACLE_HOME/bin/emctl stop iasconsole
```

On Windows systems:

```
%ORACLE_HOME%\bin\emctl stop iasconsole
```

5. Stop OPMN and all OPMN-managed processes.

To shutdown the OPMN daemon and all OPMN-managed processes, on UNIX systems:

```
$ORACLE_HOME/opmn/bin/opmnctl stopall
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl stopall
```

6. Stop the OracleAS Infrastructure database as follows:

On UNIX systems, use the following command:

```
$ORACLE_HOME/bin/sqlplus /nolog
```

On Windows systems, use the following command:

```
%ORACLE_HOME%\bin\sqlplus /nolog
```

At the SQL prompt, enter the following:

```
SQL> connect SYS as SYSDBA
SQL> shutdown
SQL> quit
```

7. Stop the OracleAS Infrastructure Net listener.

On UNIX systems, use the following command:

```
$ORACLE_HOME/bin/lsnrctl stop
```

On Windows systems, use the following command:

```
%ORACLE_HOME%\bin\lsnrctl stop
```

**See Also:** *Oracle Application Server Administrator's Guide* for information on starting and stopping middle-tier instances

### 6.3.1.3 Monitoring OracleAS Cold Failover Cluster (Identity Management)

This section describes how to monitor the OracleAS Cold Failover Cluster (Identity Management).

1. Check the status of OPMN and OPMN-managed processes:

On UNIX systems:

```
$ORACLE_HOME/opmn/bin/opmnctl status
```

On Windows systems:

```
%ORACLE_HOME%\opmn\bin\opmnctl status
```

2. Check the status of Application Server Control.

On UNIX systems:

```
$ORACLE_HOME/bin/emctl status iasconsole
```

On Windows systems:

```
%ORACLE_HOME%\bin\emctl status iasconsole
```

3. Check the status of Oracle Internet Directory:

On UNIX systems:

```
$ORACLE_HOME/ldap/bin/ldapcheck
```

On Windows systems:

```
%ORACLE_HOME%\ldap\bin\ldapcheck
```

Verify that you can log in to Oracle Internet Directory:

On UNIX systems:

```
$ORACLE_HOME/bin/oidadmin
```

On Windows systems:

```
%ORACLE_HOME%\bin\oidadmin
```

Use the following login and password:

Login: orcladmin

Passwd: <orcladmin\_password>

---

---

**Note:** After installation, the *orcladmin\_password* is the same as the *ias\_admin* password.

---

---

4. Verify you can log in to OracleAS Single Sign-On:

`http://<host>:<HTTP_port>/pls/orasso`

Login: orcladmin

Passwd: <*orcladmin\_password*>

---

---

**Note:** After installation the *orcladmin\_password* is the same as the *ias\_admin* password.

---

---

5. Verify you can log in to Delegated Administration Services:

`http://<host>:<HTTP_port>/oiddas`

Login: oracleadmin

Passwd: <*orcladmin\_password*>

---

---

**Note:** After installation the *orcladmin\_password* is the same as the *ias\_admin* password.

---

---

## 6.3.2 Configuring OracleAS Cold Failover Cluster (Identity Management)

This section covers the following topics:

- Changing Configuration For OracleAS Cold Failover Cluster (Identity Management)
- Configuring Virtual IPs For OracleAS Cold Failover Cluster (Identity Management)

### 6.3.2.1 Changing Configuration For OracleAS Cold Failover Cluster (Identity Management)

This configuration requires the installation of OracleAS Identity Management components a two-node, active-passive hardware cluster. The OracleAS Metadata Repository is installed separately and can be installed in an existing database using OracleAS Metadata Repository Creation Assistant. Hence, OracleAS Identity Management has a different Oracle home from OracleAS Metadata Repository. The two nodes for OracleAS Metadata Repository are connected to shared storage. Thus, if you need to change the configuration, you can use the standard OracleAS Identity Management administration techniques. The OracleAS Metadata Repository may be a single instance database, a cold failover database, or a Real Application Clusters database. It may or may not be on the same cluster as the OracleAS Cold Failover Cluster (Identity Management) only install.

**See Also:** *Oracle Application Server Administrator's Guide*



### 6.3.2.2 Configuring Virtual IPs For OracleAS Cold Failover Cluster (Identity Management)

The Oracle Application Server Installation Guides for your platform cover the instructions for configuring the Virtual IPs for a OracleAS Cold Failover Cluster (Identity Management).

**See Also:**

- 11.2.2 Map the Virtual Hostname and Virtual IP Address in the *Oracle Application Server Installation Guide 10g Release 2 (10.1.2) for Solaris Operating System (SPARC)*
- 11.3.2 Get a Virtual Address for the Cluster in the *Oracle Application Server Installation Guide 10g Release 2 (10.1.2) for Microsoft Windows*

### 6.3.3 Failover For OracleAS Cold Failover Cluster (Identity Management)

This section covers the following topics:

- Failover For OracleAS Cold Failover Cluster (Identity Management) On UNIX Systems
- Failover For OracleAS Cold Failover Cluster (Identity Management) On Linux Systems

#### 6.3.3.1 Failover For OracleAS Cold Failover Cluster (Identity Management) On UNIX Systems

The following shows the steps to failover from the active node to the standby node, for Solaris systems with a Veritas Volume Manager.

On the failed node:

1. Stop and if necessary, kill all processes belonging to the OracleAS Cold Failover Cluster (Identity Management) instance on this node.
2. Login as root.
3. Stop the Oracle Cluster Synchronization Services (CSS) daemon, `ocssd`, if it is running. Use the following command:

```
> /etc/init.d/init.cssd stop
```

4. Unmount the file system using the following command:

```
> umount <mount_point>
```

---

**Note:** Before executing the above commands, ensure that the file system is not busy. If it is, check which processes are using it and stop them if required.

---

5. Login as root and deport the disk group. For example, if you are using Sun Cluster with Veritas Volume Manager, deport the disk group using the following commands:

```
> su - root
> vxvg deport <disk_group_name>
```

6. If the failed node is usable, execute the following command to release the virtual IP:

```
> ifconfig <interface_name> removeif <virtual_IP>
```

On the new active node:

1. Login as root.
2. Execute the following command to assign the virtual IP to this node:

```
> ifconfig <interface_name> addif <virtual_IP> up
```

3. Import the disk group. For example, if you are using Sun Cluster with Veritas Volume Manager, use the following commands:

```
> vxdg import <disk_group_name>
> vxvol -g <disk_group_name> startall
```

4. Mount the file system using the following command:

```
> mount /dev/vx/dsk/<disk_group_name>/<volume_name> <mount_point>
```

5. If the Oracle Cluster Synchronization Services (CSS) daemon, `ocssd`, is required, run the following command as the user which installed the Oracle home:

```
> /etc/init.d/init.cssd start
```

6. Start all OracleAS Cold Failover Cluster (Identity Management) processes on this new active node with the following command:

```
> $ORACLE_HOME/opmn/bin/opmnctl startall
```

### 6.3.3.2 Failover For OracleAS Cold Failover Cluster (Identity Management) On Linux Systems

The following shows the steps to failover from the active node to the standby node, for Linux Systems.

On the failed node:

1. Make sure all processes belonging to the OracleAS Cold Failover Cluster (Identity Management) instance on the failed node are down.
2. Login as root.
3. Use the following command to stop the Oracle Cluster Synchronization Services (CSS) daemon, `ocssd`, if it is running:

```
> /etc/init.d/init.cssd stop
```

4. Unmount the file system using the following command:

```
> umount <mount_point>
```

If the file system is busy, check which processes are using the file system with the following command:

```
> fuser -muv <Shared Storage Partition>
```

Stop the processes, if required, using the following command:

```
> fuser -k <Shared Storage Partition>
```

5. If the failed node is usable, execute the following command to release the virtual IP:

```
> ifconfig <interface_name> down
```

For example,

```
> ifconfig eth1:1 down
```

On the new active node:

1. Login as root.
2. Execute the following command to assign the virtual IP to this node (the new active node):

```
> ifconfig <interface_name> netmask <subnet mask> up
```

For example,

```
> ifconfig 144.88.27.125 netmask 255.255.252.0 up
```

3. Verify that the virtual IP is up and working using `telnet` from a different host (subnet/domain).
4. Mount the file system using the following command:

```
> mount <Shared Storage Partition> <mount_point>
```

For example:

```
> mount /dev/sdc1 /oracle
```

5. If the Oracle Cluster Synchronization Services (CSS) daemon, `ocssd`, is required, run the following command as the user that installed the Oracle home:

```
> /etc/init.d/init.cssd start
```

6. Start all OracleAS Identity Management processes on this new active node with the following commands:

- a. Start OPMN and all OPMN-managed processes with the following command:

```
$ORACLE_HOME/opmn/bin/opmnctl startall
```

- b. Start the Application Server Control Console:

```
$ORACLE_HOME/bin/emctl start iasconsole
```

### 6.3.4 Backup and Recovery For OracleAS Cold Failover Cluster (Identity Management)

For backing up OracleAS Cold Failover Cluster (Identity Management) environments and recovering these backups during failures, use the general backup and recovery procedures provided in the *Oracle Application Server Administrator's Guide*.

**See Also:** *Oracle Application Server Administrator's Guide*



# Part IV

---

## Disaster Recovery

The chapter in this part describes the Oracle Application Server Disaster Recovery solution.

This part contains the following chapter:

- Chapter 7, "Oracle Application Server Disaster Recovery"



---



---

## Oracle Application Server Disaster Recovery

Disaster recovery refers to how a system recovers from catastrophic site failures caused by natural or unnatural disasters. Examples of catastrophic failures include earthquakes, tornadoes, floods, or fire. Additionally, disaster recovery can also refer to how a system is managed for planned outages. For most disaster recovery situations, the solution involves replicating an entire site, not just pieces of hardware or subcomponents. This also applies to the Oracle Application Server Disaster Recovery (OracleAS Disaster Recovery) solution.

This chapter describes the OracleAS Disaster Recovery solution, how to configure and set up its environment, and how to manage the solution for high availability. The discussion involves both OracleAS middle tiers and OracleAS Infrastructure tiers in two sites: production and standby. The standby site is configured identically to the production site. Under normal operation, the production site actively services requests. The standby site is maintained to mirror the applications and content hosted by the production site.

The sites are managed using Oracle Application Server Guard, which contains a command-line utility (asgctl) that encapsulates administrative tasks (verify, dump, instantiate, synchronize, switchover, failover, startup farm, shutdown farm, show operation, and stop operation, among others. Behind the scenes OracleAS Guard automates the use of Backup and Recovery Tool (for managing configuration files in the file system) and Oracle Data Guard (for managing the OracleAS Infrastructure database) in a distributed fashion across the farm. The following table provides a summary of the OracleAS Disaster Recovery strategy and how this Oracle software is used behind the scenes:

**Table 7-1 Overview of OracleAS Disaster Recovery strategy**

Coverage	Procedure	Purpose
Middle-tier Configuration Files	OracleAS Backup and Recovery Tool	To backup OracleAS configuration files in the production site middle-tier nodes and restore the files to the standby site middle-tier nodes.
OracleAS Infrastructure Configuration Files	OracleAS Backup and Recovery Tool	To backup OracleAS configuration files in the production site OracleAS Infrastructure node and restore them to the standby site OracleAS Infrastructure node.
OracleAS Infrastructure Database	Oracle Data Guard	To ship archive logs from production site OracleAS Infrastructure database to standby site OracleAS Infrastructure database. Note that logs are not applied immediately.

Note that your other databases must be covered in the Disaster Recovery strategy and that you must use Oracle Data Guard as the solution. In addition to the recovery strategies, configuration and installation of both sites are discussed. For these tasks, two different ways of naming the middle-tier nodes are covered as well as two ways of resolving hostnames intra-site and inter-site.

With OracleAS Disaster Recovery, planned outages of the production site can be performed without interruption of service by switching over to the standby site using the OracleAS Guard switchover operation. Unplanned outages are managed by failing over to the standby site using the OracleAS Guard failover operation. Procedures for switchover and failover are covered in this chapter in Section 7.6, "Runtime Operations -- OracleAS Guard Switchover and Failover Operations".

This chapter is organized into the following main sections:

- Section 7.1, "Oracle Application Server 10g Disaster Recovery Solution"
- Section 7.2, "Preparing the OracleAS Disaster Recovery Environment"
- Section 7.3, "Overview of Installing Oracle Application Server 10g Software"
- Section 7.4, "Overview of OracleAS Guard and asgctl"
- Section 7.5, "OracleAS Guard Operations -- Standby Instantiation and Standby Synchronization"
- Section 7.6, "Runtime Operations -- OracleAS Guard Switchover and Failover Operations"
- Section 7.7, "Wide Area DNS Operations"
- Section 7.8, "Using OracleAS Guard Command-Line Utility (asgctl)"

**See Also:** *Oracle Application Server Installation Guide* for instructions on how to install the OracleAS Disaster Recovery solution.

Geographically distributed IM Infrastructure deployment replication, though an example of an active-active configuration, shares some features similar to an OracleAS Disaster Recovery solution in that Oracle Internet Directory, OracleAS Metadata Repository, and OracleAS Single Sign-On are set up in replication and distributed across different geographic regions. Note that each OracleAS Single Sign-On site uses its own Oracle Internet Directory and OracleAS Metadata Repository located at the local site, hence the active-active configuration. The shared similarities are in case a database failure is detected at one site, Oracle Internet Directory and OracleAS Single Sign-On servers are reconfigured to route user requests to the closest geographic area and in case a OracleAS Single Sign-On middle-tier failure is detected, the network is reconfigured to route traffic to a remote middle tier. However, this solution does not provide synchronization for OracleAS Portal, OracleAS Wireless, and Distributed Configuration Management (DCM) schemas in the Infrastructure database because neither supports the replica model used for Oracle Internet Directory and OracleAS Single Sign-On information. See *Oracle Identity Management Concepts and Deployment Planning Guide* for more information about a geographically distributed identity management Infrastructure deployment.

## 7.1 Oracle Application Server 10g Disaster Recovery Solution

The Oracle Application Server Disaster Recovery solution consists of two identically configured sites - one primary (production/active) and one secondary (standby). Both sites have the same number of middle-tier and OracleAS Infrastructure nodes and the



same number and types of components installed. In other words, the installations on both sites, middle tier and OracleAS Infrastructure are identical. Both sites are usually dispersed geographically, and if so, they are connected via a wide area network.

This section describes the overall layout of the solution, the major components involved, and the configuration of these components. It has the following sections:

- Section 7.1.1, "Requirements"
- Section 7.1.2, "Topology"

## 7.1.1 Requirements

To ensure that your implementation of the OracleAS Disaster Recovery solution performs as designed, the following requirements need to be adhered to:

- On each host in the standby site, make sure the following is identical to its equivalent peer in the production site:
  - For the middle-tier hosts, physical hostnames.

---



---

**Note:** If you already have installed systems, you only need to modify the physical names for the mid tier systems at the standby site and then create a virtual hostname for the physical hostname of the OracleAS Infrastructure (see the next bullet). See Section 7.2.1, "Planning and Assigning Hostnames" for information about how to change these physical hostnames and the virtual hostname.

---



---

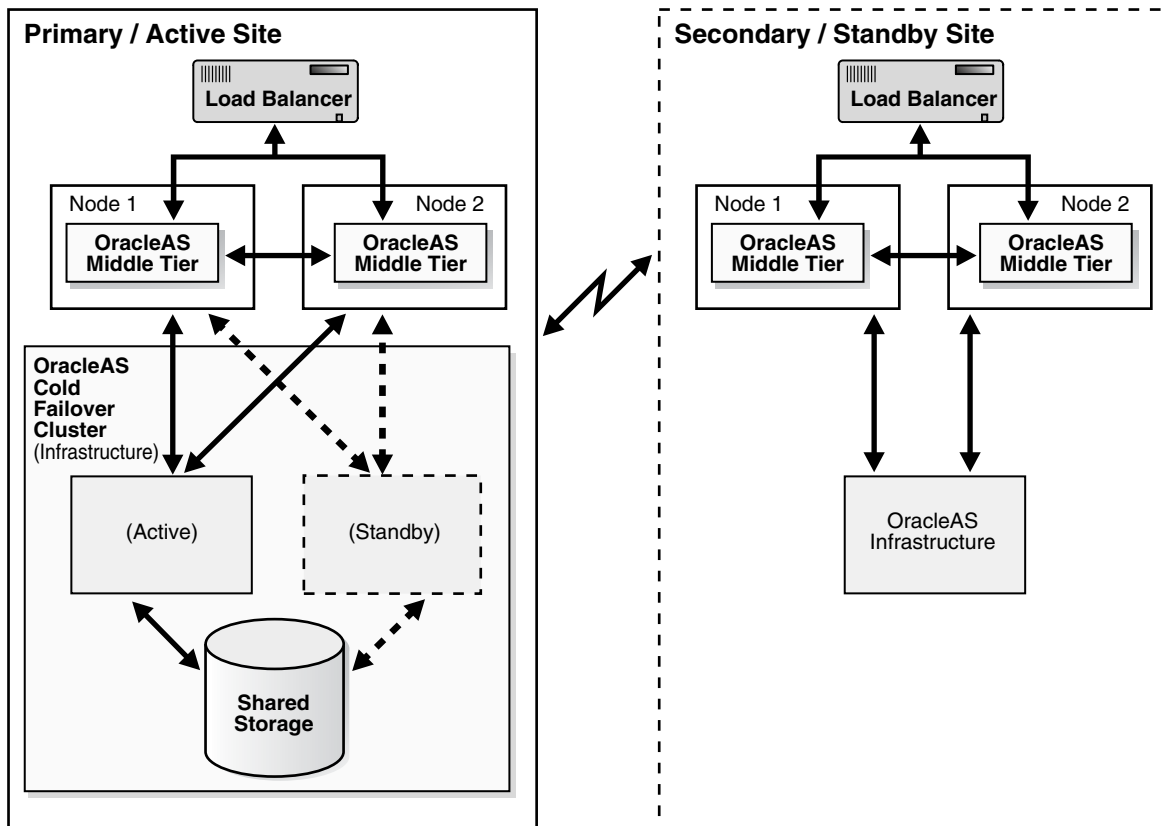
- Virtual hostname for the OracleAS Infrastructure. The virtual hostname can be specified in the Specify High Availability Addressing Page screen presented by the installer.
- Hardware platform.
- Operating system release and patch levels.
- All installations conform to the requirements listed in the *Oracle Application Server Installation Guide* to install Oracle Application Server.
- Oracle Application Server software is installed in identical directory paths between each host in the production site and its equivalent peer in the standby site.
- The following details must be the same between a host in the production site and peer in the standby site:
  - User name and password of the user who installed Oracle Application Server must be the same between a host in the production site and its peer in the standby site.
  - Numerical user ID of the user who installed Oracle Application Server on that particular node
  - Group name of the user who installed Oracle Application Server on that particular node
  - Numerical group ID of the group of the user who installed Oracle Application Server on that particular node
  - Environment profile
  - Shell (command line environment)

- Directory structure, Oracle home names, and path of the Oracle home for each OracleAS installation on a node. Do not use symbolic links anywhere in the path.
- OracleAS installation types:
  - \* Middle Tier: J2EE and Web Cache, and Portal and Wireless
  - \* OracleAS Infrastructure: Metadata Repository and Identity Management (both are required to be installed with the OracleAS Infrastructure installation type in both sites for the OracleAS Disaster Recovery solution)

## 7.1.2 Topology

Figure 7-1 depicts the topology of an example OracleAS Disaster Recovery solution.

**Figure 7-1 Example Oracle Application Server 10g site-to-site Disaster Recovery solution (load balancer appliance is optional if only one middle-tier node is present)**



The procedures and steps for configuring and operating the OracleAS Disaster Recovery solution support 1 to n number of middle-tier installations in the production site. The same number of middle-tier installations must exist in the standby site. The middle tiers must mirror each other in the production and standby sites.

For the OracleAS Infrastructure, a uniform number of installations is not required between the production and standby sites. For example, the Oracle Application Server Cold Failover Cluster solution can be deployed in the production site, and a single node installation of the OracleAS Infrastructure can be deployed in the standby site as shown in Figure 7-1. This way, the production site's OracleAS Infrastructure has protection from host failure using an OracleAS Cold Failover Cluster. This solution provides hardware redundancy by utilizing a virtual hostname. Refer to the section

Section 5.3.2, "Active-Passive High Availability Solutions" on page 5-13 for more information on OracleAS Cold Failover Clusters.

The Oracle Application Server Disaster Recovery solution is an extension to various single-site Oracle Application Server architectures. Examples of such single-site architectures include the combination of Oracle Application Server Cold Failover Cluster (Infrastructure) and active-active Oracle Application Server middle-tier architecture. For the latest information on what single-site architectures are supported, please check the following URL for the latest certification matrix.

[http://www.oracle.com/technology/products/ias/hi\\_av/index.html](http://www.oracle.com/technology/products/ias/hi_av/index.html)

The following are important characteristics of the OracleAS Disaster Recovery solution:

- Middle-tier installations are identical between the production and standby sites. In other words, each middle-tier installation in the production site has an identical installation in the standby site. More than one middle-tier node is recommended because this enables each set of middle-tier installations on each site to be redundant. Being on multiple machines, problems and outages within a site of middle-tier installations are transparent to clients.
- The OracleAS Disaster Recovery solution is restricted to identical site configuration to ensure that processes and procedures are kept the same between sites, making operational tasks easier to maintain and execute. Identical site configuration also allows for a higher success rate for manually maintaining the synchronization of Oracle Application Server 10g component configuration files between sites.
- When the production site becomes unavailable due to a disaster, the standby site can become operational within a reasonable time. Client requests are always routed to the site that is operating in the production role. After a failover or switchover operation occurs due to an outage, client requests are routed to another site that assumes the production role. The quality of service offered by the new production site should be the same as that offered by the original production site before the outage.
- The sites are set up in active-passive configuration. An active-passive setup has one primary site used for production and one secondary site that is initially passive (on standby). The secondary site is made active only after a failover or switchover operation is performed. Since the sites are symmetrical, after failover or switchover, the original standby site can be kept active as the new production site. After repairing or upgrading the original production site, it can be made into the new standby site. Either site should offer the same level of service to clients as the other.
- The site playing the standby role contains a physical standby of the Oracle Application Server Infrastructure coordinated by Oracle Data Guard, OracleAS Guard automates the configuration and use of Oracle Data Guard together with procedures for backing up and restoring OracleAS Infrastructure configuration files and provides configuration synchronization between the production and standby sites. Switchover and failover operations allow the roles to be traded between the OracleAS Infrastructures in the two sites. Refer to Section 7.5, "OracleAS Guard Operations -- Standby Instantiation and Standby Synchronization" and Section 7.8, "Using OracleAS Guard Command-Line Utility (asgctl)" for information on using the asgctl command-line interface to perform OracleAS Guard administrative tasks of instantiation, synchronization, switchover, and failover in the OracleAS Disaster Recovery solution.

## 7.2 Preparing the OracleAS Disaster Recovery Environment

Prior to the installation of OracleAS software for the OracleAS Disaster Recovery solution, a number of system level configurations are required or optional as specified. The tasks that accomplish these configurations are:

- Section 7.2.1, "Planning and Assigning Hostnames"
- Section 7.2.2, "Configuring Hostname Resolution"
- Appendix F, "Secure Shell (SSH) Port Forwarding" (optional)

This section covers the steps needed to perform these tasks.

### 7.2.1 Planning and Assigning Hostnames

Before performing the steps to set up the physical and network hostnames, plan the physical and network hostnames you wish to use with respect to the entire OracleAS Disaster Recovery solution. The overall approach to planning and assigning hostnames is to meet the following goals:

- OracleAS components in the middle tier and OracleAS Infrastructure must use the same physical hostnames in their configuration settings regardless of whether the components are in the production or standby site. In addition, you must also create a virtual hostname for the physical hostname of the OracleAS Infrastructure.

For example, if a middle-tier component in the production site uses the name "asmid1" to reach a host in the same site, the same component in the standby site must use the same name to reach `asmid1`'s equivalent peer in the standby site. Likewise, if the virtual hostname of the OracleAS Infrastructure on the production site uses the name "infra", the virtual hostname for the physical hostname of the OracleAS Infrastructure on the standby site must be named "infra".

- No changes to hostnames (physical, network, or virtual) are required when the standby site takes over the production role.

---

---

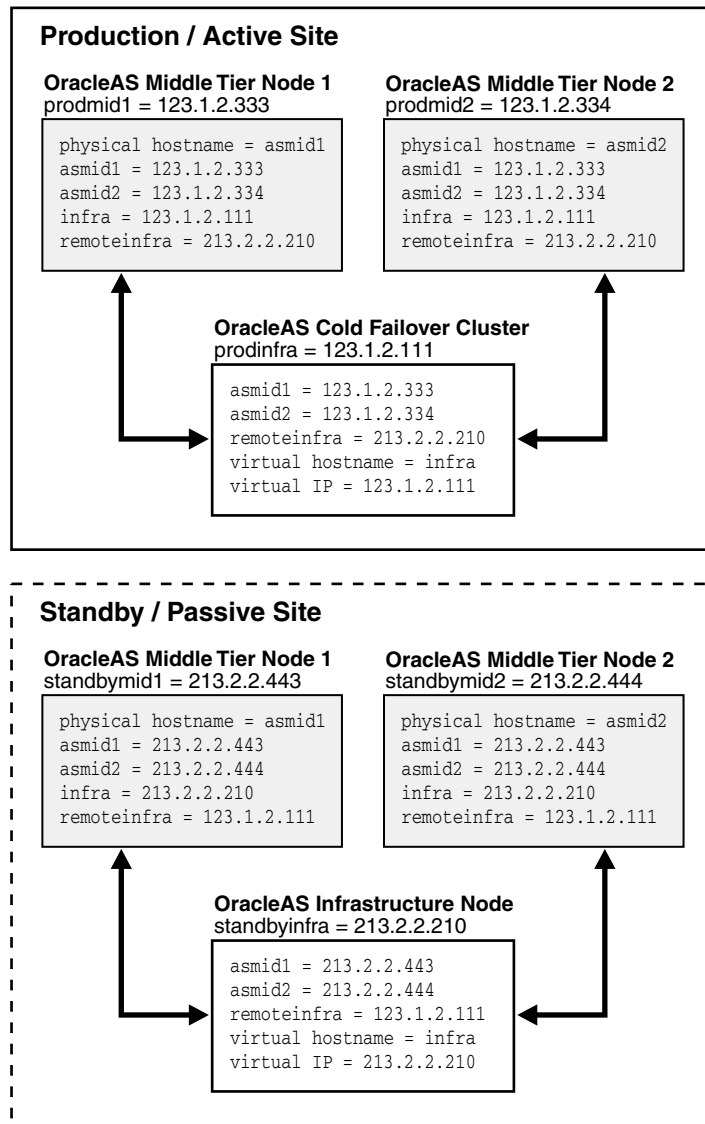
**Note:** Although the physical hostnames in the production and standby sites must remain uniform between the two sites, the resolution of these physical hostnames to the correct hosts can be different. Section 7.2.2, "Configuring Hostname Resolution" explains hostname resolution.

---

---

To illustrate what should be done to plan and assign hostnames, let us use an example as shown in Figure 7-2.

**Figure 7-2 Name assignment example in the production and standby sites**



In Figure 7-2, two middle-tier nodes exist in the production site. The OracleAS Infrastructure can be a single node or an OracleAS Cold Failover Cluster solution (represented by a single virtual hostname and a virtual IP, as for a single node OracleAS Infrastructure). The common names in the two sites are the physical hostnames of the middle-tier nodes and the virtual hostname of the OracleAS Infrastructure. Table 7-2 below details what the physical, network, and virtual hostnames are in the example:

**Table 7-2 Physical, network, and virtual hostnames in Figure 7-2**

Physical Hostnames	Virtual Hostname	Network Hostnames
asmid1	-	prodmid1, standbymid1
asmid2	-	prodmid2, standbymid2
- <sup>1</sup>	infra	prodfinfra, standbyinfra

<sup>1</sup> In this example, the physical hostname is the network hostname. Therefore, the network host name is used in the appropriate asgctl commands for the respective <host>, <host-name>, or <standby\_farm\_host> parameter arguments.

- *Co-hosting non OracleAS applications*

If the hosts in the production site are running non OracleAS applications, and you wish to co-host OracleAS on the same hosts, changing the physical hostnames of these hosts may break these applications. In such a case, you can keep these hostnames in the production site and modify the physical hostnames in the standby site to the same as those in the production site. The non OracleAS applications can then also be installed on the standby hosts so that they can act in a standby role for these applications.

As explained in Section 1.2.1, "Terminology", physical, network, and virtual hostnames have differing purposes in the OracleAS Disaster Recovery solution. They are also set up differently. Information on how the three types of hostnames are set up follows.

### 7.2.1.1 Physical Hostnames

The naming of middle-tier hosts in both the production and standby sites require the changing of the physical hostname in each host.

In Solaris, to change the physical hostname of a host:

---

**Note:** For other UNIX variants, consult your system administrator for equivalent commands in each step.

---

1. Check to see what the existing physical hostname is set to. Type:  

```
prompt> hostname
```
2. Use a text editor, such as `vi`, to edit the name in `/etc/nodename` to your planned physical hostname.
3. For each middle-tier host, reboot it for the change to take effect.
4. Repeat step 1 to verify the correct hostname has been set.
5. Repeat the above steps for each host in the production and standby sites.

In Windows, to change the physical hostname of a host:

---

**Note:** The user interface elements in your version of Windows may vary from those described in the following steps.

---

1. In the Start menu, select Control Panel.
2. Double-click the System icon.
3. Select the Advance tab.
4. Select Environment variables.
5. Under the User Environment variables for the installer account, select New to create a new variable.
6. Enter the name of the variable as `"_CLUSTER_NETWORK_NAME_"`.
7. For the value of this variable, enter the planned physical hostname.

### 7.2.1.2 Network Hostnames

The network hostnames used in the OracleAS Disaster Recovery solution are defined in DNS. These hostnames are visible in the network that the solution uses and are

resolved through DNS to the appropriate hosts via the assigned IP address in the DNS system. You need to add these network hostnames and their corresponding IP addresses to the DNS system.

Using the example in Figure 7–2, the following should be the additions made to the DNS system serving the entire network that encompasses the production and standby sites:

```

prodmid1.oracle.com      IN      A       123.1.2.333
prodmid2.oracle.com      IN      A       123.1.2.334
prodinfra.oracle.com     IN      A       123.1.2.111
standbymid1.oracle.com   IN      A       213.2.2.443
standbymid2.oracle.com   IN      A       213.2.2.444
standbyinfra.oracle.com  IN      A       213.2.2.210

```

### 7.2.1.3 Virtual Hostname

As defined in Section 1.2.1, "Terminology", virtual hostname applies to the OracleAS Infrastructure only. It is specified during installation of the OracleAS Infrastructure. When you run the OracleAS Infrastructure installation type, a screen called "Specify High Availability" appears to provide a text box to enter the virtual hostname of the OracleAS Infrastructure that is being installed. Refer to the *Oracle Application Server Installation Guide* for more details.

For the example in Figure 7–2, when you install the production site's OracleAS Infrastructure, enter its virtual hostname, "infra", when you see the Specify High Availability Addressing Page screen. Enter the same virtual hostname when you install the standby site's OracleAS Infrastructure.

---



---

**Note:** If the OracleAS Infrastructure is installed in an OracleAS Cold Failover Cluster solution, the virtual hostname is the name that is associated with the virtual IP of the OracleAS Cold Failover Cluster.

---



---

## 7.2.2 Configuring Hostname Resolution

In the Oracle Application Server Disaster Recovery solution, one of two ways of hostname resolution can be configured to resolve the hostnames you planned and assigned in Section 7.2.1, "Planning and Assigning Hostnames". These are:

- Section 7.2.2.1, "Using Local Hostnaming File Resolution"
- Section 7.2.2.2, "Using DNS Resolution"

In UNIX, the order of the method of name resolution can be specified using the "hosts" parameter in the file `/etc/nsswitch.conf`. The following is an example of the `hosts` entry:

```
hosts:      files dns nis
```

In the previous statement, local hostnaming file resolution is preferred over DNS and NIS (Network Information Service) resolutions. When a hostname is required to be resolved to an IP address, the `/etc/hosts` file (UNIX) or `C:\WINDOWS\system32\drivers\etc\hosts` file is consulted first. In the event that a hostname cannot be resolved using local hostnaming resolution, DNS is used. (NIS resolution is not used for the OracleAS Disaster Recovery solution.) Refer to your UNIX system's documentation if you wish to find out more about `/etc/nsswitch.conf`.

### 7.2.2.1 Using Local Hostnaming File Resolution

This method of hostname resolution relies on a local hostnaming file to contain the requisite hostname-to-IP address mappings. In UNIX, this file is `/etc/hosts`. In Windows, this file is `C:\WINDOWS\system32\drivers\etc\hosts`.

To use the local hostnaming file to resolve hostnames for the OracleAS Disaster Recovery solution in UNIX, for each middle-tier and OracleAS Infrastructure host in both the production and standby sites, perform the following:

1. Use a text editor, such as `vi`, to edit the `/etc/nsswitch.conf` file. With the "hosts:" parameter, specify "files" as the first choice for hostname resolution.
2. Edit the `/etc/hosts` file to include the following:
  - The physical hostnames and their correct IP addresses of all middle-tier nodes in the current site. Ensure that the first entry is the hostname and IP address of the current node.

---

---

**Note:** When making entries in the `hosts` file, make sure the intended hostname is positioned in the second column of the `hosts` file; otherwise, an `asgctl verify farm` with `<host>` operation will fail indicating that the production farm is not symmetrical with the standby farm. See Appendix A, "Troubleshooting High Availability" for more information about troubleshooting and resolving this type of problem.

---

---

For example, if you are editing the `/etc/hosts` file of a middle-tier node in the production site, enter all the middle-tier physical hostnames and their IP addresses in the production site beginning the list with the current host. (Note that you should also include fully qualified hostnames in addition to the abbreviated hostnames. See Table 7-3.)

- The virtual hostname of the OracleAS Infrastructure in the current site.

For example, if you are editing the `/etc/hosts` of a middle-tier node in the standby site, enter the virtual hostname, fully qualified and abbreviated, and IP address of the OracleAS Infrastructure host in the standby site.
3. Reboot each host after editing the above files.
  4. From each host, ping each physical hostname that is valid in its particular site to ensure that the IP addresses have been assigned correctly.

For the example in Figure 7-2, on the `asmid1` host, use the following commands in:

```
ping asmid1
```

The returned IP address should be 123.1.2.333.

```
ping asmid2
```

The returned IP address should be 123.1.2.334.

```
ping infra
```

The returned IP address should be 123.1.2.111.



---



---

**Note:** Some UNIX variants, such as Solaris, require the `-s` option to return an IP address.

---



---

In Windows, the method of ordering hostname resolution varies depending on the Windows version. Refer to the documentation for your version of Windows for the appropriate steps.

Using the example in Figure 7–2, Table 7–3 shows the `/etc/hosts` file entries on each production node contains the required entries in the of each UNIX host. The entries in the Windows `C:\WINDOWS\system32\drivers\etc\hosts` file should reflect similarly.

**Table 7–3 Network and virtual hostname entries in each `/etc/hosts` file of example in Figure 7–2**

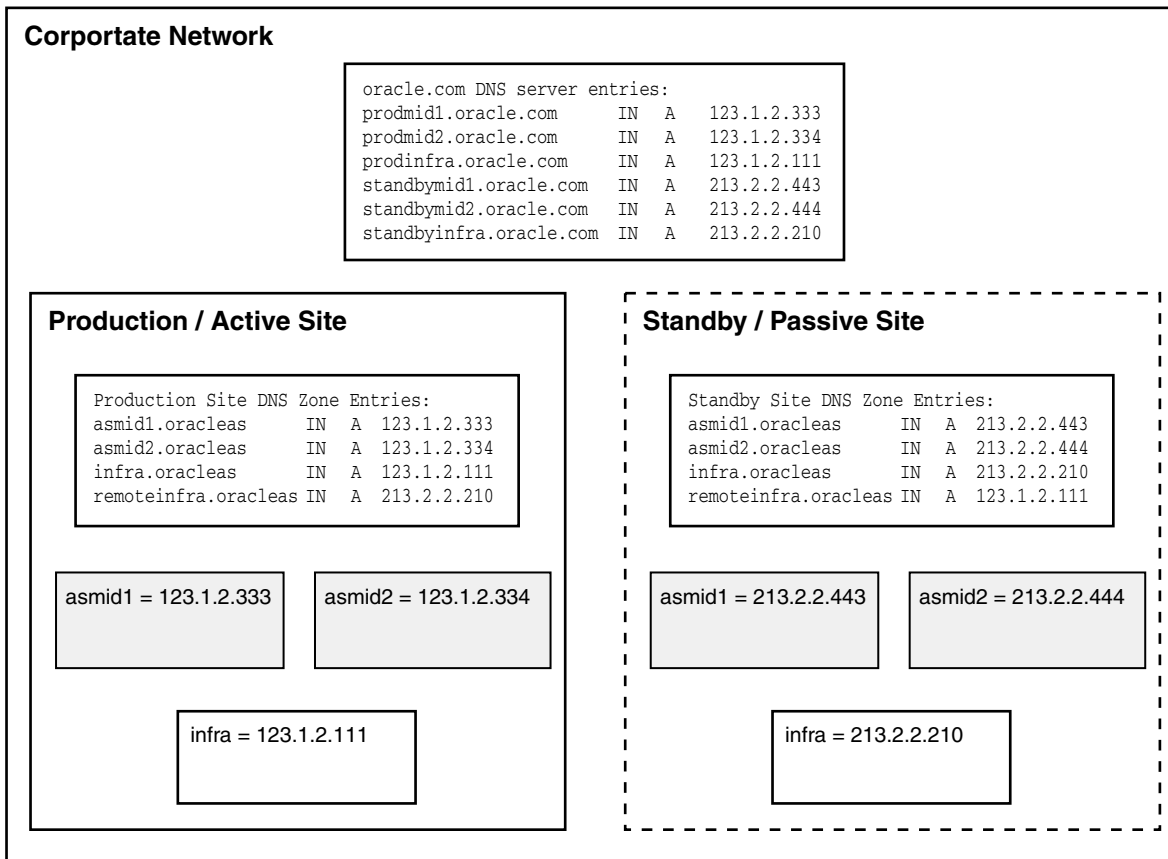
Host	Entries in <code>/etc/hosts</code>
asmid1 in production site	123.1.2.333 asmid1.oracle.com asmid1 123.1.2.334 asmid2.oracle.com asmid2 123.1.2.111 infra.oracle.com infra 213.2.2.210 remoteinfra.oracle.com remoteinfra
asmid2 in production site	123.1.2.334 asmid2.oracle.com asmid2 123.1.2.333 asmid1.oracle.com asmid1 123.1.2.111 infra.oracle.com infra 213.2.2.210 remoteinfra.oracle.com remoteinfra
infra in production site	123.1.2.111 infra.oracle.com infra 123.1.2.333 asmid1.oracle.com asmid1 123.1.2.334 asmid2.oracle.com asmid2 213.2.2.210 remoteinfra.oracle.com remoteinfra
asmid1 in standby site	213.2.2.443 asmid1.oracle.com asmid1 213.2.2.444 asmid2.oracle.com asmid2 213.2.2.210 infra.oracle.com infra 123.1.2.111 remoteinfra.oracle.com remoteinfra
asmid2 in standby site	213.2.2.444 asmid2.oracle.com asmid2 213.2.2.443 asmid1.oracle.com asmid1 213.2.2.210 infra.oracle.com infra 123.1.2.111 remoteinfra.oracle.com remoteinfra
infra in standby site	213.2.2.210 infra.oracle.com infra 213.2.2.443 asmid1.oracle.com asmid1 213.2.2.444 asmid2.oracle.com asmid2 123.1.2.111 remoteinfra.oracle.com remoteinfra

### 7.2.2.2 Using DNS Resolution

To set up the OracleAS Disaster Recovery solution to use DNS hostname resolution, site-specific DNS servers must be set up in the production and standby sites in addition to the overall corporate DNS servers (usually more than one DNS server exists in a corporate network for redundancy). Figure 7–3 provides an overview of this setup.

**See Also:** Appendix E, "Setting Up a DNS Server" for instructions on how to set up a DNS server in a UNIX environment.

Figure 7-3 DNS resolution topology overview



For the topology in Figure 7-3 to work, the following requirements and assumptions are made:

- The production and standby sites' DNS servers are not aware of each other. They make non authoritative lookup requests to the overall corporate DNS servers if they fail to resolve any hostnames within their specific sites.
- The production site and standby site DNS servers contain entries for middle-tier physical hostnames and OracleAS Infrastructure virtual hostnames. Each DNS server contain entries of hostnames within their own site only. The sites have a common domain name that is different from that of the overall corporate domain name.
- The overall corporate DNS servers contain network hostname entries for the middle-tier hosts and OracleAS Infrastructure hosts of both production and standby sites.
- In UNIX, the `/etc/hosts` file in each host does not contain any entries for the physical, network, or virtual hostnames of any host in either site. In Windows, this applies to the file `C:\WINDOWS\system32\drivers\etc\hosts`.

To set up the OracleAS Disaster Recovery solution for DNS resolution:

1. Configure each of the overall corporate DNS servers with the network hostnames of all the hosts in the production and standby sites. Using the example in Figure 7-2, the following entries are made:

```

prodmid1.oracle.com    IN  A   123.1.2.333
prodmid2.oracle.com    IN  A   123.1.2.334
    
```

```

prodinfra.oracle.com      IN    A    123.1.2.111
standbymid1.oracle.com   IN    A    213.2.2.443
standbymid2.oracle.com   IN    A    213.2.2.444
standbyinfra.oracle.com  IN    A    213.2.2.210

```

2. For each site, production and standby, create a unique DNS zone by configuring a DNS server as follows:
  - a. Select a unique domain name to use for the two sites that is different from the corporate domain name. As an example, let's use the name "oracleas" for the domain name for the two sites in Figure 7-2. The high level corporate domain name is `oracle.com`.
  - b. Configure the DNS server in each site to point to the overall corporate DNS servers for unresolved requests.
  - c. Populate the DNS servers in each site with the physical hostnames of each middle-tier host and the virtual hostname of each OracleAS Infrastructure host. Include the domain name selected in the previous step.

For the example in Figure 7-2, the entries are as follows:

For the production site's DNS:

```

asmid1.oracleas         IN    A    123.1.2.333
asmid2.oracleas         IN    A    123.1.2.334
infra.oracleas          IN    A    123.1.2.111

```

For the standby site's DNS:

```

asmid1.oracleas         IN    A    213.2.2.443
asmid2.oracleas         IN    A    213.2.2.444
infra.oracleas          IN    A    213.2.2.210

```

---

**Note:** If you are using the OracleAS Cold Failover Cluster solution for the OracleAS Infrastructure in either site, enter the cluster's virtual hostname and virtual IP address. For example, in the previous step above, `infra` is the virtual hostname and `123.1.2.111` is the virtual IP of the cluster in the production site. For more information on the OracleAS Cold Failover Cluster solution, see Section 5.3.2, "Active-Passive High Availability Solutions" on page 5-13.

---

#### 7.2.2.2.1 Additional DNS Server Entries for Oracle Data Guard

Because OracleAS Guard automates the use of Oracle Data Guard technology, which is used to synchronize the production and standby OracleAS Infrastructure databases, the production OracleAS Infrastructure must be able to reference the standby OracleAS Infrastructure and vice versa.

For this to work, the IP address of the standby OracleAS Infrastructure host must be entered in the production site's DNS server with a unique hostname with respect to the production site. Similarly, the IP address of the production OracleAS Infrastructure host must be entered in the standby site's DNS server with the same hostname. The reason for these DNS entries is that Oracle Data Guard uses TNS Names to direct requests to the production and standby OracleAS Infrastructures. Hence, the appropriate entries must be made to the `tnsnames.ora` file as well. Additionally, OracleAS Guard `asgctl` command-line commands need to reference the network hostnames.

Using the example in Figure 7–2 and assuming that the selected name for the remote OracleAS Infrastructure is "remoteinfra", the entries in the DNS server in the production site are:

```
asmid1.oracleas      IN    A    123.1.2.333
asmid2.oracleas      IN    A    123.1.2.334
infra.oracleas       IN    A    123.1.2.111
remoteinfra.oracleas IN    A    213.2.2.210
```

And, for the standby site, its DNS server should have the following entries:

```
asmid1.oracleas      IN    A    213.2.2.443
asmid2.oracleas      IN    A    213.2.2.444
infra.oracleas       IN    A    213.2.2.210
remoteinfra.oracleas IN    A    123.1.2.111
```

## 7.3 Overview of Installing Oracle Application Server 10g Software

This section provides an overview of the steps for installing the OracleAS Disaster Recovery solution. After following the instructions in Section 7.2, "Preparing the OracleAS Disaster Recovery Environment" to set up the environment for the solution, go through this section for an overview of the installation process. Thereafter, follow the detailed instructions in the *Oracle Application Server Installation Guide* to install the solution.

---

---

**Note:** To assign identical ports to be used by symmetrical hosts in the production and standby sites, static port definitions can be used. These definitions are defined in a file, for example, named `staticports.ini`. Then, you specify the `staticports.ini` file in the **Select TCP/IP Ports** screen in the installer. Detailed information on this static ports file is found in the *Oracle Application Server Installation Guide*.

---

---

The following is the overall sequence for installing the OracleAS Disaster Recovery solution:

1. Install OracleAS Infrastructure in the production site (refer to *Oracle Application Server Installation Guide*).
2. Install OracleAS Infrastructure in the standby site (refer to *Oracle Application Server Installation Guide*).
3. Start the OracleAS Infrastructure in each site before installing the middle tiers for that site.
4. Install the middle tiers in the production site (refer to *Oracle Application Server Installation Guide*).
5. Install the middle tiers in the standby site (refer to *Oracle Application Server Installation Guide*).

Note the following important points when you perform the installation:

- The OracleAS Infrastructure Identity Management and OracleAS Metadata Repository components must be installed on the same host. These components cannot be distributed over multiple hosts or on multiple Oracle homes within the same host. (This requirement also applies to the OracleAS Cold Failover Cluster.)

- Ensure that the same ports are used by equivalent peer hosts in both sites. For example, the `asmid1` host in the standby site must use the same ports as the `asmid1` host in the production site. Utilize a static ports definition file for this purpose (see note above and the next point).
- In the installer's Specify TCP/IP Ports screen, specify the full path to the `staticports.ini` file.
- In the installer's Select Configuration Options screen, ensure that you select the High Availability and Replication option.
- During OracleAS Infrastructure installation, specify the virtual address assigned to the OracleAS Infrastructure in the Specify High Availability Addressing Page screen.
- For the middle-tier hosts, any of the available middle-tier installation types can be installed. (Ensure that the OracleAS Infrastructure services have been started for a site before installing any middle tiers in that site.)
- During each middle-tier installation, specify the OracleAS Infrastructure's virtual hostname as the OracleAS Infrastructure database.
- Start the OracleAS services on the hosts in each site starting with the OracleAS Infrastructure.

## 7.4 Overview of OracleAS Guard and asgctl

This section provides an overview of OracleAS Guard and its command-line interface `asgctl`. If you are already familiar with this overview information, you can continue to Section 7.5, "OracleAS Guard Operations -- Standby Instantiation and Standby Synchronization". This section contains the following sections:

- Section 7.4.1, "Overview of `asgctl`"
- Section 7.4.2, "OracleAS Guard Client"
- Section 7.4.3, "OracleAS Guard Server"
- Section 7.4.4, "`asgctl` Operations"
- Section 7.4.5, "OracleAS Guard Integration with OPMN"
- Section 7.4.6, "Supported OracleAS Disaster Recovery Configurations"
- Section 7.4.7, "Supported Oracle Application Server Releases and Operating Systems"

### 7.4.1 Overview of `asgctl`

`asgctl` is a command-line interface that encapsulates the Oracle Application Server Guard administrative tasks to:

- Ensure a correct environment in which OracleAS can run.
- Provide simple administrative semantics to instantiate a standby site, once the install has been performed.
- Provide a simple administration synchronization operation that will perform the complex steps, in a reliable fashion, narrowing the change window for potential problems.
- Provide a simple switchover administration operation that will perform the complex steps of switching from the production site to the standby site.

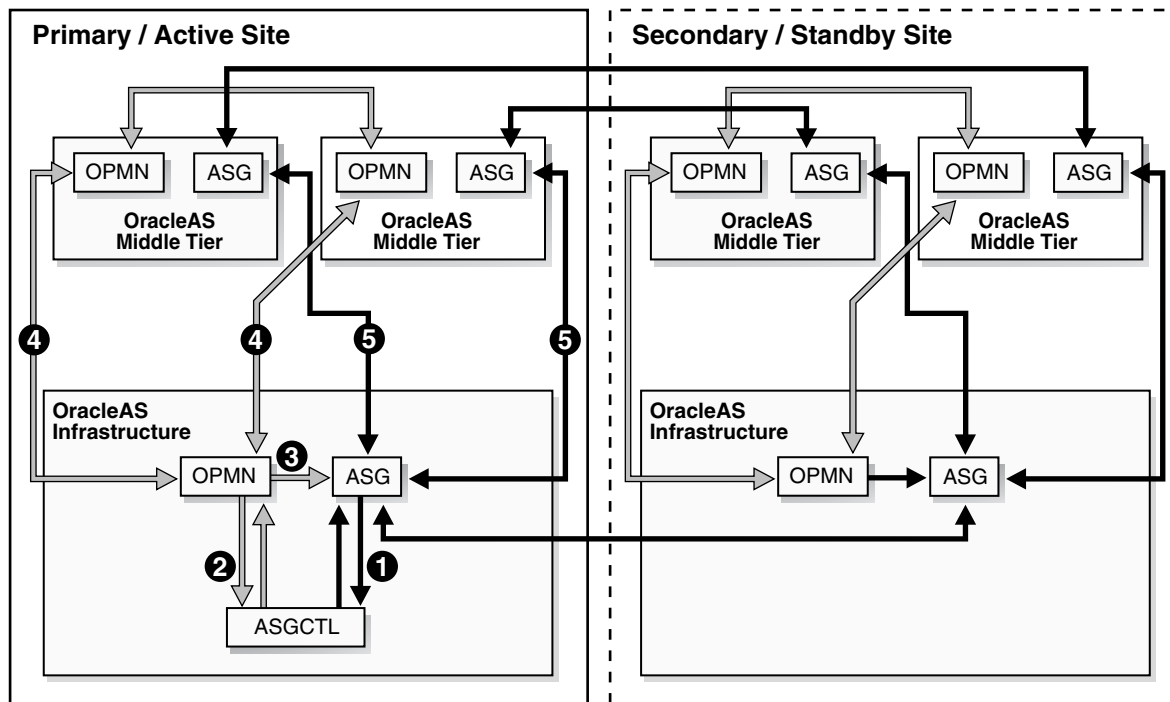
- Provide a simple failover administration operation that will perform the complex steps of recovering from an unplanned outage by failing over from the production site to the standby site.

asgctl greatly simplifies the complexity and magnitude of the number of steps involved in setting up and managing OracleAS Disaster Recovery. asgctl provides a distributed solution that consists of a client component (OracleAS Guard client) that can be installed on a machine on the farm and a transient server component (OracleAS Guard server) that is installed by default on the systems hosting the primary and standby Oracle homes that comprise the DR environment.

## 7.4.2 OracleAS Guard Client

The OracleAS Guard client is installed on every OracleAS install type. The OracleAS Guard client attempts to open and maintain a connection to the OracleAS Guard server (Step 1 in Figure 7–4). If this is unsuccessful, then the local OPMN server is contacted (Step 2) to start the local OracleAS Guard server for the instance (Step 3). The OracleAS Guard client then connects to the OracleAS Guard server and this server becomes the coordinating server in the DR configuration for all OracleAS Guard operations. For operations that require OracleAS Guard operations across the farm, the local OPMN server is contacted to start instances across the farm (Step 4) on all systems where OracleAS Guard server software is installed. The OracleAS Guard servers communicate directly (Step 5) and the coordinating OracleAS Guard server initiates and controls all communications on behalf of the OracleAS Guard client. The OracleAS Guard server will remain active until the OracleAS Guard client disconnects, or all operations are complete and the timeout value has been reached. The OracleAS Guard client provides an asgctl command-line interface (CLI) (see Section 7.8.5) consisting of a set of commands to perform instantiation, synchronization, switchover, failover, and monitoring operations.

Figure 7–4 How an ASG Server Is Started and OPMN and ASG Servers Communicate



### 7.4.3 OracleAS Guard Server

The OracleAS Guard server is a distributed server (installed by default) that runs on all the systems in a DR configuration. An OracleAS Guard server is a transient server that is started and stopped by OPMN on each system for the duration of a client session. The OracleAS Guard client maintains an active connection to the OracleAS Guard server on one system that has network connectivity in the DR configuration. This coordinating server communicates to the OracleAS Guard servers on the other systems in the DR configuration as necessary to complete processing during standby site instantiation, synchronization, and verification operations. The OracleAS Guard server carries out asgctl commands issued directly by the OracleAS Guard client or issued on behalf of the OracleAS Guard client by another OracleAS Guard server in the network for the client's session. The steps to complete an operation will execute in concert over all systems in both the production and standby farms. Most operational steps will be executed in parallel or sequentially (as required) on these systems throughout the DR configuration by the OracleAS Guard server.

### 7.4.4 asgctl Operations

asgctl operations using the asgctl commands fall into the following categories of operations:

- Standby site instantiation -- discovers the current farm definition at the production and standby sites, verifies that each complies with the OracleAS Disaster Recovery rules and restrictions of the current OracleAS software deployed on these systems prior to creation (instantiate farm command).
- Standby site synchronization -- synchronizes the standby site with the primary site to ensure that the two sites are transactionally consistent (sync farm command).
- Switchover -- switching from the production site to the standby site after the standby site is synchronized with the production site with the application of the database redo logs.
- Failover -- failing over from the production site to the standby site after restoring configuration files and restoring the OracleAS server environment to the point of the last successful backup operation.
- Verification and troubleshooting -- validates that the primary farm is running and the configuration is valid (verify farm command). If a standby farm is specified, compares the primary farm to which the local host system is a member with the standby farm to validate that they are consistent with one another and each conform to the requirements for OracleAS Disaster Recovery (verify farm command). Lets you shut down the farm (shutdown farm command) in the case that one of its members is having a problem and you need to shut everything down and then start up the farm again (startup farm command). Lets you determine what the current operations are that are running (show operations command) and lets you stop any operations that need to be halted for some reason (stop operation command).
- Use as needed -- setting asgctl credentials and identifying the OracleAS Infrastructure database on the primary farm (set primary database).

### 7.4.5 OracleAS Guard Integration with OPMN

OPMN can communicate with other OPMN servers within a farm (fine black lines and arrows in Figure 7–4), whereas OracleAS Guard servers and the OracleAS Guard client communicate across the network (coarse black lines and arrows in Figure 7–4) based on the configured topology of the service. With disaster recovery, these operations

require communication across both farms. In this case, OracleAS Guard servers must communicate across both farms and rely on OPMN services to start local OracleAS Guard servers (see Step 5 in Figure 7-4). OPMN starts the OracleAS Guard services across the production farm when an OracleAS Guard client connects to a server node. OracleAS Guard services are executed in concert with OracleAS Guard servers across the other nodes of the production farm and by default are a transient process. The results and status are communicated back to the OracleAS Guard client through the coordinating OracleAS Guard server. OPMN is used to start, stop, and monitor the OracleAS Guard server.

Because there is no way an OracleAS Guard client nor OPMN on the production site can start OracleAS Guard services on the standby site, OracleAS Guard must be started directly using `opmnctl` on the Infrastructure node in the standby farm. Connect to the node containing the OracleAS Infrastructure database and run the following OPMN command on UNIX systems.

```
> <ORACLE_HOME>/opmn/bin/opmnctl startproc ias-component=DSA
```

On Windows systems, you can issue the following OPMN command to start OracleAS Guard if your Oracle home is located on drive C:.

```
C:\<ORACLE_HOME>\opmn\bin\opmnctl startproc ias-component=DSA
```

Once this OracleAS Guard server is started it is non transient, while the remaining OracleAS Guard servers in the standby farm are transient servers. This configuration allows cross-farm communication.

---

---

**Note:** When you perform an `opmnctl status` command on a system on which OracleAS Guard is running, you will see an `ias-component` and `process-type` named `DSA`. This is the OracleAS component name and server process name for the OracleAS Guard server.

---

---

## 7.4.6 Supported OracleAS Disaster Recovery Configurations

For OracleAS 10g release (10.1.2), OracleAS Guard supports only the default OracleAS Infrastructure configuration supported on Oracle Application Server Cold Failover Cluster and single instance.

## 7.4.7 Supported Oracle Application Server Releases and Operating Systems

OracleAS Guard supports Oracle Application Server releases 10g (9.0.4) and 10g (10.1.2) and all platforms where OracleAS is supported.

## 7.5 OracleAS Guard Operations -- Standby Instantiation and Standby Synchronization

Once you have adhered to the following conditions, you are ready to use the Oracle Application Server Guard for standby instantiation and standby synchronization.

- Met the requirements for the implementation of the OracleAS Disaster Recovery solution as described in Section 7.1.1, "Requirements", Section 7.1.2, "Topology", and Section 7.2, "Preparing the OracleAS Disaster Recovery Environment".
- Installed the OracleAS Disaster Recovery (DR) solution as described in Section 7.3, "Overview of Installing Oracle Application Server 10g Software".



This section describes the following topics:

- Section 7.5.1, "Configuring OracleAS Guard and Other Relevant Information"
- Section 7.5.2, "Standby Instantiation"
- Section 7.5.3, "Standby Synchronization"

See Section 7.8 and Section 7.8.5 for OracleAS Guard command-line asgctl utility tutorial and asgctl reference information.

## 7.5.1 Configuring OracleAS Guard and Other Relevant Information

By default, OracleAS Guard and asgctl, the command-line utility for OracleAS Guard are installed for all install types with the following default configuration information, which includes:

---



---

**Note:** See the OracleAS Guard `readme.txt` file in the `<ORACLE_HOME>\dsa\doc` directory for more information about OracleAS Guard parameters that are configurable.

---



---

- OracleAS Guard command-line utility asgctl is installed in the `<ORACLE_HOME>/dsa/bin` directory on UNIX systems and `<ORACLE_HOME>\dsa\bin` directory on Windows systems on all nodes in the farm production and standby farms.
- OracleAS Guard uses a default port (port) number of 7890; for example, `port=7890`.

---



---

**Note:** If the port number must be changed for some reason (it must be unique for each OracleAS Guard server on a machine, which is automatically handled during installation), you can change its value in the `<ORACLE_HOME>/dsa/dsa.conf` file. Then, stop the OracleAS Guard server (`<ORACLE_HOME>/opmn/bin/opmnctl stopproc ias-component=DSA`) and start the OracleAS Guard server (`<ORACLE_HOME>/opmn/bin/opmnctl startproc ias-component=DSA`) to activate the change. See Section 7.4.5, "OracleAS Guard Integration with OPMN") for more information.

---



---

- OracleAS Guard has a default timeout value (`exec_timeout_secs`) for executing OS commands of 60 seconds; for example, `exec_timeout_secs=60`.
- OracleAS Guard has a default server inactive timeout value (`server_inactive_timeout`) of 600 seconds (10 minutes); for example, `server_inactive_timeout=600`. That is, after 10 minutes of inactivity, OracleAS Guard server will shut down.
- OracleAS Guard is a transient server that is started and stopped by OPMN on each system for the duration of the client session on the production and standby farm.
- Initially, on the production farm, when a OracleAS Guard client connects to a server node, OracleAS Guard server starts and this OracleAS Guard coordinating node instructs OPMN on the remaining nodes in the farm to start OracleAS Guard server on each respective node. However, on the standby farm, an OracleAS Guard server must first be started directly using `opmnctl` on the node containing the OracleAS infrastructure database (see Section 7.4.5, "OracleAS Guard Integration with OPMN" for more information), which when started will

automatically start the remaining OracleAS Guard servers on the remaining nodes in the standby farm. Once the OracleAS Guard server is started on the node containing the OracleAS infrastructure database, it is not transient and will remain running indefinitely, while the remaining OracleAS Guard servers on the remaining nodes in the standby farm are transient.

- The OracleAS Guard operation status information for a farm (from either an `asgctl show operation full` or `show operation history` command) remains available for only the life of the current OracleAS Guard client `asgctl connect` session. When the OracleAS Guard client disconnects from the OracleAS Guard server this farm's operation history information becomes unavailable.
- Once you start an `asgctl` operation, another `asgctl` command cannot be run on the same OracleAS Guard server until the previous command that is running completes or is forced to stop (see the `asgctl stop` operation command for more information.) In addition, you cannot run an `asgctl` operation in background and then quit or exit the `asgctl` utility.

## 7.5.2 Standby Instantiation

The standby instantiation operation performs a number of operations to setup up and maintain a logical mirror of the production site at the standby site. OracleAS Guard is used to coordinate the distributed operations across the production and standby sites to ensure the disaster recovery functionality is enabled. The operations performed are:

- Discovers the Oracle homes that are installed as part of the OracleAS farm at both the production and standby sites.
- Verifies the farm definitions to ensure they comply with the rules of the OracleAS DR environment.
- Configures Oracle Data Guard to maintain the DR environment for the database repository.
- Mirrors the configuration information of all the Oracle homes in the OracleAS farm to the corresponding Oracle home at the standby site.
- Reports any errors found for correction.

The procedure to perform a standby instantiation operation is as follows:

---



---

**Note: For the Windows environment only:** In a DR configuration that uses CFC on the primary farm or standby farm, or both, the following information must be considered before performing an `asgctl` `instantiate farm to`, `switchover farm to`, or `failover` command.

Before taking a cold backup or restoring the metadata repository database, the Oracle Backup and Recovery Tool shuts down the database first. In the Windows CFC environment, Oracle Fail Safe performs database polling and restarts the database if it is down. Hence, every time before the administrator performs an `instantiate`, `switchover`, or `failover` operation, the administrator must disable database polling in Oracle Fail Safe and re-enable it after the backup/restore operation (after the `instantiate`, `switchover`, or `failover` operation completes).

The steps to perform this sequence of operations are as follows:

1. Using Microsoft Cluster Administrator, open the cluster group that contains the Application Server resources. Take the following resources offline in this order: Oracle Process Manager, then Oracle Database, then Oracle Listener.
  2. Using Windows Service Control Manager, start the following services in this order: Fail Safe Listener, then the Oracle Database service.
  3. From a Windows command prompt, use `sqlplus` to startup the database.
  4. Using Windows Service Control Manager, start the Oracle Process Manager.
  5. Perform the `asgctl` commands, including the `instantiate farm to`, `switchover farm to`, or `failover` command.
  6. Using Microsoft Cluster Administrator, open up the cluster group that contains the Application Server resources and bring up the following resources online in this order: Oracle Listener, then Oracle Database, then Oracle Process Manager.
- 
- 

1. Invoke the OracleAS Guard client and connect to the OracleAS Guard server. See Section 7.8.1.1, "Invoking the OracleAS Guard Client and Connecting to the OracleAS Guard Server" for more information.

```
> ASGCTL
ASGCTL > connect asg prodinfra ias_admin/<adminpwd>
Successfully connected to prodinfra:7890
ASGCTL>
```

2. Specify the primary database. See Section 7.8.1.3, "Specifying the Primary Database" for more information.

```
ASGCTL> set primary database sys/testpwd@asdb
```

3. Verify the farm. See Section 7.8.1.4, "Verifying the Farm" for more information. Note that the network hostname is referenced.

```
ASGCTL> verify farm with standbyinfra
```

4. Instantiate the farm at the secondary site. See Section 7.8.1.5, "Instantiating the Farm at the Secondary Site" for more information. Note that the network hostname is referenced.

```
ASGCTL> instantiate farm to standbyinfra
```

5. Disconnect from the OracleAS Guard server and exit the OracleAS Guard client. See Section 7.8.1.7, "Disconnecting from the OracleAS Guard Server and Exiting the OracleAS Guard Client" for more information.

```
ASGCTL> disconnect
ASGCTL> exit
>
```

Note that whenever a standby instantiation is performed using the `asgctl instantiate farm` command that a synchronization operation is also performed, which is why you do not need to perform another synchronization operation immediately following the instantiation operation. Only if a period of time had passed following an instantiate operation would you want to ensure that both the primary and standby sites are transactionally consistent; then you would perform a `sync farm` operation to ensure that if some transaction occurred on the primary site that it is applied to the secondary site.

### 7.5.3 Standby Synchronization

The OracleAS Guard synchronization operation synchronizes the standby site with the primary site to ensure that the two sites are logically consistent. This operation is necessary whenever any of the following circumstances are met:

- Deploy a new application or the re-deployment of an existing application - both the deployment of a new application and the re-deployment of an existing application require changes to schema based information in the metadata repository as well as component configuration information distributed among the Oracle homes in an OracleAS farm. This information has to be uniformly deployed at the standby site.
- Configuration changes - specific changes to a configuration, small to large, require that these configuration changes be reflected at the standby site.
- User Provisioning - The default infrastructure installation maintains the database data for Oracle Internet Directory. As new users are added to the database, they should be synchronized to the standby site to a schedule that fulfills the business availability needs.
- Periodic full synchronization - By default, the synchronization operations only synchronizes the pieces of configuration that have changed since the last synchronization operation. During test cycles or occasional complex configuration changes, administrators may desire full refreshes of their configuration information to the standby site to ensure mirroring of these changes.

As part of the synchronization operation a `verify` operation is performed to ensure the required DR environment is maintained. Additionally, if new OracleAS instances are installed into the OracleAS farm, OracleAS Guard will discover these installations.

The procedure to perform standby synchronization is as follows:

1. Invoke the OracleAS Guard client and connect to the OracleAS Guard server. See Section 7.8.1.1, "Invoking the OracleAS Guard Client and Connecting to the OracleAS Guard Server" for more information.

```
> asgctl.sh
Application Server Guard: Release 10.1.2.0.0

(c) Copyright 2004 Oracle Corporation. All rights reserved
ASGCTL > connect asg prodinfra ias_admin/<adminpwd>
Successfully connected to prodinfra:7890
ASGCTL>
```

2. Specify the primary database. See Section 7.8.1.3, "Specifying the Primary Database" for more information.

```
ASGCTL> set primary database sys/testpwd@asdb
```

3. Synchronize the secondary site with the primary site. See Section 7.8.1.6, "Synchronizing the Secondary Site with the Primary Site" for more information.

```
ASGCTL> sync farm to standbyinfra
```

4. Disconnect from the OracleAS Guard server and exit the OracleAS Guard client. See Section 7.8.1.7, "Disconnecting from the OracleAS Guard Server and Exiting the OracleAS Guard Client" for more information.

```
ASGCTL> disconnect
```

```
ASGCTL> exit
```

```
>
```

Note that you should always perform an ongoing sync farm operation whenever any of the previously mentioned circumstances are met.

## 7.6 Runtime Operations -- OracleAS Guard Switchover and Failover Operations

Runtime operations include dealing with outages, whether they are scheduled or unscheduled (see Section 7.6.1, "Outages"), and monitoring ongoing OracleAS Guard operations using the asgctl command-line utility and troubleshooting (see Section 7.6.2, "Monitoring OracleAS Guard Operations and Troubleshooting").

### 7.6.1 Outages

Outages fall into two categories:

- Section 7.6.1.1, "Scheduled Outages"
- Section 7.6.1.2, "Unplanned Outages"

#### 7.6.1.1 Scheduled Outages

Scheduled outages are planned outages. They are required for regular maintenance of the technology infrastructure supporting the business applications and include tasks such as hardware maintenance, repair and upgrades, software upgrades and patching, application changes and patching, and changes to improve performance and manageability of systems. Scheduled outages can occur either for the production or standby site. Descriptions of scheduled outages that impact the production or standby site are:

- Site-wide maintenance

The entire site where the current production resides is unavailable. Examples of site-wide maintenance are scheduled power outages, site maintenance, and regularly planned switchover operations.

- OracleAS Cold Failover Cluster cluster-wide maintenance

This is scheduled downtime of the OracleAS Cold Failover Cluster for hardware maintenance. The scope of this downtime is the whole hardware cluster. Examples of cluster-wide maintenance are repair of the cluster interconnect and upgrade of the cluster management software.

- Testing and validating the standby site as a means to test OracleAS Disaster Recovery readiness.

For scheduled outages, a site switchover has to be performed, which is explained in Section 7.6.1.1.1, "Site Switchover Operations".

**7.6.1.1.1 Site Switchover Operations** A site switchover is performed for planned outages of the production site. Both the production and standby sites have to be available during the switchover. The application of the database redo logs is synchronized to match the backup and restoration of the configuration files for the middle-tier and OracleAS Infrastructure installations.

During site switchover, considerations to avoid long periods of cached DNS information have to be made. Modifications to the site's DNS information, specifically time-to-live (TTL), have to be performed. Refer to Section 7.7.2, "Manually Changing DNS Names" for instructions.

---

---

**Note:** All sessions to the production and standby databases need to be closed in order to perform the switchover operation. The switchover operation requires that all middle-tier and OracleAS Infrastructure instances be shut down. Also, the CJQO and QMNO database processes have sessions that need to be stopped.

Just prior to performing the switchover operation, only OPMN is running on all systems in both production and standby farms and OracleAS Guard server is explicitly started on the Infrastructure system of the standby farm. (When an opmnctl status command is performed on a system on which OracleAS Guard server is running, you will see an ias-component and process-type named DSA, which is the running OracleAS Guard server.) See Section 7.4.5, "OracleAS Guard Integration with OPMN" for more information.

Prior to performing the switchover operation, on the production farm, if there are services outside of the OPMN controlled services domain (the OPMN startall domain), such as emagent (see Step 2 that follows), user applications, or the DB job system for example, they must be manually managed (stopped) by the administrator depending on the operations.

Prior to performing the switchover operation, on the standby farm, all systems must be available on the network and basically nothing can be running other than OPMN and OracleAS Guard server.

After the OracleAS Guard asgctl switchover operation completes, any services outside of the OPMN controlled services domain (the OPMN startall domain) and user applications that are not started by default, must be manually managed (started) on this new production farm.

---

---

---

**Note: For the Windows environment only:** In a DR configuration that uses CFC on the primary farm or standby farm or both, the following information must be considered before performing an `asgctl` instantiate farm to, switchover farm to, or failover command.

Before taking a cold backup or restoring the metadata repository database, the Oracle Backup and Recovery Tool shuts down the database first. In the Windows CFC environment, Oracle Fail Safe performs database polling and restarts the database if it is down. Hence, every time before the administrator performs an instantiate, switchover, or failover operation, the administrator must disable database polling in Oracle Fail Safe and re-enable it after the backup/restore operation (after the instantiate, switchover, or failover operation completes).

The steps to perform this sequence of operations are as follows:

1. Using Microsoft Cluster Administrator, open the cluster group that contains the Application Server resources. Take the following resources offline in this order: Oracle Process Manager, then Oracle Database, then Oracle Listener.
  2. Using Windows Service Control Manager, start the following services in this order: Fail Safe Listener, then the Oracle Database service.
  3. From a Windows command prompt, use `sqlplus` to startup the database.
  4. Using Windows Service Control Manager, start the Oracle Process Manager and also shut down the OracleDBConsole service on the primary node.
  5. Perform the `asgctl` commands, including the instantiate farm to, switchover farm to, or failover command.
  6. Using Microsoft Cluster Administrator, open up the cluster group that contains the Application Server resources and bring up the following resources online in this order: Oracle Listener, then Oracle Database, then Oracle Process Manager.
- 

To switchover from the production site to the standby site, perform the following steps:

1. Reduce the wide area DNS TTL value for the site. See Section 7.7.2, "Manually Changing DNS Names" for more information.
2. On the primary infrastructure system, make sure the emagent process is stopped. Otherwise, you may run into the following error when doing a switchover operation because the emagent has a connection to the database:

```
prodinfra: -->ASG_DGA-13051: Error performing a physical standby switchover.
prodinfra: -->ASG_DGA-13052: The primary database is not in the proper state to
perform a switchover. State is "SESSIONS ACTIVE"
```

On UNIX systems, to stop the emagent process, stop the Application Server Control, which is called `iasconsole`, as follows:

```
> <ORACLE_HOME>/bin/emctl stop iasconsole
```

On UNIX systems, to check to see if there is an emagent process running, do the following:

```
> ps -ef | grep emagent
```

On UNIX systems, if after performing the stop iasconsole operation, the emagent process is still running, get its process ID (PID) as determined from the previous ps command and stop it as follows:

```
> kill -9 <emagent-pid>
```

On Windows systems, open the Services control panel. Locate the OracleAS10gASControl service and stop this service.

3. Invoke the OracleAS Guard client command-line utility asgctl (on UNIX systems, asgctl.sh is located in <ORACLE\_HOME>/dsa/bin and on Windows systems, asgctl.bat is located in <ORACLE\_HOME>\dsa\bin.) and connect to the OracleAS Guard server.

```
> asgctl.sh
Application Server Guard: Release 10.1.2.0.0
```

```
(c) Copyright 2004 Oracle Corporation. All rights reserved
ASGCTL> connect asg prodinfra ias_admin/<adminpwd>
```

4. Specify the primary database.
5. Switchover the farm to the secondary site.

```
ASGCTL> set primary database sys/testpwd@asdb
```

```
ASGCTL> switchover farm to standbyinfra
```

---

---

**Note:** As part of the OracleAS Guard switchover operation, an implicit sync farm operation is performed to make sure the farms are identical. In addition, OPMN automatically starts the OracleAS Guard server on the "new" standby Infrastructure node and this server will run indefinitely, and in turn, starts the OracleAS Guard server on the other nodes in the "new" standby farm and each of these is a transient server.

---

---

6. Disconnect from the "old" primary site OracleAS Guard server.
7. Now you can perform some other OracleAS Guard operations, such as a verify farm operation or a dump farm operation, though neither operation is required to be run.
8. Perform a wide area DNS switchover to direct requests to the new production site based on one of the options presented in Section 7.7, "Wide Area DNS Operations".
9. Adjust the wide area DNS TTL to an appropriate value.

### 7.6.1.2 Unplanned Outages

An unplanned outage that impacts either or both the production and standby sites can be one of the following:

- Site-wide failure. The entire site where the current production resides is unavailable. Examples of site-wide outages are disasters at the production or standby site such as fire, flood, earthquake, or power outages.



---

---

**Note:** A site-wide or OracleAS Cold Failover Cluster cluster-wide unplanned outage on the standby site does impact availability; a production database outage is required to restore the standby outage.

---

---

- Complete failure of the middle tier. The entire middle tier is not available. Either all nodes are down or the application server instances on all nodes are down. The last surviving node of the Oracle Application Server 10g Farm for a service is no longer available.
- OracleAS Cold Failover Cluster cluster-wide failure. The entire hardware cluster hosting the OracleAS Infrastructure is unavailable or crashes. This includes failure of nodes in the cluster as well as any other components that results in the hardware cluster and the OracleAS Infrastructure on this site not being available.

Unplanned outages warrant the failover of the production site to the standby site.

**7.6.1.2.1 Site Failover Operations** A site failover is performed for unplanned outages for the production site. Failover operations require the restoration of the configuration and Infrastructure data to a consistent point in time. OracleAS Guard ensures that the site services are brought up in a consistent fashion to the point of the last sync operation.

To failover the production site to the standby site:

---

---

**Note: For the Windows environment only:** In a DR configuration that uses CFC on the primary farm or standby farm or both, the following information must be considered before performing an `asgctl` `instantiate farm to`, `switchover farm to`, or `failover` command.

Before taking a cold backup or restoring the metadata repository database, the Oracle Backup and Recovery Tool shuts down the database first. In the Windows CFC environment, Oracle Fail Safe performs database polling and restarts the database if it is down. Hence, every time before the administrator performs an `instantiate`, `switchover`, or `failover` operation, the administrator must disable database polling in Oracle Fail Safe and re-enable it after the backup/restore operation (after the `instantiate`, `switchover`, or `failover` operation completes).

The steps to perform this sequence of operations are as follows:

1. Using Microsoft Cluster Administrator, open the cluster group that contains the Application Server resources. Take the following resources offline in this order: Oracle Process Manager, then Oracle Database, then Oracle Listener.
  2. Using Windows Service Control Manager, start the following services in this order: Fail Safe Listener, then the Oracle Database service.
  3. From a Windows command prompt, use `sqlplus` to startup the database.
  4. Using Windows Service Control Manager, start the Oracle Process Manager.
  5. Perform the `asgctl` commands, including the `instantiate farm to`, `switchover farm to`, or `failover` command.
  6. Using Microsoft Cluster Administrator, open up the cluster group that contains the Application Server resources and bring up the following resources online in this order: Oracle Listener, then Oracle Database, then Oracle Process Manager.
- 
- 

1. If you are using `asgctl` to continually synchronize the secondary (standby) site with the primary site, then both sites should already be synchronized and you can continue to Step 2.

---

---

**Note:** If you have not performed a recent sync operation prior to the loss of the production farm, a failover operation will still failover to the standby farm. The failover operation will make this standby farm the new production farm with the last sync information that was available.

If you are not using OracleAS Guard `asgctl` commands at all to ensure that the secondary (standby) site is synchronized with the primary site, you must perform regular backup operations of the primary site middle-tier and OracleAS Infrastructure configuration files as described in Section D.1.1, "Manually Backing Up the Production Site", then you will need to restore the backup configuration files as described in Section D.1.2, "Manually Restoring to Standby Site". After restoring the configuration files (OracleAS Infrastructure and Middle Tier) on the standby site, then proceed to Step 2.

---

---

2. Invoke the OracleAS Guard client command-line utility asgctl (on UNIX systems, asgctl.sh is located in <ORACLE\_HOME>/dsa/bin and on Windows systems, asgctl.bat is located in <ORACLE\_HOME>\dsa\bin.) and connect to the OracleAS Guard server on the standby site. Note that standbyinfra is the network name.

```
> asgctl.sh
Application Server Guard: Release 10.1.2.0.0

(c) Copyright 2004 Oracle Corporation. All rights reserved
ASGCTL> connect asg standbyinfra ias_admin/<adminpwd>
Successfully connected to stanfbyinfra:7890
```

3. Specify the primary database.

```
ASGCTL> set primary database sys/testpwd@asdb
```

4. Specify that the primary database on the standby site is now identified as the **new** primary database on this **new** production site. The keyword **new** is shown as bold text in the following example to indicate its importance as a key word.

```
ASGCTL> set new primary database sys/testpwd@asdb
```

5. Perform an asgctl failover operation.

```
ASGCTL> failover
```

---

---

**Note:** As part of the OracleAS Guard failover operation, OPMN automatically starts the OracleAS Guard server on the "new" standby Infrastructure node and this server will run indefinitely, and in turn, starts the OracleAS Guard server on the other nodes in the "new" standby farm and each of these is a transient server.

---

---

6. To maintain disaster failover capability, a new standby site should be set up and instantiated.
7. Specify the primary database prior to doing an instantiate operation.

```
ASGCTL> set primary database sys/testpwd@asdb
```
8. Instantiate the farm at the secondary site. Note that an instantiate operation also implicitly performs a sync operation.

---

---

**Note: For the Windows environment only:** In a DR configuration that uses CFC on the primary farm or standby farm or both, the following information must be considered before performing an `asgctl` `instantiate farm to`, `switchover farm to`, or `failover` command.

Before taking a cold backup or restoring the metadata repository database, the Oracle Backup and Recovery Tool shuts down the database first. In the Windows CFC environment, Oracle Fail Safe performs database polling and restarts the database if it is down. Hence, every time before the administrator performs an `instantiate`, `switchover`, or `failover` operation, the administrator must disable database polling in Oracle Fail Safe and re-enable it after the backup/restore operation (after the `instantiate`, `switchover`, or `failover` operation completes).

The steps to perform this sequence of operations are as follows:

1. Using Microsoft Cluster Administrator, open the cluster group that contains the Application Server resources. Take the following resources offline in this order: Oracle Process Manager, then Oracle Database, then Oracle Listener.
  2. Using Windows Service Control Manager, start the following services in this order: Fail Safe Listener, then the Oracle Database service.
  3. From a Windows command prompt, use `sqlplus` to startup the database.
  4. Using Windows Service Control Manager, start the Oracle Process Manager.
  5. Perform the `asgctl` commands, including the `instantiate farm to`, `switchover farm to`, or `failover` command.
  6. Using Microsoft Cluster Administrator, open up the cluster group that contains the Application Server resources and bring up the following resources online in this order: Oracle Listener, then Oracle Database, then Oracle Process Manager.
- 
- 

```
ASGCTL> instantiate farm to standbyinfra
```

9. Disconnect from the "new" primary site OracleAS Guard server, and exit the OracleAS Guard client.

```
ASGCTL> disconnect
ASGCTL> exit
>
```

10. Perform a wide area DNS switchover to direct requests to the new production site based on one of the options presented in Section 7.7, "Wide Area DNS Operations".

## 7.6.2 Monitoring OracleAS Guard Operations and Troubleshooting

Once your DR solution is set up, and the standby farm is instantiated, and the production and standby farms are synchronized, you can use the OracleAS Guard client command-line utility `asgctl` to issue commands through the coordinating OracleAS Guard server to monitor `asgctl` operations and perform some troubleshooting tasks. A typical OracleAS Guard monitoring or troubleshooting session may involve the following tasks:

1. Section 7.6.2.1, "Verifying the Farm"
2. Section 7.6.2.2, "Displaying the Current Operation"

3. Section 7.6.2.3, "Displaying a List of Recent Operations"
4. Section 7.6.2.4, "Stopping an Operation"
5. Section 7.6.2.5, "Tracing Tasks"
6. Section 7.6.2.6, "Writing Information About the Farm to a File"
7. Section 7.6.2.7, "Shutting Down a Farm with a Problem"
8. Section 7.6.2.8, "Starting Up a Shutdown Farm"

As `asgctl` commands are issued through the OracleAS Guard client and requests are then made to the coordinating OracleAS Guard server, the coordinating OracleAS Guard server pushes these requests out to the other OracleAS Guard servers in the production and standby farms, status messages are returned to the OracleAS Guard client as well as any error messages should a particular task encounter a problem. Section 7.6.2.9, "Error Messages" describes where you can obtain more information about these error messages.

### 7.6.2.1 Verifying the Farm

To validate that the primary farm is running and the configuration is valid, enter the following `asgctl` command at the `asgctl` prompt.

```
ASGCTL> connect asg ias_admin/iactest2
Successfully connected to prodinfra:7890
ASGCTL> verify farm
prodinfra(asr1012): Verifying each instance in the farm
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
prodinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
ASGCTL>
```

To compare a primary farm to which the local host is a member with a standby farm to ensure they are consistent with one another and that both farms conform to DR requirements, enter the following `asgctl` command at the `asgctl` prompt and specify the name of the standby host system.

```
ASGCTL> verify farm with standbyinfra
prodinfra(asr1012): Verifying each instance in the farm
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
prodinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
standbyinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
prodinfra(asr1012): Verifying farm IasFarm is symmetrical in both primary and standby configuration.
prodinfra(asr1012): Verifying instance asmid2.midtier.us.oracle.com is symmetrical in both primary and standby configuration.
prodinfra(asr1012): Host names are symmetrical for instance asmid2.midtier.us.oracle.com.
prodinfra(asr1012): Home name and path are symmetrical for instance asmid2.midtier.us.oracle.com.
prodinfra(asr1012): Verifying instance asmid1.midtier.us.oracle.com is symmetrical in both primary and standby configuration.
prodinfra(asr1012): Host names are symmetrical for instance asmid1.midtier.us.oracle.com.
prodinfra(asr1012): Home name and path are symmetrical for instance asmid1.midtier.us.oracle.com.
prodinfra(asr1012): Verifying instance asr1012.infra.us.oracle.com is symmetrical in both primary and standby configuration.
prodinfra(asr1012): Host names are symmetrical for instance asr1012.infra.us.oracle.com.
prodinfra(asr1012): Home name and path are symmetrical for instance asr1012.infra.us.oracle.com.
ASGCTL>
```

### 7.6.2.2 Displaying the Current Operation

To display the status of all the current operations running on all nodes of the farm to which the OracleAS Guard client is connected, enter the following asgctl command at the asgctl prompt:

```
ASGCTL> show operation full
*****
OPERATION: 15
  Status: running
  Elapsed Time: 0 days, 0 hours, 1 minutes, 35 secs
  TASK: instantiateFarm
      TASK: verifyFarm
```

### 7.6.2.3 Displaying a List of Recent Operations

To display only operations that are not running on all nodes of the farm to which the OracleAS Guard client is connected, enter the following asgctl command at the asgctl prompt:

```
ASGCTL> show operation history
*****
OPERATION: 69
  Status: success
  Elapsed Time: 0 days, 0 hours, 0 minutes, 7 secs

  TASK: getFarm
*****
OPERATION: 70
  Status: success
  Elapsed Time: 0 days, 0 hours, 6 minutes, 49 secs

  TASK: syncFarm
      TASK: backupFarm
          TASK: fileCopyRemote
          TASK: fileCopyRemote
          TASK: fileCopyRemote
      TASK: resolveHost
      TASK: restoreFarm
          TASK: fileCopyLocal
          TASK: fileCopyLocal
          TASK: fileCopyLocal
  .
  .
  .
```

### 7.6.2.4 Stopping an Operation

To stop a specific operation that is running on the server, enter the following asgctl command at the asgctl prompt and specify the operation number you want to stop. You can obtain the operation number you want to stop by entering a asgctl show operation full command.

```
ASGCTL> show operation full
*****
OPERATION: 15
  Status: running
  Elapsed Time: 0 days, 0 hours, 1 minutes, 35 secs
  TASK: instantiateFarm
      TASK: verifyFarm
ASGCTL> stop operation 15
```

### 7.6.2.5 Tracing Tasks

To set a trace flag for a specific event and to log the output to the asgctl log files, enter the following asgctl command at the asgctl prompt and specify the "on" keyword and enter the traceflags to be enabled. In this case, the traceflags DB indicates that trace information regarding processing in the Oracle Database environment will be displayed. See the set trace command for more information about other traceflags that can be enabled. See the set trace command for a complete list of the traceflags that can be set.

```
ASGCTL> set trace on db
```

### 7.6.2.6 Writing Information About the Farm to a File

To write detailed information about the farm to which the local host is connected, enter the following asgctl command at the asgctl prompt and specify the path name and file name where the detailed output is to be written. The output is the same as that displays shown in the dump farm command, except it is written to a file so you can save a record of it.

```
ASGCTL> dump farm to c: \dump_mid_1.txt
```

### 7.6.2.7 Shutting Down a Farm with a Problem

Occasionally, you may run into a problem with a member of the farm and the farm must be shut down and then started up again to fix the problem. To perform this task on a problem farm, enter the following asgctl commands at the asgctl prompt:

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> shutdown farm
prodinfra(asr1012): Shutting down each instance in the farm
prodinfra(asm21012): Shutting down component wireless
prodinfra(asm1012): Shutting down component WebCache
prodinfra(asm1012): Shutting down component OC4J
prodinfra(asm21012): Shutting down component WebCache
prodinfra(asm1012): Shutting down component dcm-daemon
prodinfra(asm21012): Shutting down component OC4J
prodinfra(asm1012): Shutting down component LogLoader
prodinfra(asm1012): Shutting down component HTTP_Server
prodinfra(asm21012): Shutting down component dcm-daemon
prodinfra(asm21012): Shutting down component LogLoader
prodinfra(asm21012): Shutting down component HTTP_Server
prodinfra(asr1012): Shutting down component OID
prodinfra(asr1012): Shutting down component OC4J
prodinfra(asr1012): Shutting down component dcm-daemon
prodinfra(asr1012): Shutting down component HTTP_Server
ASGCTL>
```

### 7.6.2.8 Starting Up a Shutdown Farm

Occasionally, you may run into a problem with a member of the farm and the farm must be shut down and then started up again to fix the problem. To perform this task on a shutdown farm, enter the following asgctl command at the asgctl prompt:

```
ASGCTL> startup farm
```

```
prodinfra(asr1012): Starting each instance in the farm
prodinfra(asr1012): Executing opmnctl startall command.
prodinfra(asm1012): Executing opmnctl startall command.
prodinfra(asm21012): Executing opmnctl startall command.
ASGCTL>
```

### 7.6.2.9 Error Messages

Appendix C, "OracleAS Guard Error Messages" categorizes and describes the error messages that may appear while using the OracleAS Disaster Recovery solution.

## 7.7 Wide Area DNS Operations

In order for client requests to be directed to the entry point of a production site, DNS resolution is used. When a site switchover or failover is performed, client requests have to be redirected transparently to the new site playing the production role. To accomplish this redirection, the wide area DNS that resolves requests to the production site has to be switched over to the standby site. The DNS switchover can be accomplished in one of the following two ways:

---

---

**Note:** A hardware load balancer is assumed to be front-ending each site. Check <http://metalink.oracle.com> for supported load balancers.

---

---

- Section 7.7.1, "Using a Wide Area Load Balancer"
- Section 7.7.2, "Manually Changing DNS Names"

### 7.7.1 Using a Wide Area Load Balancer

When a wide area load balancer (global traffic manager) is deployed in front of the production and standby sites, it provides fault detection services and performance-based routing redirection for the two sites. Additionally, the load balancer can provide authoritative DNS name server equivalent capabilities.

During normal operations, the wide area load balancer can be configured with the production site's load balancer name-to-IP mapping. When a DNS switchover is required, this mapping in the wide area load balancer is changed to map to the standby site's load balancer IP. This allows requests to be directed to the standby site, which should have been brought up and now has the production role.

This method of DNS switchover works for both site switchover and failover. One advantage of using a wide area load balancer is that the time for a new name-to-IP mapping to take effect can be almost immediate. The downside is that an additional investment needs to be made for the wide area load balancer.

### 7.7.2 Manually Changing DNS Names

This method of DNS switchover involves the manual change of the name-to-IP mapping that is originally mapped to the IP address of the production site's load balancer. The mapping is changed to map to the IP address of the standby site's load balancer. Follow these instructions to perform the task:

1. Note the current time-to-live (TTL) value of the production site's load balancer mapping. This mapping is in the DNS cache and will be there until the TTL



expires. For the purposes of discussion and providing an example, let's assume this TTL to be 3600 seconds.

2. Modify the TTL value to a short interval. For example, 60 seconds.
3. Wait one interval of the original TTL. This is the original TTL of 3600 seconds that we noted in step 1.
4. Ensure that the standby site is switched over to receive requests.
5. Modify the DNS mapping to resolve to the standby site's load balancer giving it the appropriate TTL value for normal operation (for example, 3600 seconds).

This method of DNS switchover works for planned site switchover operations only. The TTL value set in step 2 should be a reasonable time period where client requests cannot be fulfilled. The modification of the TTL is effectively modifying the caching semantics of the address resolution from a long period of time to a short period. Due to the shortened caching period, an increase in DNS requests can be observed.

## 7.8 Using OracleAS Guard Command-Line Utility (asgctl)

This section includes the following topics:

- Section 7.8.1, "Typical OracleAS Guard Session Using asgctl"
- Section 7.8.2, "Periodic Scheduling of OracleAS Guard asgctl Scripts"
- Section 7.8.3, "Submitting OracleAS Guard Jobs to the Enterprise Manager Job System"
- Section 7.8.4, "Specifying Different Credentials for Two or More Nodes"
- Section 7.8.5, "Reference Section: OracleAS Guard asgctl Command-line Commands"

### 7.8.1 Typical OracleAS Guard Session Using asgctl

A typical OracleAS Guard session using asgctl involves the following tasks:

1. Section 7.8.1.1, "Invoking the OracleAS Guard Client and Connecting to the OracleAS Guard Server"
2. Section 7.8.1.2, "Getting Help"
3. Section 7.8.1.3, "Specifying the Primary Database"
4. Section 7.6.2.1, "Verifying the Farm"
5. Section 7.8.1.5, "Instantiating the Farm at the Secondary Site"
6. Section 7.8.1.6, "Synchronizing the Secondary Site with the Primary Site"
7. Section 7.8.1.7, "Disconnecting from the OracleAS Guard Server and Exiting the OracleAS Guard Client"

One of the advantages of supporting an asgctl command-line interface is that you can place these asgctl commands in a proper sequence in a script as described in Section 7.8.1.8, "Creating and Executing an asgctl Script" and then execute the script as described in Section 7.8.2, "Periodic Scheduling of OracleAS Guard asgctl Scripts" and Section 7.8.3, "Submitting OracleAS Guard Jobs to the Enterprise Manager Job System".

### 7.8.1.1 Invoking the OracleAS Guard Client and Connecting to the OracleAS Guard Server

To invoke OracleAS Guard client command-line utility `asgctl`, enter the following command at the operating system command-line prompt (on UNIX systems, `asgctl.sh` is located in `<ORACLE_HOME>/dsa/bin` and on Windows systems, `asgctl.bat` is located in `<ORACLE_HOME>\dsa\bin`).

```
> asgctl.sh
```

```
Application Server Guard: Release 10.1.2.0.0
```

```
(c) Copyright 2004 Oracle Corporation. All rights reserved
```

Then enter the following `asgctl` command at the `asgctl` command-line prompt specifying the node name (in this case, `prodinfra`) of the host system for the OracleAS Guard server to which you want the OracleAS Guard client to connect and specify the `ias_admin` account name and password for the `ias_admin` account created during the Oracle Application Server installation:

```
ASGCTL > connect asg prodinfra ias_admin/<adminpwd>
Successfully connected to prodinfra:7890
ASGCTL>
```

### 7.8.1.2 Getting Help

To get help on a particular command, enter the `asgctl` command at the `asgctl` prompt specifying the command name you for which you want help information; otherwise to get help on all commands, enter the following `asgctl` command at the `asgctl` prompt:

```
ASGCTL> help
  connect asg [<host>] [<ias_admin/<password>]
  disconnect
  exit
  quit
  dump farm [to <file>]
  help [<command>]
  instantiate farm to <standby_farm_host>
  set asg credentials <host> ias_admin/<password> [for farm]
  set trace on|off <traceflags>
  set primary database <username>/<password>@<servicename> [pfile <filename> | spfile
<filename>]
  set new primary database <username>/<password>@<servicename> [pfile <filename> | spfile
<filename>]
  sync farm to <standby_farm_host> [<sync_mode>]
  startup farm
  shutdown farm
  show op[eration] [full] [[his]tory]
  stop op[eration] <op#>
  verify farm [with <host>]
  switchover farm to <standby_farm_host>
  failover
ASGCTL>
```

### 7.8.1.3 Specifying the Primary Database

To identify the OracleAS Infrastructure database on the primary farm, enter the following `asgctl` command at the `asgctl` prompt specifying the user name and password for the database account with `sysdba` privileges to access the OracleAS Infrastructure database and the TNS service name of the OracleAS Infrastructure database:

```
ASGCTL> set primary database sys/testpwd@asdb
ASGCTL>
```

The standby site uses the same values as specified for the primary database because the service name and password for both the primary and standby OracleAS Infrastructure Databases must be the same. Note that you must always set the primary database before performing an instantiate, sync, or switchover operation.

#### 7.8.1.4 Verifying the Farm

To validate that the primary farm is running and the configuration is valid, enter the following asgctl command at the asgctl prompt specifying the name of the standby host system that is a member of the standby farm (see Section 7.6.2.1, "Verifying the Farm" for an example with output):

```
ASGCTL> verify farm with standbyinfra
.
.
.
```

To display more detailed information about the farm to which the local host is connected as another way of verifying the farm, enter the following asgctl command at the asgctl prompt.

```
ASGCTL> dump farm
prodinfra(asr1012): Describing each instance in the farm
Dump Farm
  Name: IasFarm
  Instance: asmid2.midtier.us.oracle.com
  Type: Core
.
.
.
  Asg Server Port: 7890
  Instance: asmid.midtier.us.oracle.com
  Type: Core
  Oracle Home Name: asmid1
.
.
.
  Asg Server Port: 7891
  Instance: asr1012.infra.us.oracle.com
  Type: Infrastructure
.
.
.
  Asg Server Port: 7890
```

#### 7.8.1.5 Instantiating the Farm at the Secondary Site

To instantiate a farm to a standby site, enter the following asgctl command at the asgctl prompt specifying the name of the standby host system that is a member of the standby farm. Note that part way through the operation you will be prompted to answer a question regarding whether you want to shut down the database. Reply by entering *y* or *yes*.

---

---

**Note: For the Windows environment only:** In a DR configuration that uses CFC on the primary farm or standby farm or both, the following information must be considered before performing an asgctl instantiate farm to, switchover farm to, or failover command.

Before taking a cold backup or restoring the metadata repository database, the Oracle Backup and Recovery Tool shuts down the database first. In the Windows CFC environment, Oracle Fail Safe performs database polling and restarts the database if it is down. Hence, every time before the administrator performs an instantiate, switchover, or failover operation, the administrator must disable database polling in Oracle Fail Safe and re-enable it after the backup/restore operation (after the instantiate, switchover, or failover operation completes).

The steps to perform this sequence of operations are as follows:

1. Using Microsoft Cluster Administrator, open the cluster group that contains the Application Server resources. Take the following resources offline in this order: Oracle Process Manager, then Oracle Database, then Oracle Listener.
  2. Using Windows Service Control Manager, start the following services in this order: Fail Safe Listener, then the Oracle Database service.
  3. From a Windows command prompt, use sqlplus to startup the database.
  4. Using Windows Service Control Manager, start the Oracle Process Manager.
  5. Perform the asgctl commands, including the instantiate farm to, switchover farm to, or failover command.
  6. Using Microsoft Cluster Administrator, open up the cluster group that contains the Application Server resources and bring up the following resources online in this order: Oracle Listener, then Oracle Database, then Oracle Process Manager.
- 
- 

```
ASGCTL> instantiate farm to standbyinfra
prodinfra(asr1012): Instantiating each instance in the farm to standby farm
prodinfra(asr1012): Verifying each instance in the farm
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
prodinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
.
.
.
This operation requires the database to be shutdown. Do you want to continue? Yes or No
Y
.
.
.
standbyinfra(asr1012): Starting restore/synchronization of database "asdb.us.oracle.com".
standbyinfra(asr1012): Shutting down the standby database.
standbyinfra(asr1012): Starting the standby database.
standbyinfra(asr1012): Synchronizing farm completed successfully.
prodinfra(asr1012): Synchronizing farm completed successfully.
ASGCTL>
```

A synchronization operation is also performed as part of the instantiation operation.

### 7.8.1.6 Synchronizing the Secondary Site with the Primary Site

To synchronize the standby site with the primary site, enter the following asgctl command at the asgctl prompt specifying the name of the standby site host system and optionally the sync mode.

```
ASGCTL> sync farm to standbyinfra
prodinfra(asr1012): Synchronizing each instance in the farm to standby farm
prodinfra(asr1012): Starting backup of farm "IasFarm".
prodinfra(asr1012): Loading farm information.
prodinfra(asr1012): Backing up and copying data to the standby farm.
prodinfra(asr1012): Backing up each instance in the farm
prodinfra(asr1012): Starting backup of instance "asr1012.infra.us.oracle.com".
prodinfra(asr1012): Executing config_nodb option in bkp_restore.pl script.
midtier(asmid2): Starting backup of instance "asmid2.midtier.us.oracle.com".
midtier(asmid2): Executing config_nodb option in bkp_restore.pl script.
midtier(asmid1): Starting backup of instance "asmid1.midtier.us.oracle.com".
midtier(asmid1): Executing config_nodb option in bkp_restore.pl script.
prodinfra(asr1012): Executing backup_config option in bkp_restore.pl script.
prodinfra(asr1012): Deleted directory "/private1/OraHome/dsa/backup".
midtier(asmid2): Executing backup_config option in bkp_restore.pl script.
midtier(asmid2): Deleted directory "/private1/OraHome2/dsa/backup".
midtier(asmid1): Executing backup_config option in bkp_restore.pl script.
midtier(asmid1): Deleted directory "/private1/OraHome/dsa/backup".
.
.
.
prodinfra(asr1012): Starting backup/synchronization of database "asdb.us.oracle.com".
midtier(asmid2): Synchronizing farm completed successfully.
midtier(asmid1): Synchronizing farm completed successfully.
midtier(asmid1): Synchronizing farm completed successfully.
midtier(asmid2): Synchronizing farm completed successfully.
standbyinfra(asr1012): Starting restore/synchronization of database "asdb.us.oracle.com".
standbyinfra(asr1012): Shutting down the standby database.
standbyinfra(asr1012): Starting the standby database.
standbyinfra(asr1012): Synchronizing farm completed successfully.
ASGCTL>
```

Even though the instantiation operation in the previous step also performs a synchronization operation, this step showing the synchronization operation is a way to ensure that both the standby site and primary site are synchronized.

Note that you can specify a `sync_mode` parameter as being one of two values, incremental or full. By default `sync_mode` is incremental and offers the best performance. However, there may be three circumstances when `sync_mode` of full should be specified.

- When you want to force a full synchronization to happen for some reason, such as synchronizing the standby site completely with the primary site.
- When you know there are many transactional changes over a short period of time on the primary site that must be synchronized with the secondary site.
- When you know that there are a large accumulation of transactional changes over a long period of time on the primary site that must be synchronized with the secondary site.

See the `sync farm to` command for more information about the `sync_mode` parameter.

### 7.8.1.7 Disconnecting from the OracleAS Guard Server and Exiting the OracleAS Guard Client

To disconnect the OracleAS Guard client from the OracleAS Guard server to which it is currently connected, and then exiting the OracleAS Guard client, enter the following asgctl commands at the asgctl prompt:

```
ASGCTL> disconnect
ASGCTL> exit
>
```

### 7.8.1.8 Creating and Executing an asgctl Script

To create a script containing a sequence of asgctl command names and their arguments, open an edit session with your favorite editor, enter the asgctl commands in proper sequence according to the operations you want to perform, save the script file, then execute the script when you invoke asgctl as shown in the following command:

```
> ASGCTL @myasgctlscript.txt
```

See the set echo command for an example of a script containing a series of asgctl commands.

## 7.8.2 Periodic Scheduling of OracleAS Guard asgctl Scripts

For OracleAS Guard operations that you want to run periodically, such as a periodic sync farm operation to keep the standby farm synchronized with the primary farm, you can automate the periodic running of an OracleAS Guard asgctl script.

On UNIX systems, you can set up a cron job to run the asgctl script. Copy your asgctl script into the appropriate /etc subdirectory `cron.hourly`, `cron.daily`, `cron.weekly`, or `cron.monthly`. It will run either hourly, daily, weekly, or monthly, depending on the name of the subdirectory in which you choose to place your script. Or you can edit a crontab and create an entry that will be specific for the time on which you want to run the asgctl script. See the manpage on cron and crontab for more information.

On Windows systems, you can use the task scheduler or scheduled tasks from the control panel to choose the time to run the asgctl script, daily, weekly, or monthly or at certain times. You can also purchase additional scheduler software from a third party with more options and set the time and frequency to run the asgctl script. See the Windows operating system help for more information.

## 7.8.3 Submitting OracleAS Guard Jobs to the Enterprise Manager Job System

You can use the Enterprise Manager Job System to automate the execution of any asgctl script to be run at a specified time interval or at a specified time and date, or both, in addition to setting other custom settings. To do this, access the EM Job Activity page and create your own host command job to execute your asgctl script, which is called a job task. Your job task (script) would invoke asgctl to then run the asgctl commands in the order in which they are listed. After you create your OracleAS Guard job, save it to the EM Job Library, which is a repository for frequently used jobs, where it can be executed based on the custom settings and time specifications you selected. See the Enterprise Manager online help and *Oracle Enterprise Manager Concepts* for more information.

## 7.8.4 Specifying Different Credentials for Two or More Nodes

By default, the credentials you specified in the `asgctl connect` command are used whenever one OracleAS Guard server connects to another OracleAS Guard server. However, there may be cases where you want to do either of the following:

- Use different credentials for each system on a given site, see Section 7.8.4.1, "Setting asgctl Credentials".
- Use a common set of credentials in the standby farm that are the same as the credentials used in the primary farm, see Section 7.8.4.2, "Specifying the Primary Database".

If the credentials for any host system are not the same as those used in the `asgctl connect` command, you must set the ASG credentials so that the OracleAS Guard server can connect to each host system in the configuration.

### 7.8.4.1 Setting asgctl Credentials

To set different credentials for all the host systems belonging to the same farm, enter the following `asgctl` command at the `asgctl` prompt. Specify the node name of the host system to which the credentials apply and the `ias_admin` account name and password for the `ias_admin` account created during the Oracle Application Server installation, and the key words `for farm`. Note that these settings are good for the current session.

```
ASGCTL> set asg credentials standbyinfra ias_admin/<iasadminpwd> for farm
```

When you specify the key words, `for farm`, you set the credentials for all the host systems that belong to the same farm as the specified system; otherwise, the credentials will only apply for the specified host system.

The `set asg credentials` command is also useful when you want to use different credentials for a specific server on the farm. In the previous example, the same credentials were set for all nodes on the standby farm, so that these credentials are different from the credentials used in the primary farm. The following command sets the credentials for a specific node, the `standbyinfra` node, on the standby farm.

```
ASGCTL> set asg credentials standbyinfra ias_admin/<iasadminpwd>
```

To summarize, if you set the credentials for a farm, these credentials are inherited for the entire farm. If you set the credentials for an individual host on the farm, the credentials (for this host) override the default credentials set for the farm.

### 7.8.4.2 Specifying the Primary Database

To identify the OracleAS Infrastructure database on the primary farm, enter the following `asgctl` command at the `asgctl` prompt. Specify the user name and password for the database account with `sysdba` privileges to access the OracleAS Infrastructure Database on the primary farm and the TNS service name of the OracleAS Infrastructure database:

```
ASGCTL> set primary database sys/testpwd@asdb
ASGCTL>
```

The standby site uses the same values as specified for the primary database because the service name and password for both the primary and standby OracleAS Infrastructure databases must be the same.

## 7.8.5 Reference Section: OracleAS Guard asgctl Command-line Commands

The rest of this chapter contains reference information describing the asgctl commands. Table 7–4 summarizes all the asgctl commands. Subsequent sections provide detailed reference information about each command.

**Table 7–4 Summary of asgctl Commands**

Command	Description
asgctl	Invokes the OracleAS Guard client command-line utility asgctl. On UNIX systems, <code>asgctl.sh</code> is located in <code>&lt;ORACLE_HOME&gt;/dsa/bin</code> and on Windows systems, <code>asgctl.bat</code> is located in <code>&lt;ORACLE_HOME&gt;\dsa\bin</code> .
connect asg	Connects the OracleAS Guard client to the OracleAS Guard server.
disconnect	Disconnects the OracleAS Guard client from the OracleAS Guard server.
dump farm	Directs the OracleAS Guard server to write detailed information about the farm to the screen or if specified, to a file.
exit	Disconnects the OracleAS Guard client from any existing connections and exits the OracleAS Guard client. This has the same effect as the quit command.
failover	During an unscheduled outage of the production site, the standby site becomes the production site.
help	Displays help information at the command line.
instantiate farm to	Creates a farm at the standby site (after verifying that the primary and standby sites are valid for OracleAS Disaster Recovery; also synchronizes the standby site with the primary site such that the primary and standby sites are transactionally consistent).
quit	Disconnects the OracleAS Guard client from any existing connections and exits the OracleAS Guard client. This has the same effect as the exit command.
set asg credentials	Sets the credentials used to authenticate the OracleAS Guard client connections to OracleAS Guard servers and connections between OracleAS Guard servers.
set echo	Sets command-echoing on or off in a asgctl script.
set new primary database	Identifies the OracleAS Infrastructure database on the standby farm as the new primary OracleAS Infrastructure database.
set primary database	Identifies the OracleAS Infrastructure database on the primary farm.
set trace	Enables or disables tracing for the specified trace flag. When tracing for a flag is set to on, the output of the trace is written to the OracleAS Guard log files.
show operation	Shows the current operation.
shutdown farm	Shuts down a running farm.
startup farm	Starts up a shutdown farm.
stop operation	Stops the specified operation.
switchover farm to	During a scheduled outage of the production site, switches the roles of the production site with the standby site so that the standby site now becomes the production site.



**Table 7-4 (Cont.) Summary of asgctl Commands**

<b>Command</b>	<b>Description</b>
sync farm to	Synchronizes the standby site with the primary site such that the primary and standby sites are transactionally consistent.
verify farm	Verifies that the farm is running and the configuration is valid. If a standby farm is specified, this command compares the primary and standby farms to verify that they conform to the requirements for OracleAS Disaster Recovery.

## asgctl

Invokes the OracleAS Guard client from the operating system command-line prompt or runs a script, if the path name to the script is provided.

### Format

```
asgctl@[filename]
```

### Parameters

**filename = <file-path>**

The path to a file that contains asgctl commands that you want to run as a script.

### Usage Notes

On UNIX systems, `asgctl.sh` is located in `<ORACLE_HOME>/dsa/bin` and on Windows systems, `asgctl.bat` is located in `<ORACLE_HOME>\dsa\bin`.

If you get an error starting `asgctl.sh` on UNIX systems and you are not using `csch`, the C shell, start `asgctl.sh` using `csch` as follow:

```
> csch asgctl.sh
```

### Example

```
> asgctl.sh
Application Server Guard: Release 10.1.2.0.0

(c) Copyright 2004 Oracle Corporation. All rights reserved
ASGCTL>
```

---

## connect asg

Connects the OracleAS Guard client to the OracleAS Guard server.

### Format

```
connect asg <host-name>[:<port>]> <username>/<password>
```

### Parameters

**host-name = <host-name>**

Name of the host system for the OracleAS Guard server to which you want the OracleAS Guard client to connect. This is the host system on which the OracleAS Infrastructure database is located. This OracleAS Guard server will be the coordinating server for all operations performed on the systems being configured. The host name is optional if the OracleAS Guard client and OracleAS Guard server are on the same node. The host can be a system outside the farm.

**port**

The port number of the host system.

**username/password**

The user name must be the `ias_admin` account name and the password for the `ias_admin` account created during the Oracle Application Server installation. This account name must be the same as the account name on at least one of the Oracle Application Server homes.

### Usage Notes

- The OracleAS Guard client system must have network access to the OracleAS Guard host system specified with the `host-name` parameter.
- The OracleAS Guard host system must have network access to all systems in the DR configuration and OracleAS Guard server must be installed and configured on all of these systems to run under the operating system user account specified by the user name and password parameters.
- The specified `ias_admin` account name must be configured with the necessary rights and privileges to permit OracleAS DR site operations (read/write access to all required files and directories, and so forth)
- An IP address can be used in place of a host name.

### Example

The command in the following example results in the OracleAS Guard client connecting to the OracleAS Guard server running on a host named `prodinfra` using the user name and password `ias_admin` and `adminpwd` respectively.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
```

---

## disconnect

Disconnects the OracleAS Guard client from the OracleAS Guard server to which it is currently connected.

### Format

disconnect

### Usage Notes

The OracleAS Guard client must be connected to a OracleAS Guard server when you issue this command.

### Example

The command in the following example disconnects the OracleAS Guard client from the OracleAS Guard server to which it is currently connected.

```
ASGCTL> disconnect  
ASGCTL>
```

---

## dump farm

Directs asgctl to write detailed information about the farm to the specified file.

### Format

```
dump farm [to <file>]
```

### Parameters

**to <file>**

Name of file on the OracleAS Guard client node where the detailed output is to be written.

### Usage Notes

An asgctl connection must be established before issuing this command.

### Example

The following example writes detailed information about the farm to a local file.

```
ASGCTL> dump farm to c:\dump_mid_1.txt
```

Contents of file c:\dump\_mid\_1.txt are:

```
prodinfra(asr1012): Describing each instance in the farm
```

```
Dump Farm
```

```
Name: IasFarm
Instance: asmid2.midtier.us.oracle.com
  Type: Core
  Oracle Home Name: asmid2
  Oracle Home Path: /private1/OraHome2
  Version: 10.1.2.0.0
  OidHost: infra.us.oracle.com
  OidPort: 4060
  VirtualHost: midtier.us.oracle.com
  Host: midtier.us.oracle.com
  Ip: 123.1.2.333
  Operation System Arch: sparc
  Operation System Version: 5.9
  Operation System Name: SunOS
  Java Home: /private1/OraHome2/jdk/jre
  Java Version: 1.4.1_03
  Java Class Version: 48.0
  Asg Server Port: 7890
```

```
Instance: asmid1.midtier.us.oracle.com
  Type: Core
  Oracle Home Name: asmid
  Oracle Home Path: /private1/OraHome
  Version: 10.1.2.0.0
  OidHost: infra.us.oracle.com
  OidPort: 4060
  VirtualHost: midtier.us.oracle.com
  Host: midtier.us.oracle.com
  Ip: 123.1.2.334
```

Operation System Arch: sparc  
Operation System Version: 5.9  
Operation System Name: SunOS  
Java Home: /private1/OraHome/jdk/jre  
Java Version: 1.4.1\_03  
Java Class Version: 48.0  
Asg Server Port: 7891

Instance: asr1012.infra.us.oracle.com  
Type: Infrastructure  
Oracle Home Name: asr1012  
Oracle Home Path: /private1/OraHome  
Version: 10.1.2.0.0  
OidHost: infra.us.oracle.com  
OidPort: 4060  
VirtualHost: infra.us.oracle.com  
Infrastructure DB: asdb.us.oracle.com  
Host: infra.us.oracle.com  
Ip: 123.1.2.111  
Operation System Arch: sparc  
Operation System Version: 5.8  
Operation System Name: SunOS  
Java Home: /private1/OraHome/jdk/jre  
Java Version: 1.4.1\_03  
Java Class Version: 48.0  
Asg Server Port: 7890

## exit

Disconnects from any existing connections to OracleAS Guard servers and exits from the OracleAS Guard client.

### Format

exit

### Parameters

**None**

### Usage Notes

None.

### Example

```
ASGCTL> exit  
>
```

## failover

During an unscheduled outage of the production site, performs the failover operation of the production site to the standby site.

### Format

failover

### Parameters

**none**

### Usage Notes

**For the Windows environment only:** In a DR configuration that uses CFC on the primary farm or standby farm, or both, the following information must be considered before performing an `asgctl` `instantiate farm to`, `switchover farm to`, or `failover` command. Before taking a cold backup or restoring the metadata repository database, the Oracle Backup and Recovery Tool shuts down the database first. In the Windows CFC environment, Oracle Fail Safe performs database polling and restarts the database if it is down. Hence, every time before the administrator performs an `instantiate`, `switchover`, or `failover` operation, the administrator must disable database polling in Oracle Fail Safe and re-enable it after the backup/restore operation (after the `instantiate`, `switchover`, or `failover` operation completes). The steps to perform this sequence of operations are described in a note in Section 7.6.1.2.1.

The OracleAS Guard client must be connected to a OracleAS Guard server before using this command. This OracleAS Guard server will act as the coordinating server for all operations performed on the systems being configured. By default, this is the local system where the `connect asg` command is being executed. This local system must be a member of the standby site farm.

The OracleAS Guard server must be started on the standby host system (`<standby_farm_host>`). The OracleAS Guard server can be stopped and started using `opmnctl` as follows:

```
On UNIX systems:  
<ORACLE_HOME>/opmn/bin/opmnctl startproc ias-component=DSA
```

```
On Windows systems:  
<ORACLE_HOME>\opmn\bin\opmnctl stopproc ias-component=DSA
```

The standby local system must be part of an ONS farm for the site.

The standby host must be part of an ONS farm for the standby site and must be symmetrical to the topology of the production farm.

Make sure OracleAS Infrastructure database is running on the standby farm before performing a failover operation. Also, the OracleAS Infrastructure database information must be set by using the `set new primary database asgctl` command.

The global DNS names are used to direct the failover. This will be different than the HA naming utilized in the DR environment. The discovery mechanism automatically maps the topology to the corresponding peer based off local name resolution.



## Example

The following example performs a failover operation to a standby site.

```

ASGCTL> connect asg standbyinfra ias_admin/adminpwd
Successfully connected to standbyinfra:7890
ASGCTL> set primary database sys/testpwd@asdb
ASGCTL> set new primary database sys/testpwd@asdb
ASGCTL> failover
standbyinfra(asr1012): Failover each instance in the farm from standby to primary farm
midtier(asmid2): Shutting down component HTTP_Server
midtier(asmid2): Shutting down component OC4J
midtier(asmid2): Shutting down component dcm-daemon
midtier(asmid2): Shutting down component LogLoader
midtier(asmid1): Shutting down component HTTP_Server
midtier(asmid1): Shutting down component OC4J
midtier(asmid1): Shutting down component dcm-daemon
midtier(asmid1): Shutting down component LogLoader
standbyinfra(asr1012): Shutting down component OID
standbyinfra(asr1012): Shutting down component HTTP_Server
standbyinfra(asr1012): Shutting down component OC4J
standbyinfra(asr1012): Shutting down component dcm-daemon
standbyinfra(asr1012): Shutting down component LogLoader
standbyinfra(asr1012): Failing over standby database
standbyinfra(asr1012): Failover Physical Standby Database - init.
standbyinfra(asr1012): Loading credentials for new primary database.
standbyinfra(asr1012): Failover Physical Standby Database - activate the standby database.
standbyinfra(asr1012): Connecting to database
standbyinfra(asr1012): Restarting the infrastructure standby database.
standbyinfra(asr1012): Starting up database as standby
standbyinfra(asr1012): Changing the database role to primary
standbyinfra(asr1012): Shutting down the new primary database
standbyinfra(asr1012): Starting up the new primary database
standbyinfra(asr1012): Failover Physical Standby Database - new primary database is started.
standbyinfra(asr1012): Starting component OID
standbyinfra(asr1012): Starting component dcm-daemon
midtier(asmid2): Starting component dcm-daemon
midtier(asmid1): Starting component dcm-daemon
standbyinfra(asr1012): Verifying each instance in the farm
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
standbyinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
ASGCTL>

```

## help

Displays help information.

### Format

help [<command>]

### Parameters

**command**

Name of the command for which you want help.

### Usage Notes

None.

### Example

The following example displays help about all dump farm commands.

```
ASGCTL> help
  connect asg [<host>] [ias_admin/<password>]
  disconnect
  exit
  quit
  dump farm [to <file>]
  help [<command>]
  instantiate farm to <standby_farm_host>
  set asg credentials <host> ias_admin/<password> [for farm]
  set trace on|off <traceflags>
  set primary database <username>/<password>@<servicename> [pfile <filename> | spfile
<filename>]
  set new primary database <username>/<password>@<servicename> [pfile <filename> | spfile
<filename>]
  sync farm to <standby_farm_host> [<sync_mode>]
  startup farm
  shutdown farm
  show op[eration] [full] [[his]tory]
  stop op[eration] <op#>
  verify farm [with <host>]
  switchover farm to <standby_farm_host>
  failover
ASGCTL>
```

---

## instantiate farm to

Instantiates a farm to a standby site by discovering the current farm definition at the production and standby sites, verifying that each complies with the OracleAS Disaster Recovery rules and restrictions of the current OracleAS software deployed on these systems prior to creation. Also synchronizes the standby site with the primary site such that the primary and standby sites are transactionally consistent.

### Format

```
instantiate farm to <standby_farm_host>[:<port>]
```

### Parameters

#### **standby\_farm\_host**

Name of the standby host system. This is the host system on which the OracleAS Infrastructure database is located. This parameter is required because it directs the coordinating OracleAS Guard server instance to discover the instances that make up the standby site. This host system must be a member of the standby farm.

#### **port**

The port number of the host system.

### Usage Notes

You must always set the primary database before performing an instantiate, sync, or switchover operation.

**For the Windows environment only:** In a DR configuration that uses CFC on the primary farm or standby farm, or both, the following information must be considered before performing an `asgctl instantiate farm to`, `switchover farm to`, or `failover` command. Before taking a cold backup or restoring the metadata repository database, the Oracle Backup and Recovery Tool shuts down the database first. In the Windows CFC environment, Oracle Fail Safe performs database polling and restarts the database if it is down. Hence, every time before the administrator performs an instantiate, switchover, or failover operation, the administrator must disable database polling in Oracle Fail Safe and re-enable it after the backup/restore operation (after the instantiate, switchover, or failover operation completes). The steps to perform this sequence of operations are described in a note in Section 7.5.2, "Standby Instantiation".

The OracleAS Guard client must be connected to a OracleAS Guard server before using this command. This OracleAS Guard server will act as the coordinating server for all operations performed on the systems being configured. By default, this is the local system where the `connect asg` command is being executed. This local system must be a member of the production site farm.

The OracleAS Guard server must be started on the standby host system (<standby\_farm\_host>). The OracleAS Guard server can be stopped and started using `opmnctl` as follows:

On UNIX systems:

```
<ORACLE_HOME>/opmn/bin/opmnctl startproc ias-component=DSA
```

On Windows systems:

```
<ORACLE_HOME>\opmn\bin\opmnctl stopproc ias-component=DSA
```

The production local system must be part of an ONS farm for the site.

The standby host must be part of an ONS farm for the standby site and must be symmetrical to the topology of the production farm.

Make sure OracleAS Infrastructure database is running on the primary farm before performing an instantiating farm operation. Also, the OracleAS Infrastructure database information must be set by using the set primary database asgctl command.

The global DNS names are used to direct the instantiation. This will be different than the HA naming utilized in the DR environment. The discovery mechanism automatically maps the topology to the corresponding peer based off local name resolution.

## Example

The following example instantiates a standby farm by attaching the coordinating OracleAS Guard server and discovering the topology of the production and standby sites, and performing site verification, establishing a DR environment with the farm containing the standby farm host known by DNS as standbyinfra. Note that part way through the operation you will be prompted to answer a question regarding whether you want to shut down the database. Reply by entering `y` or `yes`.

```
ASGCTL> instantiate farm to standbyinfra
prodinfra(asr1012): Instantiating each instance in the farm to standby farm
prodinfra(asr1012): Verifying each instance in the farm
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
prodinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
.
.
.
This operation requires the database to be shutdown. Do you want to continue? Yes or No
Y
.
.
.
standbyinfra(asr1012): Starting restore/synchronization of database "asdb.us.oracle.com".
standbyinfra(asr1012): Shutting down the standby database.
standbyinfra(asr1012): Starting the standby database.
standbyinfra(asr1012): Synchronizing farm completed successfully.
prodinfra(asr1012): Synchronizing farm completed successfully.
ASGCTL>
```

---

## quit

Instructs the OracleAS Guard client to disconnect from any existing connections and exit from asgctl.

### Format

quit

### Parameters

**None**

### Usage Notes

None.

### Example

The following example exits from asgctl.

```
ASGCTL> quit  
>
```

## set asg credentials

Sets the credentials used to authenticate the OracleAS Guard connections to OracleAS Guard servers.

### Format

```
set asg credentials <host>[:<port>] <username>/<password> [for farm]
```

### Parameters

**host**

Name of the host system to which the credentials apply. When OracleAS Guard connects to that host, it will use these credentials.

**port**

The port number of the host system.

**username/password**

The user name must be the `ias_admin` account name and the password for the `ias_admin` account created during the Oracle Application Server installation. This account name must be the same as the account name on at least one of the Oracle Application Server homes.

**for farm**

A keyword, that if present in the command line, directs OracleAS Guard to set the credentials for all of the host systems that belong to the same farm as the local host system.

### Usage Notes

By default, the credentials used in the `asgctl connect` command are used whenever a OracleAS Guard server needs to connect to another OracleAS Guard server. However, there may be cases where you want to use different credentials for a specific server. This command allows you to use the same credentials for all nodes in a farm. For example, you may want to use a common set of credentials in the standby farm that is different from the credentials used in the primary farm.

If you set the credentials for a farm, these credentials are inherited for the entire farm. If you set the credentials for an individual host on the farm, the credentials (for this host) override the default credentials set for the farm.

The OracleAS Guard client must be connected to a OracleAS Guard server before using this command.

An IP address can be used in place of a host name.

### Example

The following example sets the OracleAS Guard credentials of host system `standbyinfra` to all host systems that belong to this farm.

```
ASGCTL> set asg credentials standbyinfra <username>/<password> for farm
```

---

## set echo

Sets command-echoing on or off in a asgctl script.

### Format

```
set echo on | off
```

### Parameters

#### **on | off**

Specifying "on" turns on command-echoing in a asgctl script. Specifying "off" turns off command-echoing in a asgctl script.

### Usage Notes

This command is useful when running large asgctl scripts. For example, if the asgctl script has error test cases with comments entered before each test case or before each asgctl command, setting echo on displays the comment before each test case or before each asgctl command that is run to give you an explanation of what the test case is or what asgctl command is about to be run.

This command also works with nested scripts.

### Example

The following example is a asgctl script that turns on command-echoing, runs a test case, connects to a OracleAS Guard server, displays detailed information about the farm, then turns echo off, disconnects from the OracleAS Guard server, and exits from the OracleAS Guard client.

```
> ASGCTL @myasgctltestscript.txt

# myasgctltestscript.txt
# turn on echo
set echo on

# make sure you're not connected
disconnect

# not connected, should get an error message
dump farm

# connect to a DSA server
connect asg prodinfra ias_admin/adminpwd

#display detailed info about the farm
dump farm

#disconnect
disconnect

# turn off echo
echo off
exit
```

## set new primary database

Identifies the OracleAS Infrastructure database on the standby farm as the new primary database preceding a failover operation. This command is only used as part of a failover operation.

### Format

```
set new primary database <username>/<password>@<servicename> [pfile <filename> | spfile  
<filename>]
```

### Parameters

**username/password**

User name and password for the database account with sysdba privileges.

**servicename**

The TNS service name of the OracleAS Infrastructure database. The name must be defined on the OracleAS Infrastructure host system; it does not need to be defined on the OracleAS Guard client host system.

**pfile filename**

The filename of the primary (OracleAS Infrastructure) database initialization file that will be used when the primary database is started.

**spfile filename**

The filename of the server (OracleAS Infrastructure) initialization file that will be used when the database is started.

### Usage Notes

The OracleAS Guard client must be connected to a OracleAS Guard server before using this command.

Before performing a failover operation, you are required to connect to the Infrastructure node of the standby farm and define the new primary database. Once the Oracle Infrastructure database on the standby site is identified as the new primary database, then you can proceed to begin the failover operation.

### Example

The following example sets the OracleAS Infrastructure database information for the standby farm as the new primary/production farm preceding a failover operation.

```
ASGCTL> connect asg standbyinfra ias_admin/adminpwd  
Successfully connected to standbyinfra:7890  
ASGCTL> set new primary database sys/testpwd@asdb  
ASGCTL> failover  
. . .  
ASGCTL>
```



## set primary database

Identifies the OracleAS Infrastructure database on the primary farm.

### Format

```
set primary database <username>/<password>@<servicename> [pfile <filename> | spfile <filename>]
```

### Parameters

**username/password**

User name and password for the database account with sysdba privileges.

**servicename**

The TNS service name of the OracleAS Infrastructure database. The name must be defined on the OracleAS Infrastructure host system; it does not need to be defined on the OracleAS Guard client host system.

**pfile filename**

The filename of the primary (OracleAS Infrastructure) database initialization file that will be used when the primary database is started.

**spfile filename**

The filename of the server (OracleAS Infrastructure) initialization file that will be used when the database is started.

### Usage Notes

The OracleAS Guard client must be connected to a OracleAS Guard server before using this command.

You must always set the primary database before performing an instantiate, sync, or switchover operation.

### Example

The following example sets the OracleAS Infrastructure database information for the primary/production farm.

```
ASGCTL> set primary database sys/testpwd@asdb  
ASGCTL>
```

## set trace

Sets a trace flag on or off to log output to the OracleAS Guard log files.

### Format

```
set trace on | off <traceflags>
```

### Parameters

#### **on | off**

Specifying "on" enables tracing. Specifying "off" disables tracing.

#### **traceflags**

The traceflags to be enabled. Two or more specified traceflags entries must be separated by a comma (.). The traceflags are as follows:

- DB -- trace information regarding processing in the Oracle Database environment
- HOME -- trace information with regard to Oracle homes
- IAS -- trace information regarding processing in Oracle Application Server
- OPMN -- trace information regarding access to OracleAS OPMN calls
- IP -- trace information regarding network access and address translation

### Usage Notes

This command applies to all hosts that might be involved in a asgctl command during the lifetime of the connection.

The OracleAS Guard client must be connected to a OracleAS Guard server before using this command.

### Example

The following example turns on trace for database operations.

```
ASGCTL> set trace on db
```

---

## show operation

Shows all operations on all nodes of the farm to which the OracleAS Guard client is connected.

### Format

```
show operation [full] [history]
```

### Parameters

#### full

For all operations, shows the operation number, the job name, the job owner's users name, the job ID, the time the operation began, the time the operation ended, the elapsed time for the operation, as well as all tasks belonging to this job.

#### history

For only operations that are not running, shows the operation number and the job name.

### Usage Notes

None.

### Example

The following examples show the status of the current operation.

```
ASGCTL> show operation full
*****
OPERATION: 15
  Status: running
  Elapsed Time: 0 days, 0 hours, 1 minutes, 35 secs
  TASK: instantiateFarm
      TASK: verifyFarm
```

```
ASGCTL> show operation
There are no operations in progress
```

The following example shows the history of all operations.

```
ASGCTL> show operation history
*****
OPERATION: 69
  Status: success
  Elapsed Time: 0 days, 0 hours, 0 minutes, 7 secs

  TASK: getFarm
*****
OPERATION: 70
  Status: success
  Elapsed Time: 0 days, 0 hours, 6 minutes, 49 secs

  TASK: syncFarm
      TASK: backupFarm
          TASK: fileCopyRemote
          TASK: fileCopyRemote
          TASK: fileCopyRemote
```

show operation

---

```
TASK: resolveHost
TASK: restoreFarm
  TASK: fileCopyLocal
  TASK: fileCopyLocal
  TASK: fileCopyLocal
.
```

---

## shutdown farm

Shuts down a running farm.

### Format

```
shutdown farm
```

### Parameters

**none**

### Usage Notes

This is a useful troubleshooting command to use when the farm is having a problem with one of its members and must be shut down. Use the startup farm command to start it up again.

### Example

The following example shuts down the prodinfra production farm.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> shutdown farm
prodinfra(asr1012): Shutting down each instance in the farm
prodinfra(asm21012): Shutting down component wireless
prodinfra(asm1012): Shutting down component WebCache
prodinfra(asm1012): Shutting down component OC4J
prodinfra(asm21012): Shutting down component WebCache
prodinfra(asm1012): Shutting down component dcm-daemon
prodinfra(asm21012): Shutting down component OC4J
prodinfra(asm1012): Shutting down component LogLoader
prodinfra(asm1012): Shutting down component HTTP_Server
prodinfra(asm21012): Shutting down component dcm-daemon
prodinfra(asm21012): Shutting down component LogLoader
prodinfra(asm21012): Shutting down component HTTP_Server
prodinfra(asr1012): Shutting down component OID
prodinfra(asr1012): Shutting down component OC4J
prodinfra(asr1012): Shutting down component dcm-daemon
prodinfra(asr1012): Shutting down component HTTP_Server
ASGCTL>
```

## startup farm

Starts up a shutdown farm.

### Format

startup farm

### Parameters

**none**

### Usage Notes

This is a useful troubleshooting command to use when the farm is having a problem with one of its members and must be shut down, then started up again.

### Example

The following example starts up the production farm.

```
ASGCTL> startup farm
prodinfra(asr1012): Starting each instance in the farm
prodinfra(asr1012): Executing opmnctl startall command.
prodinfra(asm1012): Executing opmnctl startall command.
prodinfra(asm21012): Executing opmnctl startall command.
ASGCTL>
```

---

## stop operation

Stops a specific operation that is running on the server.

### Format

```
stop operation <op #>
```

### Parameters

**op #**

The number of the operation.

### Usage Notes

The number of the operation that is running on the server can be determined from a show operation command.

### Example

The following example first shows the running operation (15) on the server and then the stop operation command stops this operation.

```
ASGCTL> show operation
*****
OPERATION: 15
  Status: running
  Elapsed Time: 0 days, 0 hours, 1 minutes, 35 secs
  TASK: instantiateFarm
        TASK: verifyFarm

ASGCTL> stop operation 15
```

## switchover farm to

During a scheduled outage of the production site, performs the switchover operation from the production site to the standby site. The switchover operation to the standby site makes the application of the database redo logs synchronized to match the backup and restoration of the configuration files for the middle-tier and OracleAS Infrastructure installations.

### Format

```
switchover farm to <standby_farm_host>[:<port>]
```

### Parameters

#### **standby\_farm\_host**

Name of the standby host system. This is the host system on which the OracleAS Infrastructure database is located. This parameter is required because it directs the coordinating OracleAS Guard server instance to discover the instances that make up the standby site. This host system must be a member of the standby farm.

#### **port**

The port number of the standby host system.

### Usage Notes

You must always set the primary database before performing an instantiate, sync, or switchover operation.

**For the Windows environment only:** In a DR configuration that uses CFC on the primary farm or standby farm, or both, the following information must be considered before performing an `asgctl instantiate farm to`, `switchover farm to`, or `failover` command. Before taking a cold backup or restoring the metadata repository database, the Oracle Backup and Recovery Tool shuts down the database first. In the Windows CFC environment, Oracle Fail Safe performs database polling and restarts the database if it is down. Hence, every time before the administrator performs an instantiate, switchover, or failover operation, the administrator must disable database polling in Oracle Fail Safe and re-enable it after the backup/restore operation (after the instantiate, switchover, or failover operation completes). The steps to perform this sequence of operations are described in a note in Section 7.6.1.1.1, "Site Switchover Operations".

**For the Windows environment only:** Before doing a switchover operation, the administrator must use the Windows Service Control Manager and shut down the OracleDBConsole service on the primary node.

On the primary infrastructure system, make sure the emagent process is stopped. Otherwise, you may run into the following error when doing a switchover operation because the emagent has a connection to the database:

```
prodinfra: -->ASG_DGA-13051: Error performing a physical standby switchover.  
prodinfra: -->ASG_DGA-13052: The primary database is not in the proper state to  
perform a switchover. State is "SESSIONS ACTIVE"
```

On UNIX systems, to stop the emagent process, stop the Application Server Control, which is called `iasconsole`, as follows:

```
> <ORACLE_HOME>/bin/emctl stop iasconsole
```



On UNIX systems, to check to see if there is an emagent process running, do the following:

```
> ps -ef | grep emagent
```

On UNIX systems, if after performing the stop iasconsole operation, the emagent process is still running, get its process ID (PID) as determined from the previous ps command and stop it as follows:

```
> kill -9 <emagent-pid>
```

On Windows systems, open the Services control panel. Locate the OracleAS10gASControl service and stop this service.

The OracleAS Guard client must be connected to a OracleAS Guard server before using this command. This OracleAS Guard server will act as the coordinating server for all operations performed on the systems being configured. By default, this is the local system where the `connect asg` command is being executed. This local system must be a member of the production site farm.

The OracleAS Guard server must be started on the standby host system (<standby\_farm\_host>). The OracleAS Guard server can be stopped and started using `opmnctl` as follows:

On UNIX systems:

```
<ORACLE_HOME>/opmn/bin/opmnctl startproc ias-component=DSA
```

On Windows systems:

```
<ORACLE_HOME>\opmn\bin\opmnctl stopproc ias-component=DSA
```

The production local system must be part of an ONS farm for the site.

The standby host must be part of an ONS farm for the standby site and must be symmetrical to the topology of the production farm.

Make sure OracleAS Infrastructure database is running on the primary farm before performing a switchover operation. Also, the OracleAS Infrastructure database information must be set by using the `set primary database asgctl` command.

The global DNS names are used to direct the switchover. This will be different than the HA naming utilized in the DR environment. The discovery mechanism automatically maps the topology to the corresponding peer based off local name resolution.

As part of the OracleAS Guard switchover operation, an implicit sync farm operation is performed to make sure the farms are identical. In addition OPMN automatically starts the OracleAS Guard server on the "new" standby Infrastructure node and this server will run indefinitely, and in turn, starts the OracleAS Guard server on the other nodes in the "new" standby farm and each of these is a transient server.

## Example

The following example performs a switchover operation to a standby site known by DNS as `standbyinfra`.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> set primary database sys/testpwd@asdb
ASGCTL> switchover farm to standbyinfra
prodinfra(asr1012): Switchover each instance in the farm to standby farm
midtier(asmid2): Shutting down component HTTP_Server
midtier(asmid1): Shutting down component HTTP_Server
```

```

midtier(asmid2): Shutting down component OC4J
midtier(asmid1): Shutting down component OC4J
midtier(asmid2): Shutting down component dcm-daemon
midtier(asmid1): Shutting down component WebCache
midtier(asmid1): Shutting down component dcm-daemon
midtier(asmid1): Shutting down component HTTP_Server
midtier(asmid1): Shutting down component OC4J
midtier(asmid1): Shutting down component dcm-daemon
midtier(asmid1): Shutting down component LogLoader
midtier(asmid2): Shutting down component HTTP_Server
midtier(asmid2): Shutting down component OC4J
midtier(asmid2): Shutting down component dcm-daemon
midtier(asmid2): Shutting down component LogLoader
midtier(asmid1): Shutting down component LogLoader
midtier(asmid2): Shutting down component LogLoader
prodinfra(asr1012): Shutting down component OID
standbyinfra(asr1012): Shutting down component OID
standbyinfra(asr1012): Shutting down component HTTP_Server
standbyinfra(asr1012): Shutting down component OC4J
standbyinfra(asr1012): Shutting down component dcm-daemon
standbyinfra(asr1012): Shutting down component LogLoader
prodinfra(asr1012): Shutting down component HTTP_Server
prodinfra(asr1012): Shutting down component OC4J
prodinfra(asr1012): Shutting down component dcm-daemon
prodinfra(asr1012): Shutting down component LogLoader
prodinfra(asr1012): Synchronizing farm.
prodinfra(asr1012): Synchronizing each instance in the farm to standby farm
prodinfra(asr1012): Starting backup of farm "IasFarm".
prodinfra(asr1012):   Loading farm information.
prodinfra(asr1012):   Backing up and copying data to the standby farm.
prodinfra(asr1012): Backing up each instance in the farm.
.
.
.
standbyinfra(asr1012): Verifying each instance in the farm
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
standbyinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
prodinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
standbyinfra(asr1012): Verifying farm IasFarm is symmetrical in both primary and standby
configuration.
standbyinfra(asr1012): Verifying instance asmid1.midtier.us.oracle.com is symmetrical in both
primary and standby configuration.
standbyinfra(asr1012): Host names are symmetrical for instance asmid.midtier.us.oracle.com.
standbyinfra(asr1012): Home name and path are symmetrical for instance
asmid.midtier.us.oracle.com.
standbyinfra(asr1012): Verifying instance asmid2.midtier.us.oracle.com is symmetrical in both
primary and standby configuration.
standbyinfra(asr1012): Host names are symmetrical for instance asmid2.midtier.us.oracle.com.
standbyinfra(asr1012): Home name and path are symmetrical for instance
asmid2.midtier.us.oracle.com.
standbyinfra(asr1012): Verifying instance asr1012.infra.us.oracle.com is symmetrical in both
primary and standby configuration.
standbyinfra(asr1012): Host names are symmetrical for instance asr1012.infra.us.oracle.com.
standbyinfra(asr1012): Home name and path are symmetrical for instance
asr1012.infra.us.oracle.com.
ASGCTL>

```

## sync farm to

Synchronizes the standby site with the primary site to ensure that the two sites are transactionally consistent.

### Format

```
sync farm to <standby_farm_host>[:<port>] [<sync_mode>]
```

### Parameters

#### **standby\_farm\_host**

Name of the standby site host system. This is the host system on which the OracleAS Infrastructure database is located. This parameter is required because it directs the coordinating OracleAS Guard server instance to discover the instances that make up the standby site. This host system must be a member of the standby farm.

#### **port**

The port number of the standby host system.

#### **sync\_mode**

The sync mode can be either "full" or "incremental". The default is "incremental".

### Usage Notes

The OracleAS Guard client must be connected to a OracleAS Guard server before using this command.

You must always set the primary database before performing an instantiate, sync, or switchover operation.

By default `sync_mode` is incremental and offers the best performance. However, there may be three circumstances when specifying a `sync_mode` of full should be used.

- When you want to force a full synchronization to happen, such as synchronizing the standby site completely at a specific point in time (currently) with the primary site.
- When you know there are many transactional changes over a short period of time on the primary site that must be synchronized with the secondary site.
- When you know that there are a large accumulation of transactional changes over a long period of time on the primary site that must be synchronized with the secondary site.

### Example

The following example synchronizes the specified standby site with the coordinating OracleAS Guard server (the primary site). By default the sync mode is incremental.

```
ASGCTL> sync farm to standbyinfra
prodinfra(asr1012): Synchronizing each instance in the farm to standby farm
prodinfra(asr1012): Starting backup of farm "IasFarm".
prodinfra(asr1012): Loading farm information.
prodinfra(asr1012): Backing up and copying data to the standby farm.
prodinfra(asr1012): Backing up each instance in the farm
prodinfra(asr1012): Starting backup of instance "asr1012.infra.us.oracle.com".
prodinfra(asr1012): Executing config_nodb option in bkp_restore.pl script.
midtier(asmid2): Starting backup of instance "asmid2.midtier.us.oracle.com".
```

```
midtier(asmid2): Executing config_nodb option in bkp_restore.pl script.
midtier(asmid1): Starting backup of instance "asmid1.midtier.us.oracle.com".
midtier(asmid1): Executing config_nodb option in bkp_restore.pl script.
prodinfra(asr1012): Executing backup_config option in bkp_restore.pl script.
prodinfra(asr1012): Deleted directory "/private1/OraHome/dsa/backup".
midtier(asmid2): Executing backup_config option in bkp_restore.pl script.
midtier(asmid2): Deleted directory "/private1/OraHome2/dsa/backup".
midtier(asmid1): Executing backup_config option in bkp_restore.pl script.
midtier(asmid1): Deleted directory "/private1/OraHome/dsa/backup".
.
.
.
prodinfra(asr1012): Starting backup/synchronization of database "asdb.us.oracle.com".
midtier(asmid2): Synchronizing farm completed successfully.
midtier(asmid1): Synchronizing farm completed successfully.
midtier(asmid1): Synchronizing farm completed successfully.
midtier(asmid2): Synchronizing farm completed successfully.
standbyinfra(asr1012): Starting restore/synchronization of database "asdb.us.oracle.com".
standbyinfra(asr1012): Shutting down the standby database.
standbyinfra(asr1012): Starting the standby database.
standbyinfra(asr1012): Synchronizing farm completed successfully.
ASGCTL>
```

## verify farm

Validates that the primary farm is running and the configuration is valid. If a standby farm is specified, compares the primary farm to which the local host system is a member with the standby farm to validate that they are consistent with one another and conform to the requirements for OracleAS Disaster Recovery.

### Format

```
verify farm [with <host>[:<port>]]
```

### Parameters

#### host

Name of the standby host system. This is the host system on which the OracleAS Infrastructure database is located. This parameter is required because it directs the coordinating OracleAS Guard server instance to discover the instances that make up the standby site. This host system must be a member of the standby farm.

#### port

The port number of the host system.

### Usage Notes

The OracleAS Guard client must be connected to a OracleAS Guard server before using this command. This OracleAS Guard server will act as the coordinating server for all operations performed on the systems being verified. By default, the name is the local system where the `connect asgctl` command is being executed. The local host system must be a member of the production site farm.

If the host system name is not specified, the farm in which the local host system participates will be verified for local OracleAS Disaster Recovery rules.

If the standby host system name is specified, the farm at the standby site will be verified along with the production farm for both local rules and distributed OracleAS Disaster Recovery rules.

### Example

The following example validates that the primary farm is running and the configuration is valid.

```
ASGCTL> connect asg ias_admin/iactest2
Successfully connected to prodinfra:7890
ASGCTL> verify farm
prodinfra(asr1012): Verifying each instance in the farm
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
prodinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
ASGCTL>
```

The following example validates that the farm to which the local host system is a member is consistent with the standby farm to which the host system standbyinfra is a member.

```
ASGCTL> verify farm with standbyinfra
prodinfra(asr1012): Verifying each instance in the farm
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
```

```
prodfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
midtier(asmid1): HA directory exists for instance asmid1.midtier.us.oracle.com.
midtier(asmid2): HA directory exists for instance asmid2.midtier.us.oracle.com.
standbyinfra(asr1012): HA directory exists for instance asr1012.infra.us.oracle.com.
prodfra(asr1012): Verifying farm IasFarm is symmetrical in both primary and standby configuration.
prodfra(asr1012): Verifying instance asmid2.midtier.us.oracle.com is symmetrical in both primary and standby
configuration.
prodfra(asr1012): Host names are symmetrical for instance asmid2.midtier.us.oracle.com.
prodfra(asr1012): Home name and path are symmetrical for instance asmid2.midtier.us.oracle.com.
prodfra(asr1012): Verifying instance asmid1.midtier.us.oracle.com is symmetrical in both primary and standby
configuration.
prodfra(asr1012): Host names are symmetrical for instance asmid1.midtier.us.oracle.com.
prodfra(asr1012): Home name and path are symmetrical for instance asmid1.midtier.us.oracle.com.
prodfra(asr1012): Verifying instance asr1012.infra.us.oracle.com is symmetrical in both primary and standby
configuration.
prodfra(asr1012): Host names are symmetrical for instance asr1012.infra.us.oracle.com.
prodfra(asr1012): Home name and path are symmetrical for instance asr1012.infra.us.oracle.com.
ASGCTL>
```

# Part V

---

## Appendices

The information in this part is supplementary to the previous chapters of the book and is organized into the following appendices:

- Appendix A, "Troubleshooting High Availability"
- Appendix B, "Manually Managed Oracle Application Server Cluster"
- Appendix C, "OracleAS Guard Error Messages"
- Appendix D, "Manual Sync Operations"
- Appendix E, "Setting Up a DNS Server"
- Appendix F, "Secure Shell (SSH) Port Forwarding"





---

---

# Troubleshooting High Availability

This appendix describes common problems that you might encounter when deploying and managing Oracle Application Server in high availability configurations, and explains how to solve them. It contains the following topics:

- Problems and Solutions
- Need More Help?

## A.1 Problems and Solutions

This section describes common problems and solutions. It contains the following topics:

- Section A.1.1, "Cluster Configuration Assistant Fails During Installation"
- Section A.1.2, "Oracle Ultra Search Configuration Assistant is Unable to Connect to Oracle Internet Directory During High Availability Infrastructure Installation"
- Section A.1.3, "Unable to Perform Online Database Backup and Restore in OracleAS Cold Failover Cluster Environment"
- Section A.1.4, "odisrv Process Does Not Failover"
- Section A.1.5, "Oracle Ultra Search Web Crawler Does Not Failover"
- Section A.1.6, "Unable to Restore OracleAS Metadata Repository to a Different Host"
- Section A.1.7, "Cannot Connect to Database for Restoration (Windows)"
- Section A.1.8, "Unpredictable Behavior from Oracle Application Server Cluster (Identity Management) Configuration"
- Section A.1.9, "Wrong Name Specified for Load Balancer"
- Section A.1.10, "OracleAS Disaster Recovery: Standby Site Not Synchronized"
- Section A.1.11, "OracleAS Disaster Recovery: Failure to Bring Up Standby Instances After Failover or Switchover"
- Section A.1.12, "OracleAS Disaster Recovery: Unable to Start Standalone OracleAS Web Cache Installations at the Standby Site"
- Section A.1.13, "OracleAS Disaster Recovery: Standby Site Middle-tier Installation Uses Wrong Hostname"
- Section A.1.14, "OracleAS Disaster Recovery: Failure of Farm Verification Operation with Standby Farm"

- Section A.1.15, "OracleAS Disaster Recovery: Sync Farm Operation Returns Error Message"

### A.1.1 Cluster Configuration Assistant Fails During Installation

Problems encountered during the clustering of components using the Cluster Configuration Assistant are addressed here.

#### Problem

During the installation of distributed Oracle Identity Management configurations, the OracleAS Single Sign-On and Oracle Delegated Administration Services components are installed in two of their own nodes separate from the other Oracle Identity Management components. The Cluster Configuration Assistant may attempt to cluster the two resulting OracleAS Single Sign-On/Oracle Delegated Administration Services instances together. However, the error message "Instances containing disabled components cannot be added to a cluster" may appear. This message appears because Enterprise Manager cannot cluster instances with disabled components.

#### Solution

If the Cluster Configuration Assistant fails, you can cluster the instance after installation. In this case, to cluster the instance, you must use the "dcmctl joincluster" command instead of Application Server Control Console. You cannot use Application Server Control Console in this case because it cannot cluster instances that contain disabled components. In this case, the "home" OC4J instance is disabled.

### A.1.2 Oracle Ultra Search Configuration Assistant is Unable to Connect to Oracle Internet Directory During High Availability Infrastructure Installation

During high availability Infrastructure installation, the Oracle Ultra Search Configuration Assistant cannot connect to an Oracle Internet Directory instance at port 3060 of the virtual hostname provided in the virtual hostname addressing screen.

#### Problem

A common mistake can be made when virtual hostname addressing is used during Infrastructure installation. The load balancer virtual server name is entered, and the load balancer is set up correctly to assume this name. However, the Infrastructure node is not set up correctly to resolve this name. Thus, when the Oracle Ultra Search Configuration Assistant on the Infrastructure node tries to connect to the load balancer virtual server name, the Configuration Assistant cannot find the load balancer.

#### Solution

The solution is to set up name resolution correctly on the Infrastructure machine for the load balancer virtual server name. This procedure is platform dependent. Check your operating system manual for an accurate procedure. In Unix, this usually involves editing the /etc/hosts file and making sure this file is used for name resolution by editing the /etc/nsswitch.conf file. In Windows, this usually involves editing the C:\WINDOWS\system32\drivers\etc\hosts file.

---

### A.1.3 Unable to Perform Online Database Backup and Restore in OracleAS Cold Failover Cluster Environment

Issues with online database backup and restore are noted here. This information pertains to the OracleAS Cold Failover Cluster environment.

#### Problem

Unable to perform online recovery of Infrastructure database due to dependencies and cluster administrator trying to bring the database down and then up during the recovery phase by the Backup and Recovery Tool.

#### Solution 1

To perform a clean recovery, use the following steps:

1. Bring all resources offline using the cluster administrator (for Windows, use Oracle Failsafe).
2. Perform a normal shutdown of the Infrastructure database.
3. Start only the database service using the following command:  

```
net start OracleService<SID>
```
4. Run the Backup and Recovery Tool to perform the recovery of the database.

#### Solution 2

For Windows, the following steps can be used to perform a recovery:

1. In Oracle Failsafe, under "Cluster Resources", select "ASDB(DB Resource)" in the "Database" tab.
2. For "Database Polling", select "Disabled" from the drop down list.
3. Using the Backup and Recovery Tool, perform an online restore of the Infrastructure database.

The database is not accessible for a brief period while the Backup and Recovery Tool stops and starts the database. Once the database starts up, it can be accessed by middle-tier and Infrastructure components.

### A.1.4 `odisrv` Process Does Not Failover

Issues with `odisrv` process failover between nodes are documented here.

#### Problem

In any OracleAS Cluster (Identity Management) solution, when `opmnctl stopall` is executed to stop all OPMN-managed processes on that node, `odisrv` is not started automatically on the second node because `opmnctl stopall` is a normal administrative shutdown, not an actual node failure. In a true node failure, `odisrv` is started on the remaining node upon death detection of the original `odisrv` process.

#### Solution

If planned maintenance is required for an OracleAS Cluster (Identity Management), use the `oidctl` command to explicitly stop and start `odisrv`.

On the node where `odisrv` is running, use the following command to stop it:

```
oidctl connect=<dbConnect> server=odisrv inst=1 stop
```

On the remaining active node, start `odisrv` using the following command:

```
oidctl connect=<dbConnect> server=odisrv inst=1 falgs="..." start
```

## A.1.5 Oracle Ultra Search Web Crawler Does Not Failover

For Real Application Clusters that do not use a Cluster File System, the Oracle Ultra Search web crawler does not failover to an available node.

### Problem

Currently, the Oracle Ultra Search web crawler is configured so that it can be run only from one node in a Real Application Cluster. If that node (or the database) goes down, the web crawler will not startup on an available node. This situation occurs for non Cluster File System Real Application Clusters.

### Solution

When Real Application Clusters use a Cluster File System, Oracle Ultra Search crawler can be launched from any of the Real Application Clusters nodes. At least one node has to be running.

When a Cluster File System is not used, the Oracle Ultra Search crawler always runs on a specified node. If this node stops operating, you must run the `wk0reconfig.sql` script to move Oracle Ultra Search to another Real Application Clusters node. This script can be run as follows:

```
> sqlplus wksys/wksys_passwd
> ORACLE_HOME/ultrasearch/admin/wk0reconfig.sql <instance_name> <connect_url>
```

where `<instance_name>` is the name of the Real Application Clusters instance that Oracle Ultra Search uses for crawling. This name can be obtained by using the following SQL statement after connecting to the database:

```
SELECT instance_name FROM v$instance
```

`<connect_url>` is the JDBC connection string that guarantees a connection only to the specified instance, such as:

```
(DESCRIPTION=
  (ADDRESS_LIST=
    (ADDRESS= (PROTOCOL=TCP)
      (HOST=<nodename>
        (PORT=<listener_port>)))
    (CONNECT_DATA= (SERVICE_NAME=<service_name>)))
```

Note that when Oracle Ultra Search is switched from one Real Application Clusters node to another, the contents of the cache will be lost. After switching instances, force a re-crawl of the documents to re-populate the cache.

---

## A.1.6 Unable to Restore OracleAS Metadata Repository to a Different Host

The backing up and restoration of an OracleAS Metadata Repository using the Backup and Recovery Tool from one host to another fails if the ORACLE SID in the new host is different from that of the old host.

### Problem

The Backup and Recovery Tool does not work with different ORACLE SID values.

The following is an example of the error message that appears when the restoration fails due to an inconsistent ORACLE SID:

Assume two nodes: A and B. The OracleAS Metadata Repository in machine A is backed up using the Backup and Recovery Tool. When attempting to restore it on machine B using the same tool, the following message appears:

```
Oracle instance started
RMAN-00571: =====
RMAN-00569: ===== ERROR MESSAGE STACK FOLLOWS =====
RMAN-00571: =====
RMAN-00579: the following error occurred at 09/08/2003 16:29:15
RMAN-06003: ORACLE error from target database: ORA-01103: database name 'M16REP1' in controlfile is
not 'M16MR2'
RMAN-06097: text of failing SQL statement: alter database mount
RMAN-06099: error occurred in source file: krmk.pc, line: 4124
```

Notice that "M16REP1" is the ORACLE SID of the database that was backed up.

### Solution

Non at this time. Restoring the OracleAS Metadata Repository to a database with a different ORACLE SID is currently not supported.

## A.1.7 Cannot Connect to Database for Restoration (Windows)

Unable to connect to idle OracleAS Metadata Repository database to restore it after it is shutdown using Microsoft Cluster Administrator.

### Problem

When you stop the OracleAS Metadata Repository database using Microsoft Cluster Administrator, the latter performs the strictest and fastest abort that shuts down the database Windows service. After the shutdown, attempts to connect to the service fail.

The following steps can be used to illustrate the problem:

1. Access an OracleAS Metadata Repository that is used for testing.
2. Corrupt a database file (note: do not modify the `ts$` table).
3. Issue a SQL query to ensure that the database is corrupted.
4. Using Microsoft Cluster Administrator, verify that the database is online.
5. Using Oracle Fail Safe Manager, disable database polling.
6. Using Microsoft Cluster Administrator, take the database offline. This also takes OPMN and Application Server Control Console offline as they are dependencies of the database.
7. Try connecting as `sysdba`. The connection should fail.

**Solution**

Use the Oracle Failsafe Manager to shutdown the database. To do so:

1. In the Oracle Failsafe Manager, right click on the "ASDB" resource (default if not changed), and select "immediate".
2. Start the database service using the Windows Service Manager
3. Connect to the database as `sysdba`. The connection should be successful.

### A.1.8 Unpredictable Behavior from Oracle Application Server Cluster (Identity Management) Configuration

Unpredictable behavior from OracleAS Cluster (Identity Management) nodes if system time on all nodes is not synchronized.

**Problem**

In a OracleAS Cluster (Identity Management) configuration, the Oracle Internet Directory Monitor (OIDMON) on each node updates the directory database every 10 seconds with metadata. At the same time, it queries the database to verify that all other directory servers are running.

If an OIDMON does not update the database for 250 seconds, the other nodes assume that that node has failed. This delay can be manifested erroneously by nodes with their system clocks set with a difference of more than 250 seconds from the other nodes. When this happens, OIDMON on one of the other nodes will initiate failover operations, which include locally bringing up processes that were running on the failed node. The node where these processes are started continue processing the operations that were underway in the failed node.

As an example, assume a OracleAS Cluster (Identity Management) configuration with nodes A and B. The system clock in node B is 300 seconds behind node A's clock. Node B updates its metadata in the directory database, which includes the system clock value. Node A queries the database for active Oracle Internet Directory servers and determines that node B has failed because its time value is 300 seconds. Node A then initiates failover operations by locally starting all Oracle Internet Directory server processes that were running on node B.

**Solution**

The system clock value on all nodes in the OracleAS Cluster (Identity Management) configuration should be synchronized using Greenwich mean time so that there is a discrepancy of no more than 250 seconds between them.

Refer to the chapters on Rack-Mounted directory server configurations in the *Oracle Internet Directory Administrator's Guide*.

### A.1.9 Wrong Name Specified for Load Balancer

If a load balancer is deployed in front of OracleAS instances that are clustered together, configuration files of the instances may not have the correct load balancer virtual server name specified.

**Problem**

For a cluster of OracleAS instances front-ended by a load balancer, a redirect back to the cluster may not contain the load balancer virtual server name. Dynamic pages created by a servlet or JSP may also not use the correct load balancer virtual server name. In both cases, the local hostname is most likely used instead.

---

To correctly specify the load balancer virtual server name to be used, modifications have to be made to the `httpd.conf` and `default-web-site.xml` file for each instance.

### Solution

At **each** OracleAS instance, perform the following instructions:

1. Perform the following steps for Oracle HTTP Server:
  - a. Stop the Oracle HTTP Server using the following command:

```
opmnctl stopproc ias_component=HTTP_Server
```
  - b. In Oracle HTTP Server's `httpd.conf` file, change the value for the directive `ServerName` to the virtual server name of your load balancer. For example, if you use `"localhost"`, change it to the virtual server name of your load balancer.
  - c. In the same `httpd.conf` file, change the value of the `Port` directive to the port number your load balancer is configured with for incoming requests. For example, if the port number specified is `7777`, change it to port `80` if that is configured on your load balancer.
  - d. Execute the following command to update the DCM repository with the above changes:

```
dcmctl updateConfig -ct ohs
```
  - e. Start the Oracle HTTP Server using the following command:

```
opmnctl startproc ias_component=HTTP_Server
```
2. Perform the following steps for OC4J:
  - a. Stop the OC4J processes for each OracleAS instance using the following command:

```
opmnctl stopproc ias_component=OC4J
```
  - b. Edit the file `default-web-site.xml` to include the following line:

```
<frontend host="load_balancer_name" port="port_number" />
```

Replace `"load_balancer_name"` with the virtual server name of your load balancer and `"port_number"` with the port number that is configured for incoming requests in your load balancer (these values are similar to those you entered for `httpd.conf` above).
  - c. Execute the following command to update the DCM repository with the changes you made in the `default-web-site.xml` file:

```
dcmctl updateconfig -ct oc4j
```
  - d. Start the OC4J instances using the following command:

```
opmnctl startproc ias_component=OC4J
```

### A.1.10 OracleAS Disaster Recovery: Standby Site Not Synchronized

In the OracleAS Disaster Recovery standby site, you may find that the site's OracleAS Metadata Repository is not synchronized with the OracleAS Metadata Repository in the primary site.

#### **Problem**

The OracleAS Disaster Recovery solution requires manual configuration and shipping of data files from the primary site to the standby site. Also, the data files (archived database log files) are not applied automatically in the standby site, that is, OracleAS Disaster Recovery does not use managed recovery in Oracle Data Guard.

#### **Solution**

The archive log files have to be applied manually. The steps to perform this task is found in Chapter 7, "Oracle Application Server Disaster Recovery".

### A.1.11 OracleAS Disaster Recovery: Failure to Bring Up Standby Instances After Failover or Switchover

Standby instances are not started after a failover or switchover operation.

#### **Problem**

IP addresses are used in instance configuration. OracleAS Disaster Recovery setup does not require identical IP addresses in peer instances between the production and standby site. OracleAS Disaster Recovery synchronization does not reconcile IP address differences between the production and standby sites. Thus, if you use explicit IP address xxx.xx.xxx.xx in your configuration, the standby configuration after synchronization will not work.

#### **Solution**

Avoid using explicit IP addresses. For example, in OracleAS Web Cache and Oracle HTTP Server configurations, use ANY or host names instead of IP addresses as listening addresses

### A.1.12 OracleAS Disaster Recovery: Unable to Start Standalone OracleAS Web Cache Installations at the Standby Site

OracleAS Web Cache cannot be started at the standby site possibly due to misconfigured standalone OracleAS Web Cache after failover or switchover.

#### **Problem**

OracleAS Disaster Recovery synchronization does not synchronize standalone OracleAS Web Cache installations.

#### **Solution**

Use the standard Oracle Application Server full CD image to install the OracleAS Web Cache component



---

### A.1.13 OracleAS Disaster Recovery: Standby Site Middle-tier Installation Uses Wrong Hostname

A middle-tier installation in the standby site uses the wrong hostname even after the machine's physical hostname is changed.

#### Problem

Besides modifying the physical hostname, you also need to put it as the first entry in `/etc/hosts` file. Failure to do the latter will cause the installer to use the wrong hostname.

#### Solution

Put the physical hostname as the first entry in the `/etc/hosts` file. See Section 7.2.2, "Configuring Hostname Resolution" on page 7-9 for more information.

### A.1.14 OracleAS Disaster Recovery: Failure of Farm Verification Operation with Standby Farm

When performing a verify farm with standby farm operation, the operation fails with an error message indicating that the middle-tier machine instance cannot be found and that the standby farm is not symmetrical with the production farm.

#### Problem

The verify farm with standby farm operation is trying to verify that the production and standby farms are symmetrical to one another, that they are consistent, and conform to the requirements for disaster recovery.

The verify operation is failing because it sees the middle-tier instance as `mid_tier.<hostname>` and not as `mid_tier.<physical_hostname>`. You might suspect that this is a problem with the environmental variable `_CLUSTER_NETWORK_NAME_`, which is set during installation. However, in this case, it is not because a check of the `_CLUSTER_NETWORK_NAME_` environmental variable setting finds this entry to be correct. However, a check of the contents of the `/etc/hosts` file, indicates that the entries for the middle tier in question are incorrect. That is, all middle-tier installations take the hostname from the second column of the `/etc/hosts` file.

For example, assume the following scenario:

- Two environments are used: `examp1` and `examp2`
- OracleAS Infrastructure (Oracle Identity Management and OracleAS Metadata Repository) is first installed on `examp1` and `examp2` as host `infra`
- OracleAS middle-tier (OracleAS Portal and OracleAS Wireless) is then installed on `examp1` and `examp2` as host `node1`
- Basically, these are two installations (OracleAS Infrastructure and OracleAS middle-tier) on a single node
- Updated the latest `duf.jar` and `backup_restore` files on all four Oracle homes
- Started OracleAS Guard (`asgctl`) on all four Oracle homes (OracleAS Infrastructure and OracleAS middle-tier on two nodes)
- Performed `asgctl` operations: `connect asg, set primary, dump farm`
- Performed `asgctl verify farm with standby farm` operation, but it fails because it sees the instance as `mid-tier.examp1` and not as `mid_tier.node1.us.oracle.com`

A check of the `/etc/hosts` file shows the following entry:

```
123.45.67.890 exampl node1.us.oracle.com node1 infra
```

Then `ias.properties` and `farms` shows the following and the `verify` operation is failing:

```
IASname=midtier_inst.examp1
```

However, the `/etc/hosts` file should actually be the following:

```
123.45.67.890 node1.us.oracle.com node1 infra
```

Then `ias.properties` and `farms` shows the following and the `verify` operation succeeds:

```
IASname=midtier_inst.node1.us.oracle.com
```

### Solution

Check and change the second column entry in your `/etc/hosts` file to match the hostname of the middle-tier node in question as described in the previous explanation.

## A.1.15 OracleAS Disaster Recovery: Sync Farm Operation Returns Error Message

A `sync farm to` operation returns the error message: "Cannot Connect to asdb"

### Problem

Occasionally, an administrator may forget to set the primary database using the `asgctl` command line utility in performing an operation that requires that the `asdb` database connection be established prior to an operation. The following example shows this scenario for a `sync farm to` operation:

```
ASGCTL> connect asg hsunnab13 ias_admin/iastest2
Successfully connected to hsunnab13:7890
ASGCTL>
.
.
.
<Other asgctl operations may follow, such as verify farm, dump farm,
<and show operation history, and so forth that do not require the connection
<to the asdb database to be established or a time span may elapse of no activity
<and the administrator may miss performing this vital command.
.
.
.
ASGCTL> sync farm to usunnaa11
prodinfra(asr1012): Synchronizing each instance in the farm to standby farm
prodinfra: -->ASG_ORACLE-300: ORA-01031: insufficient privileges
prodinfra: -->ASG_DUF-3700: Failed in SQL*Plus executing SQL statement: connect
null/*****@asdb.us.oracle.com as sysdba;.
prodinfra: -->ASG_DUF-3502: Failed to connect to database asdb.us.oracle.com.
prodinfra: -->ASG_DUF-3504: Failed to start database asdb.us.oracle.com.
prodinfra: -->ASG_DUF-3027: Error while executing Synchronizing each instance in
the farm to standby farm at step - init step.
```

---

### **Solution**

Perform the `asgctl set primary database` command. This command sets the connection parameters required to open the asdb database in order to perform the `sync farm to operation`. Note that the `set primary database` command must also precede the `instantiate farm to command` and `switchover farm to command` if the primary database has not been specified in the current connection session.

## **A.2 Need More Help?**

In case the information in the previous section is not sufficient, you can find more solutions on Oracle *MetaLink*, <http://metalink.oracle.com>. If you do not find a solution for your problem, log a service request.

### **See Also:**

- *Oracle Application Server Release Notes*, available on the Oracle Technology Network:  
<http://www.oracle.com/technology/documentation/index.html>



---

---

## Manually Managed Oracle Application Server Cluster

Clustering components of a system allows the components to be viewed, functionally, as a single entity from the perspective of a client. This appendix describes the procedures you use to create and configure Manually Managed OracleAS Clusters in the Oracle Application Server middle-tier.

This appendix covers the following topics:

- Section B.1, "Overview of Manually Managed OracleAS Cluster"
- Section B.2, "Configuring Manually Managed OracleAS Cluster"

---

---

**Note:** This appendix only covers information on Manually Managed Oracle Application Server Clusters. Read this appendix when using a DCM-Managed OracleAS Cluster is either not desirable or not feasible.

---

---

**See Also:**

- Chapter 2, "Oracle Application Server High Availability Framework"
- Chapter 3, "Middle-tier High Availability"
- Chapter 5, "Oracle Application Server Infrastructure High Availability"

### B.1 Overview of Manually Managed OracleAS Cluster

Oracle Application Server 10g consists of many components that can be deployed in distributed topologies. Clustering, which unites various Oracle Application Server components to provide scalable and unified functionality with redundancy should any individual components fail, enables high availability for Oracle Application Server.

Review the following terms from Section 1.2.1, "Terminology" on page 1-4 before reading this appendix:

- Oracle Application Server Instance
- Oracle Application Server Farm
- Oracle Application Server Cluster
- Oracle Application Server Cluster (OC4J)

In addition, we use the following term in this appendix:

**Component Instance:** Component instances include a single Oracle HTTP Server process or multiple Oracle Application Server Containers for J2EE (OC4J) instances.

### B.1.1 Oracle Application Server Manually Managed Clusters

The Oracle Application Server components that constitute an OracleAS Cluster include:

- Oracle Process Manager and Notification Server (OPMN) which provides process death detection and process restarting in the event that problems are detected for monitored processes, and channels notifications between the different processes.
- Distributed Configuration Management (DCM) distributes the configuration information across the components in the OracleAS Cluster, and also saves the configuration to the repository.
- DCM Repository is a vital part of the whole OracleAS Infrastructure that provides the Management Services. The DCM Repository stores the configuration information for the Oracle Application Server Instances, OracleAS Cluster and the whole OracleAS Farm.
- mod\_oc4j plugs into Oracle HTTP Server, provides configurable and intelligent routing for incoming requests to all OC4J instances using AJP. Requests are routed only to processes and components that mod\_oc4j determines to be alive, through communication with OPMN.

### B.1.2 What Are Manually Managed OracleAS Clusters?

Manually Managed Oracle Application Server Clusters rely on the administrators to manually configure each instance within the OracleAS Cluster. With a Manually Managed Oracle Application Server Cluster, it is the administrator's job to make a group of application server instances function as an OracleAS Cluster.

Manually Managed OracleAS Clusters provide the following load balancing and high-availability services:

- Load-balance requests across all instances of the application server in the Manually Managed OracleAS Cluster.
- Replicate session state across all clustered instances in the Manually Managed OracleAS Cluster.
- In the event of a node or container failure, transparently fail-over requests to a surviving node or container in the Manually Managed OracleAS Cluster.

Using Manually Managed OracleAS Clusters, each Oracle Application Server Instance must be managed independently. The administrator needs to make configuration changes manually to each instance and the changes need to be identical for each instance. Applications deployed to one instance must be individually deployed to all other instances in the Manually Managed OracleAS Cluster.

In essence, a Manually Managed Oracle Application Server Cluster provides scalability and availability, but not manageability. The administrator has the responsibility to synchronize the configuration of the Oracle Application Server Instances across the Manually Managed OracleAS Cluster.

### B.1.3 When Do I Need To Use A Manually Managed OracleAS Cluster

In environments where using a DCM-Managed OracleAS Cluster is either not desirable or not feasible, you have the option to use a Manually Managed OracleAS Cluster.

This section covers the following:

- Section B.1.3.1, "No Database Requirement for Manually Managed OracleAS Cluster"
- Section B.1.3.2, "Tiered Deployment Requirement for Manually Managed OracleAS Cluster"
- Section B.1.3.3, "Tiered Deployment With Security Requirement"

---



---

**Note:** We recommend using a DCM-Managed OracleAS Cluster wherever possible. A DCM-Managed OracleAS Cluster provides manageability along with scalability and availability and takes care of synchronization of application server instance configuration across the OracleAS Cluster.

---



---

#### B.1.3.1 No Database Requirement for Manually Managed OracleAS Cluster

Starting with Oracle Application Server 10g, you have the option of using an OracleAS File-based Farm, which in many cases allows you to avoid using a Manually Managed OracleAS Cluster. In releases prior to Oracle Application Server 10g, a primary reason why people used a Manually Managed OracleAS Cluster was to avoid using a database for storing the DCM Repository. In Oracle Application Server 10g, using an OracleAS File-based Farm, you can configure a DCM-Managed OracleAS Cluster without storing the DCM Repository in a database. To avoid using a database to store DCM configuration repository information, you should use a DCM-Managed OracleAS Cluster with a OracleAS File-based Farm. In this case, a Manually Managed OracleAS Cluster is not required.

#### B.1.3.2 Tiered Deployment Requirement for Manually Managed OracleAS Cluster

Using a DCM-Managed OracleAS Cluster you can also deploy components in a tiered deployment, where the OracleAS Web Cache, Oracle HTTP Server, OC4J and database are running on different tiers. In this case, depending on how you configure each tier, it is possible to use a DCM-Managed OracleAS Cluster instead of a Manually Managed OracleAS Cluster. However, in some cases a tiered deployment does require the use of a Manually Managed OracleAS Cluster.

While you cannot selectively install the Oracle Application Server components that you want to use on a particular tier, you can either stop or disable non-required components on a particular tier. For example, a J2EE & Web Cache type install installs Oracle HTTP Server and OC4J, but if you do not want to run OC4J on that tier, you can either stop or disable the OC4J component in that Oracle Application Server Instance.

If you decide to stop the non-required components on a tier, as compared to disabling the components, then you can use a DCM-Managed OracleAS Cluster.

- **Stopping Non-Required Components**

If you don't have the requirement to disable the tiered deployment non-required Oracle Application Server components on a particular tier, then you can simply stop them and put all your tiers in one OracleAS Farm. Using this configuration option, for a tiered deployment, you can create a DCM-Managed OracleAS Cluster and avoid using a Manually Managed OracleAS Cluster.

- **Disabling Non-Required Components**

In cases where it is a business requirement to disable the non-required tiered deployment components on a particular tier, you need to use a Manually Managed OracleAS Cluster.

Using a DCM-Managed OracleAS Cluster, you cannot selectively disable components in an Oracle Application Server Instance. In this configuration, disabling a component in one Oracle Application Server Instance would also disabled the component in all the Oracle Application Server Instances in the DCM-Managed OracleAS Cluster. For example, if you disable Oracle HTTP Server in one Oracle Application Server Instance, DCM disables Oracle HTTP Servers in all Oracle Application Server Instances that are in the DCM-Managed OracleAS Cluster.

Thus, if it is required to disable the components in a particular tier, then you cannot use a DCM-Managed OracleAS Cluster.

However, using this type of configuration you can put all of the tiers in an OracleAS Farm, as standalone instances that are not part of a DCM-Managed OracleAS Cluster (this can simplify Manually Managed OracleAS Cluster configuration by eliminating the step of associating application server instances that is required to configure a Manually Managed OracleAS Cluster.)

**See Also:** Section B.2.1, "Associating Oracle Application Server Instances Together" on page B-5

- **B.1.3.3 Tiered Deployment With Security Requirement**

In cases where a tiered deployment includes special security requirements, the configuration could prevent the use of a DCM-Managed OracleAS Cluster. In this case, you may need to configure and use a Manually Managed OracleAS Cluster to support a highly available system.

## B.2 Configuring Manually Managed OracleAS Cluster

This section covers the steps you need to follow to configure a Manually Managed OracleAS Cluster.

For this section, we assume the following use case:

- A Manually Managed OracleAS Cluster is required for the purpose of a J2EE application, `myapp`.
- The Manually Managed OracleAS Cluster will have two standalone instances, not part of an OracleAS Farm. The instances are named `inst1` and `inst2`.
- The two Oracle Application Server Instances (`inst1` and `inst2`) are installed on either two different nodes or on one node but in two separate `ORACLE_HOME` directories.
- J2EE application, `myapp` is deployed on an OC4J instance named `home`, that is not used by any of the components internally.

This section covers the following:

- Section B.2.1, "Associating Oracle Application Server Instances Together"
- Section B.2.2, "Configuring OC4J Instances for State Replication"
- Section B.2.3, "Configuring the J2EE Application Properties"



- Section B.2.4, "Configuring Oracle HTTP Server for Failover and Load Balancing"

## B.2.1 Associating Oracle Application Server Instances Together

For this use case we assume that the Oracle Application Server Instances are standalone instances, meaning they are not associated with an OracleAS Farm. But if in your case the Oracle Application Server Instances are associated with an OracleAS Farm then you can skip this section.

1. On both `inst1` and `inst2`, run following command, on UNIX systems:

```
$ORACLE_HOME/dcm/bin/dcmctl getOPMNPort
```

On Windows systems:

```
%ORACLE_HOME%\dcm\bin\dcmctl getOPMNPort
```

This returns the hostname and the ONS remote port for the local application server instance (from the local `ons.conf` file). The output will be of the form:

IPaddress:PortNumber,

For example:

```
127.2.141.148:6200
```

We will refer the output from `inst1` as `<IP of inst1:Port of inst1>` and output from `inst2` as `<IP of inst2:Port of inst2>`.

2. On `inst1`, run following command on UNIX systems:

```
$ORACLE_HOME/dcm/bin/dcmctl addOPMNLink <IP of inst2:Port of inst2>
```

On Windows systems:

```
%ORACLE_HOME%\dcm\bin\dcmctl addOPMNLink <IP of inst2:Port of inst2>
```

3. On `inst2`, run following command on UNIX systems:

```
$ORACLE_HOME/dcm/bin/dcmctl addOPMNLink <IP of inst1:Port of inst1>
```

On Windows systems:

```
%ORACLE_HOME%\dcm\bin\dcmctl addOPMNLink <IP of inst1:Port of inst1>
```

Running these three steps configures the OPMN processes to communicate with each other, allowing OPMN to monitor Oracle Application Server components across the instances.

When you associate Oracle Application Server instances, note the following:

- To run `addOPMNLink`, all instances must be J2EE and Web Cache instances and the instances must not be part of an OracleAS Farm (associated with a repository); otherwise, the command will fail.
- If you would like to change the ONS remote port for an instance in a Manually Managed OracleAS Cluster, you must remove the instance from the cluster using `removeOPMNLink`, change the remote port, and add the instance to the cluster again using `addOPMNLink`. You must repeat the command in every Oracle home that is part of the Manually Managed OracleAS Cluster.
- If you create a cluster and then want to add another instance to the cluster, you must run the command again in all Oracle homes that are part of the Manually Managed Oracle Application Server Cluster.

## B.2.2 Configuring OC4J Instances for State Replication

To assure that Oracle Application Server maintains, across the Manually Managed OracleAS Cluster, the state of stateful applications you need to configure state replication for Web and EJB applications. Replication properties are at an OC4J instance level and not at the J2EE application level, meaning that once enabled these are enabled for all the J2EE applications deployed on the OC4J Instance.

Session state for Web applications is replicated in OC4J islands with the same name across application boundaries and across the OracleAS Cluster. To assure high availability with stateful applications, the OC4J island names must be the same in each OC4J instance across the cluster.

---

---

**Note:** If you do not require session state replication or if your J2EE application is stateless, you can skip this section.

---

---

This section covers the following topics:

- Section B.2.2.1, "Configuring State Replication for Web Applications"
- Section B.2.2.2, "Configuring State Replication for EJB Applications"

### B.2.2.1 Configuring State Replication for Web Applications

To configure state replication for stateful Web applications, do the following on both instances `inst1` and `inst2`:

1. Invoke Application Server Control Console and navigate to the Home Page of the "home" OC4J instance.
2. Select the Administration link.
3. Select Replication Properties in the Instance Properties column.
4. Scroll down to the Web Applications section.
5. Select the Replicate session state checkbox.

Optionally, you can provide the multicast host IP address and port number. If you do not provide the host and port for the multicast address, it defaults to host IP address 230.0.0.1 and port number 9127. The host IP address must be between 224.0.0.2 through 239.255.255.255. Do not use the same multicast address for both HTTP and EJB multicast addresses.

**See Also:** "Configuring Web Application State Replication With OracleAS Cluster (OC4J)" on page 4-24 for more details.

### B.2.2.2 Configuring State Replication for EJB Applications

To configure state replication for EJB applications, do the following on both instances `inst1` and `inst2`:

1. Invoke Application Server Control Console and navigate to the Home Page of the "home" OC4J instance.
2. Select the Administration link.
3. Select Replication Properties in the Instance Properties column.
4. In the EJB Applications section, select the Replicate State checkbox.

Optionally, you can provide the multicast host IP address and port number. If you do not provide the host and port for the multicast address, it defaults to host IP

address 230.0.0.1 and port number 23791. The host IP address must be between 224.0.0.2 through 239.255.255.255. Do not use the same multicast address for both HTTP and EJB multicast addresses.

5. Provide the username and password, which is used to authenticate itself to other hosts in the cluster. If the username and password are different for other hosts in the cluster, they will fail to communicate. You can have multiple username and password combinations within a multicast address. Those with the same username/password combinations will be considered a unique cluster.
6. Provide RMI Server Host name; this is usually the name of the machine where the OC4J instance is running.
7. Click Apply and then OK to confirm.

**See Also:** "Configuring EJB Application State Replication With OracleAS Cluster (OC4J-EJB)" on page 4-26 for more details.

### B.2.3 Configuring the J2EE Application Properties

For J2EE applications to be cluster aware, you need to make the following changes in each of the J2EE applications on instances `inst1` and `inst2`.

1. Add `<distributable>` tag in `web.xml`.

If the Web application is serializable, you must add this tag to the `web.xml` file.

The following shows an example of this tag added to `web.xml`:

```
<web-app>
  <distributable/>
  <servlet>
    ...
  </servlet>
</web-app>
```

2. Add `<cluster-config>` Tag in `orion-web.xml`.

The following shows an example of this tag added to `orion-web.xml`:

```
<orion-web-app ...>
  ...
  <cluster-config/>
</orion-web-app>
```

3. Add replication attribute in `orion-ejb-jar.xml`.

Modify the `orion-ejb-jar.xml` file to add the state replication configuration for stateful session beans. Since you configure the replication type for the stateful session bean within the bean deployment descriptor, each bean can use a different type of replication (either `VMTermination` or `EndOfCall`).

The following shows an example of this attributed added to `orion-ejb-jar.xml`:

```
<session-deployment ... replication="EndOfCall" />
```

## B.2.4 Configuring Oracle HTTP Server for Failover and Load Balancing

The module `mod_oc4j` in the Oracle HTTP Server identifies the requests it needs to act on, determines which OC4J to route those requests to, and communicates with a particular process. Every J2EE (web) application when deployed needs to be associated with a root context. This root context, that is the URL prefix, acts as the identifier of requests that need to be handled by `mod_oc4j`. Requests are routed only to OC4J instances and processes that `mod_oc4j` determines to be alive, through communication with the OPMN.

All `mod_oc4j` related configuration information is kept in the `mod_oc4j.conf` file. Oracle HTTP Server uses an `Oc4jMount` directive to map the root context to the OC4J instance where application was deployed.

Oracle HTTP Server keeps a table of available OC4J instances and load balancing information. To configure a Manually Managed OracleAS Cluster, you need to update the `mod_oc4j.conf` configuration to make `mod_oc4j` aware of the other instances so that Oracle HTTP Servers in the Manually Managed OracleAS Cluster can send requests to the OC4J instances in other Oracle Application Server Instances across the Manually Managed OracleAS Cluster (incase the local OC4J instance goes down).

This section covers the following:

- Section B.2.4.1, "Understanding `mod_oc4j` Request Routing"
- Section B.2.4.2, "Identifying the Instance Names"
- Section B.2.4.3, "Configuring `mod_oc4j` Request Routing"

**See Also:** *Oracle HTTP Server Administrator's Guide*

### B.2.4.1 Understanding `mod_oc4j` Request Routing

Oracle HTTP Server uses the `Oc4jMount` directives defined in `mod_oc4j.conf` file to route requests containing a particular path to a destination. A destination can be a single OC4J process or a set of OC4J instances.

The syntax for the `Oc4jMount` directive is as follows:

```
Oc4jMount path [destination]
```

Where *path* is the context root, it must be the same as the application context root specified during application deployment and where *destination* is one of these types:

```
ajp13_dest
cluster_dest
instance_dest
```

A portion of the `Oc4jMount` section of a default `mod_oc4j.conf` file is shown:

```
Oc4jMount /j2ee/*
Oc4jMount /webapp home
Oc4jMount /webapp/* home
```

**See Also:** *Oracle HTTP Server Administrator's Guide*

### B.2.4.2 Identifying the Instance Names

Identify the fully qualified names of each Oracle Application Server Instances that will be participating in the Manually Managed OracleAS Cluster.

On both `inst1` and `inst2`, run following command, on UNIX systems:

```
$ORACLE_HOME/dcm/bin/dcmctl whichInstance
```

On Windows systems:

```
%ORACLE_HOME%\dcm\bin\dcmctl whichInstance
```

This command returns the name of the local application server instance that you should use in the following section (the output from `inst1` is `<inst1_name>` and output from `inst2` as `<inst2_name>`).

### B.2.4.3 Configuring `mod_oc4j` Request Routing

The default mount points only indicate the local OC4J instance. For this configuration it is necessary to edit the `Oc4jMount` directives to point to each Oracle Application Server Instance and OC4J instance in the Manually Managed OracleAS Cluster.

The following edits should be done after an application is deployed, not before, since part of the deployment process an `Oc4jMount` directive is written to the `mod_oc4j.conf` file. Do the following on both Oracle Application Server Instances `inst1` and `inst2` (or do it only on one instance, for example `inst1`, if you wish to use Oracle HTTP Server of `inst1` and would never use Oracle HTTP Server of `inst2`):

1. Invoke Application Server Control Console and navigate to the Home Page of the "home" OC4J and navigate to Home Page of "HTTP\_Server".
2. Select the Administration link and then select Advance Server Properties.
3. Select `mod_oc4j.conf` in the File Name column.
4. In the editor, scroll down and find the `Oc4jMount` directives for the application you deployed and change these as follows:

```
Oc4jMount /myapp instance://<inst1_name>:home, <inst2_name>:home
Oc4jMount /myapp/* instance://<inst1_name>:home, <inst2_name>:home
```

This configuration automatically makes these instances (`inst1` and `inst2`) fail-over candidates for each other.

5. Click Apply and then OK to confirm.

This step requires restarting, or stopping and then starting the Oracle HTTP Server component.



---

---

## OracleAS Guard Error Messages

The following sections describe the OracleAS Guard error messages. Though not shown, OracleAS Guard error messages are preceded by an ASG prefix. Error messages are categorized into the following groups and subgroups:

- DGA Error Messages
  - LRO Error Messages
  - Undo Error Messages
  - Create Template Error Messages
  - Switchover Physical Standby Error Messages
- Duf Error Messages
  - Database Error Messages
  - Connection and Network Error Messages
  - SQL\*Plus Error Messages
  - JDBC Error Messages
  - OPMN Error Messages
  - Net Services Error Messages
  - System Error Messages
  - Warning Error Messages
  - OracleAS Database Error Messages
  - OracleAS Farm Error Messages
  - OracleAS Backup and Restore Error Messages
  - OracleAS Guard Synchronize Error Messages
  - OracleAS Guard Instantiate Error Messages

### C.1 DGA Error Messages

The following are DGA error messages.

---

---

**Note:** The symbols {0}, {1}, and {2} are variables that will be replaced by the name of the object.

---

---

**12001, Error while creating a DGA template.**

**Cause:** An error occurred while creating a template file.

**Action:** See secondary error.

**12500, Standby database instance {0} already exists on host {1}.**

**Cause:** The standby database instance specified already exists on target host.

**Action:** Either select a new instance or remove the current instance.

## **C.1.1 LRO Error Messages**

The following are LRO error messages.

**13000, Error during Create Physical Standby: Prepare-init.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13001, Error during Create Physical Standby: Prepare-check standby.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13002, Error during Create Physical Standby: Prepare-primary processing.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13003, Error during Create Physical Standby: Prepare-standby processing.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13004, Error during Create Physical Standby: Prepare-sqlnet configuration.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13005, Error during Create Physical Standby: Copy-init.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13006, Error during Create Physical Standby: Copy-validate standby.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13007, Error during Create Physical Standby: Copy-file copy.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13008, Error during Create Physical Standby: Finish-init.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13009, Error during Create Physical Standby: Finish-prepare primary.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.



**13010, Error during Create Physical Standby: Finish-configure primary.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13011, Error during Create Physical Standby: Finish-configure standby.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

## C.1.2 Undo Error Messages

The following are undo error messages.

**13015, Error trying to Undo Create Physical Standby: Prepare.**

**Cause:** Error occurred during undo of the prepare task.

**Action:** See secondary error.

**13016, Error trying to Undo Create Physical Standby: Copy.**

**Cause:** Error occurred during undo of the copy task.

**Action:** See secondary error.

**13017, Error trying to Undo Create Physical Standby: Finish.**

**Cause:** Error occurred during undo of the finish task.

**Action:** See secondary error.

## C.1.3 Create Template Error Messages

The following are create template error messages.

**13020, Error during Create Template: init.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13021, Error during Create Template: primary processing.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13022, Error during Create Template: standby processing.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

**13023, Error during Create Template: finish.**

**Cause:** Error occurred during specified step.

**Action:** See secondary error.

## C.1.4 Switchover Physical Standby Error Messages

The following are switchover physical standby error messages.

**13051, Error performing a physical standby switchover.**

**Cause:** Error occurred in performing a switchover.

**Action:** See secondary error.

**13052, The primary database is not in the proper state to perform a switchover.**

**Cause:** The switchover status of the primary database must be either "TO STANDBY" or "SESSIONS ACTIVE".

**Action:** Make sure the SWITCHOVER\_STATUS of the V\$DATABASE table is either "TO STANDBY" or "SESSIONS ACTIVE".

**13053, The standby database is not in the proper state to perform a switchover.**

**Cause:** The switchover status of the standby database must be either "TO PRIMARY" or "SWITCHOVER PENDING".

**Action:** Make sure the SWITCHOVER\_STATUS of the V\$DATABASE table is either "TO PRIMARY" or "SWITCHOVER PENDING".

**13061, Error failing over physical standby database.**

**Cause:** Error occurred in performing a failover of a standby database.

**Action:** See secondary error.

## C.2 Duf Error Messages

The following are Duf error messages.

**3000, Server error {0}.**

**Cause:** Invalid argument was supplied.

**Action:** Pass in a valid argument.

**3001, Invalid argument {0}.**

**Cause:** Invalid argument was supplied.

**Action:** Pass in a valid argument.

**3002, Invalid log path {0}.**

**Cause:** Invalid log path specification.

**Action:** Specify a valid log path.

**3003, Invalid command line value {0} specified.**

**Cause:** Invalid command line specification.

**Action:** Correct the command line option and retry.

**3004, Invalid command action {0} specified.**

**Cause:** Invalid command action specification.

**Action:** Correct the command line action and retry.

**3005, Invalid command argument {0} specified, commands must begin with a hyphen.**

**Cause:** Command argument did not start with a hyphen.

**Action:** Enter a correct command line argument.

**3006, Command line argument {0} missing a required value.**

**Cause:** Command argument missing a required value.

**Action:** Enter a correct command line argument value.

**3007, Command line argument {0} given an incorrect value {1}.**

**Cause:** Command argument value is incorrect.

- Action:** Enter a correct command line argument value.
- 3008, Command line argument {0} is required but missing.**  
**Cause:** Command argument value is missing.  
**Action:** Enter a correct command line argument value.
- 3009, Invalid session ID.**  
**Cause:** The client passed an invalid session ID.  
**Action:** Enter a correct command line argument value.
- 3010, Duplicate session ID.**  
**Cause:** The session ID is already in use.  
**Action:** Enter a correct command line argument value.
- 3011, Unsatisfied link error for {0} in library DufNatives.**  
**Cause:** An attempt to make a call using the DufNatives library failed.  
**Action:** Make sure the DufNatives library is correctly installed.
- 3012, Checksum error in password.**  
**Cause:** The login password has a checksum error.  
**Action:** Try to reconnect.
- 3013, Operation failed.**  
**Cause:** The specified operation failed.  
**Action:** See secondary error.
- 3014, Invalid command line specified.**  
**Cause:** Invalid command line specification.  
**Action:** Correct command line option and retry.
- 3015, Error getting local host name.**  
**Cause:** Error trying to get local host name.  
**Action:** See secondary error.
- 3016, No encrypt key.**  
**Cause:** No encrypt key supplied. Encryption requires an encrypt key.  
**Action:** This is an internal programming error.
- 3017, Error encrypting data.**  
**Cause:** Failed to encrypt the given data.  
**Action:** See secondary error.
- 3018, Error missing plan for specified request {0}, cannot process.**  
**Cause:** Could not find plan for specified request.  
**Action:** Either specify a valid request or supply valid plan.
- 3019, Server does not recognize application ID.**  
**Cause:** Client specified an application ID that the server does not support.  
**Action:** Contact Oracle support.

**3020, Failed to authenticate user {0}. Please enter the correct user name and password.**

**Cause:** Client supplied incorrect OS user name or password or both.

**Action:** Make sure the correct OS user name and password are used.

**3021, No user name and/or password are supplied for authentication.**

**Cause:** Client did not supply a user name or password or both through the "connect duf" command.

**Action:** Make sure user issue a "connect duf" command before any other commands.

**3022, Failed to authorize user {0}. User must have administrator privilege on the server system.**

**Cause:** The user name provided by the client to connect to DUF server must have administrator privilege on the server system. This error is applicable on Windows system only.

**Action:** Make sure the user account belongs to the administrator group on the server system.

**3023, Error: There is no connection to a DUF server.**

**Cause:** You must connect to DUF server before issues other commands.

**Action:** Connect to the DUF server.

**3024, Failed to authorize user {0}. The owner account of the Oracle Home must be used.**

**Cause:** The user name provided by the client to connect to DUF server must be the same user from which the Oracle home is installed. This error is applicable on UNIX system only.

**Action:** Make sure the user account is the same as that of the Oracle home.

**3025, The operation has been cancelled.**

**Cause:** The operation has been cancelled by either the user or the DUF internal software.

**Action:** None.

**3026, The {0} task must complete successfully before running the {1} task.**

**Cause:** An attempt was made to run the specified task before the required previous task has successfully completed.

**Action:** Rerun the required previous task.

**3027, Error while executing {0} at step - {1}.**

**Cause:** Error during specified step of specified operation.

**Action:** Check secondary error.

**3028, Failed to start DUF server on host {0}.**

**Cause:** Error during specified step of specified operation.

**Action:** Check DUF log file for more information.

**3029, Failed to start {0} server with exception.**

**Cause:** Error trying to start the server.

**Action:** See secondary error.

**3030, Error, cannot resolve host name {0}.**

**Cause:** Error trying to resolve specified host name.

**Action:** Check that the host name is correctly specified.

**3031, Error, Invalid user name {0}. Only {1} account can connect to a DSA server.**

**Cause:** Only ias\_admin can connect to a DSA server.

**Action:** Please use ias\_admin to connect to a DSA server.

**3032, Failed to start {0} server on host {1}. Start server on specified host and reconnect.**

**Cause:** Error trying to start the server on the specified host while trying to connect.

**Action:** Start the server manually and retry the connect.

**3033, Error, the server is shutting down.**

**Cause:** Error communicating with the server.

**Action:** Retry the operation.

**3100, Error reading file {0}.**

**Cause:** Error trying to read from file.

**Action:** See secondary error.

**3101, Error writing file {0}.**

**Cause:** Error trying to write file.

**Action:** See secondary error.

**3102, Error creating file {0}.**

**Cause:** Error trying to create specified file.

**Action:** See secondary error.

**3103, Error deleting file {0}.**

**Cause:** Error trying to delete a file.

**Action:** See secondary error.

**3104, Error opening file {0}.**

**Cause:** Error trying to open file.

**Action:** See secondary error.

**3105, File {0} not found.**

**Cause:** Error trying to open file.

**Action:** See secondary error.

**3106, No read access to file {0}.**

**Cause:** Error trying to open file.

**Action:** See secondary error.

**3107, No write access to file {0}.**

**Cause:** Error trying to open file.

**Action:** See secondary error.

**3108, File specification {0} must be absolute.**

**Cause:** Error trying to open file.

**Action:** See secondary error.

**3109, Error closing file {0}.**

**Cause:** Error trying to close the file.

**Action:** See secondary error.

**3110, Error creating dir {0}.**

**Cause:** Error trying to create specified directory.

**Action:** See secondary error.

**3111, Error deleting dir {0}.**

**Cause:** Error trying to delete specified directory.

**Action:** See secondary error.

**3112, Error expanding file wildcard specification {0}.**

**Cause:** Error trying to process file wildcard specification.

**Action:** See secondary error.

**3120, Error opening configuration file {0}.**

**Cause:** Error trying to open configuration file.

**Action:** Make sure configuration file exists or specified correctly.

**3121, Error creating zip file {0}.**

**Cause:** Error trying to create a zip file.

**Action:** See secondary error.

**3122, There are no files to be zipped.**

**Cause:** The directory to be zipped has no files in it.

**Action:** Make sure the directory to be zipped has files in it.

**3123, Error adding files in directory {0} to zip file.**

**Cause:** Error adding files in the given directory to zip file.

**Action:** See secondary error.

**3124, No zip file is specified.**

**Cause:** No zip file is specified.

**Action:** Internal error.

**3125, Error extracting files from zip file {0}.**

**Cause:** Error extracting files from a zip file.

**Action:** See secondary error.

**3400, Error processing XML document.**

**Cause:** Error processing XML document.

**Action:** See secondary error.

**3401, Error processing XML node.**

**Cause:** Error processing XML node.

**Action:** See secondary error.

**3402, Error parsing XML request message.**

**Cause:** There was an error parsing the XML request message.

**Action:** Contact Oracle support.

**3403, Error parsing XML response message.**

**Cause:** There was an error parsing the XML request message.

**Action:** Contact Oracle support.

**3404, Error parsing XML body string.**

**Cause:** There was an error parsing the XML body string.

**Action:** Contact Oracle support.

**3405, Error writing the body to an XML DOM.**

**Cause:** There was an error writing the XML body string.

**Action:** Contact Oracle support.

**3406, Error reading the body from an XML DOM.**

**Cause:** There was an error reading the XML body string.

**Action:** Contact Oracle support.

**3407, Error writing a work item to an XML DOM.**

**Cause:** There was an error reading the XML body string.

**Action:** Contact Oracle support.

**3408, Error reading a work item from an XML DOM.**

**Cause:** There was an error reading the XML body string.

**Action:** Contact Oracle support.

**3409, Error parsing an XML string.**

**Cause:** There was an error parsing the XML string.

**Action:** Contact Oracle support.

**3410, Error converting XML DOM to string.**

**Cause:** There was an error converting the DOM tree to a XML string.

**Action:** Contact Oracle support.

**3411, Error reading XML DOM tree.**

**Cause:** There was an error reading the XML DOM tree.

**Action:** Contact Oracle support.

## C.2.1 Database Error Messages

The following are database error messages.

**3501, Failed to initialize DufDb class.**

**Cause:** There was an error creating the DufDb class.

**Action:** See secondary error.

**3502, Failed to connect to database {0}.**

**Cause:** There was an error connecting to the database.

**Action:** See secondary error.

**3503, Failed to verify database {0}.**

**Cause:** There was an error verifying the database.

**Action:** See secondary error.

**3504, Failed to start database {0}.**

**Cause:** There was an error starting the database.

**Action:** See secondary error.

**3505, Failed to create pfile to include spfile.**

**Cause:** There was an error creating the given pfile.

**Action:** See secondary error.

**3506, Failed to turn on archivelog mode for the database.**

**Cause:** There was an error turning on archivelog mode.

**Action:** See secondary error.

**3507, Failed to create the standby database control file.**

**Cause:** There was an error creating the standby database control file.

**Action:** See secondary error.

**3508, Failed to create the pfile.**

**Cause:** There was an error creating the database init parameter file.

**Action:** See secondary error.

**3509, Failed to create the spfile.**

**Cause:** There was an error creating the database spfile.

**Action:** See secondary error.

**3510, Output reader thread for {0} terminated.**

**Cause:** The output reader thread is terminated.

**Action:** Contact Oracle support.

**3511, Error creating local worker on node {0}.**

**Cause:** This is an internal error.

**Action:** Contact Oracle support.

**3512, Error creating remote worker on node {0}.**

**Cause:** There is a problem communicating with the remote server.

**Action:** Make sure that the remote server is accessible.

**3513, Database is not started {0}.**

**Cause:** The specified database has not been started.

**Action:** Start the specified database.

**3514, Failed to stop database {0}.**

**Cause:** There was an error stopping the database.

**Action:** See secondary error.

**3515, Failed querying database to determine current archivelog mode.**

**Cause:** There was an error querying the database to determine current archive mode.



- Action:** See secondary error.
- 3516, Failed to query redo log information for database.**  
**Cause:** There was an error querying the database redo log information.  
**Action:** See secondary error.
- 3517, Failed to drop standby redo log for database.**  
**Cause:** There was an error dropping the standby redo log.  
**Action:** See secondary error.
- 3518, Failed to start managed recovery for standby database.**  
**Cause:** There was an error starting managed recovery for the standby database.  
**Action:** See secondary error.
- 3519, Failed to cancel managed recovery for standby database.**  
**Cause:** There was an error cancelling managed recovery for the standby database.  
**Action:** See secondary error.
- 3520, Failed to determine the existence of database instance.**  
**Cause:** There was an error determining the existence of the given database instance.  
**Action:** See secondary error.
- 3521, Invalid database instance {0} specified in the template file; DUF found instance {1}.**  
**Cause:** The standby database instance specified in the template file is different from the one DUF found on the system.  
**Action:** Please either rerun the prepare and copy phases with the new standby instance or specify the correct standby database instance found on the system.
- 3522, The pfile {0} needed to generate an spfile is missing.**  
**Cause:** A pfile needed to create the spfile used by the standby database is missing.  
**Action:** Please either rerun the prepare and copy phases to generate the pfile or manually create one with the correct values.
- 3523, The standby database cannot have the same service name as the primary database.**  
**Cause:** The standby service name is the same as the primary.  
**Action:** Change the standby service name.
- 3524, Error: The primary database is not set.**  
**Cause:** The primary database is not defined.  
**Action:** Set the primary database first.
- 3525, Error: The standby database is not set.**  
**Cause:** The standby database is not defined.  
**Action:** Set the standby database first.
- 3526, Set the primary database before setting the standby database.**  
**Cause:** The standby service name is the same as the primary on the same host.  
**Action:** Change the standby service name.

**3527, The database tablespace map is NULL.**

**Cause:** This is an internal error.

**Action:** Contact Oracle support.

**3528, Error initializing init parameter file {0}.**

**Cause:** An error occurred trying to initialize the parameter file.

**Action:** See secondary error.

**3529, Error writing init parameter file {0}.**

**Cause:** An error occurred trying to write the parameter file.

**Action:** See secondary error.

**3530, Error in setting the protection mode for database {0}.**

**Cause:** An error occurred trying to set the protection mode.

**Action:** See secondary error.

**3531, Error opening database in read only mode for database {0}.**

**Cause:** An error occurred trying to open the database in read only mode.

**Action:** See secondary error.

**3532, Failed to get init parameter value from {0}.**

**Cause:** Error trying to get the parameter value from the init parameter file.

**Action:** See secondary error.

**3533, No user name and/or password is specified for database {0}.**

**Cause:** Error trying to get the parameter value from the init parameter file.

**Action:** User must specify the user name and password to be used to connect to the database using "set primary database" or "set standby database" command.

**3534, The standby database cannot have the same host as the primary database.**

**Cause:** The standby host is the same as the primary.

**Action:** Change the standby or primary database host name.

**3535, Failed to create standby redo log.**

**Cause:** An error occurred trying to create a standby redo log.

**Action:** See secondary error.

**3536, Failed to get a list of standby database(s) from log archive destination.**

**Cause:** An error occurred trying to get a list of standby databases from the log archive destination parameters.

**Action:** See secondary error.

**3537, Failed to add standby database as a log archive destination.**

**Cause:** An error occurred trying to add a standby database as a log archive destination.

**Action:** See secondary error.

**3538, Failed to remove standby database as a log archive destination.**

**Cause:** An error occurred trying to remove a standby database as a log archive destination.

**Action:** See secondary error.

- 3539, Error: The new primary database is not set.**  
**Cause:** The new primary database is not defined.  
**Action:** Set the new primary database first.
- 3540, Error processing template file {0}.**  
**Cause:** Error trying to process template file.  
**Action:** Correct protection and retry operation.
- 3541, Invalid database protection specified in template file {0}.**  
**Cause:** Error trying to process protection value in template file.  
**Action:** Correct protection and retry operation.
- 3542, Failed to query database role.**  
**Cause:** Error trying to query the database role.  
**Action:** See secondary error.
- 3543, Error processing command, must be connected to a OracleAS Guard server in the primary farm.**  
**Cause:** User is connected to server on a farm that is not the primary farm.  
**Action:** Connect to primary farm node.
- 3544, Error processing command, must be connected to a OracleAS Guard server in the standby farm.**  
**Cause:** User is connected to server on a farm that is not the standby farm.  
**Action:** Connect to primary farm node.
- 3550, Failed to find a valid Oracle Home.**  
**Cause:** A valid Oracle home was not found for this operation.  
**Action:** Create a valid Oracle home.
- 3551, Oracle Data Guard Home must have the same owner as the database server home.**  
**Cause:** The Oracle Data Guard Home is owned by a different user than the database server home.  
**Action:** Reinstall Oracle Data Guard user from the owner of Oracle database server.
- 3552, Specified Oracle Home {0} could not be found.**  
**Cause:** The specified Oracle home could not be found.  
**Action:** Please specify a valid Oracle home.
- 3553, An error occurred getting the list of Oracle Homes on the system.**  
**Cause:** The list of Oracle homes could not be read.  
**Action:** Make sure the Oracle inventory is valid.

## C.2.2 Connection and Network Error Messages

The following are connection and network error messages.

- 3600, Error connecting to server: Unknown node {0}.**  
**Cause:** The server host is unknown to the client.  
**Action:** Contact Oracle support.

**3601, Error connecting to server node {0}.**

**Cause:** The client cannot connect to the server.

**Action:** Contact Oracle support.

**3602, File Copy protocol error.**

**Cause:** There was an internal protocol error while copying files.

**Action:** Contact Oracle support

**3603, Error sending data across network.**

**Cause:** There was a network error.

**Action:** Retry operation.

**3604, Error receiving data across network.**

**Cause:** There was a network error.

**Action:** Retry operation.

**3605, The file copy operation has been terminated.**

**Cause:** The copy aborted due to an error.

**Action:** Retry operation.

**3606, Error connecting to file copy server {0} on port {0}.**

**Cause:** The copy server is not running.

**Action:** Contact Oracle support.

**3607, Error opening file copy server socket on {0} with port {0}.**

**Cause:** The copy aborted due to an error.

**Action:** Retry operation.

**3608, Error connecting to clipboard.**

**Cause:** There is no connection to the clipboard server.

**Action:** Retry operation.

**3609, Error while copying {0} to {1}.**

**Cause:** Error occurred during a file copy.

**Action:** See secondary error.

**3610, Error starting online backup.**

**Cause:** Error occurred while putting tablespace in online backup mode.

**Action:** See secondary error.

**3611, Error ending online backup.**

**Cause:** Error occurred while restore tablespace from online backup mode.

**Action:** See secondary error.

**3612, Error listening on server port {0}.**

**Cause:** Error occurred while listening on port.

**Action:** Check if server is already running.

**3613, Network Buffer Overflow Detected.**

**Cause:** The network protocol detected a buffer overflow due to a bug or attack.

**Action:** Call Oracle Support.

### C.2.3 SQL\*Plus Error Messages

The following are SQL\*Plus error messages.

**3700, Failed in SQL\*Plus executing SQL statement: {0}.**

**Cause:** Failed to execute the specified SQL statement.

**Action:** See secondary error.

**3701, Failed starting SQL\*Plus : {0}.**

**Cause:** Failed to execute the specified SQL statement.

**Action:** See secondary error.

### C.2.4 JDBC Error Messages

The following are JDBC error messages.

**3751, Failed to register Oracle JDBC driver: oracle.jdbc.OracleDriver.**

**Cause:** Failed to register the Oracle JDBC driver.

**Action:** Make sure that Oracle JDBC driver is installed on the local system.

**3752, There is no JDBC connection to the database.**

**Cause:** There is no connection to the database server.

**Action:** Connect to a database server first, then try the operation again.

**3753, Failed to connect to the database.**

**Cause:** Unable to connect to the database server.

**Action:** See secondary error.

**3754, Failed to disconnect from the database.**

**Cause:** Unable to disconnect from the database server.

**Action:** See secondary error.

**3755, Failed to execute the SQL statement.**

**Cause:** Failed to execute the SQL statement.

**Action:** See secondary error.

**3756, Failed to run the SQL query.**

**Cause:** Failed to run the SQL query statement.

**Action:** See secondary error.

**3757, Failed to close the Oracle result set or the Statement object.**

**Cause:** Failed to close the Oracle result set or the Statement object.

**Action:** See secondary error.

**3758, This method can not be used to verify the physical standby database.**

**Cause:** This is a programming error.

**Action:** Contact Oracle support.

**3759, Verify DB query returned no data.**

**Cause:** Verify database query returned no data.

**Action:** See secondary error.

**3760, Failed to query the archive log destination information.**

**Cause:** Failed to query the archive log destination information.

**Action:** See secondary error.

**3761, Failed to query the redo log information.**

**Cause:** Failed to query the redo log information.

**Action:** See secondary error.

**3762, Failed to process the results from SQL statement.**

**Cause:** Failed to process the results from the SQL statement.

**Action:** See secondary error.

**3763, Failed to query the data files of the database.**

**Cause:** Failed to query the data files from the database.

**Action:** See secondary error.

**3764, Failed to query the log files used by the database.**

**Cause:** Failed to query the log files used by the database.

**Action:** See secondary error.

**3765, Failed to query table space information.**

**Cause:** Failed to query tablespace information from the database.

**Action:** See secondary error.

## C.2.5 OPMN Error Messages

The following are OPMN error messages.

**3800, Failed trying to connect to OPMN Manager.**

**Cause:** Error trying to connect to OPMN manager.

**Action:** Make sure OPMN manager is started.

**3801, Failed trying to get farm information from OPMN Manager on {0}.**

**Cause:** Error trying to get farm information from OPMN manager.

**Action:** Make sure OPMN manager is started and working correctly.

**3802, Failed trying to stop OPMN Component {0}.**

**Cause:** Failed trying to stop the specified OPMN component.

**Action:** See secondary error.

**3803, Failed trying to start OPMN Component {0}.**

**Cause:** Failed trying to start the specified OPMN component.

**Action:** See secondary error.

## C.2.6 Net Services Error Messages

The following are Net Services error messages.

**4000, Failed trying to get Net Services default domain for {0}.**

**Cause:** Failed trying to get the Net Services default domain.

**Action:** See secondary error.

- 4001, Error trying to add net service name entry for {0}.**  
**Cause:** Failed trying to add the specified service name.  
**Action:** See secondary error.
- 4002, Error trying to get net service name entry for {0}.**  
**Cause:** Failed trying to get the specified service name.  
**Action:** See secondary error.
- 4003, Error trying to get host name from net service entry for {0}.**  
**Cause:** Failed trying to get the host name from the net service entry.  
**Action:** See secondary error.
- 4004, Error trying to get host name from net service description.**  
**Cause:** Failed trying to get the host name from the net service description.  
**Action:** See secondary error.
- 4005, Error trying to get net service listener information.**  
**Cause:** Failed trying to get the net service listener information.  
**Action:** See secondary error.
- 4006, Error trying to create a net service default listener.**  
**Cause:** Failed trying to create a default listener.  
**Action:** See secondary error.
- 4007, Error trying to add SID entry {0} to net service listener {1}.**  
**Cause:** Failed trying to add a SID entry to the listener.  
**Action:** See secondary error.
- 4008, Error generating a command a script for the net service listener command: {0}.**  
**Cause:** Failed generating a command script for the listener.  
**Action:** See secondary error.
- 4009, Error running the command script for the net service listener command: {0}.**  
**Cause:** Failed running the command script for the listener.  
**Action:** See secondary error.
- 4010, Error adding the net service TNS entry for {0}.**  
**Cause:** Failed adding a TNS entry.  
**Action:** See secondary error.
- 4011, Error trying to delete SID entry {0} to the net service listener {1}.**  
**Cause:** Failed trying to delete the SID entry to the listener.  
**Action:** See secondary error.
- 4012, Error trying to save the listener configuration.**  
**Cause:** Listener information was modified and an attempt to save information failed.  
**Action:** See secondary error.
- 4013, Error deleting net service TNS entry for {0}.**  
**Cause:** Failed deleting a TNS entry.

**Action:** See secondary error.

**4014, Error starting the TNS listener using the lsnrctl command.**

**Cause:** Failed to start the TNS listener.

**Action:** See secondary error.

**4030, The command \"{0}\" failed due to timeout.**

**Cause:** Command timed out.

**Action:** Increase timeout values in configuration file.

**4040, Error executing the external program or script.**

**Cause:** The execution of the specified command failed.

**Action:** See secondary error.

**4041, Failed to get the value of {0} from the TNS name descriptor {1}.**

**Cause:** Failed to get the value for the given parameter from the TNS name descriptor.

**Action:** See secondary error.

**4042, Failed to update the value of {0} for the TNS name descriptor {1}.**

**Cause:** Failed to update the value of a given parameter in the TNS name descriptor.

**Action:** See secondary error.

**4043, Failed to compare the TNS descriptor entry {0} with entry {1}.**

**Cause:** Failed to compare the two TNS entries.

**Action:** See secondary error.

**4044, Failed to generate a remote TNS name descriptor for the service name.**

**Cause:** Failed to generate a remote TNS name descriptor for the given local database.

**Action:** See secondary error.

**4045, Failed to get the remote TNS service name for the service name.**

**Cause:** Failed to get the remote TNS service name for the given local database.

**Action:** See secondary error.

## C.2.7 System Error Messages

The following are system error messages.

**4900, An exception occurred on the server.**

**Cause:** A server exception occurred.

**Action:** See secondary error.

**4901, A null pointer exception occurred on the server.**

**Cause:** Software error.

**Action:** See secondary error.

**4902, Object not found in clipboard for key {0}.**

**Cause:** Software error.

**Action:** See secondary error.



## C.2.8 Warning Error Messages

The following are warning error messages.

**15305, Warning: Problem gathering summary information for backup.**

**Cause:** Error during the gatherInfo step of the backup farm operation.

**Action:** Check secondary error.

**15306, Warning during undo processing.**

**Cause:** Error occurred during undo processing.

**Action:** Check secondary error.

## C.2.9 OracleAS Database Error Messages

The following are OracleAS database error messages.

**15601, Error storing DB Credentials in the clipboard of the server.**

**Cause:** Failed to store DB credentials in the clipboard on the specified server.

**Action:** Internal error.

**15602, Error storing DB info in the clipboard of the server.**

**Cause:** Failed to store DB information in the clipboard on the specified server.

**Action:** Internal error.

**15603, Error cleaning up the database on the standby host.**

**Cause:** Failed to clean up the database on the standby host.

**Action:** See secondary error.

**15604, Error finishing up creating the physical standby database.**

**Cause:** Failed to finish creating the standby database.

**Action:** See secondary error.

**15605, Error creating the physical standby database.**

**Cause:** Failed to the create the standby database.

**Action:** See secondary error.

**15606, Failed to perform a sync database operation on the primary farm.**

**Cause:** Failed to perform a sync database operation on the primary farm.

**Action:** See secondary error.

**15607, Failed to perform a sync database operation on the standby farm.**

**Cause:** Failed to perform a sync database operation on the standby farm.

**Action:** See secondary error and log file for more information.

**15608, Invalid backup mode specified in the template file {0}.**

**Cause:** Error trying to process a backup mode value in the template file.

**Action:** Correct backup mode and retry the operation.

**15609, Failed to get database backup files.**

**Cause:** Error trying to get the database backup files.

**Action:** See secondary error.

## C.2.10 OracleAS Farm Error Messages

The following are OracleAS farm error messages.

**15620, An Invalid Farm was specified.**

**Cause:** Error trying to process a farm object.

**Action:** Retrieve a valid farm object.

**15621, Error trying to verify farm {0}.**

**Cause:** The specified farm had an error during the verify operation.

**Action:** See secondary error for more information.

**15622, Error trying to verify instance {0}.**

**Cause:** The specified instance had an error during the verify operation.

**Action:** See secondary error for more information.

**15623, Farm {0} is not symmetrical with farm {1}.**

**Cause:** The specified farms are not symmetrical.

**Action:** See secondary error for more information.

**15624, An Invalid Farm was specified. Farm {0} does not contain any valid instances.**

**Cause:** Error trying to process a farm object. Farm object did not contain a valid instance.

**Action:** Retrieve a valid farm object with at least one instance.

**15625, Could not find matching instance {0} in Farm {1}.**

**Cause:** Could not get matching instances, Farms do not appear to be symmetrical.

**Action:** Make the farms symmetrical.

**15626, Farms are not symmetrical because farm name {0} is not same as farm name {1}.**

**Cause:** Farm names are not the same and therefore farms are not symmetrical.

**Action:** Make the farms symmetrical.

**15627, Instance {0} is not symmetrical because of different Oracle Home names {1}.**

**Cause:** Instance Home names are not symmetrical in the specified farms.

**Action:** Make the farms symmetrical.

**15628, Instance {0} is not symmetrical because of different Oracle Home paths {1}.**

**Cause:** Instance Home paths are not symmetrical in the specified farms.

**Action:** Make the farms symmetrical.

**15629, Instance {0} is not symmetrical, because of different host names {1}, {2}.**

**Cause:** Instance host names are not symmetrical in the specified farms.

**Action:** Make the farms symmetrical.

**15630, The specified instance {0} could not be found.**

**Cause:** The specified instance information could not be found on this node.

**Action:** Either the wrong instance name or host name was specified on the request to the server.

**15631, The primary and standby farms appear to be identical because both have instance {0} on host {1}.**

**Cause:** An instance can only be in a member of one farm, it appears that the primary and standby farms are the same.

**Action:** Specify a primary and separate standby farm.

**15632, The Home that contains instance {0} could not be found.**

**Cause:** The specified instance could not be found in any Home on this node.

**Action:** The Oracle home information on the system is incorrect.

**15633, An Invalid Farm was specified. Farm contains a duplicate instance named {0}.**

**Cause:** Farm information obtained from OPMN contains a duplicate instance.

**Action:** Check OPMN to ensure that the farm information listed is correct.

## C.2.11 OracleAS Backup and Restore Error Messages

The following are OracleAS backup and restore error messages.

**15681, Must specify a backup directory.**

**Cause:** A backup directory must be specified for the operation to complete successfully.

**Action:** Check secondary error.

**15682, Failed to initialize configure file: {0}.**

**Cause:** Failed to initialize the configure file for backup script.

**Action:** Check secondary error.

**15683, The ha directory does not exist in Oracle Home {0}.**

**Cause:** The ha directory does not exist in the OracleAS Oracle home.

**Action:** Make sure the ha directory which contains the backup and restore scripts is copied to the OracleAS Oracle home.

**15684, Failed to generate the configuration file for the backup and restore script.**

**Cause:** Failed to generate the configure file for the backup and restore script.

**Action:** Check secondary error.

**15685, Failed to backup configuration data for instance {0}.**

**Cause:** Failed to backup configuration data for the specified instance.

**Action:** Check secondary error.

**15686, Failed to restore configuration data for instance {0}.**

**Cause:** Failed to restore configuration data for the specified instance.

**Action:** Check secondary error.

**15687, Failed to get the database backup files.**

**Cause:** Failed to get the database backup file names from the log.

**Action:** Check secondary error.

**15688, Error running the config script.**

**Cause:** Failed to run the config script.

**Action:** Check the log file generated by the config script.

**15689, Error running the backup script.**

**Cause:** Failed to run the backup script.

**Action:** Check the log file generated by the backup script.

**15690, Error running the restore script.**

**Cause:** Failed to run the restore script.

**Action:** Check the log file generated by the restore script.

**15691, No zip file was found.**

**Cause:** No zip file was found.

**Action:** Make sure a successful backup has been performed.

**15692, The config file {0} is empty.**

**Cause:** The specified configure file is empty.

**Action:** Copy the original configure file from the "ha" directory where backup restore scripts are located.

**15693, No zip file was specified.**

**Cause:** User did not specify a zip file for the unzip operation.

**Action:** Internal error.

**15694, Error executing step - {0} of Backup farm.**

**Cause:** Backup farm failed at the specified step.

**Action:** Check secondary error.

**15695, Error executing step - {0} of Restore farm.**

**Cause:** Restore farm failed at the specified step.

**Action:** Check secondary error.

**15696, Error initializing backup farm operation.**

**Cause:** Error initializing backup farm operation.

**Action:** Check secondary error.

**15697, Error during backup farm operation - backup step.**

**Cause:** Error during backup step processing of backup farm.

**Action:** Check secondary error.

**15698, Error during backup farm operation - copy step.**

**Cause:** Error during copy step processing of backup farm.

**Action:** Check secondary error.

**15699, Error initializing restore farm operation.**

**Cause:** Error initializing restore farm operation.

**Action:** Check secondary error.

**15700, No backup file was found.**

**Cause:** No backup file was found.

**Action:** Make sure a successful backup has been performed.

**15701, Failed to restore configuration with the DCM-resyncforce option for instance {0}.**

**Cause:** Failed to restore configuration with the DCM-resyncforce option.

**Action:** Check secondary error.

## C.2.12 OracleAS Guard Synchronize Error Messages

The following are OracleAS Guard synchronize error messages.

**15721, Failed to initialize a DUF database object.**

**Cause:** Failed to initialize a DufDb object.

**Action:** Check secondary error.

**15722, No farm information is available to perform the farm operation.**

**Cause:** No farm information is available to perform the farm operation.

**Action:** Check secondary error.

**15723, No instances are found in the farm's backup list.**

**Cause:** The farm's backup list is empty.

**Action:** Check secondary error.

**15724, Failed to get the standby host list.**

**Cause:** Failed to get the standby host list.

**Action:** Check secondary error.

**15725, Failed to backup OracleAS configuration data for farm {0}.**

**Cause:** Failed to backup OracleAS farm configuration data.

**Action:** Check secondary error.

**15726, Failed to restore OracleAS configuration data for farm.**

**Cause:** Failed to restore OracleAS farm configuration data.

**Action:** Check secondary error.

**15727, Failed to backup OracleAS infrastructure database {0}.**

**Cause:** Failed to backup OracleAS farm infrastructure database.

**Action:** Check secondary error.

**15728, Failed to restore OracleAS infrastructure database {0}.**

**Cause:** Failed to restore OracleAS farm infrastructure database.

**Action:** Check secondary error.

**15729, Failed to perform the sync farm operation.**

**Cause:** Failed to perform the sync farm operation.

**Action:** Check secondary error.

## C.2.13 OracleAS Guard Instantiate Error Messages

The following are OracleAS Guard instantiate error messages.

**15751, Error executing step {0} of instantiate farm operation.**

**Cause:** The instantiate farm operation failed at the specified step.

**Action:** Check secondary error.

**15752, Failed to load remote farm information.**

**Cause:** Failed to load remote farm information.

**Action:** Make sure the user specified the correct host name for the farm and that the OPMN processes are running on the farms.

**15753, Error preparing to instantiate farm on host {0}.**

**Cause:** Error preparing to instantiate farm.

**Action:** Check secondary error.

**15754, Error instantiating database {0}.**

**Cause:** Error instantiating the database.

**Action:** Check secondary error.

**15755, Error finishing up instantiating database {0}.**

**Cause:** Error finishing up instantiating the database.

**Action:** Check secondary error.

**15756, Error initializing instantiate farm operation.**

**Cause:** Error initializing the instantiate farm operation.

**Action:** Check secondary error.

---

---

## Manual Sync Operations

The following manual sync operations must be performed if for some reason the secondary (standby) site is not synchronized with the primary site and you are performing regular backup operations of the primary site middle tier and OracleAS Infrastructure configuration files as described in Section D.1.1, "Manually Backing Up the Production Site". Then you will need to restore the backup configuration files as described in Section D.1.2, "Manually Restoring to Standby Site". After restoring the configuration files (OracleAS Infrastructure and Middle Tier) on the standby site, then proceed to Step 2 as described in Section 7.6.1.2.1, "Site Failover Operations".

### D.1 Manually Synchronizing Baseline Installation with Standby Site Without Using OracleAS Guard `asgctl` Command-line Utility

---

---

**Note:** This section and Section D.1.1, "Manually Backing Up the Production Site" and Section D.1.2, "Manually Restoring to Standby Site" are retained here for the special case as described in Step 1b in Section 7.6.1.2.1, "Site Failover Operations" where the standby site is not synchronized with the primary site. In this case, on the standby site, you must restore the most recently backed up configuration files as described in Section D.1.2, "Manually Restoring to Standby Site".

If you are using `asgctl` to continually synchronize the secondary (standby) site with the primary site, then both sites should already be synchronized and you do not need to manually perform a restore operation and you can begin with Step 2 in Section 7.6.1.2.1, "Site Failover Operations" to recover from an unplanned outage.

---

---

Once Oracle Data Guard has been set up between the production and standby sites, the procedure for synchronizing the two sites can be carried out. An initial synchronization should be done, before the production site is used, in order to obtain a baseline snapshot of the post-installation production site onto the standby site. This baseline can then be used to recover the production site configuration on the standby site if needed later.

In order to obtain a consistent point-in-time snapshot of the production site, the information stored in the OracleAS Infrastructure database and the Oracle Application Server-related configuration files in the middle-tier and OracleAS Infrastructure hosts must be synchronized at the same time. Synchronization of the configuration files can be done by backing up the files and restoring them on the standby hosts using the Oracle Application Server Backup and Recovery Tool. For the OracleAS Infrastructure database, synchronization is done using Oracle Data Guard by shipping the archive

logs to the standby OracleAS Infrastructure and applying these logs in coordination with the restoration of the configuration files.

The sequence of steps for the baseline synchronization (which can also be used for future synchronizations) are:

- Shipping OracleAS Infrastructure Database Archive Logs
- Backing Up Configuration Files (OracleAS Infrastructure and Middle Tier)
- Restoring Configuration Files (OracleAS Infrastructure and Middle Tier)
- Restoring the OracleAS Infrastructure Database - Applying Log Files

These steps are detailed in the following two main sections.

### D.1.1 Manually Backing Up the Production Site

The main strategy and approach to synchronizing configuration information between the production and standby sites is to synchronize the backup of OracleAS Infrastructure and middle-tier configuration files with the application of log information on the standby OracleAS Infrastructure database.

For Oracle Application Server, not all the configuration information is in the OracleAS Infrastructure database. The backup of the database files needs to be kept synchronized with the backup of the middle-tier and OracleAS Infrastructure configuration files. Due to this, log-apply services should not be enabled on the standby database. The log files from the production OracleAS Infrastructure are shipped to the standby OracleAS Infrastructure but are not applied.

The backup process of the production site involves backing up the configuration files in the middle-tier and OracleAS Infrastructure nodes. Additionally, the archive logs for the OracleAS Infrastructure database are shipped to the standby site.

The procedures to perform the backups and the log ship are discussed in the following sections:

- Shipping OracleAS Infrastructure Database Archive Logs
- Backing Up Configuration Files (OracleAS Infrastructure and Middle Tier)

---

---

**IMPORTANT:** Ensure that no configuration changes are going to be made to the Oracle Application Server system (underlying configuration files and OracleAS Infrastructure database) as you perform the steps in this section.

---

---

---

---

**Note:** At the minimum, the backup and restoration steps discussed in this section and the "Manually Restoring to Standby Site" section should be performed whenever there is any administration change in the production site (inclusive of changes to the OracleAS Infrastructure database and configuration files on the middle-tier and OracleAS Infrastructure nodes). On top of that, scheduled regular backups and restorations should also be done (for example, on a daily or twice weekly basis). See the *Oracle Application Server Administrator's Guide* for more backup and restore procedures.

---

---



### D.1.1.1 Shipping OracleAS Infrastructure Database Archive Logs

After installing the OracleAS Disaster Recovery solution, Oracle Data Guard should have been installed in both the production and standby databases. The steps for shipping the archive logs from the production OracleAS Infrastructure database to the standby OracleAS Infrastructure database involve configuring Oracle Data Guard and executing several commands for both the production and standby databases. Execute the following steps to ship the logs for the OracleAS Infrastructure database:

1. If not disabled already, disable log-apply services by running the following SQLPLUS statement on the standby host:

```
SQL> alter database recover managed standby database cancel;
```

2. Run the following command to perform a log switch on the production OracleAS Infrastructure database. This ensures that the latest log file is shipped to the standby OracleAS Infrastructure database

```
SQL> alter system switch logfile;
```

3. In normal operation of the production site, the production database frequently ships log files to the standby database but are not applied. At the standby site, you want to apply the logs that are consistent up to the same time that the production site's configuration files are backed up. The following SQL statement encapsulates all OracleAS Infrastructure database changes into the latest log and allows the Oracle Data Guard transport services to transport this log to the OracleAS Infrastructure in the standby site:

```
SQL> select first_change# from v$log where status='CURRENT';
```

A SCN or sequence number is returned, which essentially represents the timestamp of the transported log.

4. Note down the SCN number as you will need this for the restoration of the production database changes on the standby site.

Continue to the next section to back up the configuration files on the middle-tier host(s) and OracleAS Infrastructure host.

### D.1.1.2 Backing Up Configuration Files (OracleAS Infrastructure and Middle Tier)

Use the instructions in this section to back up the configuration files. The instructions require the use of the Oracle Application Server Backup and Recovery Tool. They assume you have installed and configured the tool on each OracleAS installation (middle tier and OracleAS Infrastructure) as it needs to be customized for each installation. Refer to *Oracle Application Server Administrator's Guide* for more details about that tool, including installation and configuration instructions.

For each middle-tier and OracleAS Infrastructure installation, perform the following steps (the same instructions can be used for the middle-tier and OracleAS Infrastructure configuration files):

1. After performing the installation and configuration steps detailed in the *Oracle Application Server Administrator's Guide*, for the Oracle Application Server Backup and Recovery Tool, the variables `oracle_home`, `log_path`, and `config_backup_path` in the tool's configuration file, `config.inp`, should have the appropriate values. Also, the following command for the tool should have been run to complete the configuration:

```
perl bkp_restore.pl -m configure_nodb
```

In Windows, the Perl executable can be found in `<ORACLE_HOME>\perl\<perl_version>\bin\MSWin32-x86`.

If you have not completed these tasks, do so before continuing with the ensuing steps.

2. Execute the following command to back up the configuration files from the current installation:

```
perl bkp_restore.pl -v -m backup_config
```

This command creates a directory in the location specified by the `config_backup_path` variable specified in the `config.inp` file. The directory name includes the time of the backup. For example: `config_bkp_2003-09-10_13-21`.

3. A log of the backup is also generated in the location specified by the `log_path` variable in the `config.inp` file. Check the log files for any errors that may have occurred during the backup process.
4. Copy the OracleAS Backup and Recovery Tool's directory structure and contents from the current node to its equivalent in the standby site. Ensure that the path structure on the standby node is identical to that on the current node.
5. Copy the backup directory (as defined by `config_backup_path`) from the current node to its equivalent in the standby site. Ensure that the path structure on the standby node is identical to that on the current node.
6. Repeat the steps above for each Oracle Application Server installation in the production site (middle tier and OracleAS Infrastructure).

---

---

**Note:** There are two important items that should be maintained consistently between the production and standby sites. The directory names should be the same and the correlation of SCN to a given backup directory should be noted at both sites in administration procedures.

---

---

## D.1.2 Manually Restoring to Standby Site

After backing up the configuration files from the middle-tier Oracle Application Server instances and OracleAS Infrastructure together with the OracleAS Infrastructure database, restore the files and database in the standby site using the instructions in this section, which consists of the following sub-sections:

- Restoring Configuration Files (OracleAS Infrastructure and Middle Tier)
- Restoring the OracleAS Infrastructure Database - Applying Log Files

### D.1.2.1 Restoring Configuration Files (OracleAS Infrastructure and Middle Tier)

Restoring the backed up files from the production site requires the Oracle Application Server Backup and Recovery Tool that was used for the backup. The instructions in this section assume you have installed and configured the tool on each OracleAS installation in the standby site, both in the middle-tier and OracleAS Infrastructure nodes. Refer to *Oracle Application Server Administrator's Guide* for instructions on how to install the tool.

For each middle-tier and OracleAS Infrastructure installation in the standby site, perform the following steps (the same instructions can be used for the middle-tier and OracleAS Infrastructure configuration files):

1. Check that the OracleAS Backup and Recovery Tool's directory structure and the backup directory from the equivalent installation in the production site are present in the current node.
2. Stop the Oracle Application Server instances and their processes so that no modification of configuration files can occur during the restoration process. Use the following OPMN command:

In UNIX:

```
<ORACLE_HOME>/opmn/bin/opmnctl stopall
```

In Windows:

```
<ORACLE_HOME>\opmn\bin\opmnctl stopall
```

Check that all relevant processes are no longer running. In UNIX, use the following command:

```
ps -ef | grep <ORACLE_HOME>
```

In Windows, press <ctrl><alt><del> to bring up the Task Manager and verify that the processes have stopped.

3. Configure the backup utility for the Oracle home.

This can be accomplished either by configuring the OracleAS Backup and Recovery Tool for the Oracle home or copying the backup configuration file, `config.inp`, from the production site peer. Below is an example of running the OracleAS Backup and Recovery Tool configuration option:

```
perl bkp_restore.pl -v -m configure_nodb
```

In Windows, the Perl executable can be found in `<ORACLE_HOME>\perl\<perl_version>\bin\MSWin32-x86`.

4. Execute the following command to view a listing of the valid configuration backup locations:

```
perl bkp_restore.pl -v -m restore_config
```

5. Restore the configuration files using the following command:

```
perl bkp_restore.pl -v -m restore_config -t <backup_directory>
```

where `<backup_directory>` is the name of the directory with the backup files that was copied from the production site. For example, this could be `config_bkp_2003-09-10_13-21`.

6. Check the log file specified in `config.inp` for any errors that may have occurred during the restoration process.
7. Repeat the steps above for each Oracle Application Server installation in the production site (middle tier and OracleAS Infrastructure).

### D.1.2.2 Restoring the OracleAS Infrastructure Database - Applying Log Files

During the backup phase, you executed several instructions to ship the database log files from the production site to the standby site up to the SCN number that you recorded as per instructed. To restore the standby database to that SCN number, apply the log files to the standby OracleAS Infrastructure database using the following SQLPLUS statement:

```
SQL> alter database recover automatic from '/private/oracle/oracleas/standby/' standby
```

```
database until change <SCN>;
```

(In Windows, substitute the path shown above appropriately.)

With this command executed and the instructions to restore the configuration files completed on each middle-tier and OracleAS Infrastructure installation, the standby site is now synchronized with the production site. However, there are two common problems that can occur during the application of the log files: errors caused by the incorrect specification of the path and gaps in the log files that have been transported to the standby site.

The following are methods of resolving these problems:

**1. Find the correct log path.**

On the standby OracleAS Infrastructure database, try to determine location and number of received archive logs using the following SQLPLUS statement:

```
SQL> show parameter standby_archive_dest
```

NAME	TYPE	VALUE
standby_archive_dest	string	/private/oracle/oracleas/standby/

(The previous example shows the UNIX path. The Windows equivalent path is shown in Windows systems.)

**2. Use the log path obtained from the previous step to ensure that all log files have been transported.**

At the standby OracleAS Infrastructure database, perform the following:

```
standby> cd /private/oracle/oracleas/standby
standby> ls
1_13.dbf 1_14.dbf 1_15.dbf 1_16.dbf 1_17.dbf 1_18.dbf 1_19.dbf
```

(In Windows, use the command `cd` to change to the appropriate directory and `dir` to view the directory contents.)

At the production OracleAS Infrastructure database, execute the following SQLPLUS statement:

```
SQL> show parameter log_archive_dest_1
```

NAME	TYPE	VALUE
log_archive_dest1	string	LOCATION=/private/oracle/oracleas/oradata
MANDATORY		
log_archive_dest_10	string	

(The previous example shows the UNIX path. The Windows equivalent path is shown in Windows systems.)

**3. Using the path specified in step 1, note the number and sequence of the log files. For example:**

```
production> cd /private/oracle/oracleas/oradata
production> ls
1_10.dbf 1_12.dbf 1_14.dbf 1_16.dbf 1_18.dbf asdb
1_11.dbf 1_13.dbf 1_15.dbf 1_17.dbf 1_19.dbf
```

(In Windows, use the command `cd` to change to the appropriate directory and `dir` to view the directory contents.)

In the previous example, note the discrepancy where the standby OracleAS Infrastructure is missing files `1_10.dbf` through `1_12.dbf`. Since this gap in the log files happened in the past, it could be due to a problem with the historic setup involving the network used for the log transport. This problem has obviously been corrected and subsequent logs have been shipped. To correct the problem, copy (FTP) the log files to the corresponding directory on the standby OracleAS Infrastructure database host and re-attempt the SQLPLUS recovery statement shown earlier in this section.



---

---

## Setting Up a DNS Server

This appendix provides instructions on setting up a DNS server in UNIX. These instructions are applicable for setting up the site-specific DNS zones used for hostname resolution in the example in Figure 7-3, "DNS resolution topology overview".

---

---

**Note:** The DNS setup information provided in this appendix is an example to aid in the understanding of OracleAS Disaster Recovery operations. It is generic to DNS, and other appropriate DNS documentation should be consulted for comprehensive DNS information.

---

---

For the discussion in this chapter, the DNS server that is set up creates and services a new DNS zone with the unique domain `oracleas`. Within the zone, this DNS server resolves all requests for the `oracleas` domain and forwards other requests to the overall wide area company DNS server(s).

On the UNIX host that will act as the DNS zone server, perform the following steps:

1. Create the name server configuration file `/var/named.conf`. Assuming the wide area company DNS server IP address is `123.1.15.245`, the contents of this file should be as follows:

```
options {
    directory "/var/named";
    forwarders {
        123.1.15.245;
    };
};

zone "." in {
    type hint;
    file "named.ca";
};

zone "oracleas" {
    type master;
    file "oracleas.zone";
};

zone "0.0.127.IN-ADDR.ARPA" {
    type master;
    file "127.zone";
};
```

- 
2. Create the root hint file `/var/named/named.ca`, which has the following contents (123.1.2.117 is the IP of the zone DNS server):

```
.          999999   IN       NS       ourroot.private.
ourroot.private.  IN       A        123.1.2.117
```

3. Create the loopback address file `/var/named/127.zone`, which has the following contents (assume the zone DNS server's hostname is `aszone1`):

```
$ORIGIN    0.0.127.IN-ADDR.ARPA.
0.0.127.IN-ADDR.ARPA.  IN       SOA     aszone1.oracleas.  root.aszone1.oracleas.
(
    25          ; serial number
    900         ; refresh
    600         ; retry
    86400       ; expire
    3600        ) ; minimum TTL

0.0.127.IN-ADDR.ARPA.  IN       NS      aszone1.oracleas.
1                      IN       PTR     localhost.oracleas.
```

4. Create the zone data file `/var/named/oracleas.dns`, which has the following contents (values shown are applicable to the example of the production site in Figure 7-3):

```
;
; Database file oracleas.dns for oracleas zone.
; Zone version: 25
;
$ORIGIN oracleas.
oracleas.      IN       SOA     aszone1.oracleas.  root.aszone1.oracleas (
    25          ; serial number
    900         ; refresh
    600         ; retry
    86400       ; expire
    3600        ) ; minimum TTL

;
; Zone NS records
;
oracleas.      IN       NS      aszone1.oracleas.

;
; Zone records
;
localhost      IN       A        127.0.0.1

asmid1         IN       A        123.1.2.333
asmid2         IN       A        123.1.2.334
infra          IN       A        123.1.2.111
remoteinfra    IN       A        213.2.2.210
```

5. Run the following command to start the name server:

```
/sbin/in.named
```

6. On all the hosts in the domain that is serviced by this DNS server, edit the domain and nameserver settings in the file `/etc/resolv.conf` as follows (all previous nameserver settings should be removed; 123.1.2.117 is assumed to be the zone DNS server's IP address):



---

```
domain oracleas  
nameserver 123.1.2.117
```



---

---

# Secure Shell (SSH) Port Forwarding

This appendix describes how secure shell (SSH) port forwarding may be used with Oracle Data Guard.

## F.1 SSH Port Forwarding

OracleAS Guard automates the use of Oracle Data Guard, which sends redo data across the network to the standby system using Oracle Net. SSH tunneling may be used with Oracle Data Guard as an integrated way to encrypt and compress the redo data before it is transmitted by the production system and subsequently decrypt and uncompress the redo data when it is received by the standby system.

**See Also:**

- Implementing SSH port forwarding with Data Guard:  
<http://metalink.oracle.com/metalink/plsql/showdoc?db=NOT&id=225633.1>
- Troubleshooting Data Guard network issues:  
<http://metalink.oracle.com/metalink/plsql/showdoc?db=NOT&id=241925.1>



---

---

# Index

## A

---

active-active, 1-4, 5-7, 5-14  
    OracleAS Cluster, 2-1  
    OracleAS Cluster (Middle-Tier), 3-1  
active-passive, 1-4, 2-1, 5-7, 5-14  
    OracleAS Cold Failover Cluster, 2-1, 2-2  
    site-level, 5-8  
agsctl help command, 7-36  
AJP  
    port number, 4-23  
AJP port, 4-14  
Application Server Control, 4-6, 4-11, 4-22, 5-6, 6-12,  
    6-21, 6-29  
    stop, 6-3, 6-13, 6-24  
archive logs, 7-1, D-2  
    shipping, D-3  
asgctl commands  
    asgctl, 7-44  
    connect asg, 7-45  
    disconnect, 7-46  
    dump farm, 7-47  
    exit, 7-49  
    failover, 7-27, 7-50  
    help, 7-52  
    instantiate farm to, 7-53  
    quit, 7-55  
    set asg credentials, 7-56  
    set echo, 7-57  
    set new primary database, 7-58  
    set primary database, 7-59  
    set trace, 7-60  
    show operation, 7-61  
    shutdown farm, 7-63  
    startup farm, 7-64  
    stop operation, 7-65  
    switchover farm to, 7-24, 7-66  
    sync farm to, 7-69  
    verify farm, 7-71

## C

---

centralized repository, 5-3  
cluster agent, 1-5  
Cluster File System, A-4  
clustering

    configure Web application state replication, 4-24  
    configuring  
        EJB application state replication, 4-26  
        OC4J processes, 4-28  
    EJB applications, 4-26  
    OracleAS Cluster  
        configuration, 4-2  
        managing, 4-2  
    removing application server instance, 4-9  
clusterware, 1-5  
cold failover database, 5-7, 5-26, 5-28, 6-26  
command-line options, OC4J, 4-14  
common configuration, establishing, 4-13  
component instance, 1-6  
config.inp, D-3  
configuration  
    common, 4-13  
configuration cloning, 3-4  
configuration files, 7-1, 7-5  
    backup, D-3  
connect asg command, 7-36, 7-45  
connecting to the DSA server, 7-36  
creating and executing asgctl scripts, 7-40

## D

---

database polling, 5-29  
database provider, 3-21  
DCM, 5-2, 5-6  
    configuration management, 2-6  
    configuration synchronization, 2-6  
    daemon, 3-18  
    dcmctl, 3-18  
    file-based repository, 5-2  
dcmctl, 3-18, 4-9, 4-11, 4-22, 4-24  
DCM-Managed OracleAS Cluster  
    configuration, 4-2  
directory service, 5-3  
disaster recovery, 5-8, 7-1  
disconnect command, 7-40, 7-46  
disconnecting from OracleAS Guard server, 7-40  
displaying detailed farm information, 7-33, 7-37  
displaying operation history on all nodes, 7-32  
displaying the current operation, 7-32  
DNS, 7-8, 7-9, 7-11, 7-12  
    mapping, 7-35

- switchover, 7-26,7-30
- DNS resolution, 7-11
- dump farm command, 7-33,7-37,7-47

## E

---

- EJB
  - client routing, 3-11
  - replication, 3-10
  - stateful session, 3-10
- EJB application state replication, 4-26
- error messages, 7-34
- exit command, 7-40,7-49
- exiting OracleAS Guard client, 7-40

## F

---

- failback, 1-5
- failover, 1-5
- failover command, 7-5,7-27,7-50
- Farm, 1-6
- file
  - opmn.xml, 4-15
- file-based repository, 5-2

## G

---

- getting help, 7-36

## H

---

- hardware cluster, 1-5
- help command, 7-52
- hostname
  - logical, 7-8
  - network, 1-6,7-8
  - physical, 1-6,7-7,7-8,7-10
  - virtual, 1-7,7-7,7-9,7-10
- hostname resolution, 7-9
- HTTP, 4-38
- HTTPS, 4-38

## I

---

- identifying infrastructure database on primary farm, 7-41
- identity management metadata, 5-2
- Infrastructure
  - backup and recovery, 5-29
- installation type
  - J2EE and Web Cache, 5-2,7-4
  - Portal and Wireless, 5-2,7-4
- instance-specific parameters, 4-14
- instance-specific parameters
  - ApacheVirtualHost, 4-14
  - Groups, 4-14
  - Listen, 4-14
  - NameVirtualHost, 4-14
  - OPMN
    - MissingLocalValuePolicy, 4-15

- OpnmHostPort, 4-14
- PerlBlob, 4-14
- Port, 4-14
- ServerName, 4-14
- User, 4-14

- instantiate farm to command, 7-20,7-37,7-53
- instantiating the farm at secondary site, 7-20,7-37
- invoke asgctl, 7-36

## J

---

- J2EE and Web Cache, 5-2
- J2EE and Web Cache installation type, 7-4
- Java Object Cache, 3-11
- JMS
  - port number, 4-23
- JMS port, 4-14
- JNDI namespace, 3-10
  - replication, 3-10

## K

---

- Kerberos Security Tickets, 5-4

## L

---

- LDAP, 5-3
- load balancer, 7-4
- load balancing
  - mod\_oc4j, 3-4
- local affinity, 3-5
- log apply, D-2
- logical hostname, 7-8

## M

---

- management metadata, 5-2
- managing
  - application server instances in a cluster
    - adding, 4-7
    - removing, 4-9
  - OracleAS Cluster, 4-2
- metadata
  - identity management, 5-2
  - management, 5-2
  - product, 5-2
- metric-based, 3-5
- Microsoft Cluster Server, 5-21
- middle-tier
  - backup and recovery, 3-19
- MissingLocalValuePolicy flag, 4-15
- mod\_oc4j, 3-6,3-20
  - configuring load balancing, 4-10
  - load balancing, 3-4,3-5
- mod\_oc4j.conf, 4-11
- mod\_oradav, 3-20
- mod\_osso, 3-20
- mod\_plsql, 3-20
- monitoring asgctl operations, 7-30
- multicast, 3-10
- multicast address, 4-25,4-27

## N

---

Netegrity Site Minder, 5-4  
network hostname, 1-6, 7-8  
NIS, 7-9  
number of OC4J processes, 4-14

## O

---

OC4J, 5-5  
  cluster-wide parameters, 4-22  
  distributed caching, 3-11  
  instance, 3-10, 4-23  
  instance-specific parameters, 4-22, 4-28  
    islands, 4-23  
    number of processes, 4-23  
    port numbers, 4-23  
  island, 4-23, 4-28  
  islands  
    processes, 4-29  
  Java Object Cache, 3-11  
  load balancing, 3-4  
  Oracle Delegated Administration Services  
    instance, 5-6  
    port numbers, 4-29  
    process, 3-6, 3-7, 3-8, 3-10, 4-10, 4-22, 4-23, 4-28  
    processes  
      configuring, 4-28  
OC4J command-line options, 4-14  
OC4J island definitions, 4-14  
OC4J\_SECURITY, 5-25  
OPMN, 5-7, 5-17, 6-12, 6-20, 6-29  
  custom processes, 4-35  
opmnctl, 4-11, D-5  
opmn.xml file, 4-15  
Oracle Application Server  
  Backup and Recovery Tool, 7-1, D-1, D-3, D-4  
  Certificate Authority, 5-2  
  Cluster, 3-10, 4-10, 4-11, 4-23, 5-6  
    adding instances, 4-7  
    using file-based repository, 3-18, 4-3, 4-9  
    using with Single Sign-On, 4-37  
  Cold Failover Cluster, 7-4, 7-14, 7-23, 7-27  
  Disaster Recovery, 7-1  
  Discoverer, 3-34  
    load balancing, 3-34  
    process monitoring and restart, 3-34  
  Farm, 1-6, 4-2, 4-6, 7-27  
  Infrastructure, 5-1, 7-27  
    stopping, 6-13  
  instance, 1-6, 3-3  
  Java Object Cache, 4-17  
  Metadata Repository, 5-2, 5-6, 7-4, 7-14  
  Portal, 3-20  
    DAD, 3-20  
    repository, 3-21  
  Single Sign-On, 3-20, 4-37, 5-2, 5-3, 5-6  
    third party authentication, 5-4  
  Web Cache, 3-3  
  Web Cache clusters, 3-2  
Oracle Application Server Farm, 1-6  
Oracle Application Server instance, 1-6  
Oracle Data Guard, 7-1, 7-5, 7-13, D-3, F-1  
Oracle Delegated Administration Services, 3-20, 5-3,  
  5-5, 5-6, 5-7, 5-16, 5-25  
Oracle Directory Integration and Provisioning, 3-20,  
  5-6, 5-16, 5-25  
Oracle Directory Manager, 5-3  
Oracle Enterprise Manager, 5-17  
  Application Server Control Console, 5-6  
Oracle Fail Safe, 5-19, 5-21  
Oracle Fail Safe Manager, 5-29  
Oracle HTTP Server, 1-6, 3-6, 3-8, 3-9, 4-10, 4-11, 5-5,  
  5-6, 5-17  
  stateful load balancing, 3-3  
  stateless load balancing, 3-3  
Oracle Identity Management, 3-20, 5-5, 5-6, 5-7, 5-15,  
  5-16, 7-4  
Oracle Internet Directory, 3-20, 5-3, 5-16, 5-25, 5-27  
Oracle Management Services, 5-6  
Oracle Net, F-1  
Oracle Net listener, 5-6  
OracleAS Backup and Recovery Tool, 5-29  
OracleAS Cluster, 1-6, 2-1, 3-9  
  advantages, 2-2  
  DCM-managed, 3-17, 4-1  
    configuring, 4-10  
    membership, 4-13  
  manually managed, 3-19, B-2  
  OC4J, 3-7  
  upgrading, 4-36  
OracleAS Cluster (Identity Management), 2-2, 5-7,  
  5-8, 5-9  
  Application Server Control Console, 6-10  
  backup and recovery, 6-9  
  configuring, 6-6, A-6  
  distributed, 5-7, 5-8, 5-11  
  failover, 6-8  
  managing, 6-1  
  monitoring, 6-4  
  starting, 6-2  
  stopping, 6-3  
OracleAS Cluster (J2EE), 2-2  
OracleAS Cluster (Middle-Tier), 3-1, 4-37  
OracleAS Cluster (OC4J), 2-6, 3-7, 3-8, 3-9, 3-10, 3-11,  
  3-24  
  configuring, 3-10  
  EJB state replication, 3-10  
  session state replication, 3-7, 3-8, 3-9  
  state replication, 2-6  
OracleAS Cluster (OC4J-EJB), 3-10  
OracleAS Cluster (OC4J-JMS), 3-12, 3-13  
  manual failover, 4-35  
OracleAS Cluster (Portal), 2-2  
OracleAS Cluster (Web Cache), 3-23  
OracleAS Cold Failover Cluster, 2-1, 5-27, 7-4, 7-9,  
  7-13, 7-14  
  advantages, 2-3  
  backup and recovery, 5-29  
  cluster-wide maintenance, 7-23  
  cluster-wide, site-wide unplanned outage, 7-27

- environment, 1-5
  - online database backup and restore, A-3
  - solutions for Metadata Repository and Identity Management, 5-13
  - OracleAS Cold Failover Cluster (Identity Management), 5-7, 5-14, 5-25
    - backup and recovery, 6-29
    - cold failover database, 5-26
    - configuring, 6-26
    - distributed, 5-7, 5-27
      - cold failover database, 5-28
    - failover, 6-27
    - managing, 6-22
    - monitoring, 6-25
    - starting, 6-23
    - stopping, 6-24
  - OracleAS Cold Failover Cluster (Infrastructure), 3-15, 3-26, 5-7, 5-13, 5-15
    - backup and recovery, 6-21
    - co-located OracleAS Cold Failover Cluster (Middle-Tier), 5-21
    - configuring, 6-15
    - distributed, 5-7, 5-13, 5-22, 5-24
    - failover, 6-16
    - managing, 6-11
    - monitoring, 6-14
    - node failure protection, 5-19
    - normal operation, 5-16
    - starting, 6-11
    - stopping, 6-13
    - using Application Server Control Console, 6-21
    - Windows solution, 5-19
  - OracleAS Cold Failover Cluster (Middle-Tier) using Application Server Control Console for, 4-32
  - OracleAS Cold Failover Cluster (Middle-Tier), 3-12, 3-15, 3-16, 4-33
    - backup and recovery, 4-32
    - co-located OracleAS Cold Failover Cluster (Infrastructure), 5-21
    - configuration and deployment, 4-31
    - managing, 4-31
    - managing failover, 4-32
    - manual failover, 4-33
    - using with OracleBI Discoverer, 4-32
  - OracleAS Farm, 1-6
  - OracleAS Guard, 7-18
    - asgctl command, 7-36
    - asgctl command-line interface overview, 7-15
    - asgctl commands description summary, 7-42
    - client, 7-16
    - connecting to DSA server, 7-36
    - creating and executing asgctl scripts, 7-40
    - disconnecting from OracleAS Guard server, 7-40
    - displaying detailed farm information, 7-33, 7-37
    - displaying operation history, 7-32
    - displaying the current operation, 7-32
    - error messages, 7-34
    - exiting OracleAS Guard client, 7-40
    - failing over to standby farm, 7-27
    - getting help, 7-36
    - identifying infrastructure database on primary farm, 7-41
    - instantiating farm at secondary site, 7-20, 7-37
    - invoking asgctl, 7-36, 7-44
    - monitoring asgctl operations, 7-30
    - operations, 7-17
    - role of OPMN, 7-18
    - server, 7-17
    - setting asg credentials, 7-41
    - shutting down a farm, 7-33
    - specifying the primary database, 7-36
    - starting up a shutdown farm, 7-33
    - stopping operations, 7-32
    - supported disaster recovery configurations, 7-18
    - supported OracleAS releases, 7-18
    - switching over to standby farm, 7-24
    - synchronizing secondary site with primary site, 7-22, 7-39
    - tracing tasks, 7-33
    - typical asgctl session, 7-35
    - validating primary farm configuration is valid, 7-31
    - validating primary farm is running, 7-31
    - verifying the farm, 7-37
  - OracleAS Integration B2B, 3-24, 3-26
  - OracleAS JMS, 3-16, 4-35
    - distributed destinations, 3-12
  - OracleAS Metadata Repository, 3-26, 5-14, 5-15, 5-16, 5-22, 5-25, 5-27
    - high availability options, 5-14
  - OracleAS Metadata Repository Creation Assistant, 5-25
  - OracleAS Single Sign-On, 4-37, 5-7, 5-16, 5-25
    - configuring, 4-38
  - OracleAS Wireless
    - clustering, 3-24
    - stateful requests, 3-24
    - stateless requests, 3-24
  - OracleBI Discoverer, 4-32
  - OracleBI Discoverer Preferences Server, 3-35
  - ORASSO DAD, 3-20
  - orion-ejb-jar.xml, 4-27
- ## P
- 
- physical hostname, 1-6, 7-7, 7-8, 7-10
  - port numbers, 4-23, 4-29
  - Portal and Wireless, 5-2
  - Portal and Wireless installation type, 7-4
  - Portal Page Engine, 3-20
  - primary node, 1-6
  - product metadata, 5-2
  - production site, 7-2, 7-12
    - backup, D-2
  - Public Key Infrastructure, 5-4
- ## Q
- 
- quit command, 7-55



## R

---

RAID disks, 4-17  
random, 3-5  
Real Application Clusters, 3-7, 3-11, 3-13, 3-29, 3-30, 3-31, 3-32, 3-33, 4-12, 5-3, 5-7, 5-9, 5-11, 5-14, 5-25, 5-27, 6-26, A-4  
redundancy, 2-1  
repository host, 3-18  
RMI  
    port number, 4-23  
RMI port, 4-14  
round robin, 3-5

## S

---

scheduled outages, 7-23  
SCN, D-3, D-5  
secondary node, 1-6  
secure shell port forwarding, F-1  
sequence number, D-3, D-5  
serializable, 4-25  
session state replication, 3-7, 3-8, 3-9  
set asg credentials command, 7-41, 7-56  
set echo command, 7-57  
set new primary database command, 7-58  
set primary database command, 7-36, 7-41, 7-59  
set trace command, 7-60  
set trace off command, 7-33  
set trace on command, 7-33  
setting asg credentials, 7-41  
shared disk, 5-21, 5-25  
shared storage, 1-5, 2-2  
show operation command, 7-61  
show operation full command, 7-32  
show operation history command, 7-32  
shutdown farm command, 7-33, 7-63  
shutting down a farm, 7-33  
single point of failure, 3-18  
specifying the primary database, 7-36  
SQLPLUS, D-5  
SSH tunneling, F-1  
ssoreg.sh script, 4-37, 4-38  
SSOSDK, 5-3  
standby site, 7-2, 7-3, 7-12  
    restoration, D-4  
starting up a shutdown farm, 7-33  
startup farm command, 7-33, 7-64  
state replication  
    configuring for EJB applications, 4-26  
    configuring for Web applications, 4-24  
    JVM termination, 4-28  
state safe applications, 3-7  
stateful applications, 3-8  
stateful OC4J applications, clustering and, 4-14  
stateful session EJB, 4-27, 4-28  
static ports file, 7-14  
staticports.ini, 7-14  
stop operation command, 7-32, 7-65  
stopping asgctl operations, 7-32  
switchback, 1-7

switchover, 1-7  
switchover farm to command, 7-5, 7-24, 7-66  
sync farm to command, 7-69  
synchronize farm to command, 7-22, 7-39  
synchronizing secondary site with primary site, 7-22, 7-39

## T

---

time-to-live, 7-34  
TNS listener, 5-21  
TNS Names, 7-13  
tracing asgctl tasks, 7-33  
Transparent Application Failover, 3-11  
True Application Failover, 3-12  
typical asgctl session, 7-35

## U

---

unplanned outages, 7-26

## V

---

validating primary farm configuration is valid, 7-31  
validating primary farm is running, 7-31  
verify farm command, 7-31, 7-37, 7-71  
verifying the farm, 7-37  
virtual hostname, 1-7, 5-11, 5-17, 5-19, 5-20, 5-21, 5-25, 5-27, 7-3, 7-4, 7-6, 7-7, 7-8, 7-9, 7-10, 7-11, 7-12, 7-15, A-2  
virtual IP, 1-7, 1-8, 2-8, 3-15, 3-16, 4-34, 5-17, 5-19, 5-20, 5-21, 5-27, 6-12, 6-13, 6-15, 6-16, 6-27, 6-28, 6-29, 7-7, 7-9, 7-13  
    failover, 4-33  
volume management software, 6-12, 6-13

## W

---

Web application session state replication, 3-7, 3-9  
Web provider, 3-21  
web.xml, 4-25  
weighted routing, 3-5  
wide area network, 7-3

## X

---

X.509 certificate, 5-4

