Oracle® Application Server

High Availability Guide

10*g* Release 3 (10.1.3)

B15977-02

February 2006

Oracle Application Server High Availability Guide, 10*g* Release 3 (10.1.3)

B15977-02

Contents

Part III Disaster Recovery

5 OracleAS Disaster Recovery

Preface

This preface contains these sections:

- Intended Audience
- Documentation Accessibility
- Related Documentation
- Conventions

Intended Audience

The *Oracle Application Server High Availability Guide* is intended for administrators, developers, and others whose role is to deploy and manage Oracle Application Server 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. Accessibility 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 more information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation

Screen readers 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, some screen readers 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.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documentation

For more information, see these Oracle resources:

- *Oracle Application Server Concepts*
- *Oracle Application Server Installation Guide*
- *Oracle Application Server Administrator's Guide*

Conventions

The following text conventions are used in this document:

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

Part I

Overview

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

1

Introduction to High Availability

This release of Oracle Application Server extends and improves upon the high availability solutions that were available in earlier releases. New flexible and automated high availability solutions for Oracle Application Server have been tested and are described in this guide. 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. Sections in this chapter:

- Section 1.1, "What is High Availability"
- Section 1.2, "Oracle Application Server High Availability Concepts"
- Section 1.3, "High Availability Information in Other Documentation"

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

Availability Percentage	Approximate Downtime Per Year
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 (includes hardware and software changes in areas such as operating system, application server, configuration, and application changes)
	Data maintenance

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

1.1.2 High Availability Solutions

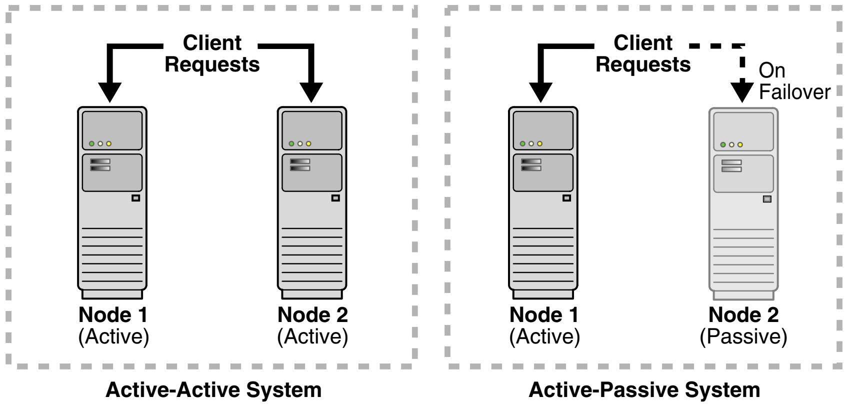
High availability solutions can be categorized into local high availability solutions that provide high availability in a single data center deployment, and disaster recovery solutions, which are usually geographically distributed deployments that protect your applications from disasters such as floods 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 operating system vendor-specific clusterware. Generally, vendor-specific cluster agents are also available to automatically monitor and failover between cluster nodes, so that when the active instance fails, an agent shuts down the active instance completely, brings up the passive instance, and application services can successfully resume processing. As a result, the active-passive roles are now switched. The same procedure can be done manually for planned or unplanned down time. Active-passive solutions are also generally referred to as cold failover clusters.

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

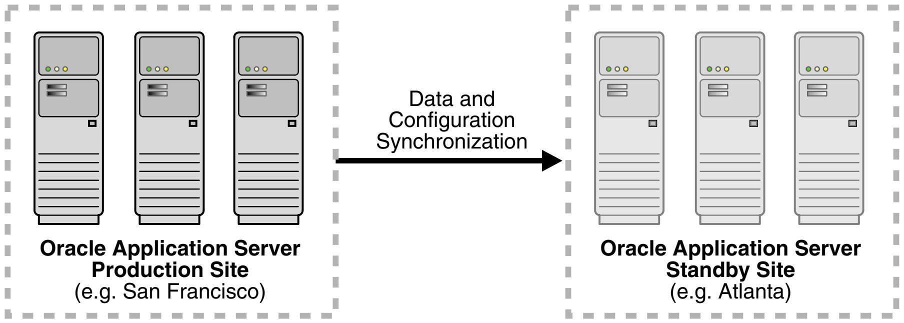
  When multiple instances of identical server components are available, client requests to these components can be load balanced to ensure that the instances have roughly the same workload. With a load balancing mechanism in place, the instances are redundant. If any of the instances fail, requests to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


Data and Configuration Synchronization
Oracle Application Server Production Site (e.g. San Francisco)
Oracle Application Server Standby Site (e.g. Atlanta)

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, "Using Oracle Identity Management with Oracle Application Server Release 3 (10.1.3)"
- Section 1.2.4, "Oracle Application Server High Availability Architectures"
- Section 1.2.5, "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 non 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**: Although each node in a hardware cluster 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 either node in the hardware cluster. 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.

In OracleAS Cold Failover Cluster (Middle-Tier) environments, you can install the Oracle home directory on a shared storage system or on a local storage of each node in the hardware cluster.

- **primary node**: The node that is actively running Oracle Application Server at any given time. If this node fails, Oracle Application Server is failed over to the secondary node. Because the primary node runs the active Oracle Application Server installation(s), it is considered the "hot" node. See the definition for "secondary node" in this section.
- **secondary node**: This is the node that runs Oracle Application Server if the primary node fails. Because the secondary node does not originally run Oracle Application Server, 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 (on UNIX), `C:\WINDOWS\system32\drivers\etc\hosts` file (on 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.
- **physical hostname**: This guide differentiates between the terms physical hostname and network hostname. This guide uses physical hostname to refer to the "internal name" of the current machine. On UNIX, this is the name returned by the `hostname` command.

  Physical hostname is used by Oracle Application Server 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.

Note: Whenever the term "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.

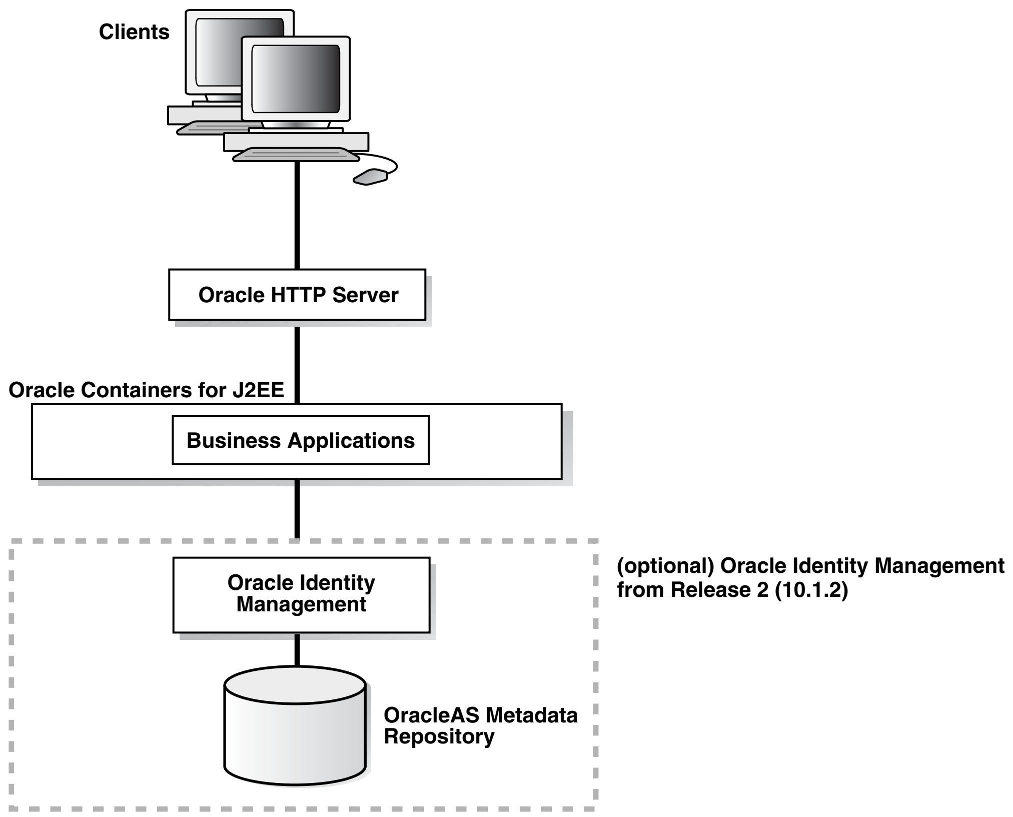
  A load balancer also uses a virtual IP as the entry point to a set of servers. These servers tend to be active at the same time. This virtual IP address is not assigned to any individual server but to the load balancer which acts as a proxy between servers and their clients.

1.2.2 Oracle Application Server Base Architecture

Before creating a highly available Oracle Application Server installation, you should understand Oracle Application Server's base architecture. Then, to make Oracle Application Server highly available, you examine every component and connection path between components and make each one of them highly available. This produces a highly available architecture by 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


Clients
Oracle HTTP Server
Oracle Containers for J2EE
Business Applications
Oracle Identity Management
OracleAS Metadata Repository
(optional) Oracle Identity Management from Release 2 (10.1.2)

At a high level, Oracle Application Server consists of Oracle HTTP Server and Oracle Containers for J2EE ("OC4J"). OC4J provides the J2EE containers on which you deploy your business applications.

You can use Oracle Identity Management from Oracle Application Server Release 2 (10.1.2), if needed. See the next section for details.

1.2.3 Using Oracle Identity Management with Oracle Application Server Release 3 (10.1.3)

This release of Oracle Application Server includes only the J2EE middle tier; it does not include Oracle Identity Management or the OracleAS Metadata Repository.

If your business applications need services provided by the Oracle Identity Management components, you can deploy and run your business applications on J2EE middle tier from Release 3 (10.1.3), and the applications can access Oracle Identity Management from Release 2 (10.1.2.x). See Figure 1–3.

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

- OracleAS Single Sign-On
- Oracle Delegated Administration Services
- Oracle Internet Directory
- Oracle Directory Integration and Provisioning
- OracleAS Certificate Authority

In a highly available environment, you need middle tiers, Oracle Identity Management, and OracleAS Metadata Repository all to be highly available. This guide describes how to set up highly available J2EE middle tiers. For information on setting up highly available Oracle Identity Management and OracleAS Metadata Repository, see the *Oracle Application Server High Availability Guide* from Release 2 (10.1.2) for details.

Note that although you have an OracleAS Metadata Repository database associated with the Release 2 (10.1.2) Oracle Identity Management, the OracleAS Metadata Repository database is not used to store DCM or Release 3 (10.1.3) instance configuration information. There is no DCM in Release 3 (10.1.3). The OracleAS Metadata Repository is used only for the Oracle Identity Management components.

1.2.4 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. Although both types of solutions provide high availability, active-active solutions generally offer higher scalability and faster failover, but 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.

1.2.5 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 (Oracle Identity Management)[1]	Y
Y	Y	Y	Active-active (middle tier) Active-passive and active-active (Oracle Identity Management)[1]	Y

[1] Oracle Identity Management is from Release 2 (10.1.2). OracleAS Disaster Recovery supports the base, active-passive, and active-active architectures for Oracle Identity Management. For additional scalability in a base, active-passive, or active-active architecture, extra computing power can be added to the infrastructure hardware (for example, high capacity CPUs, more memory).

In the following paragraphs, Oracle Identity Management is from Release 2 (10.1.2).

Although you can choose different high availability architectures for your middle-tier and Oracle Identity Management, their local high availability and disaster recovery requirements should be identical. Scalability requirements should be evaluated separately for middle-tier and Oracle Identity Management. Oracle Identity Management does not usually need to be as scalable as the middle tier because it handles fewer requests.

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

1.3 High Availability Information in Other Documentation

Table 1–4 lists Oracle Application Server guides (other than this guide) that contain high availability information. This information pertains to high availability of various Oracle Application Server components.

Table 1–4 High Availability Information in* Oracle Application Server *Documentation

Component	Location of Information
Oracle installer	In the chapter for installing in a high availability environment in *Oracle Application Server Installation Guide*.
OracleAS Backup and Recovery Tool	In the backup and restore part of *Oracle Application Server Administrator's Guide*.
Oracle Process Manager and Notification Server (OPMN)	*Oracle Process Manager and Notification Server Administrator's Guide*
OC4J	*Oracle Containers for J2EE Configuration and Administration Guide* *Oracle Containers for J2EE Services Guide* *Oracle Containers for J2EE Enterprise JavaBeans Developer's Guide*
Oracle HTTP Server (load balancing to OC4J processes)	*Oracle HTTP Server Administrator's Guide*

2 Oracle Application Server High Availability Framework

Whereas Chapter 1 provided an overview of high availability in general, this chapter describes the Oracle Application Server features that are important in high availability topologies. It contains the following sections:

2.1 Process Management through OPMN

An Oracle Application Server instance consists of many different running processes that 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 Oracle Process Manager and Notification Server (OPMN) component of Oracle Application Server provides the following process management services:

- When OPMN detects Oracle Application Server processes that are down, unresponsive, or unreachable, it automatically restarts them. It monitors the processes by pinging them or through notification methods.
- OPMN starts processes in the proper order: processes that depend on other processes are not started up until the dependent processes are started.

You can also use OPMN to perform tasks such as starting and stopping processes, and also to check the status of processes. See the *Oracle Process Manager and Notification Server Administrator's Guide* for details.

OPMN manages the following Oracle Application Server processes:

- Oracle HTTP Server
- Oracle Containers for J2EE (OC4J)

- OracleAS Log Loader
- OracleAS Guard (for disaster recovery)
- OracleAS Port Tunnel

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

You also use OPMN to cluster Oracle Application Server instances so that Oracle HTTP Server can direct requests to any OC4J instance in a cluster. This clustering is needed in active-active topologies. See Chapter 3, "Active-Active Topologies" for details.

OracleAS Clusters also enable you to manage all the Oracle Application Server instances in the cluster. For example, you can issue an OPMN command on one machine to start all processes or a specific process type across all local and remote Oracle Application Server instances in the cluster.

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

For details on OPMN, see the *Oracle Process Manager and Notification Server Administrator's Guide*.

2.2 Replication of State Information

One of the advantages of distributing applications is that multiple redundant processes can all handle requests from clients. In the event that one of these processes becomes unavailable, another 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 replicate this application state across multiple processes. Oracle Application Server enables the replication of state in J2EE applications through application-level clustering. In an application cluster, 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 instances in the cluster.

A J2EE application can maintain two types of state information:

- HTTP session state (updated by servlets and JSPs)
- Stateful session EJB state (updated by stateful session EJB instances)

Within an OracleAS Cluster (OC4J), application-level clustering enables the replication of both types of state information. You can control how the state information is replicated, the frequency at which the state information is replicated between OC4J instances, and which state information is replicated. See Table 2–1:

Table 2–1 Procedures for Replicating State Information

To do this:	See
Set up how state information is replicated	You can replicate state information using multicasting, peer-to-peer, or database: ■ Section 3.2.2, "Setting up Multicast Replication" ■ Section 3.2.3, "Setting up Peer-to-Peer Replication" ■ Section 3.2.4, "Setting up Replication to a Database"
Specify how frequently state information is replicated, or specify which information is replicated	See Section 3.2.5, "Setting the Replication Policy".

See also the "Application Clustering in OC4J" chapter in the *Oracle Containers for J2EE Configuration and Administration Guide* for more information.

2.3 Load Balancing in Oracle Application Server

Load balancing involves distributing requests among two or more processes. Features of a software or hardware load balancer include:

- load balancing algorithm

  The algorithm specifies how to allocate requests across the different processes. 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.

  The algorithms usually include processes in the local instance as well as in remote instances. This enables one component to direct requests to another component running on a remote machine.

- death detection

  The load balancer must be able to recognize failed requests to one or more processes and be able to mark these processes as inactive so that no further requests will be forwarded to them.

In Oracle Application Server, routing of requests between some components involve load balancing mechanisms. Table 2–2 shows some examples:

Table 2–2 Routing of Requests Between Components

From	To	Description
OracleAS Web Cache (from Release 2 (10.1.2))	Oracle HTTP Server	OracleAS Web Cache can route requests to any Oracle Application Server instance that includes Oracle HTTP Server. If OracleAS Web Cache detects failures in the replies returned by Oracle HTTP Server, it routes new requests to the available Oracle HTTP Servers. You configure the routing algorithm in the OracleAS Web Cache component. See the *Oracle Application Server Web Cache Administrator's Guide* from Release 2 (10.1.2) for details.

Table 2–2 (Cont.) Routing of Requests Between Components

From	To	Description
Oracle HTTP Server	OC4J	Oracle HTTP Server routes requests for J2EE applications to OC4J processes. The mod_oc4j module in Oracle HTTP Server performs the routing. mod_oc4j can route a request to any OC4J process in the OracleAS Clusters. 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. You configure the routing algorithm using directives in Oracle HTTP Server configuration files. See Section 3.2.12, "Setting mod_oc4j Load Balancing Options" for details.
Oracle HTTP Server	Database	Oracle HTTP Server routes requests for PL/SQL applications to the database. The mod_plsql module in Oracle HTTP Server performs the routing. mod_plsql is able to detect failure in the database and it routes requests only to available database nodes.
OC4J	OC4J	Within OC4J itself, OC4J routes requests from the presentation layer components (servlets and JSPs) to the business layer components (EJBs). If OC4J detects failures in the RMI invocations to the EJB tier, it fails over communication to available EJB nodes.
OC4J	Database	OC4J drivers are enabled to detect failures of database nodes and re-route requests to available nodes.

2.4 OracleAS Clusters

Note that Oracle Application Server supports different types of clustering. This section describes OracleAS Clusters. For information on application-level clustering, see Section 2.2, "Replication of State Information".

You can group Oracle Application Server instances in OracleAS Clusters. OracleAS Clusters provide the following benefits:

- In active-active topologies, you need to group Oracle Application Server instances in the same cluster so that Oracle HTTP Server instances in the topology know which OC4J instances are also in the same topology. These are the OC4J instances that Oracle HTTP Server can forward application requests to.

  When Oracle HTTP Server applies the load balancing algorithm specified in the mod_oc4j module, it applies the algorithm to the OC4J instances in the cluster. To set the mod_oc4j load balancing algorithm, see Section 3.2.12, "Setting mod_oc4j Load Balancing Options".

- You can manage the Oracle Application Server instances collectively by using the `@cluster` parameter in the `opmnctl` command. For example, you can stop Oracle HTTP Server in all the instances belonging to the cluster with the following command:

  ```
  > opmnctl @cluster stopproc ias-component=HTTP_Server
  ```

  For a list of `opmnctl` commands that accept the `@cluster` parameter, see the *Oracle Process Manager and Notification Server Administrator's Guide*.

- If you are using application-level clustering with dynamic peer-to-peer replication, you need to set up OracleAS Clusters because OracleAS Clusters determine the members of the cluster.

  For information on peer-to-peer replication, see Section 3.2.3, "Setting up Peer-to-Peer Replication".

2.5 External Load Balancers

To load balance requests among many Oracle Application Server instances in an active-active topology, you should use an external load balancer.

When several Oracle Application Server instances are clustered and are fronted by a load balancer, the load balancer hides the multiple instance configuration by being the entry point to the system. External load balancers can send requests to any Oracle Application Server instance in the cluster, as any instance can service any request. You can raise the capacity of the system by introducing additional Oracle Application Server instances. These instances can be installed on separate nodes to allow for redundancy in case of node failure.

There are different types of external load balancers you can use with Oracle Application Server. Table 2–3 summarizes the different types.

Table 2–3 Types of External Load Balancers

Load Balancer Type	Description
Hardware load balancer	Hardware load balancing involves placing a hardware load balancer in front of a group of Oracle Application Server instances or OracleAS Web Cache (from Release 2 (10.1.2)). The hardware load balancer routes requests to the Oracle HTTP Server or OracleAS Web Cache instances in a client-transparent fashion.
Software load balancer	Software load balancer involves using some process that intercepts the different calls to an application server and routes those requests to redundant components.
Lvs network load balancer for Linux	With some Linux operating systems, you can use the operating system to perform network load balancing.
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 enables you to send requests to different machines that share the same virtual IP or MAC address. The servers themselves to do not need to be clustered at the operating system level.

External Load Balancer Requirements

Oracle does not provide external load balancers. You can get external load balancers from other companies.

To ensure that your external load balancer can work with Oracle Application Server, check that your external load balancer meets the requirements listed in Table 2–4.

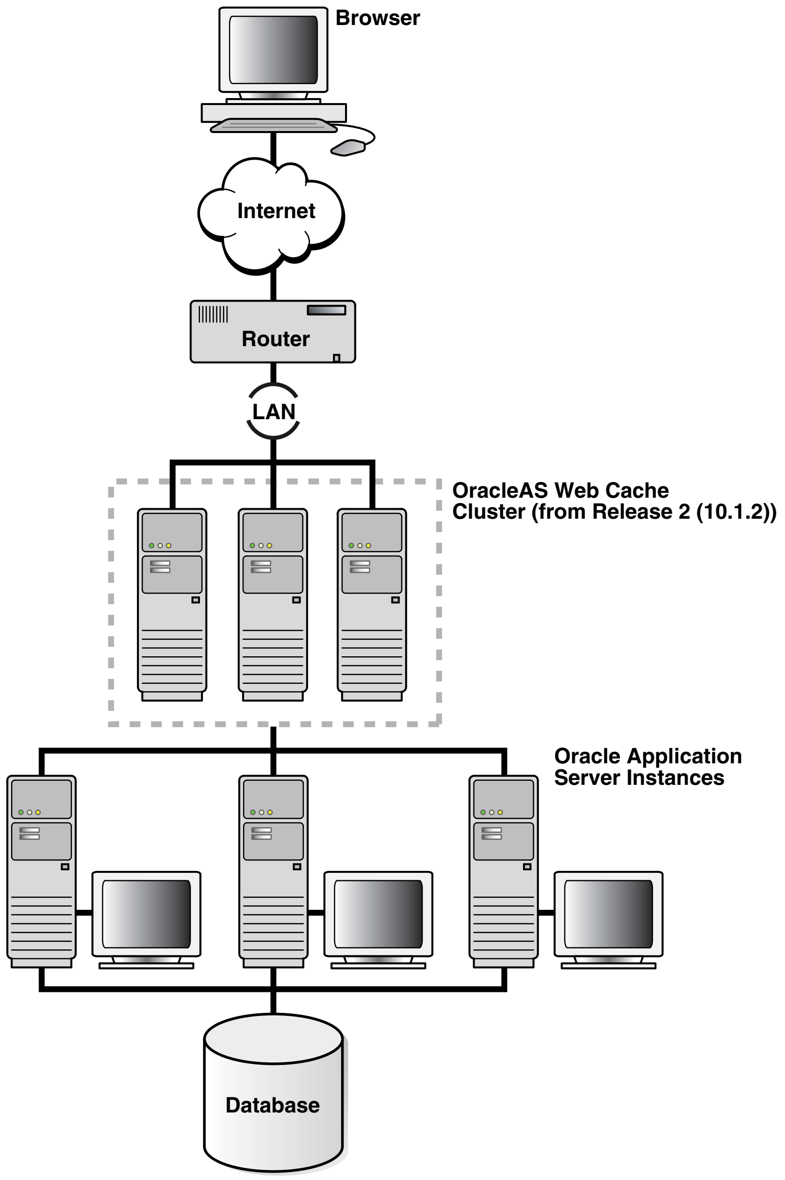
Note that you may not need all the requirements listed in the table. The requirements for an external load balancer depend on the topology being considered, and on the Oracle Application Server components that are being load balanced.

Table 2–4 External Load Balancer Requirements

External Load Balancer Requirement	**Description**
Virtual servers and port configuration	You need to be able to configure virtual server names and ports on your external load balancer, and the virtual server names and ports must meet the following requirements: ■ The load balancer should allow configuration of multiple virtual servers. For each virtual server, the load balancer should allow configuration of traffic management on more than one port. For example, for OracleAS Clusters, the load balancer needs to be configured with a virtual server and ports for HTTP and HTTPS traffic. ■ The virtual server names must be associated with IP addresses and be part of your DNS. Clients must be able to access the external load balancer through the virtual server names.
Resource monitoring / port monitoring / process failure detection	You need to set up the external load balancer to detect service and node failures (through notification or some other means) and to stop directing non-Oracle Net traffic to the failed node. If your external load balancer has the ability to automatically detect failures, you should use it.
Fault tolerant mode	It is highly recommended that you configure the load balancer to be in fault-tolerant mode.
Other	It is highly recommended that you configure the load balancer virtual server to return immediately to the calling client when the backend services to which it forwards traffic are unavailable. This is preferred over the client disconnecting on its own after a timeout based on the TCP/IP settings on the client machine.

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


Browser
Internet
Router
LAN
OracleAS Web Cache Cluster (from Release 2 (10.1.2))
Oracle Application Server Instances
Database

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 external 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 external load balancer can send requests to another active instance.

2.6 Backup and Recovery

Protecting against data loss in any system component is critical to maintaining a highly available environment. You should perform regular backups of all Oracle Application Server instances in your environment.

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 the Oracle system files on each host in your environment.
- If you are using Oracle Identity Management from Release 2 (10.1.2), then you also need to perform a full backup of:
  - all files in the Oracle Identity Management's Oracle home (this includes Oracle software files and configuration files)
  - the OracleAS Metadata Repository used by the Oracle Identity Management

2.6.1 Oracle Application Server Backup and Recovery Tool

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

You can use the OracleAS Backup and Recovery Tool to back up and recover the following installation types:

- J2EE server
- Oracle HTTP Server
- Integrated (J2EE server and Oracle HTTP Server)

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

For details on the OracleAS Backup and Recovery Tool, see the *Oracle Application Server Administrator's Guide*.

2.7 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 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.

2.7.1 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 5.4, "Overview of OracleAS Guard and asgctl"

2.8 High Availability Topologies: Overview

Oracle Application Server provides redundancy by supporting multiple instances for the same workload. These redundant configurations provide increased availability through a distributed workload, 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 Clusters or an active-passive configuration using OracleAS Cold Failover Cluster.

The following sections describe the basics of these configurations:

- Section 2.8.1, "Active-Active Topologies"
- Section 2.8.2, "Active-Passive Topologies: OracleAS Cold Failover Clusters"

2.8.1 Active-Active Topologies

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

The instances are front-ended by an external load balancer, which directs requests to any of the active instances. Instead of an external load balancer, you can also run a software load balancer to distribute the requests. In production environment, however, a hardware load balancer is recommended.

Common properties of an active-active topology include:

- Similar instance configuration

  The instances need to serve the same workload or applications. Some configuration properties should have similar values across instances so that the instances can deliver the same reply to the same request. Other configuration properties may be instance-specific, such as local host name information.

  If you make a configuration change to one instance, you should also make the same change to the other instances in the active-active topology. The "Configuring and Managing Clusters" chapter in the *Oracle Containers for J2EE Configuration and Administration Guide* lists the files that contain properties that should be replicated.

- Independent operation

  If one Oracle Application Server instance in an active-active topology fails, the other instances in the cluster continue to serve requests. The load balancer directs requests only to instances that are alive.

Advantages of an active-active topology include:

- Increased availability

  An active-active topology 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 share a workload among different machines and processes. You can scale the topology by adding new instances as the number of requests increase.

2.8.2 Active-Passive Topologies: OracleAS Cold Failover Clusters

Oracle Application Server provides an active-passive model for all its components in OracleAS Cold Failover Clusters. In an OracleAS Cold Failover Cluster topology, two Oracle Application Server instances are configured to serve the same application workload but only one is active at any particular time. The passive instance runs (that is, becomes active) only when the active instance fails. These instances run on nodes that are in a hardware cluster.

Common properties of an OracleAS Cold Failover Cluster topology include:

- Hardware cluster

  In an OracleAS Cold Failover Cluster topology, you run Oracle Application Server on machines that are in a hardware cluster, with vendor clusterware running on the machines.

- Shared storage

  You install the Oracle home for the Oracle Application Server instance on storage shared by the machines in the hardware cluster.

  The active node in the OracleAS Cold Failover Cluster topology mounts the shared storage so that it has access to the Oracle home. If it fails, the passive instance mounts the shared storage and accesses the same Oracle home.

- Virtual hostname

  The virtual hostname gives clients a single system view of the Oracle Application Server middle tier. Clients use the virtual hostname to access the Oracle Application Server middle tier.

  The virtual hostname is associated with a virtual IP. This name-IP entry must be added to the DNS that the site uses. 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 virtual hostname `apps.mycompany.com`. In the DNS, `apps` maps to a virtual IP address that floats between `node1` and `node2` via a hardware cluster. Clients access Oracle Application Server using `apps.mycompany.com`; they do not know which physical node is active and actually servicing a particular request.

  You can specify the virtual hostname during installation. See the *Oracle Application Server Installation Guide*.

- Failover procedure

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

Advantages of an OracleAS Cold Failover Cluster topology 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 topology only one set of processes is up and serving requests. Managing the active instance is generally easier than managing an array of active instances.

- Application independence

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

Part II

Middle-tier High Availability

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

- Chapter 3, "Active-Active Topologies"
- Chapter 4, "Active-Passive Topologies"

3 Active-Active Topologies

This chapter describes active-active topologies. It contains the following sections:

3.1 About Active-Active Topologies

An active-active topology consists of redundant middle-tier instances that deliver greater scalability and availability than a single instance. Active-active topologies remove the single point of failure that a single instance poses. While a single Oracle Application Server instance leverages the resources of a single host, a cluster of middle-tier instances spans multiple hosts, distributing application execution over a greater number of CPUs. A single Oracle Application Server instance is vulnerable to the failure of its host and operating system, but an active-active topology continues to function despite the loss of an operating system or a host, hiding any such failure from clients.

In active-active topologies, all the instances are active at the same time. This is different from active-passive topologies, where only one instance is active at any time.

The nodes in the active-active topologies are not in a hardware cluster.

Load Balancer Requirements

Active-active topologies use a load balancer to direct requests to one of the Oracle Application Server instances in the topology. In other words, the Oracle Application Server instances are fronted by the load balancer.

You configure the load balancer with virtual server names for HTTP and HTTPS traffic. Clients use the virtual server names in their requests. The load balancer directs requests to an available Oracle Application Server instance.

See Section 2.5, "External Load Balancers" for a list of features that your load balancer should have.

Figures of Active-Active Topologies

The following figures show two active-active topologies. The difference in the topologies is whether you install Oracle HTTP Server and OC4J in the same Oracle home or in separate Oracle homes.

Figure 3–1 shows an active-active topology with Oracle HTTP Server and OC4J in the same Oracle home. Figure 3–2 shows an active-active topology with Oracle HTTP Server and OC4J in separate Oracle homes.

Figure 3–1 Active-Active Topology with Oracle HTTP Server and OC4J in the Same Oracle Home

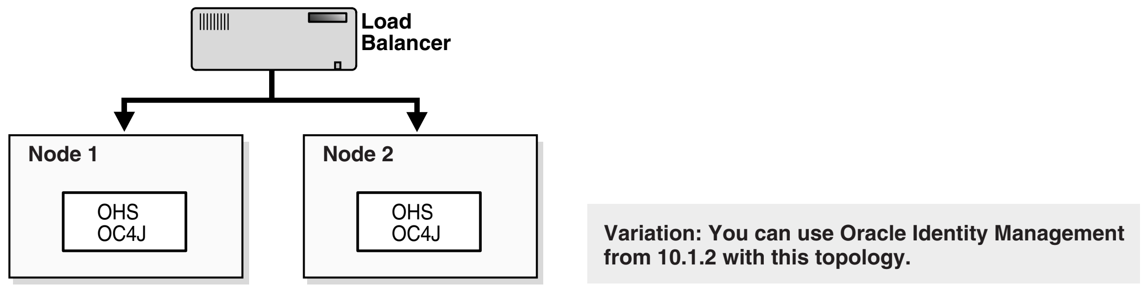
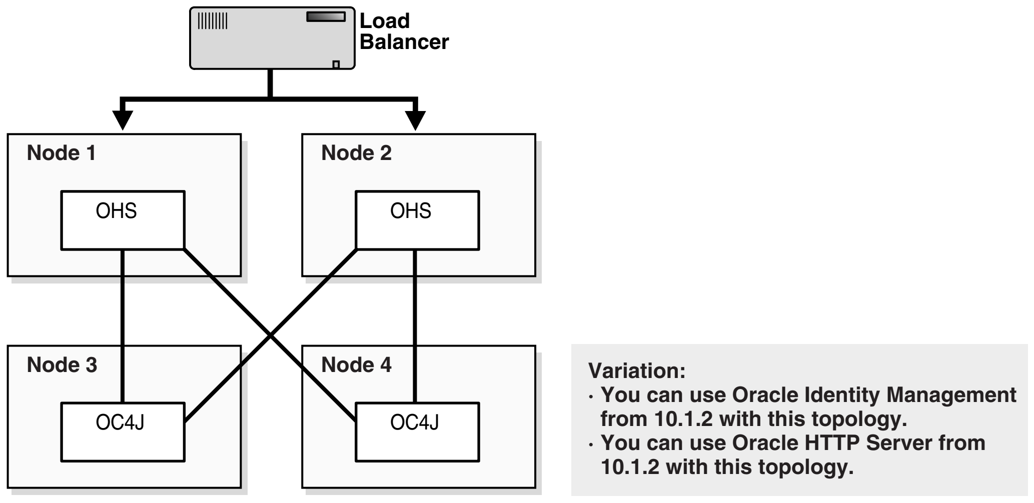

Load Balancer
Node 1
OHS OC4J
Node 2
OHS OC4J
Variation: You can use Oracle Identity Management from 10.1.2 with this topology.

Figure 3–2 Active-Active Topology with Oracle HTTP Server and OC4J in Separate Oracle Homes


Load Balancer
Node 1
OHS
Node 2
OHS
Node 3
OC4J
Node 4
OC4J
Variation:
· You can use Oracle Identity Management from 10.1.2 with this topology.
· You can use Oracle HTTP Server from 10.1.2 with this topology.

3.1.1 OracleAS Clusters in Active-Active Topologies

All the Oracle Application Server instances in an active-active topology belong to the same cluster. Oracle HTTP Server forwards application requests only to OC4J instances that belong to the same cluster.

You can group instances in an cluster using one of the following ways:

- All the instances use the same multicast IP address and port.
- All the instances are chained to the same discovery server.
- Each instance specifies all other instances in the opmn.xml configuration file.
- If the instances run on nodes that are on different subnets, you have to designate a node to be the gateway server, which bridges the instances on the different subnets.

OracleAS Clusters also enable you to use the `@cluster` parameter in some `opmnctl` commands. Commands that use the `@cluster` parameter apply to all instances in the cluster. For example, you can use the `@cluster` parameter to start all components in all instances in the cluster.

OC4J instances in a cluster have the following features:

- OC4J instances have cluster-wide properties as well as instance-specific properties. Cluster-wide properties are properties whose values should be similar for all OC4J instances in the cluster. Instance-specific properties are properties that have different values for each OC4J instance. For a list of cluster-wide properties, see the "Configuring and Managing Clusters" chapter in the *Oracle Containers for J2EE Configuration and Administration Guide*.
- If you modify a cluster-wide property in one OC4J instance, you should propagate the change to all other OC4J instances in the cluster.
- When you deploy an application to an OC4J instance, you also need to deploy it on all other OC4J instances in the cluster.
- The number of OC4J processes is an instance-specific property: it can be different for each OC4J instance. This must be configured for each Oracle 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.

For details, see the "Configuring and Managing Clusters" chapter in the *Oracle Containers for J2EE Configuration and Administration Guide*.

3.1.2 Application-Level Clustering in Active-Active Topologies

For stateful Web applications and stateful session EJBs running on OC4J, a client communicates with the same OC4J process over a series of HTTP requests and responses. However, if the OC4J process that is running the application terminates or hangs, or if the node fails, the state associated with a client request may be lost.

To protect against such software and hardware failures, you need to do these steps:

- Run OC4J instances on multiple nodes.
- Cluster the OC4J instances in the same OracleAS Cluster (OC4J).
- Cluster the applications at the application-level.

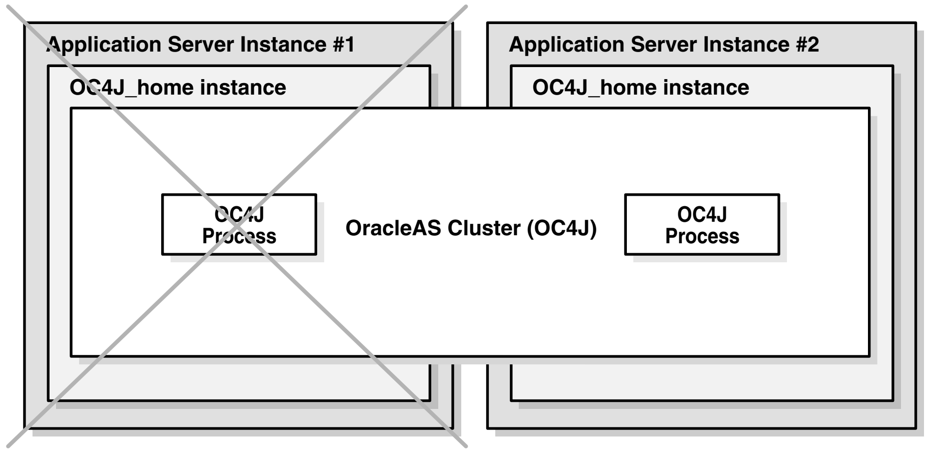
In application-level clustering in an OracleAS Cluster (OC4J), OC4J processes replicate their session state among each other. This configuration provides failover and high availability by replicating state across multiple OC4J processes running on different Oracle Application Server instances. In the event of a failure, Oracle HTTP Server forwards requests to active (alive) OC4J process within the OracleAS Cluster (OC4J).

To protect against hardware failures such as node failures, cluster OC4J instances on different nodes in the same OracleAS Cluster (OC4J). OC4J processes that run on different nodes but within the same OracleAS Cluster (OC4J) can share session state information. When an OC4J instance fails or becomes unavailable, Oracle HTTP Server forwards requests to an available OC4J process. Oracle HTTP Server forwards requests only to active (alive) OC4J processes within the cluster.

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, OPMN notifies mod_oc4j of the failure, and mod_oc4j routes requests to another OC4J process in the same cluster. The session state for the client is preserved and the client does not notice any loss of service.

Figure 3–3 shows an OracleAS Cluster (OC4J) configured across two Oracle Application Server instances, with OC4J processes running on each instance. If Oracle Application Server instance 1 fails, the OC4J process on instance 2 contains the session state information and can handle the requests. This configuration enables web application session state replication failover within an OracleAS Cluster (OC4J).

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



Minimum Number of Instances and Processes Needed

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 Oracle Application Server instances, instance 1 and instance 2, you can configure two OC4J processes on each 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 same Oracle Application Server instance. State is preserved and the client does not notice any irregularity.
- If Oracle Application Server instance 1 terminates abnormally, then the client is redirected to an OC4J process on Oracle Application Server instance 2. The state is preserved and the client does not notice any irregularity.

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

Note: Use of EJB replication OracleAS Cluster (OC4J-EJB) for high availability is independent of OracleAS Cluster (OC4J) and can involve multiple Oracle Application Server instances installed across nodes that are or are not part of OracleAS Cluster (OC4J).

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.

For more information, see chapter 24, "Configuring OC4J EJB Application Clustering Services", in the *Oracle Containers for J2EE Enterprise JavaBeans Developer's Guide*.

3.1.3 Properties of Oracle Application Server Instances in Active-Active Topologies

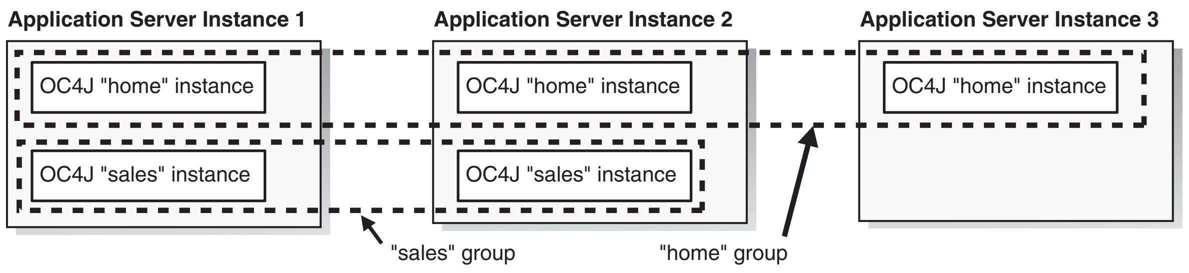
Because the load balancer can send a request to any Oracle Application Server instance in the topology, you need to ensure that the instances are configured in the same manner so that clients get the same response regardless of which instance handles the request. This includes the following:

- Deploy the same applications on each OC4J instance in the topology.
- Ensure that you replicate state and stateful session bean information across OC4J instances so that in the event that an OC4J instance fails, another OC4J instance contains the state information and can continue the session.
- Ensure that configuration properties for all the OC4J instances in the topology are identical. These configuration properties are listed in chapter 8, "Configuring and Managing Clusters", in section "Replicating Changes Across a Cluster", in the *Oracle Containers for J2EE Configuration and Administration Guide*.

3.1.4 About Groups

In an OracleAS Cluster (OC4J), OC4J instances with the same name are considered to be in a group. For example, if you have three Oracle Application Server instances, and each of these instances has an OC4J instance called "home", then these OC4J instances belong to the same group called "home". Figure 3–4 shows this example. The figure also shows an OC4J instance called "sales" on two of the Oracle Application Server instances.

Figure 3–4 Groups of OC4J Instances



Oracle Application Server creates groups automatically when it finds OC4J instances with the same name in an OracleAS Cluster (OC4J). The "home" instance is the default OC4J instance that is created during installation.

Note that a group does not need to have OC4J instances in all the Oracle Application Server instances in the cluster. For example, in Figure 3–4, the "sales" OC4J instance exists only in two of the Oracle Application Server instances. This is valid.

Oracle Application Server does not enforce that the instances in a group be configured identically. However, you should ensure that some configurations (such as data sources, JMS resources, and security provider settings) are identical so that your applications return the same response to the client regardless of which instance is processing the request.

3.1.4.1 Creating Additional OC4J Instances

You can create additional OC4J instances using the `ORACLE_HOME/bin/createinstance` command. The syntax is:

```
createinstance -instanceName name [-port httpPort]
```

`name` specifies the name of the OC4J instance you want to create.

The optional `port` parameter is useful if the Oracle Application Server instance does not contain Oracle HTTP Server. Setting the HTTP port enables you to access the home page of the new OC4J instance directly. For more information on the `createinstance` command, see chapter 8, "Configuring and Managing Clusters", in section "Creating and Managing Additional OC4J Instances", in the *Oracle Containers for J2EE Configuration and Administration Guide*.

For example, to create the "sales" OC4J instance, you can use the following command:

```
createinstance -instanceName sales
```

The command prompts you to set the oc4jadmin password for the "sales" instance. This password can be different from the oc4jadmin password for the "home" instance. When you access the "sales" instance, you will need to enter its password.

The command creates the instance but does not start it. You can start it from Application Server Control Console or from the `opmnctl` command.

After creating the instance, you should reload OPMN so that it is aware of the new instance:

```
opmnctl reload
```

3.1.4.2 Managing Instances in a Group

For a group of OC4J instances, you can manage them collectively or you can still manage them individually. You can start and stop all the applications and instances in a group or individually, and you can also deploy / redeploy / undeploy applications on all the instances in a group or on only some instances.

Deploying applications only on some instances in a group is not recommended. You should deploy the same application on all instances in a group.

You can manage groups using Application Server Control Console or `admin_client.jar`. In Application Server Control Console, you can perform these operations on groups:

Table 3–1 Performing Operations on Groups using Application Server Control Console

Operation	Steps
Start, stop, deploy, undeploy, redeploy applications in a group	**1.** On the Cluster Topology page in Application Server Control Console, scroll down to the Groups section. **2.** Click the group you want to manage. This displays the Group: *groupname* page. **3.** Select the **Applications** tab. **4.** Select the application that you want to start, stop, undeploy, or redeploy. **5.** Click the **Start**, **Stop**, **Deploy**, **Undeploy**, or **Redeploy** button.

Table 3–1 (Cont.) Performing Operations on Groups using Application Server Control Console

Operation	Steps
Configure JDBC resources and JMS providers	1. On the Cluster Topology page in Application Server Control Console, scroll down to the Groups section. 2. Click the group you want to manage. This displays the Group: *groupname* page. 3. Select the **Administration** tab. 4. To configure JDBC resources, click the "go to task" icon in the JDBC row. This displays the JDBC Resources page. For information on JDBC, see chapter "Data Sources" in the *Oracle Containers for J2EE Services Guide*. To configure JMS providers, click the "go to task" icon in the JMS providers row. This displays the OracleAS JMS page. For information on JMS, see chapter "Java Message Service (JMS)" in the *Oracle Containers for J2EE Services Guide*.
Manage OC4J instances individually	1. On the Cluster Topology page in Application Server Control Console, scroll down to the Groups section. 2. Click the group you want to manage. This displays the Group: *groupname* page. 3. Select the **OC4J Instances** tab. This displays all the instances in the group. You can click on an instance to manage it. **Note:** The operation that you perform on the instance affects that instance only. The operation is not applied to all instances in the group.
Manage OC4J instances as a group	1. On the Cluster Topology page in Application Server Control Console, click the **Cluster MBean Browser** link. 2. On the left side, expand **ias** > **J2EEServerGroup**. 3. Under J2EEServerGroup, click the group you want to manage. 4. On the right side, click the **Operations** tab. 5. To start or stop the OC4J instances in the group, click **start** or **stop** on the right side. 6. Click **Invoke Operation** to perform the operation.

3.1.4.3 Deploying Applications to a Group Using admin_client.jar

You can use the `admin_client.jar` utility to deploy applications to a group. The syntax looks like the following:

```
> cd ORACLE_HOME/j2ee/home
> java admin_client.jar
             deployer:cluster:opmn://<host>:<opmnPort>/<groupName>
             <adminID> <adminPassword>
             -deploy -file <pathToArchiveFile> -deploymentName <appName>
```

For *`<host>`*, you can specify any host in the group.

For *`<opmnPort>`*, you specify the port at which OPMN is listening. This port is listed in the `opmn.xml` file.

For *`<groupName>`*, you specify the name of the group. This is the name of the OC4J instance (for example, `home`).

For *<adminID>* and *<adminPassword>*, you specify the administrator's ID and password. Typically the adminID is `oc4jadmin`.

Note: For deployment to work across all instances in a group, the administrator's password must be the same for all instances in the group.

For *<pathToArchiveFile>*, you specify the full path to the EAR, WAR, or JAR file that you want to deploy.

For *<appName>*, you specify the application name.

There are other options that you can specify on the command-line for deployment. For details, see the *Oracle Containers for J2EE Deployment Guide*.

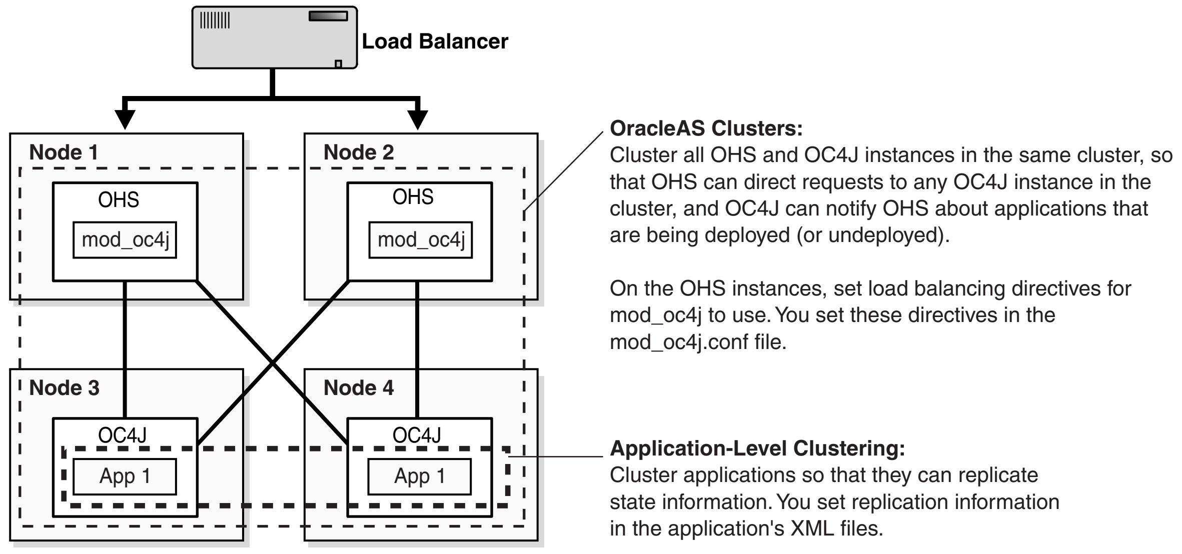
3.1.5 How Oracle HTTP Server Routes Requests to OC4J

When Oracle HTTP Server receives a request for a J2EE application, it forwards the request to the mod_oc4j module located within Oracle HTTP Server itself. Following a load balancing algorithm, mod_oc4j forwards the request to an OC4J instance in the same cluster. See Figure 3–5.

The default load balancing algorithm that mod_oc4j uses to distribute the requests is a simple round robin algorithm. You can use a different algorithm by setting a directive in the mod_oc4j.conf file. See Section 3.2.12, "Setting mod_oc4j Load Balancing Options" for details.

Note that requests that are part of a session are sent to the same OC4J instance. If the OC4J instance becomes unavailable after the first request, mod_oc4j locates another instance to process the request, and subsequent requests in the same session are also sent to that instance.

Figure 3–5 OracleAS Clusters and Load Balancing Directives


Load Balancer
Node 1
Node 2
OHS
mod_oc4j
OHS
mod_oc4j
Node 3
Node 4
OC4J
App 1
OC4J
App 1
OracleAS Clusters:
Cluster all OHS and OC4J instances in the same cluster, so that OHS can direct requests to any OC4J instance in the cluster, and OC4J can notify OHS about applications that are being deployed (or undeployed).
On the OHS instances, set load balancing directives for mod_oc4j to use. You set these directives in the mod_oc4j.conf file.
Application-Level Clustering:
Cluster applications so that they can replicate state information. You set replication information in the application's XML files.

3.1.6 Using Oracle Identity Management with Active-Active Topologies

Oracle Application Server Release 3 (10.1.3) does not provide Oracle Identity Management, but you can use Oracle Application Server Release 3 (10.1.3) with Oracle Identity Management from Release 2 (10.1.2).

You can associate the active-active topology with Oracle Identity Management, if you need Oracle Identity Management services such as Oracle Internet Directory, OracleAS Single Sign-On, OracleAS Certificate Authority, and Oracle Directory Integration and Provisioning.

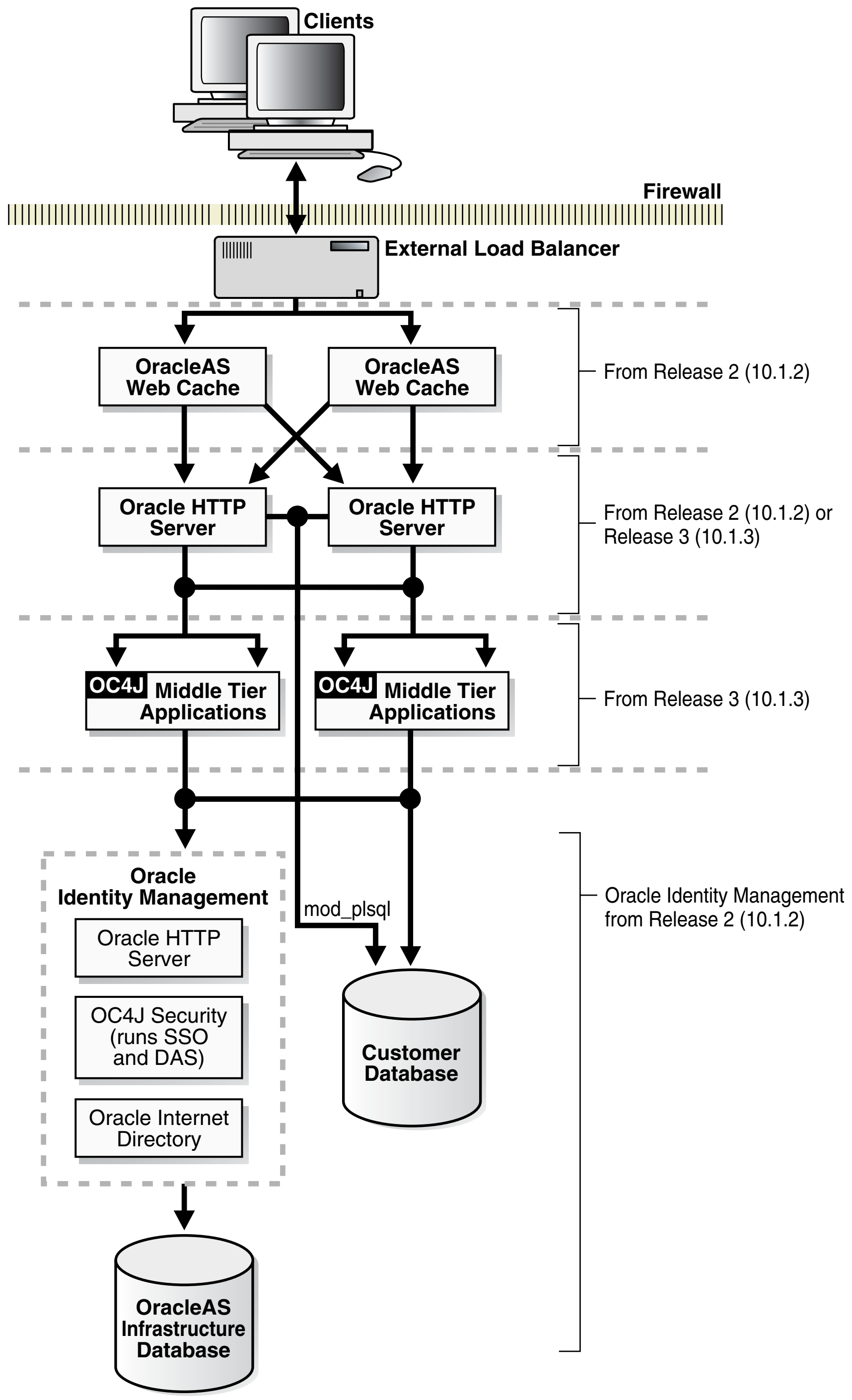
You install the Release 3 (10.1.3) active-active topology and the Oracle Identity Management instances separately. After installation, you use Application Server Control Console to associate the Release 3 (10.1.3) instances with the Oracle Internet Directory in the Oracle Identity Management.

For instructions on how to associate Oracle Identity Management with Release 3 (10.1.3) instances, see the section "Configuring Instances to Use 10.1.2 and 9.0.4 Oracle Identity Management" in the *Oracle Application Server Administrator's Guide*.

In high availability environments, you need both your Release 3 (10.1.3) instances and your Release 2 (10.1.2) Oracle Identity Management instances to be highly available. This guide describes high availability for the Release 3 (10.1.3) instances only. For Oracle Identity Management Release 2 (10.1.2), see the *Oracle Application Server High Availability Guide* for Release 2 (10.1.2).

Figure 3–6 shows Oracle Application Server Release 3 (10.1.3) instances in an active-active topology running with Oracle Identity Management from Release 2 (10.1.2).

Figure 3–6 Oracle Application Server Middle Tiers with Oracle Identity Management


Clients
Firewall
External Load Balancer
OracleAS Web Cache
OracleAS Web Cache
From Release 2 (10.1.2)
Oracle HTTP Server
Oracle HTTP Server
From Release 2 (10.1.2) or Release 3 (10.1.3)
OC4J Middle Tier Applications
OC4J Middle Tier Applications
From Release 3 (10.1.3)
Oracle Identity Management
Oracle HTTP Server
OC4J Security (runs SSO and DAS)
Oracle Internet Directory
mod_plsql
Customer Database
Oracle Identity Management from Release 2 (10.1.2)
OracleAS Infrastructure Database

3.1.7 Using Oracle HTTP Server 10.1.2 with Active-Active Topologies

Instead of using Oracle HTTP Server from Release 3 (10.1.3), you can use Oracle HTTP Server from Release 2 (10.1.2) in the active-active topology. You might want to do this for the following reasons:

- You are already using Oracle HTTP Server from Release 2 (10.1.2) with Oracle Application Server middle tiers from Release 2 (10.1.2).

- You want to use Application Server Control Console to manage Oracle HTTP Server. Application Server Control Console from Release 2 (10.1.2) provides more management capabilities for Oracle HTTP Server than Application Server Control Console from Release 3 (10.1.3).

You can use Oracle HTTP Server from Release 2 (10.1.2) only for the distributed active-active topology (Figure 3–2).

3.1.8 Using OracleAS Web Cache Release 2 (10.1.2) with Active-Active Topologies

You can use OracleAS Web Cache from Release 2 (10.1.2) with Oracle Application Server Release 3 (10.1.3), as shown in Figure 3–6.

OracleAS Web Cache is used as a reverse proxy server. The reverse proxy server can consist of an instance of OracleAS Web Cache or multiple instances configured as a cluster (called OracleAS Web Cache cluster).

For information about configuring OracleAS Web Cache (single instance or cluster) as a reverse proxy server, see section "Configuring 10.1.2 or 9.0.4 OracleAS Web Cache as a Reverse Proxy" in the *Oracle Application Server Administrator's Guide*.

3.2 Managing the Active-Active Topology

This section describes some common procedures that you may need to perform to maintain the active-active topology:

- Section 3.2.1, "Setting up OracleAS Clusters"
- Section 3.2.2, "Setting up Multicast Replication"
- Section 3.2.3, "Setting up Peer-to-Peer Replication"
- Section 3.2.4, "Setting up Replication to a Database"
- Section 3.2.5, "Setting the Replication Policy"
- Section 3.2.6, "Specifying the Number of Nodes to Replicate To"
- Section 3.2.7, "Checking Status of Components"
- Section 3.2.8, "Starting / Stopping Components in the Topology"
- Section 3.2.9, "Deploying Applications to a Cluster"
- Section 3.2.10, "Adding Instances to an Active-Active Topology"
- Section 3.2.11, "Removing Instances from an Active-Active Topology"
- Section 3.2.12, "Setting mod_oc4j Load Balancing Options"
- Section 3.2.13, "Configuring Java Message Service (JMS) for High Availability"

3.2.1 Setting up OracleAS Clusters

There are different ways to create OracleAS Clusters. This section describes only two methods:

- Section 3.2.1.1, "Dynamic Discovery Method"
- Section 3.2.1.2, "Discovery Server Method"

For additional methods, see the "Configuring and Managing Clusters" chapter in the *Oracle Containers for J2EE Configuration and Administration Guide*.

3.2.1.1 Dynamic Discovery Method

In this method, you define the same multicast address and port for each Oracle Application Server instance in the cluster. An advantage in using this method is that you do not have to specify the name of each Oracle Application Server instance in the cluster. You can add or remove instances from the cluster by editing the multicast address and port.

1. For each Oracle Application Server instance that you want to group in the same cluster, run the following command:

   ```
   opmnctl config topology update discover="*<multicastAddress>:<multicastPort>"
   ```

   `multicastAddress` specifies the multicast address that you want to use for this cluster. The multicast address must be within the valid address range, which is 224.0.0.1 to 239.255.255.255. Note that the multicast address is preceded by a `*` character in the command.

   `multicastPort` can be any unused port number.

   Example:

   ```
   > ORACLE_HOME/opmn/bin/opmnctl config topology update
                            discover="*225.0.0.20:8001"
   ```

   In distributed installations (Oracle HTTP Server and OC4J on different Oracle homes), you need to cluster all the Oracle Application Server instances into the same cluster. You need to use the same multicast IP and port for all the instances.

2. For each Oracle Application Server instance where you ran the "`opmnctl config topology update`" command, run the "`opmnctl reload`" command to force OPMN to read the updated `opmn.xml` file.

   ```
   > ORACLE_HOME/opmn/bin/opmnctl reload
   ```

3.2.1.2 Discovery Server Method

If you do not want to use the multicast method, you can define a cluster by specifying the names of the nodes running the Oracle Application Server instances in the `opmn.xml` file of each instance.

Example: if you want to cluster four instances (inst1.node1.mycompany.com, inst2.node2.mycompany.com, inst3.node3.mycompany.com, inst4.node4.mycompany.com), you would perform these steps:

1. Designate at least one of the instances to serve as the "discovery server". The discovery server maintains the topology for the cluster.

   This example assumes that inst1.node1.mycompany.com and inst2.node2.mycompany.com will be the discovery servers for the cluster.

   In distributed installations (Oracle HTTP Server and OC4J on different Oracle homes), any instance, whether running Oracle HTTP Server or OC4J, can serve as the discovery server.

2. In the `opmn.xml` file for all instances in the cluster, specify the nodes that are running the discovery servers (node1.mycompany.com and node2.mycompany.com in the example).

   In the example, the `opmn.xml` file is changed to include the following lines:

   ```
   <notification-server>
      <topology>
   ```

```
            <discover
                list="node1.mycompany.com:6201,node2.mycompany.com:6201"/>
        </topology>
    ...
    </notification-server>
```

The 6201 specifies the port number at which the notification server is listening. You can find this value in the `opmn.xml` file of that instance.

If you have more than one discovery server, you separate them with the comma character.

3. On all the instances, run "`opmnctl reload`" to force OPMN to read the updated `opmn.xml` file.

```
> ORACLE_HOME/opmn/bin/opmnctl reload
```

3.2.2 Setting up Multicast Replication

Multicast replication is the default replication type. To set up an application to use multicast replication, you can just add the empty `<cluster/>` tag to the application's `orion-application.xml` file or to the global `ORACLE_HOME/j2ee/home/config/application.xml` file. For example:

```
<orion-application ... >
   ...
   <cluster/>
</orion-application>
```

You need to add the `<cluster/>` tag on all nodes where the application is deployed.

By default, multicast replication uses multicast address 230.230.0.1 and port 45566. If you want to change these values, you specify the desired values in the `ip` and `port` attributes of the `multicast` element. For example, the following snippet shows the `ip` and `port` attributes set to customized values:

```
<orion-application ... >
   ...
   <cluster allow-colocation="false">
      <replication-policy trigger="onShutdown" scope="allAttributes"/>
      <protocol>
         <multicast ip="225.130.0.0" port="45577" bind-addr="226.83.24.10"/>
      </protocol>
   </cluster>
</orion-application>
```

The multicast address must be between 224.0.0.1 and 239.255.255.255.

Description of other tags and attributes used in the snippet above:

- `allow-colocation`: specifies whether or not application state is replicated to other Oracle Application Server instances running on the same host. The default is true.
- `trigger` and `scope`: see Section 3.2.5, "Setting the Replication Policy".
- `bind-addr`: specifies the IP of the network interface card (NIC) to bind to. This is useful if the host machine has multiple NICs, each with its own IP address.

3.2.3 Setting up Peer-to-Peer Replication

Oracle Application Server supports two types of peer-to-peer replication: dynamic and static.

- In dynamic peer-to-peer replication, OC4J discovers other OC4J instances through OPMN. You do not have to list the names of the instances in a configuration file.
- In static peer-to-peer replication, you list the names of the instances that you want to be involved in the replication.

Dynamic Peer-to-Peer Replication

To specify dynamic peer-to-peer replication, you include an empty `<opmn-discovery/>` tag in the application's `orion-application.xml` file or in the global `ORACLE_HOME/j2ee/home/config/application.xml` file

```
<orion-application ... >
   ...
   <cluster allow-colocation="false">
      <replication-policy trigger="onShutdown" scope="allAttributes"/>
      <protocol>
         <peer>
            <opmn-discovery/>
         </peer>
      </protocol>
   </cluster>
</orion-application>
```

You defined how OPMN discovers instances in a cluster when you set up the OracleAS Clusters. See Section 3.2.1, "Setting up OracleAS Clusters" for details.

Static Peer-to-Peer Replication

To specify static peer-to-peer replication, you list the names of the hosts in the `<node>` element in the application's `orion-application.xml` file or in the global `ORACLE_HOME/j2ee/home/config/application.xml` file. For each node, you specify another node in the active-active topology such that all the nodes in the topology are connected in the chain. For example, if you have three Oracle Application Server instances in your topology, node 1 can specify node 2, node 2 can specify node 3, and node 3 can specify node 1.

Example:

On node 1, the `<node>` tag specifies node 2:

```
<orion-application ... >
   ...
   <cluster allow-colocation="false">
      <replication-policy trigger="onShutdown" scope="allAttributes"/>
      <protocol>
         <peer start-port="7900" range="10" timeout="6000">
            <node host="node2.mycompany.com" port="7900"/>
         </peer>
      </protocol>
   </cluster>
</orion-application>
```

On node 2, the `<node>` tag specifies node 3:

```
<orion-application ... >
   ...
   <cluster allow-colocation="false">
```

```
      <replication-policy trigger="onShutdown" scope="allAttributes"/>
      <protocol>
         <peer start-port="7900" range="10" timeout="6000">
            <node host="node3.mycompany.com" port="7900"/>
         </peer>
      </protocol>
   </cluster>
</orion-application>
```

On node 3, the `<node>` tag specifies node 1:

```
<orion-application ... >
   ...
   <cluster allow-colocation="false">
      <replication-policy trigger="onShutdown" scope="allAttributes"/>
      <protocol>
         <peer start-port="7900" range="10" timeout="6000">
            <node host="node1.mycompany.com" port="7900"/>
         </peer>
      </protocol>
   </cluster>
</orion-application>
```

Another way of doing this is to have all the nodes specify the same node. In a three-node example, you could also have nodes 1 and 2 specify node 3, and node 3 can specify either node 1 or node 2.

Description of the tags and attributes used in the example above:

- `start-port`: specifies the first port on the local node that Oracle Application Server tries to bind to for peer communication. If this port is already in use, Oracle Application Server increments the port number until it finds an available port. The default is 7800.
- `timeout`: specifies the length of time in milliseconds to wait for a response from the specified peer node. The default is 3000 milliseconds.
- `host`: specifies the name of the peer node.
- `port`: specifies the port to use on the specified host (in the `host` attribute) for peer communication. The default is 7800.
- `range`: specifies the number of times to increment the port specified on the `port` (not `start-port`) attribute. The default is 5.

Note the following:

- In static peer-to-peer replication, the application's `orion-application.xml` file is different for each instance. When you deploy your application, you have to make sure that you update the `orion-application.xml` accordingly.

3.2.4 Setting up Replication to a Database

In this replication mechanism, the replicated data is saved to a database. You specify the database in the `<database>` tag in the application's `orion-application.xml` file or in the global `ORACLE_HOME/j2ee/home/config/application.xml` file. For example:

```
<orion-application ... >
   ...
   <cluster allow-colocation="false">
      <replication-policy trigger="onShutdown" scope="allAttributes"/>
      <protocol>
         <database data-source="jdbc/MyOracleDS"/>
      </protocol>
   </cluster>
</orion-application>
```

The value for the `data-source` attribute must match the data source's `jndi-name` as specified in the `data-sources.xml` file. See the *Oracle Containers for J2EE Services Guide* for details on creating and using data sources.

3.2.5 Setting the Replication Policy

Attributes in the `<replication-policy>` tag enable you to specify which data is to be replicated and how frequently the data is replicated.

The trigger attribute

The `trigger` attribute specifies when replication occurs. Table 3–2 describes supported values for this attribute:

Table 3–2 Values for the trigger Attribute

Value	HttpSession	Stateful Session Bean
`onSetAttribute`	Replicate each change made to an HTTP session attribute at the time the value is modified. From a programmatic standpoint, replication occurs each time `setAttribute()` is called on the `HttpSession` object. This option can be resource intensive in cases where the session is being extensively modified.	Not applicable.
`onRequestEnd` (default)	Queue all changes made to HTTP session attributes, then replicate all changes just before the HTTP response is sent.	Replicate the current state of the bean after each EJB method call. The state is replicated frequently, but offers higher reliance.
`onShutdown`	Replicate the current state of the HTTP session whenever the JVM is terminated gracefully, such as with Control-C. State is not replicated if the host is terminated unexpectedly, as in the case of a system crash. Because session state was not previously replicated, all session data is sent across the network at once upon JVM termination, which can impact network performance. This option can also significantly increase the amount of time needed for the JVM to shut down.	Replicate the current state of the bean whenever the JVM is terminated gracefully. State is not replicated if the host is terminated unexpectedly, as in case of a system crash. Because bean state was not previously replicated, all state data is sent across the network at once upon JVM termination, which can impact network performance. This option may also significantly increase the amount of time needed for the JVM to shut down.

The scope attribute

The `scope` attribute specifies which data is replicated. Table 3–3 describes supported values for the attribute:

Table 3–3 Values for the scope Attribute

Value	HttpSession	Stateful Session Bean
`modifiedAttributes`	Replicate only the modified HTTP session attributes. This is the default replication setting for HttpSession.	Not applicable.
`allAttributes`	Replicate all attribute values set on the HTTP session.	Replicate all member variable values set on the stateful session bean. This is the default replication setting for stateful session beans.

3.2.6 Specifying the Number of Nodes to Replicate To

To specify the number of nodes to replicate to, use the `write-quota` attribute of the `<cluster>` tag. For example, the following snippet specifies that the replicated data is replicated to two other nodes.

```
<orion-application ... >
   ...
   <cluster allow-colocation="false" write-quota="2">
      <replication-policy trigger="onShutdown" scope="allAttributes"/>
      <protocol>
         <peer>
            <opmn-discovery/>
         </peer>
      </protocol>
   </cluster>
</orion-application>
```

The default is 1.

Recommendations: For a two-node active-active topology, set `write-quota` to `1`, so that the data is replicated to the other node.

For topologies with three or more nodes, set `write-quota` to at least `2` to ensure that the data is replicated to at least two other nodes.

To replicate data to all nodes in the topology, set `write-quota` to the total number of nodes in the topology. It is possible to write back to the same node if there is another instance running on that node.

The `write-quota` attribute is not used if you are replicating to database.

3.2.7 Checking Status of Components

To check the status of instances in the active-active topology, run the following command from any instance in the topology:

```
> cd ORACLE_HOME/opmn/bin
> opmnctl @cluster status
```

3.2.8 Starting / Stopping Components in the Topology

You can use the `opmnctl` command to start and stop components in the topology. To start and stop components on all Oracle Application Server instances in the topology, you need to use the `@cluster` parameter in `opmnctl`. You can run the `opmnctl` command from any instance in the topology.

For example, to start the Oracle HTTP Server component on all instances in the topology, run the following command from any instance in the topology:

```
> cd ORACLE_HOME/opmn/bin
> opmnctl @cluster startproc ias-component=HTTP_Server
```

3.2.9 Deploying Applications to a Cluster

You can deploy applications using Application Server Control Console or using commands that you run from the command-line.

If you want to deploy your application to all instances within a cluster, you can do so using `admin_client.jar`, as follows:

```
> cd ORACLE_HOME/j2ee/home
> java admin_client.jar
             deployer:cluster:opmn://<host>:<opmnPort>/<oc4jInstanceName>
             <adminID> <adminPassword>
             -deploy -file <pathToArchiveFile> -deploymentName <appName>
```

For *`<host>`*, you can specify any host in the cluster.

For *`<opmnPort>`*, you specify the port at which OPMN is listening. This port is listed in the `opmn.xml` file.

For *`<oc4jInstanceName>`*, you specify the OC4J instance to which you want to deploy your application. Example: to deploy on the "home" instance, you specify `home`.

For *`<adminID>`* and *`<adminPassword>`*, you specify the administrator's ID and password. Typically the adminID is `oc4jadmin`.

Note: For deployment to work across all instances in a cluster, the administrator's password must be the same for all instances in the cluster.

For *`<pathToArchiveFile>`*, you specify the full path to the EAR, WAR, or JAR file that you want to deploy.

For *`<appName>`*, you specify the application name.

There are other options that you can specify on the command-line for deployment. For details, see the *Oracle Containers for J2EE Deployment Guide*.

3.2.10 Adding Instances to an Active-Active Topology

To add instances to an existing topology:

- Use the existing cluster method that the topology is using. For example, if the topology is using the dynamic discovery method, then you need to configure the new instance to use the same multicast IP and port.
- To cluster the OC4J component, follow the existing model already in use by the topology.
- Deploy the same applications on the new instances.
- Configure your load balancer to direct requests to the new node.

3.2.11 Removing Instances from an Active-Active Topology

To remove an instance from a topology:

- Reconfigure your load balancer so that it no longer directs requests to the removed instance.
- Remove the instance from OracleAS Clusters by removing the tags that you added to the instance. You added these tags when you set up the cluster. See Section 3.2.1, "Setting up OracleAS Clusters" for details.
- Remove the instance from application-level clustering by removing these tags:
  - The `<distributable/>` tag from the `web.xml` file for all Web modules that are part of an application configured for clustering
  - The `<cluster>` tag that you added to the application's `orion-application.xml` file or to the global `ORACLE_HOME/j2ee/home/config/application.xml` file

3.2.12 Setting mod_oc4j Load Balancing Options

The mod_oc4j module within Oracle HTTP Server delegates requests to OC4J processes. Whenever Oracle HTTP Server receives a request for a URL that is intended for OC4J, Oracle HTTP Server routes the request to the mod_oc4j module, which then routes the request to an OC4J process. 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.

You can configure mod_oc4j to load balance requests to OC4J processes. Oracle HTTP Server, through mod_oc4j, supports different load balancing policies. Load balancing policies provide performance benefits along with failover and high availability, depending on the network topology and host machine capabilities.

You can specify different load balancing routing algorithms for mod_oc4j 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.

By default, all OC4J instances have the same weight (all instances have a weight of 1), and mod_oc4j uses the round robin method to select an OC4J instance to forward a request to. An OC4J instance's weight is taken as a ratio compared to the weights of the other available OC4J instances in the topology to define the number of requests the instance should service. If the request belongs to an established session, mod_oc4j forwards the request to the same OC4J instance and the same OC4J process that started the session.

The mod_oc4j load balancing options do not take into account the number of OC4J processes running on an OC4J instance when determining which OC4J instance to send a request to. OC4J instance selection is based on the configured weight for the instance, and its availability.

To modify the mod_oc4j load balancing policy, set the `Oc4jSelectMethod` and the `Oc4jRoutingWeight` directives in the `ORACLE_HOME/Apache/Apache/conf/mod_oc4j.conf` file:

1. In the `mod_oc4j.conf` file on each Oracle Application Server instance, within the `<IfModule mod_oc4j.c>` section, set the `Oc4jSelectMethod` directive to one of the values shown in Table 3–4.

   If you set the `Oc4jSelectMethod` directive to either `roundrobin:weighted` or `random:weighted`, you may also need to set the `Oc4jRoutingWeight` directive to specify the weight (see the next step).

   See "Choosing a mod_oc4j Load Balancing Algorithm" on page 3-21 for tips on choosing a routing algorithm.

Table 3–4 Values for Oc4jSelectMethod

Value	Description
`roundrobin` (default)	mod_oc4j places all the OC4J processes in the topology in a list, and it selects processes in order from the list.
`roundrobin:local`	Similar to `roundrobin`, but the list includes only local OC4J processes. If no local OC4J processes are available, then it selects a remote OC4J process.
`roundrobin:weighted`	mod_oc4j distributes the total request load to each OC4J instance based on routing weight configured on each instance. It then selects OC4J processes from the local instance in a round robin manner. You configure the weight using the `Oc4jRoutingWeight` directive.
`random`	mod_oc4j randomly selects an OC4J process from a list of all OC4J processes in the topology.
`random:local`	Similar to `random`, but mod_oc4j gives preference to local OC4J processes. If no local OC4J processes are available, then it selects a remote OC4J process.
`random:weighted`	mod_oc4j selects an OC4J process based on the weight configured for each instance in the topology. You configure the weight using the `Oc4jRoutingWeight` directive.
`metric`	mod_oc4j routes requests based on runtime metrics that indicate how busy a process is.
`metric:local`	Similar to metric, but mod_oc4j gives preference to local OC4J processes. If no local OC4J processes are available, then it routes to a remote OC4J process.

   Example:

   ```
   Oc4jSelectMethod random:local
   ```

   For information on how to set up metric-based load balancing, see the "Load Balancing Using mod_oc4j" appendix in the *Oracle HTTP Server Administrator's Guide*.

2. If you set the `Oc4jSelectMethod` directive to a weight-based method (that is, `roundrobin:weighted` or `random:weighted`), you may also need to set the `Oc4jRoutingWeight` directive to specify the weight.

   If you do not set the `Oc4jRoutingWeight` directive, it defaults to 1.

   Example: If you have a topology that consists of three instances (A, B, and C), and you want B and C to get twice as many requests as A, set the following directives for B and C:

   ```
   Oc4jSelectMethod roundrobin:weighted
   Oc4jRoutingMethod 2
   ```

   For A, you can just set the `Oc4jSelectMethod` directive. Setting `Oc4jRoutingMethod` is optional because the default value is 1.

3. Restart Oracle HTTP Server on all instances in the topology for the changes to take effect.

   ```
   > opmnctl @cluster restartproc ias-component=HTTP_Server
   ```

Choosing a mod_oc4j Load Balancing Algorithm

Use the following guidelines to help determine which mod_oc4j load balancing option to use:

- In a topology with identical machines running Oracle HTTP Server and OC4J in the same Oracle home, the round robin with local affinity algorithm is preferred. 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 distributed deployment, where one set of machines runs Oracle HTTP Server and another set runs OC4J instances that handle requests, the preferred algorithms are simple round robin and simple metric-based. To determine which of these two works better 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 Oracle Application Server instances run on nodes that have different characteristics, the weighted round robin algorithm is preferred. In addition to setting the weight for each instance, remember to tune the number of OC4J processes running on each Oracle Application Server instance to achieve the maximum benefit. For example, a machine with a weight of 4 gets four times as many requests as a machine with a weight of 1, but you need to ensure that the system with a weight of 4 is running four 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.

3.2.13 Configuring Java Message Service (JMS) for High Availability

For information on how to configure JMS for high availability, see chapter 3, "Java Message Service (JMS)", in the *Oracle Containers for J2EE Services Guide*.

3.3 Summary of High Availability Features in Oracle HTTP Server and OC4J

Table 3–5 summarizes some of the high availability features in Oracle HTTP Server and OC4J.

Table 3–5 Summary of High Availability Features in Oracle HTTP Server and OC4J

Item	Description
Protection from Node Failure	**Oracle HTTP Server**: A load balancer deployed in front of Oracle HTTP Server instances protects from node failure. The load balancer can be an external load balancer or OracleAS Web Cache (OracleAS Web Cache is from Release 2 (10.1.2)). **OC4J**: mod_oc4j routes requests only to OC4J instances that are alive. You should install and run OC4J instances on different nodes to provide greater probability that OC4J is alive and running on at least one node at any time.
Protection from Service Failure	**Oracle HTTP Server**: A load balancer deployed in front of Oracle HTTP Server instances sends request to another Oracle HTTP Server if first one does not respond or is deemed failed through URL pings. The load balancer can be an external load balancer or OracleAS Web Cache. **OC4J**: OPMN monitors OC4J processes and restarts them upon process failure. OPMN also notifies mod_oc4j if the restart was not successful so that mod_oc4j will send requests only to OC4J processes that are alive.
Protection from Process Failure	**Oracle HTTP Server**: 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 topology fails. **OC4J**: OPMN monitors OC4J processes and restarts them upon process failure. OPMN also notifies mod_oc4j if the restart was not successful so that mod_oc4j will send requests only to OC4J processes that are alive.
Automatic Re-routing	**Oracle HTTP Server**: A load balancer deployed in front of Oracle HTTP Server instances automatically re-routes to another Oracle HTTP Server if the first one does not respond. **OC4J**: mod_oc4j automatically re-routes to another OC4J process if the first one does not respond.

See Also:

- Section 2.1, "Process Management through OPMN"

3.4 Miscellaneous Topics

- Section 3.4.1, "JNDI Namespace Replication"
- Section 3.4.2, "EJB Client Routing"
- Section 3.4.3, "OC4J Distributed Caching Using Java Object Cache"

3.4.1 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 Clusterss (OC4J) in an OC4J instance have visibility into the same replicated JNDI namespace.

See the *Oracle Containers for J2EE Services Guide* for details on JNDI.

3.4.2 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.

3.4.3 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 on 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 enables 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 Clusters (OC4J), Oracle Application Server instance, or overall Oracle Application Server Cluster.

By using Java Object Cache, performance can be improved because 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 is still available.

See the "Java Object Cache" chapter in the *Oracle Containers for J2EE Services Guide* for details on using Java Object Cache.

4 Active-Passive Topologies

This chapter describes how to configure and manage active-passive topologies. It contains the following sections:

- Section 4.1, "About Active-Passive Topologies"
- Section 4.2, "Managing Active-Passive Topologies"
- Section 4.3, "Summary of High Availability Features in Oracle HTTP Server and OC4J in Active-Passive Topologies"

4.1 About Active-Passive Topologies

An active-passive topology consists of the following:

- Two nodes in a hardware cluster
- A virtual hostname and IP address
- A shared storage, to be shared between the two nodes

You install the Oracle home on the shared storage. During runtime in an active-passive topology, only one node is active. The other node is passive. The active node mounts the shared storage so that it can access the files and runs all the processes and handles all the requests. Clients access the active node through the virtual hostname. Clients do not need to know the physical hostnames of the nodes in the topology.

If the active node fails for any reason, a failover event occurs and the passive node takes over and becomes the active node. It mounts the shared storage and runs all the processes and handles all the requests. The virtual hostname and IP now point to the passive node. Clients, because they access the nodes using the virtual hostname, do not know that it is the passive node that is servicing their requests.

The nodes need to be in hardware cluster to enable failover.

Note: Installing the Oracle home on the local storage of each node in the OracleAS Cold Failover Cluster topology is not supported. You have to install it on the shared storage.

Vendor Clusterware

The two nodes in an active-passive topology are in a hardware cluster, which typically includes some vendor clusterware. For a list of certified clusterware, visit the Oracle Technology Network website (`http://www.oracle.com/technology`).

If you are running on Windows, you need the following products for the cluster:

- Oracle Fail Safe
- Microsoft Cluster Server

These products must be installed on both nodes (active and passive) in the topology.

Active-Passive Topologies: Advantages

- Easier to implement because they do not require a load balancer, which is required in active-active topologies
- Easier to configure than active-active topologies because you do not have to configure options such as load balancing algorithms, clustering, and replication.
- Better simulates a one-instance topology than active-active topologies

Active-Passive Topologies: Disadvantages

- Does not scale as well as active-active topologies. You cannot add nodes to the topology to increase capacity.
- State information (from HTTP session state and EJB stateful session bean) is not replicated, and thus becomes lost when a node terminates unexpectedly unless you save the state information to persistent storage such as a database.

Figures of Active-Passive Topologies

Figure 4–1 shows a diagram of an active-passive topology with the Oracle Application Server Oracle home installed on the shared storage. The Oracle home contains both Oracle HTTP Server and OC4J. Figure 4–2 shows a distributed active-passive topology, where Oracle HTTP Server and OC4Jare installed on different Oracle home.

Figure 4–1 Active-Passive Topology with Oracle HTTP Server and OC4J in the Same Oracle Home

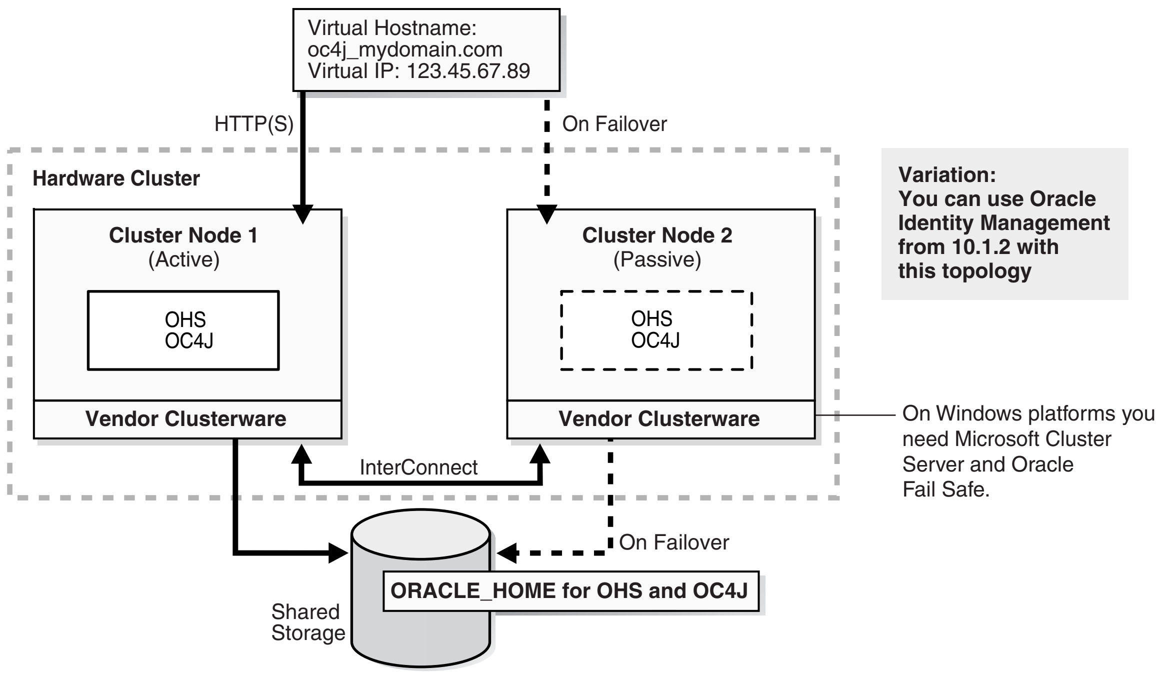
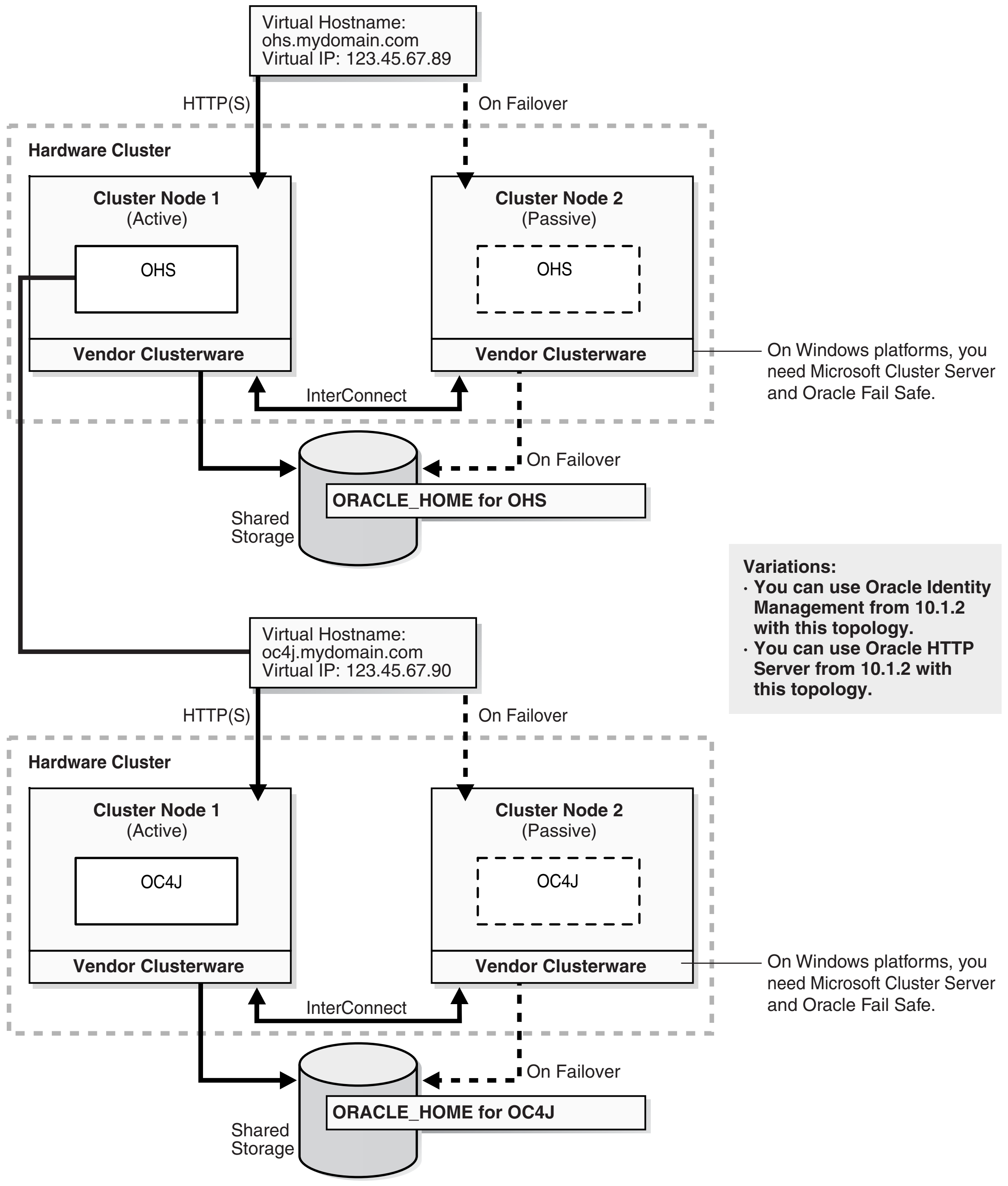


Figure 4–2 Active-Passive Topology with Oracle HTTP Server and OC4J in Separate Oracle Homes



4.2 Managing Active-Passive Topologies

Managing active-passive topologies is very similar to managing single instance Oracle Application Server topologies because there is only one Oracle home to manage.

Topics covered in this section:

- Section 4.2.1, "Managing through Application Server Control Console"
- Section 4.2.2, "Starting / Stopping Components"

- Section 4.2.3, "Deploying Applications"

4.2.1 Managing through Application Server Control Console

You can use Application Server Control Console to manage active-passive topologies. To access Application Server Control Console, use the virtual hostname instead of the physical hostname.

4.2.2 Starting / Stopping Components

You can use the opmnctl command to start and stop components in the topology. For example:

```
opmnctl startall
```

You can also use the Application Server Control Console to perform the start and stop operations.

4.2.3 Deploying Applications

You deploy applications in the normal manner. You just deploy the applications through the active node.

For details on deploying applications, see the *Oracle Containers for J2EE Deployment Guide*.

4.3 Summary of High Availability Features in Oracle HTTP Server and OC4J in Active-Passive Topologies

Table 4–1 summarizes some of the high availability features in Oracle HTTP Server and OC4J in active-passive topologies.

Table 4–1 Summary of High Availability Features in Oracle HTTP Server and OC4J in Active-Passive Topologies

Item	Description
Protection from Node Failure	**Oracle HTTP Server**: If the active node fails, the passive node takes over and runs the Oracle HTTP Server processes. **OC4J**: If the active node fails, the passive node takes over and runs the OC4J processes.
Protection from Service Failure	**Oracle HTTP Server** and **OC4J**: On Windows, Oracle Fail Safe monitors the services on the active node. If a service goes down, Oracle Fail Safe tries to restart the service. If the service fails to start, then Oracle Fail Safe fails over the instance to node 2 (the passive node) and starts the services. On UNIX, vendor clusterware provides similar services as Oracle Fail Safe.
Protection from Process Failure	**Oracle HTTP Server**: 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 topology fails. **OC4J**: OPMN monitors OC4J processes and restarts them upon process failure. OPMN also notifies mod_oc4j if the restart was not successful so that mod_oc4j will send requests only to OC4J processes that are alive. See Section 2.1, "Process Management through OPMN" for more information on OPMN.

Part III

Disaster Recovery

The chapter in this part describes the Oracle Application Server Disaster Recovery solution.

This part contains the following chapter:

5 OracleAS 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 either identically and symmetrically (same number of instances) or asymmetrically to the production site (fewer instances or OracleAS Disaster Recovery capability for the Infrastructure services is only supported). Under normal operation, the production site actively services requests. The standby site is maintained to mirror or closely mirror the applications and content hosted by the production site.

The OracleAS Disaster Recovery aspects of these sites are managed using Oracle Application Server Guard, which contains a command-line utility (asgctl) that encapsulates administrative tasks (see Chapter 6, "OracleAS Guard asgctl Command-line Reference" for reference information about these administrative commands). The OracleAS Disaster Recovery solution leverages the following services among other system services that are available across the entire site. Behind the scenes OracleAS Guard automates the use of OracleAS Recovery Manager (for managing configuration files in the file system) and Oracle Data Guard (for managing the OracleAS Infrastructure database) in a distributed fashion across the topology. Table 5–1 provides a summary of the OracleAS Disaster Recovery strategy and how this Oracle software is used behind the scenes:

Table 5–1 Overview of OracleAS Disaster Recovery strategy

Coverage	Procedure	Purpose
Middle-tier Configuration Files	OracleAS Recovery Manager	To back up and clone 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 Recovery Manager	To back up and clone OracleAS configuration files in the production site OracleAS Infrastructure node and restore them to the standby site OracleAS Infrastructure node.

Table 5–1 (Cont.) Overview of OracleAS Disaster Recovery strategy

Coverage	Procedure	Purpose
OracleAS Infrastructure Database	Oracle Data Guard	To ship archive logs from production site OracleAS Infrastructure database to standby site OracleAS Infrastructure database. Logs are not applied immediately.

Beginning with OracleAS release 10.1.2.0.2, to simplify the concepts presented to describe the OracleAS Disaster Recovery solution, the term topology is introduced to mean all farms on either the production or standby site with the same Oracle Internet Directory instance. With the exception of the OracleAS Disaster Recovery solution for OracleAS release 10.1.3, the term topology replaces the previous concept of a farm as described in the OracleAS Disaster Recovery solution documentation for OracleAS release 10.1.2.0.0. The term topology refers to all instances that share the same Oracle Internet Directory for a production site. The discover topology command queries Oracle Internet Directory to determine the list of instances and then generates a topology XML file that describes the production topology. The discover topology within farm command is used in cases where Oracle Internet Directory is not available and then OracleAS Guard uses OPMN to discover the topology within the farm.

Note: Your other databases must be covered in the overall disaster recovery strategy, and 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 5.12, "Runtime Operations -- OracleAS Guard Switchover and Failover Operations".

This chapter is organized into the following sections:

- Section 5.10, "OracleAS Guard Operations -- Standby Site Cloning of One or More Production Instances to a Standby System"
- Section 5.11, "OracleAS Guard Operations -- Standby Instantiation and Standby Synchronization"
- Section 5.12, "Runtime Operations -- OracleAS Guard Switchover and Failover Operations"
- Section 5.13, "Monitoring OracleAS Guard Operations and Troubleshooting"
- Section 5.14, "Wide Area DNS Operations"
- Section 5.15, "Using OracleAS Guard Command-Line Utility (asgctl)"
- Section 5.16, "Special Considerations for Some OracleAS Metadata Repository Configurations"
- Section 5.17, "Special Considerations for OracleAS Disaster Recovery Environments"

> **See Also:** *Oracle Application Server Installation Guide* for instructions about how to install the OracleAS Disaster Recovery solution.

Geographically Distributed Identity Management Infrastructure Deployment

Geographically distributed Identity Management (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 (OID), OracleAS Metadata Repository (MR), and OracleAS Single Sign-On (SSO) are set up in replication and distributed across different geographic regions. Each OracleAS Single Sign-On site uses its own Oracle Internet Directory and OracleAS Metadata Repository located at the local site, thus resulting in the active-active configuration. The shared similarities serve two purposes. First, 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. Second, in case an 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.

5.1 Oracle Application Server 10*g* Disaster Recovery Solution

The Oracle Application Server Disaster Recovery solution consists of two configured sites - one primary (production/active) and one secondary (standby). Both sites may or may not have the following: same number of middle tiers and the same number of OracleAS Infrastructure nodes, and the same number of components installed. In other words, the installations on both sites, middle tier and OracleAS Infrastructure could be identical (symmetrical topology) or not identical (asymmetrical topology). Both sites are usually dispersed geographically, and if so, they are connected through a wide area network.

Some important points to emphasize for the Oracle Application Server Disaster Recovery solution are the following:

- The number of instances required on the standby site to run your site can be identical to (symmetric) or fewer (asymmetric) than the production site.
- The set of instances needed must be created and installed on the standby site for Disaster Recovery set up.
- The standby site needs the minimum set of instances required to run your site.

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 5.1.1, "OracleAS Disaster Recovery Requirements"
- Section 5.1.2, "Supported Oracle Application Server Releases and Operating Systems"
- Section 5.1.3, "Supported Topologies"

5.1.1 OracleAS Disaster Recovery Requirements

To ensure that your implementation of the OracleAS Disaster Recovery solution performs as designed, the following requirements must 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 middle-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 5.2.1, "Planning and Assigning Hostnames" for information about how to change these physical hostnames and the virtual hostname. See Section 1.2.1, "Terminology" for a description of physical hostname and virtual hostname.

  - Virtual hostname and port numbers for the OracleAS Infrastructure. The virtual hostname can be specified in the **Specify Virtual Hostname** 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 a 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 for 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 directory for each OracleAS installation on a node. Do not use symbolic links anywhere in the path.
- Oracle Application Server installation types (Any instance installed on the standby system must be identical to that installed on the production system):
    * Middle Tier: J2EE and Web Cache, and Portal and Wireless
    * OracleAS Infrastructure: Metadata Repository (MR) and Identity Management (IM)

5.1.2 Supported Oracle Application Server Releases and Operating Systems

OracleAS Guard supports Oracle Application Server releases 10g (9.0.4) and 10g (10.1.2.0.0, 10.1.2.0.2, and 10.1.3.0.0).

The OracleAS Guard kit located on the Companion CD #2 must be installed on all systems with Oracle homes or Oracle Application Server instances in the topology. See the OracleAS Disaster Recovery installation information in *Oracle Application Server Installation Guide* for more information.

OracleAS Guard is installed by default on every 10.1.2 and 10.1.3 install type that already installs a JDK or JRE environment. In environments in which OracleAS Guard is not installed by default, for example Web Server and Process Management in OracleAS 10g release (10.1.3) and OracleAS Portal in OracleAS 10g release (10.1.2.0.2), the standalone OracleAS Guard kit must be installed from the Companion CD #2 into these Oracle homes.

OracleAS Guard supports mixed environments that are supported by the underlying services. For example, OracleAS Guard supports a mixed OracleAS 10g (9.0.4) and OracleAS 10g (10.1.2) or higher release environment, such as may occur during an upgrade scenario. In this case, you may have upgraded the standby site to OracleAS 10g (10.1.2) Oracle homes in the middle tier but the Infrastructure is still an OracleAS 10g (9.0.4) Infrastructure. This mixed release environment will work as long as the OracleAS Disaster Recovery peer home for the given production middle tier Oracle homes are also upgraded to the same release, for example OracleAS 10g (10.1.2) and as long as its Infrastructure is still an earlier release, for example OracleAS 10g (9.0.4) Infrastructure. So the rule of thumb is that all Oracle home peer middle tiers within the topology must match exactly in release number for all peers (that is, on both the standby and production sites). Also the Infrastructures must match exactly in release number on both the standby and production sites and Oracle AS Guard must be the same version on both sites. However, as an upgrade based requirement, the Infrastructure can be at a lower release than the middle tiers because this happens during an upgrade scenario.

5.1.3 Supported Topologies

OracleAS Disaster Recovery supports a number of basic topologies for the configuration of the Infrastructure and middle tier on production and standby sites. OracleAS Disaster Recovery supports these basic topologies:

- Symmetrical Topologies - Strict Mirror of the Production Site with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure
- Asymmetrical Topologies - Simple Asymmetric Standby Topology with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure
- Separate OracleAS Metadata Repository for OracleAS Portal with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure (the Departmental Topology)
- Distributed Application OracleAS Metadata Repositories with Non Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure
- Redundant Multiple OracleAS 10.1.3 Homes J2EE Topology
- Redundant Single OracleAS 10.1.3 Oracle Home J2EE Topology Integrated with an Existing Oracle Identity Management 10.1.2.0.2 Topology

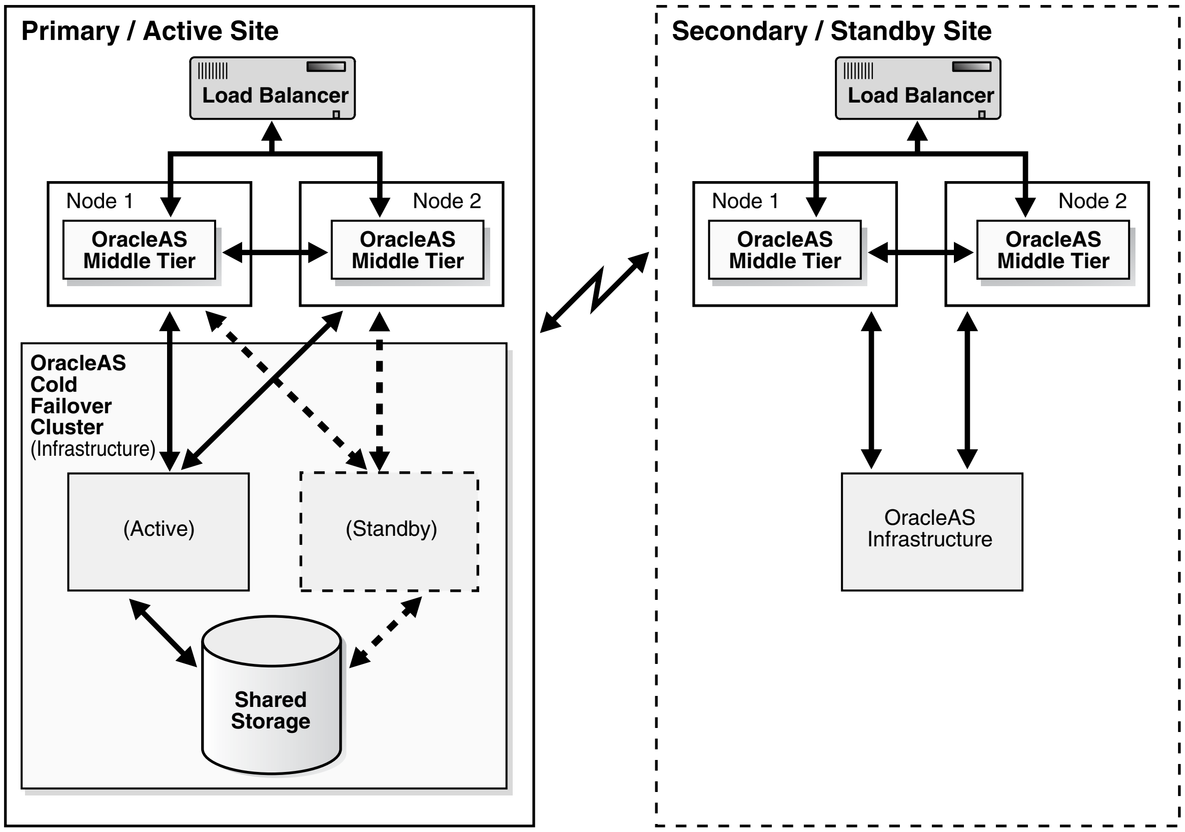
5.1.3.1 Symmetrical Topologies - Strict Mirror of the Production Site with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure

For OracleAS Disaster Recovery Release 10.1.2.0.1, only the OracleAS Disaster Recovery symmetrical topology environment was supported. This OracleAS Disaster Recovery environment has two major requirements:

- The deployment must use a single default Infrastructure install that contains a collocated OracleAS Metadata Repository and Oracle Identity Management.
- The standby site has to be a strict mirror of the production site with the same number of instances (symmetrical topology).

Figure 5–1 depicts an example OracleAS Disaster Recovery solution having a symmetrical topology with a Cold Failover Cluster on the primary site. This is considered a symmetrical topology because from an Oracle Application Server perspective both sites contain two OracleAS middle tiers and one Infrastructure.

Figure 5–1 Example Oracle Application Server 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 (names or instances must be equal) between the production and standby sites. For example, the OracleAS Cold Failover Cluster (Infrastructure) 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 5–1. 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 6.2.2, "Active-Passive High Availability Topologies" on page 6-5 of the *Oracle Application Server High Availability Guide* in the OracleAS Release 10.1.2.0.2 documentation set for more information on OracleAS Cold Failover Clusters.

The OracleAS Disaster Recovery solution is an extension to various single-site Oracle Application Server architectures. Examples of such single-site architectures include the combination of OracleAS Cold Failover Cluster (Infrastructure) and active-active Oracle Application Server middle-tier architecture. For the latest information on what single-site architectures are supported, check the Oracle Technology Network (OTN) Web site for the latest certification matrix.

`http://www.oracle.com/technology/products/ias/hi_av/index.html`

The following are important characteristics of the symmetric 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. Because they are 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 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. For a symmetric topology, 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.
- From the standpoint of a single site, 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 as long as the OracleAS Disaster Recovery site requirements are maintained. Either site should offer the same level of service to clients as the other. Note that in an active-passive setup, the standby site can be comprised of different Oracle homes that can be active on the same hosts as long as the Oracle homes being used in the Disaster Recovery environment are passive (inactive).
- For a database recovery (DBR) site as explained shortly (an OracleAS 10g release (10.1.3) site is most likely not involved, whereas an OracleAS 10g release (10.1.2.0.2) site is involved), 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 5.10, "OracleAS Guard Operations -- Standby Site Cloning of One or More Production Instances to a Standby System", Section 5.11, "OracleAS Guard Operations -- Standby Instantiation and Standby Synchronization", Section 5.12, "Runtime Operations -- OracleAS Guard Switchover and Failover Operations", and Section 5.15, "Using OracleAS Guard Command-Line Utility (asgctl)" for information about using the asgctl command-line interface to perform OracleAS Guard administrative tasks of cloning, instantiation, synchronization, switchover, and failover in the OracleAS Disaster Recovery solution.

5.1.3.2 Asymmetrical Topologies - Simple Asymmetric Standby Topology with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure

Beginning with OracleAS Disaster Recovery Release 10.1.2.0.2, support for asymmetric topologies includes support for the following simple asymmetric standby topologies:

- A standby site having reduced resources (fewer middle tiers); this means support for all production services except the scaling. This approach guarantees all services are maintained, but not scaled (see Figure 5–2 for an example of this OracleAS Disaster Recovery solution).

Figure 5–2 Simple Asymmetric Standby with Reduced Resources

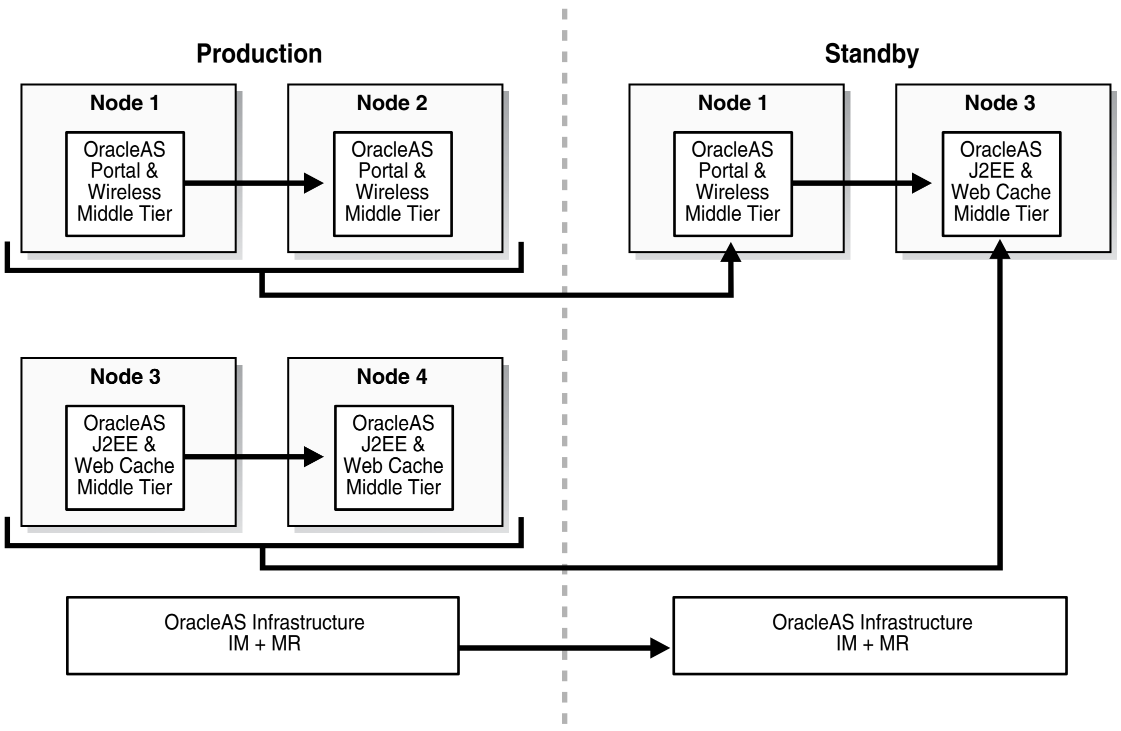

Production
Standby
Node 1
OracleAS Portal & Wireless Middle Tier
Node 2
OracleAS Portal & Wireless Middle Tier
Node 1
OracleAS Portal & Wireless Middle Tier
Node 3
OracleAS J2EE & Web Cache Middle Tier
Node 3
OracleAS J2EE & Web Cache Middle Tier
Node 4
OracleAS J2EE & Web Cache Middle Tier
OracleAS Infrastructure IM + MR
OracleAS Infrastructure IM + MR

Figure 5–2 shows a production site of four middle tier instances and one Infrastructure (collocated Oracle Identity Management and OracleAS Metadata Repository). In this example, the services and applications deployed to middle tier 1 are scaled to include middle tier 2. In addition, the services and applications deployed to middle tier 3 are scaled to include middle tier 4. To satisfy the requirements for reduced resources for disaster recovery, the scaling is not necessary at the standby site. Therefore, the services deployed at production middle tiers 1 and 2 are satisfied by a disaster recovery peer middle tier 1 at the standby site, which will be synchronized with the production middle tier 1. Likewise, the services deployed at production middle tiers 3 and 4 are satisfied by a disaster recovery peer middle tier 3 at the standby site, which will be synchronized with the production middle tier 3.

- A standby site that maintains OracleAS Disaster Recovery support for the Infrastructure services only, while the middle-tier instances are supported only through production site management. This approach guarantees that only the Infrastructure services are maintained (see Figure 5–3 for an example of this OracleAS Disaster Recovery solution).

Figure 5–3 Simple Asymmetric Standby with Guaranteed Infrastructure

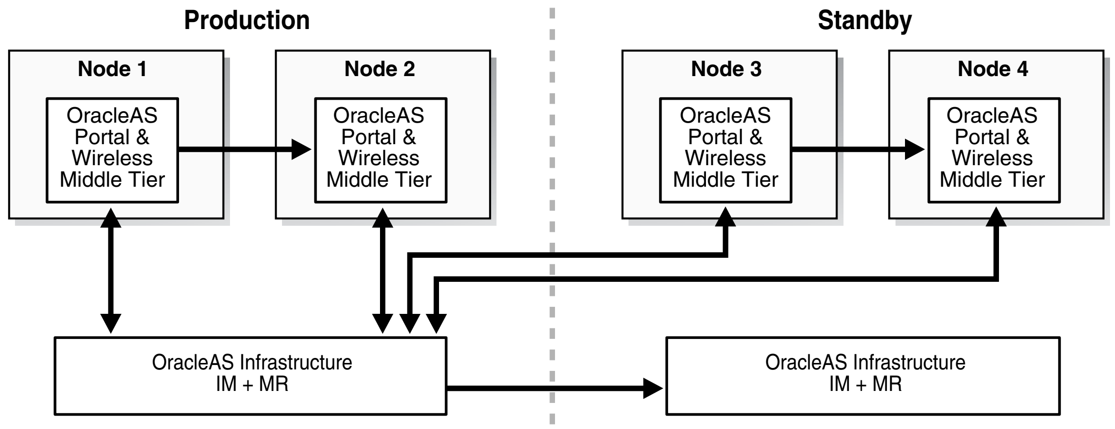

Production
Standby
Node 1
Node 2
Node 3
Node 4
OracleAS Portal & Wireless Middle Tier
OracleAS Portal & Wireless Middle Tier
OracleAS Portal & Wireless Middle Tier
OracleAS Portal & Wireless Middle Tier
OracleAS Infrastructure IM + MR
OracleAS Infrastructure IM + MR

Figure 5–3 shows a production site consisting of four middle tiers instances, with two middle tiers (1 and 2) collocated with the production Infrastructure services and two middle tiers (3 and 4) remotely located at the standby site. The standby site is used to provide disaster recovery capability for only the Infrastructure services. In this configuration, middle tier resources are configured in an active/active model and technically as a single production topology.

Under normal conditions, application requests can be serviced from middle tiers 1 through 4. This model assumes that the services and applications deployed to middle tiers 3 and 4 can tolerate the latency, firewall, and network issues associated with this topology. For disaster recovery operations, only the Infrastructure services must be maintained, while the deployment and maintenance of the middle tier instances is done through routine production site management.

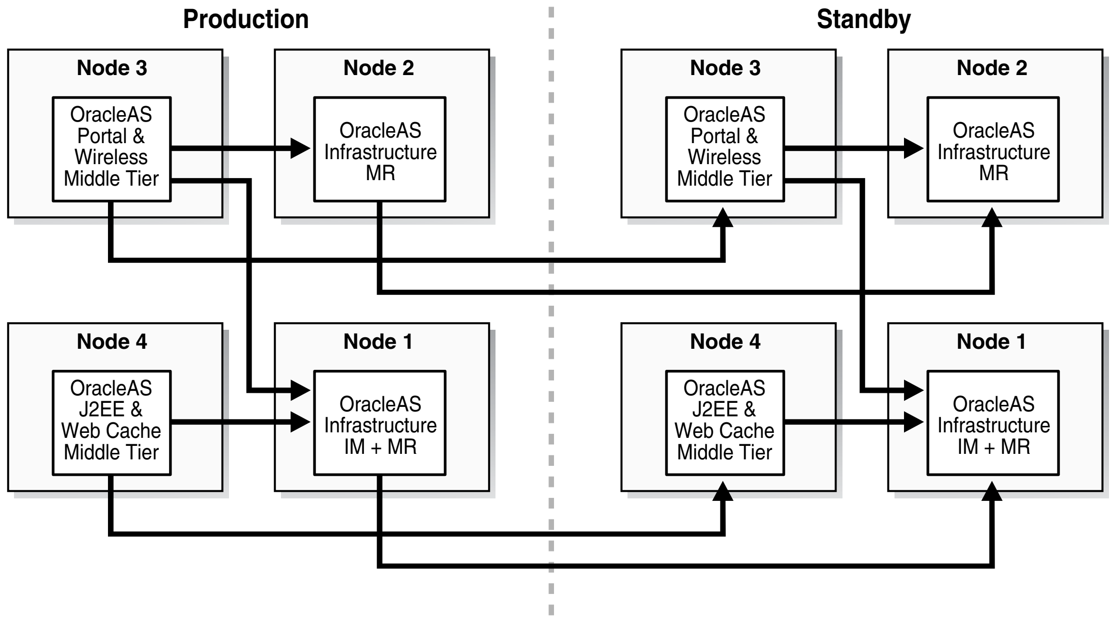
In general, support for asymmetrical topologies means that the OracleAS Disaster Recovery standby site has or potentially has reduced resources, maintains a reduction of Oracle homes, and also guarantees a certain minimum level of service capability.

5.1.3.3 Separate OracleAS Metadata Repository for OracleAS Portal with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure (the Departmental Topology)

This topology (Figure 5–4), consists of an OracleAS Infrastructure with two OracleAS Metadata Repositories and multiple middle tiers. One OracleAS Metadata Repository is used by Oracle Identity Management components, such as Oracle Internet Directory and OracleAS Single Sign-On. All middle tiers use this OracleAS Metadata Repository for Oracle Identity Management services, as well as any additional middle tiers that might be added to this topology as it expands. The other OracleAS Metadata Repository is used for product metadata by the OracleAS Portal and OracleAS Wireless middle tier components. With two metadata repositories, this deployment can best be described as two DCM production farms.

An OracleAS Disaster Recovery standby configuration could be set up as either a symmetrical topology as described in Section 5.1.3.1, "Symmetrical Topologies - Strict Mirror of the Production Site with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure", thereby requiring two DCM standby farms be configured or as a simple asymmetric topology as described in Section 5.1.3.2, "Asymmetrical Topologies - Simple Asymmetric Standby Topology with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure", with service guaranteed requiring minimally that a single DCM standby farm be configured.

Figure 5–4 Collocated Oracle Identity Management and OracleAS Metadata Repository with a Separate OracleAS Metadata Repository


Production
Standby
Node 3
OracleAS Portal & Wireless Middle Tier
Node 2
OracleAS Infrastructure MR
Node 4
OracleAS J2EE & Web Cache Middle Tier
Node 1
OracleAS Infrastructure IM + MR
Node 3
OracleAS Portal & Wireless Middle Tier
Node 2
OracleAS Infrastructure MR
Node 4
OracleAS J2EE & Web Cache Middle Tier
Node 1
OracleAS Infrastructure IM + MR

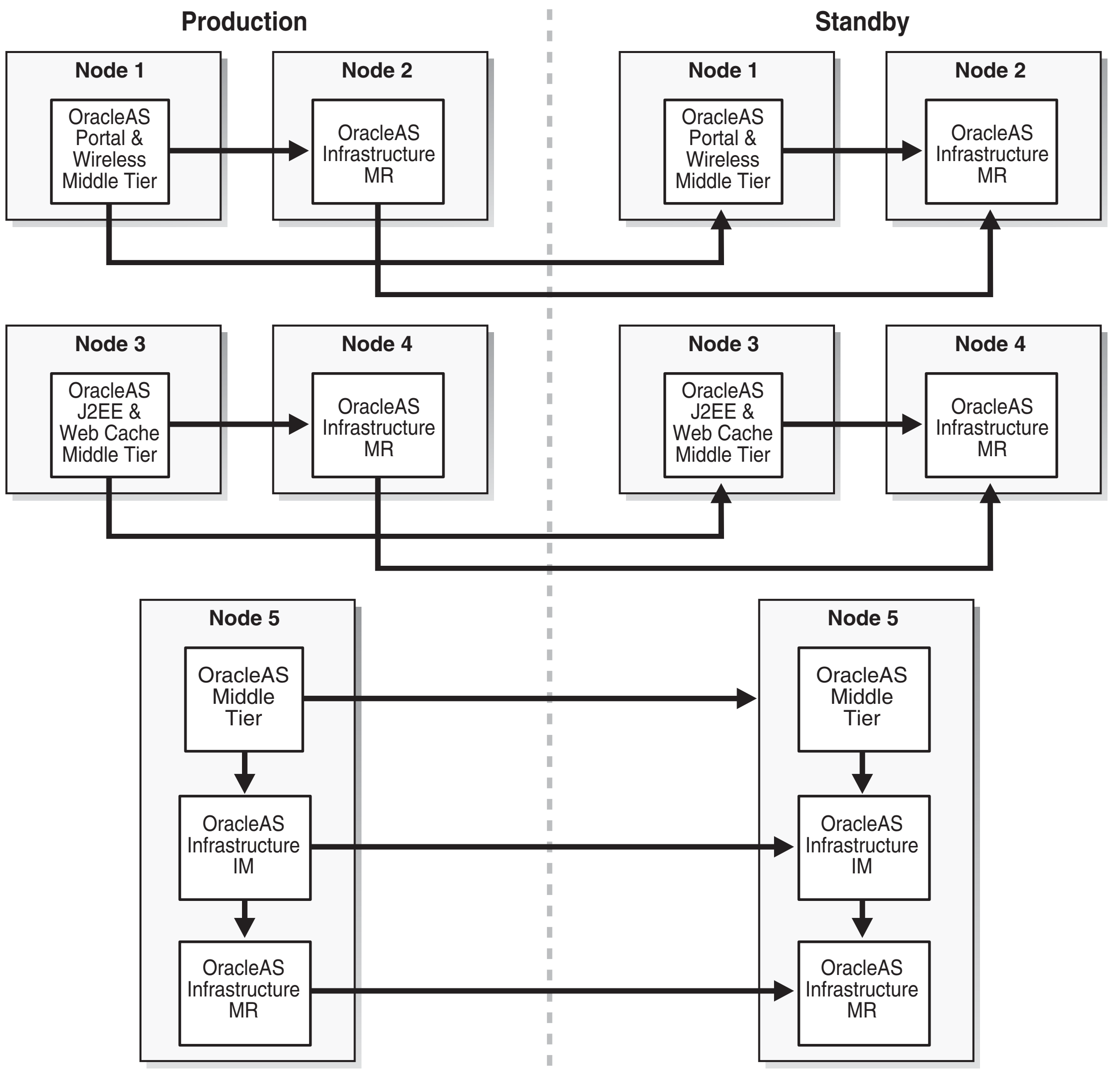
5.1.3.4 Distributed Application OracleAS Metadata Repositories with Non Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure

The topologies in Section 5.1.3.1, "Symmetrical Topologies - Strict Mirror of the Production Site with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure", Section 5.1.3.2, "Asymmetrical Topologies - Simple Asymmetric Standby Topology with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure", and Section 5.1.3.3, "Separate OracleAS Metadata Repository for OracleAS Portal with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure (the Departmental Topology)" describe a deployment for a default database repository collocated for both the Oracle Identity Management and OracleAS Metadata Repository Infrastructure, while Section 5.1.3.3, "Separate OracleAS Metadata Repository for OracleAS Portal with Collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure (the Departmental Topology)" also describes a topology with a separate OracleAS Metadata Repository.

In a topology with distributed application OracleAS Metadata Repositories and non collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure, the Oracle Identity Management Infrastructure and one OracleAS Metadata Repository Infrastructure are installed on separate hosts, and other OracleAS Metadata Repositories are installed to reside with respective applications on different hosts. Thus, one OracleAS Metadata Repository can be the result of a deployment using a single default Infrastructure install, while one or more OracleAS Metadata Repositories can be the result of an OracleAS user using a tool, such as the OracleAS Metadata Repository Creation Assistant, to install one or more application OracleAS Metadata Repositories on one or more systems with the application data, for management or policy reasons, or both.

Figure 5–5 shows an example OracleAS Disaster Recovery solution having non collocated Oracle Identity Management and OracleAS Metadata Repository Infrastructure and distributed OracleAS Metadata Repositories.

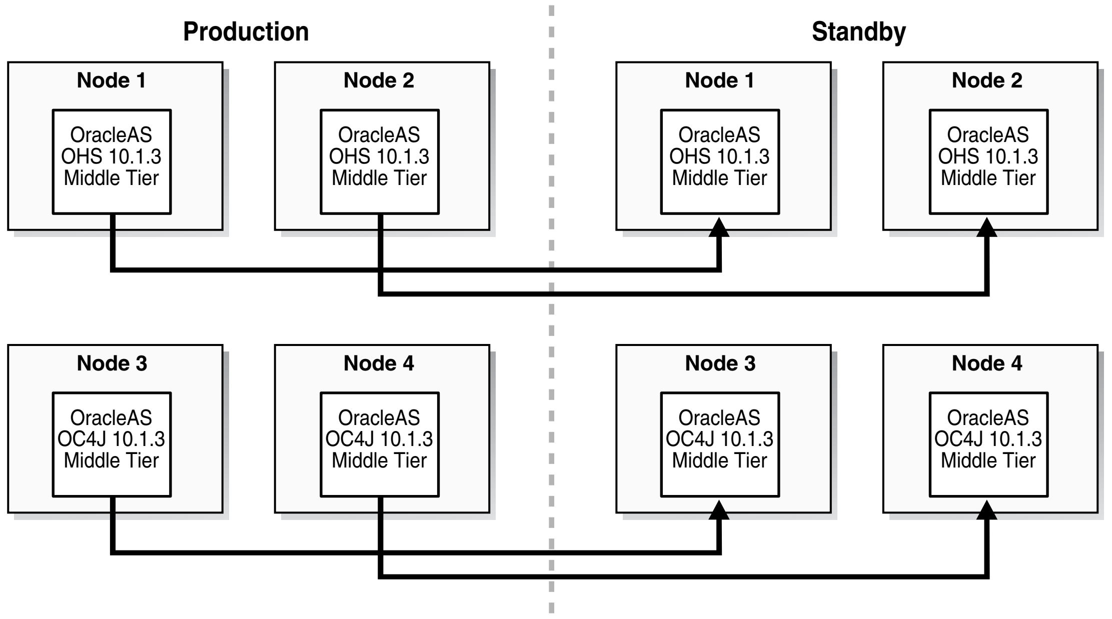
Figure 5–5 Non-Collocated Oracle Identity Management (IM) and OracleAS Metadata Repository (MR) Infrastructure Topology with Distributed OracleAS Metadata Repositories


Production
Standby
Node 1
OracleAS Portal & Wireless Middle Tier
Node 2
OracleAS Infrastructure MR
Node 1
OracleAS Portal & Wireless Middle Tier
Node 2
OracleAS Infrastructure MR
Node 3
OracleAS J2EE & Web Cache Middle Tier
Node 4
OracleAS Infrastructure MR
Node 3
OracleAS J2EE & Web Cache Middle Tier
Node 4
OracleAS Infrastructure MR
Node 5
OracleAS Middle Tier
OracleAS Infrastructure IM
OracleAS Infrastructure MR
Node 5
OracleAS Middle Tier
OracleAS Infrastructure IM
OracleAS Infrastructure MR

5.1.3.5 Redundant Multiple OracleAS 10.1.3 Homes J2EE Topology

Figure 5–6 shows an example OracleAS Disaster Recovery solution having a configuration supporting high availability of J2EE servers in an OracleAS 10.1.3 topology consisting of four nodes, known as a cluster (also called an OPMN instance). The install type of Web Server (Oracle HTTP Server or OHS) and Process Management (OPMN) is installed on nodes 1 and 2 and the install type of J2EE Server (OC4J) and Process Management (OPMN) is installed on nodes 3 and 4. There is no Identity Management.

Figure 5–6 Redundant Multiple OracleAS 10.1.3 Homes J2EE Topology


Production
Standby
Node 1
OracleAS OHS 10.1.3 Middle Tier
Node 2
OracleAS OHS 10.1.3 Middle Tier
Node 1
OracleAS OHS 10.1.3 Middle Tier
Node 2
OracleAS OHS 10.1.3 Middle Tier
Node 3
OracleAS OC4J 10.1.3 Middle Tier
Node 4
OracleAS OC4J 10.1.3 Middle Tier
Node 3
OracleAS OC4J 10.1.3 Middle Tier
Node 4
OracleAS OC4J 10.1.3 Middle Tier

Beginning with OracleAS 10.1.3, a new dynamic node discovery mechanism is operational within Oracle Notification Server (ONS), a component of OPMN. Dynamic node discovery enables the cluster to manage itself. When a new ONS node is added to the cluster, each existing ONS node announces its presence with a multicast message and each existing node then adds the new node and its connection information to its map of the current cluster, while at the same time the new ONS node adds all the existing nodes to its map. This fulfills one of the requirements for OracleAS Disaster Recovery that the OracleAS cluster be currently configured. In this case, the OPMN configuration file, `opmn.xml`, is updated whenever a new OracleAS server node is added to or removed from the cluster. This clustering configuration applies to all instances of OracleAS Server components, including OHS and OC4J, installed on the node.

By default, OracleAS Guard is installed in each Oracle Home on each node in the OracleAS cluster. When an OracleAS Guard client connects to an OracleAS Guard server, and the Disaster Recovery Administrator performs a discover topology within farm command, OracleAS Guard utilizes OPMN to discover all the instances on nodes within the cluster, creates the Disaster Recovery XML topology file on the OracleAS Guard server, and then propagates this file to all systems across the Disaster Recovery production topology. Any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, and switchover will automatically propagate the production topology file across the standby topology.

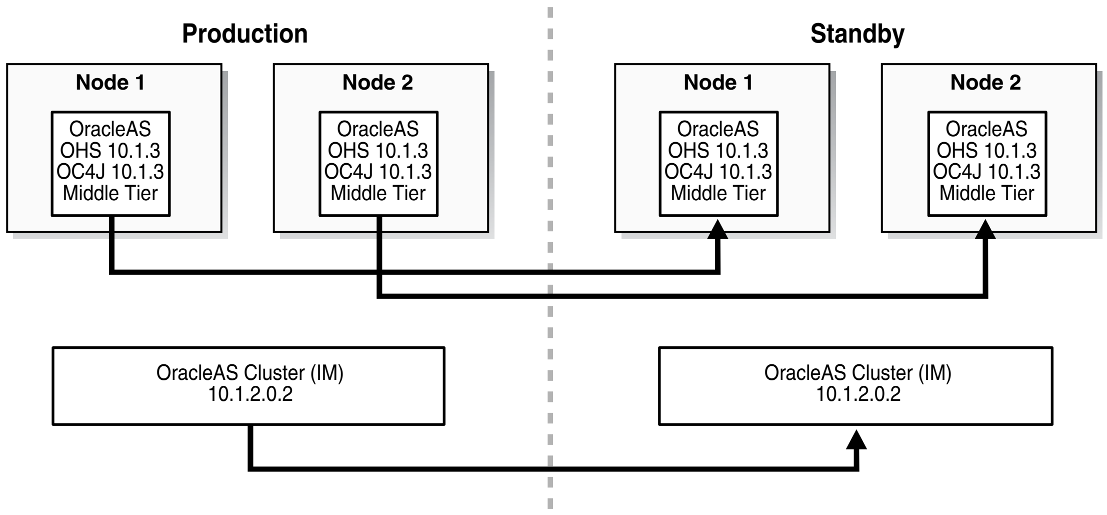
Thereafter, the Disaster Recovery Administrator can use the discover topology within farm command following the addition or removal of one or more nodes to the OracleAS cluster knowing that the ONS dynamic node discovery mechanism will automatically manage the cluster configuration information and keep it current. See *Oracle Containers for J2EE Configuration and Administration Guide* for more information about the ONS dynamic discovery mechanism. You can also manage instances in the local topology file using the add instance and remove instance commands, and if specified propagate this updated local topology file to all instances in the Disaster Recovery production environment. Any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, and switchover will automatically propagate the production topology file across the standby environment. See

Section 5.8, "Discovering OracleAS 10.1.3 Instances in Redundant Multiple OracleAS 10.1.3 Homes J2EE Topology" for a use case example. Thus, the key point is that you must perform a discover topology within farm command if any nodes have been added to the cluster and if any instances have been added to these nodes that are part of your Disaster Recovery environment.

5.1.3.6 Redundant Single OracleAS 10.1.3 Oracle Home J2EE Topology Integrated with an Existing Oracle Identity Management 10.1.2.0.2 Topology

Figure 5–7 shows an example OracleAS Disaster Recovery solution supporting a mixed version topology, where OracleAS 10.1.3 instances are integrated into an existing OID based topology (OracleAS Cluster (IM) 10.1.2.0.2). In this case, multiple OracleAS 10g (10.1.3) installations of the Integrated Web server, J2EE Server, and OPMN installation option in a single Oracle Home on individual systems create the redundant single OracleAS 10.1.3 Oracle Home J2EE topology.

Figure 5–7 Redundant Single Oracle Homes J2EE Topology Plus OracleAS Cluster (IM) 10.1.2.0.2 (OracleAS Mixed Version Topology)


Production
Standby
Node 1
Node 2
OracleAS OHS 10.1.3 OC4J 10.1.3 Middle Tier
OracleAS Cluster (IM) 10.1.2.0.2

By default, OracleAS Guard is installed in each Oracle Home. When an OracleAS Guard client connects to an OracleAS Guard server in the IM 10.1.2.0.2 OracleAS topology, and performs a discover topology command, OracleAS Guard utilizes OID and automatically recognizes all OracleAS 10.1.2.0.2 instances within the existing OID based topology. The discover topology operation creates the Disaster Recovery topology file and propagates it to all instances across the production topology. Any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, and switchover will automatically propagate the production topology file across the standby topology.

A Disaster Recovery Administrator can use the asgctl add instance or remove instance command to add or remove from the local topology file single OracleAS 10.1.3 J2EE instances to or from an existing OID based 10.1.2.0.2 production topology. With either operation, the local topology file is updated and if specified, the local updated topology file is propagated to all instances across the production topology. Any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, and switchover will automatically propagate the production topology file across the standby topology. The resulting topology is then described as a redundant single OracleAS 10.1.3 Oracle Home J2EE topology integrated with an existing OID based topology (OracleAS Cluster (IM) 10.1.2.0.2). See Section 5.9, "Adding or Removing

OracleAS 10.1.3 Instances to Redundant Single OracleAS 10.1.3 Home J2EE Topology Integrated with an Existing Oracle Identity Management 10.1.2.0.2 Topology" for a use case example. See Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information about topology files.

5.2 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 5.2.1, "Planning and Assigning Hostnames"
- Section 5.2.2, "Configuring Hostname Resolution"
- Chapter 10, "Secure Shell (SSH) Port Forwarding" (optional)

This section covers the steps needed to perform these tasks for the symmetrical topology. These same steps are also applicable to simple asymmetrical standby sites as well as to topologies for non collocated Oracle Identity Management and OracleAS Metadata Repository with or without distributed OracleAS Metadata Repositories.

5.2.1 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 must 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.

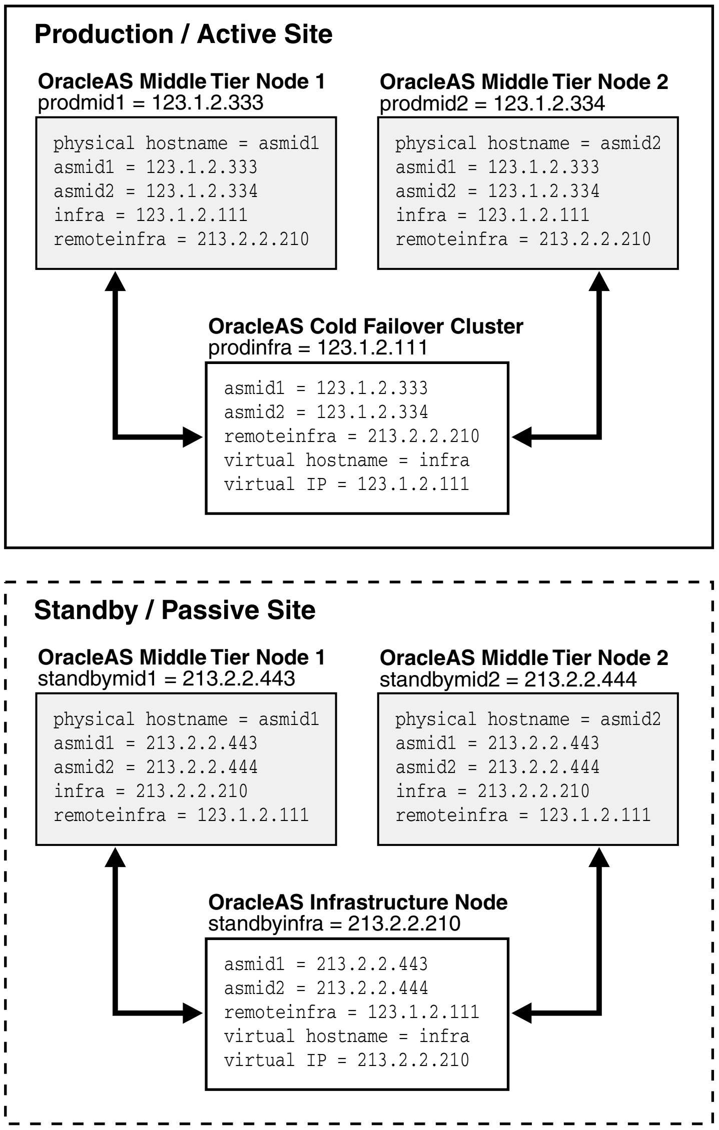
  For example, if a middle-tier component in the production site uses the name "`asmid1`" to reach a host in the same site, the same component in the standby site must use the same name to reach `asmid1`'s equivalent peer in the standby site. Likewise, if the virtual hostname of the OracleAS Infrastructure 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. However, a DNS switchover must be performed, see Section 5.14, "Wide Area DNS Operations" for more information.

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 5.2.2, "Configuring Hostname Resolution" explains hostname resolution.

Figure 5–8 illustrates the process of planning and assigning hostnames.

Figure 5–8 Name Assignment Example in the Production and Standby Sites



In Figure 5–8, 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 5–2 details what the physical, network, and virtual hostnames are in the example:

Table 5–2 Physical, network, and virtual hostnames in Figure 5–8

Physical Hostnames	Virtual Hostname	Network Hostnames
`asmid1`	-	`prodmid1, standbymid1`
`asmid2`	-	`prodmid2, standbymid2`
- [1]	`infra`	`prodinfra, standbyinfra`

[1] 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_topology_host> parameter arguments.

- *Cohosting non OracleAS applications*

  If the hosts in the production site are running non OracleAS applications, and you wish to cohost 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 match 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 different purposes in the OracleAS Disaster Recovery solution. They are also set up differently. The following sections provide information about how the three types of hostnames are set up.

5.2.1.1 Physical Hostnames

The naming of middle-tier hosts in both the production and standby sites requires changing 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 the setting for the existing hostname as follows:

   ```
   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 previous steps for each host in the production and standby sites.

In Windows, to change the physical hostname of a host, follow these steps:

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.

5.2.1.2 Network Hostnames

The network hostnames used in the OracleAS Disaster Recovery solution are defined in domain name system (DNS). These hostnames are visible in the network that the solution uses and are resolved through DNS to the appropriate hosts by 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 5–8, the following additions should be 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
```

5.2.1.3 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 Virtual Hostname** 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 5–8, when you install the production site's OracleAS Infrastructure, enter its virtual hostname, "`infra`", when you see the **Specify Virtual Hostname** 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.

For high availability deployment with multiple OracleAS instances using a load balancer on the production site, but no load balancer on the standby site, the virtual hostname is the DNS based virtual hostname for the load balancer, for example `lbr01.us.oracle.com`. For more information, see the white paper on using load balancers and OracleAS High Availability on the Oracle Technology Network (OTN).

5.2.2 Configuring Hostname Resolution

In the OracleAS Disaster Recovery solution, you can configure hostname resolution in one of two ways to resolve the hostnames you planned and assigned in Section 5.2.1, "Planning and Assigning Hostnames". These are:

- Section 5.2.2.1, "Using Local Hostnaming File Resolution"
- Section 5.2.2.2, "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 documentation to find out more about name resolution using the file `/etc/nsswitch.conf`.

5.2.2.1 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 steps:

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 the correct IP addresses for all middle-tier nodes in the current site. The first entry must be 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 topology with <host>` operation will fail indicating that the production topology is not symmetrical with the standby topology. 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. (You should also include fully qualified hostnames in addition to the abbreviated hostnames. See Table 5–3.)
   - 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 the IP address of the OracleAS Infrastructure host in the standby site.
3. Reboot each host after editing the files mentioned in the previous steps.
4. From each host, use the ping command for 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 5–8, on the `asmid1` host, use the following commands in succession:

   ```
   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 5–8, Table 5–3 shows that 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 be similar.

Table 5–3 Network and Virtual Hostname Entries in Each `/etc/hosts` File of Example in Figure 5–8

Host	Entries in /etc/hosts

`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`

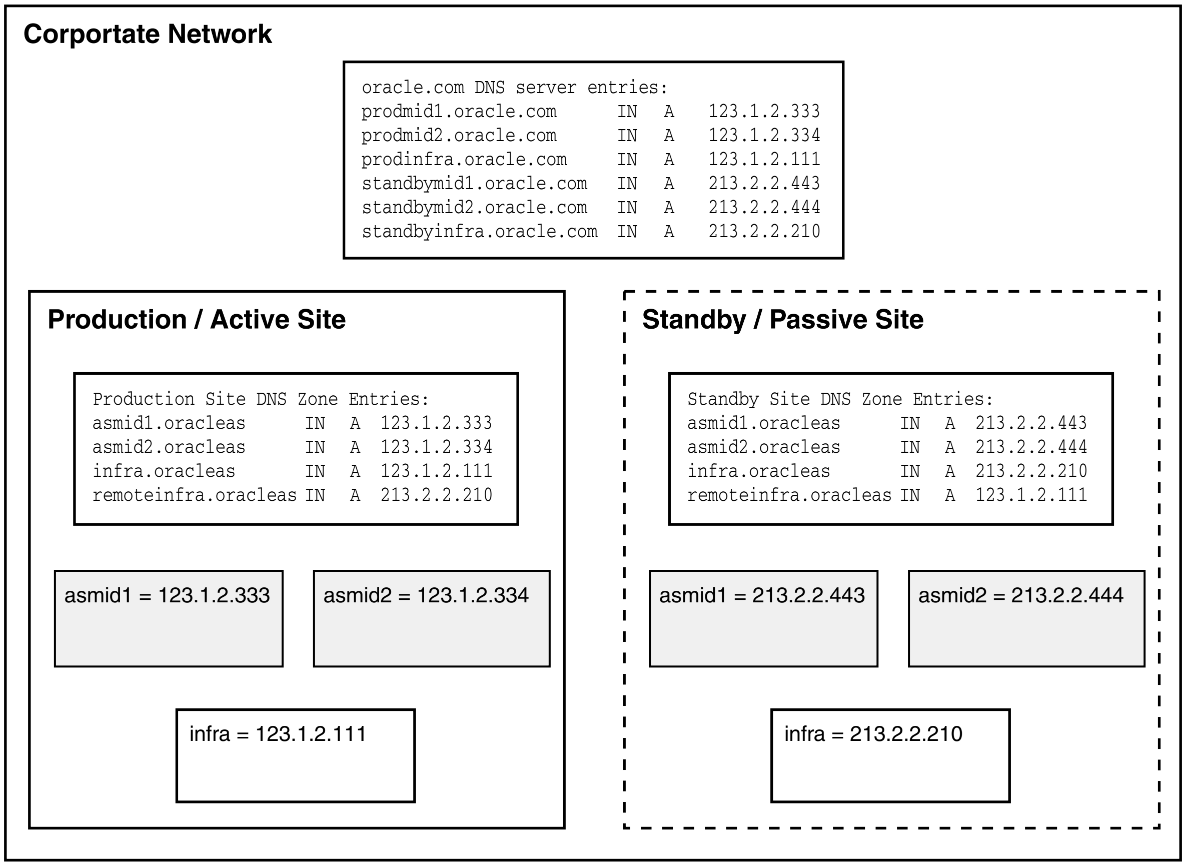
`infra` in standby site	`213.2.2.210 infra.oracle.com infra`
	`213.2.2.443 asmid1.oracle.com asmid1`
	`213.2.2.444 asmid2.oracle.com asmid2`
	`123.1.2.111 remoteinfra.oracle.com remoteinfra`

5.2.2.2 Using DNS Resolution

To set up the OracleAS Disaster Recovery solution to use DNS hostname resolution, you must set up site-specific DNS servers 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 5–9 provides an overview of this setup.

> **See Also:** Chapter 9, "Setting Up a DNS Server" for instructions on how to set up a DNS server in a UNIX environment.

Figure 5–9 DNS Resolution Topology Overview



```
oracle.com DNS server entries:
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
```

```
Production Site DNS Zone Entries:
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
```

```
Standby Site DNS Zone 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
```

For the topology in Figure 5–9 to work, the following requirements and assumptions must be made:

- The DNS servers for the production and standby sites must not be 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 must contain entries for middle-tier physical hostnames and OracleAS Infrastructure virtual hostnames. Each DNS server contains entries of only the hostnames within its own site. 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 entries for the physical, network, or virtual hostnames of any host in either the production or standby 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, follow these steps:

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 presented in Figure 5–8, 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, use the name "`oracleas`" for the domain name for the two sites in Figure 5–8. 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 5–8, the entries are as follows:

      For the DNS server on the production site:

```
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 DNS server on the standby site:

```
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 a load balancer, you must alias the IP addresses inside the host file with DNS based virtual hostnames. This is essential for OracleAS Guard on the local host to perform a local write of the topology file from a discover topology within farm command and to correctly perform an add instance command and update the topology file.

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, `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 6.2.2, "Active-Passive High Availability Topologies" on page 6-5 of the *Oracle Application Server High Availability Guide* in the OracleAS Release 10.1.2.0.2 documentation set.

5.2.2.2.1 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 conversely.

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 hostname that is unique 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. These DNS entries are required because Oracle Data Guard uses TNS Names to direct requests to the production and standby OracleAS Infrastructures. Hence, the appropriate entries must also be made to the `tnsnames.ora` file. Additionally, OracleAS Guard asgctl command-line commands must reference the network hostnames.

Using the example in Figure 5–8 and assuming that the selected name for the remote OracleAS Infrastructure is "`remoteinfra`," the entries for 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, in the standby site, the DNS server entries should be as follows:

```
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
```

5.3 Overview of Installing Oracle Application Server

This section provides an overview of the steps for installing the OracleAS Disaster Recovery solution. These steps are applicable to the topologies described in Section 5.1.3, "Supported Topologies". After following the instructions in Section 5.2, "Preparing the OracleAS Disaster Recovery Environment" to set up the environment for the solution, read this section for an overview of the installation process. Then, follow the detailed instructions in the *Oracle Application Server Installation Guide* to install the solution.

Note: To assign identical ports for use by symmetrical hosts in the production and standby sites, you can use static port definitions. These definitions are defined in a file, (for example, named `staticports.ini`). Then, specify the `staticports.ini` file in the **Specify Ports Configuration Options** screen in the installer. (Detailed information on the static ports file is found in the *Oracle Application Server Installation Guide*.)

The following steps represent the overall sequence for installing the OracleAS Disaster Recovery solution:

1. Install OracleAS Infrastructure in the production site (see *Oracle Application Server Installation Guide*).

2. Install OracleAS Infrastructure in the standby site (see *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 (see *Oracle Application Server Installation Guide*).
5. Install the middle tiers in the standby site (see *Oracle Application Server Installation Guide*).

The following points are important when you perform the installation:

- 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. Use a static ports definition file. (see the previous note in this section and the following point).
- Specify the full path to the `staticports.ini` file in the installer's **Specify Ports Configuration Options** screen.
- Ensure that you select the High Availability and Replication option in the installer's **Select Configuration Options** screen.
- Specify the virtual address assigned to the OracleAS Infrastructure in the **Specify Virtual Hostname** screen during OracleAS Infrastructure installation.
- Install for the middle-tier hosts, any of the available middle-tier installation types. (Ensure that the OracleAS Infrastructure services have been started for a site before installing any middle tiers in that site.)
- Specify the OracleAS Infrastructure's virtual hostname as the OracleAS Infrastructure database during each middle-tier installation.
- Start the OracleAS services on the hosts in each site starting with the OracleAS Infrastructure.

5.4 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, go to Section 5.5, "Authentication of Databases". This section contains the following subsections:

- Section 5.4.1, "Overview of asgctl"
- Section 5.4.2, "OracleAS Guard Client"
- Section 5.4.3, "OracleAS Guard Server"
- Section 5.4.4, "asgctl Operations"
- Section 5.4.5, "OracleAS Guard Integration with OPMN"
- Section 5.4.6, "Supported OracleAS Disaster Recovery Configurations"
- Section 5.4.7, "Configuring OracleAS Guard and Other Relevant Information"

5.4.1 Overview of asgctl

The asgctl command-line utility greatly simplifies the complexity and magnitude of the steps involved in setting up and managing OracleAS Disaster Recovery. This utility provides a distributed solution that consists of a client component and a server component. The client component (OracleAS Guard client) can be installed on a system on the topology. The server component (OracleAS Guard server) is installed by default on the systems hosting the primary and standby Oracle homes that comprise the OracleAS Disaster Recovery environment.

5.4.2 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.

The OracleAS Guard client provides an asgctl command-line interface (CLI) (see Chapter 6, "OracleAS Guard asgctl Command-line Reference") consisting of a set of commands to perform administrative tasks described in Section 5.4.4, "asgctl Operations".

5.4.3 OracleAS Guard Server

The OracleAS Guard server is a distributed server (installed by default) that runs on all the systems in an OracleAS Disaster Recovery configuration. The OracleAS Guard client maintains an active connection to the OracleAS Guard server on one system that has network connectivity in the OracleAS Disaster Recovery configuration. This coordinating server communicates to the OracleAS Guard servers on the other systems in the OracleAS Disaster Recovery configuration as necessary to complete processing during standby site cloning, instantiation, synchronization, verification, switchover, and failover 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 session. The steps to complete an operation will execute throughout all systems in both the production and standby topologies. Most operational steps will be executed either in parallel or sequentially (as required) on these systems throughout the OracleAS Disaster Recovery configuration by the OracleAS Guard server.

5.4.4 asgctl Operations

Major asgctl operations using the asgctl commands belong in the following categories of operations:

- Authentication -- Identifies the OracleAS Infrastructure database on the primary topology (set primary database command). If there are topologies with multiple Infrastructures, each must be identified using this command prior to performing an operation involving both production and standby topologies.

  Identifies the new OracleAS Infrastructure database on the standby topology (set new primary database command) prior to a failover operation.

  Sets the credentials (set asg credentials command) used to authenticate the OracleAS Guard client connections to OracleAS Guard servers and the connections between OracleAS Guard servers to a specific host. See the set asg credentials command for an example, and see Section 5.16.1.1, "Setting asgctl Credentials" for more information.

  When OracleAS Guard discovers the topology (discover topology command), it requires you provide Oracle Internet Directory authentication credentials (Oracle Internet Directory password) in order to query Oracle Internet Directory to obtain instance information for the production site.

- Discover the topology -- Discovers (discover topology command) by querying Oracle Internet Directory all instances within the topology that share the same

Oracle Internet Directory for a production site and generates a topology XML file that describes the topology and replicates this file to all instances in the topology. See Section 5.6, "Discovering, Dumping, and Verifying the Topology" for more information.

The command discover topology within farm discovers the topology using OPMN at a production site for special cases where Oracle Internet Directory is not available, such as in an OracleAS 10.1.3 OPMN topology.

- Adding or removing an instance from the local topology file -- pulls an instance into or drops an instance from an existing local topology file. This is particularly useful in an OracleAS 10.1.3 only topology, where an OracleAS Guard client can connect to an existing OracleAS 10.1.3 instance, and either perform an add instance command that adds the specified OracleAS 10.1.3 instance to the topology file, or performs a remove instance command that removes the specified OracleAS 10.1.3 instance from the local topology file, and in either case, if specified, propagates the updated topology file to all instances in the Disaster Recovery production environment. Any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, and switchover will automatically propagate the production topology file across the standby environment.

  This command is also useful for adding an OracleAS 10.1.3 J2EE instance to an OID based 10.1.2.0.2 local topology file to support a mixed version Disaster Recovery environment. For example, you can use the add instance command to add an OracleAS 10.1.3 J2EE instance to your OID based 10.1.2.0.2 local topology file. See Section 5.9, "Adding or Removing OracleAS 10.1.3 Instances to Redundant Single OracleAS 10.1.3 Home J2EE Topology Integrated with an Existing Oracle Identity Management 10.1.2.0.2 Topology" for a use case.

- Standby site cloning -- Clones a single production instance to a standby system (clone instance command) or clones two or more production instances to standby systems (clone topology command). See Section 5.10, "OracleAS Guard Operations -- Standby Site Cloning of One or More Production Instances to a Standby System" for more information. The standby site cloning operation eliminates the task of having to install these Oracle instances on the standby middle tier systems and perform an instantiate operation.

- Standby site instantiation -- Creates the disaster recovery environment. It establishes the relationship between standby and production instances, mirrors the configuration, creates the standby Infrastructure, then synchronizes the standby site with the primary site (instantiate topology command). See Section 5.11.1, "Standby Instantiation" for more information.

- Standby site synchronization -- Applies database redo logs for OracleAS Infrastructures to the standby site in conjunction with synchronizing external configuration files across the topology (sync topology command). See Section 5.11.2, "Standby Synchronization" for more information.

- Switchover -- Switches 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 (switchover topology command). See Section 5.12.1.1, "Scheduled Outages" for more information.

- Failover -- Makes the standby site the production site after restoring configuration files and restoring the OracleAS server environment to the point of the last successful sync operation (failover command). See Section 5.12.1.2, "Unplanned Outages" for more information.

- Verification -- Validates that the primary topology is running and the configuration is valid (verify topology command) or if a standby topology is specified, compare the primary topology to which the local host system is a member with the standby topology to validate that they are consistent with one another and conforms to the requirements for OracleAS Disaster Recovery. See Section 5.13.1, "Verifying the Topology" for more information.

- Using a policy file -- Used as a filter to filter out unnecessary instances for supporting asymmetric topologies. The dump policies command writes detailed, default policy information to respective XML formatted files for a select set of asgctl commands. You can then edit each respective XML policy file and use it in the `using policy <file>` parameter with any one of these select set of asgctl commands: dump topology, verify topology, clone topology, failover, instantiate topology, switchover topology, and sync topology to define by instance the domain of execution operations that are permitted for each of these asgctl commands. Each instance list entry in an XML policy file logically tags a production-standby peer combination with a particular attribute that defines the success requirement for its successful operation. For example, you may want to omit a node in a symmetric topology while performing one of the operations previously mentioned. Use the policy file to specify the node to be ignored. See Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information.
- Instance management -- Enables you to shut down (shutdown topology command) and start up the topology (startup topology command).
- Troubleshooting -- Uses the dump topology command to write detailed information about the topology to the screen or to a file. Lets you determine the current operations that are running (show operation command) and stop any operations that need to be halted (stop operation command).

Table 5–4 describes the OracleAS Disaster Recovery production and standby site environment before and after performing an asgctl clone, instantiate, sync, failover, and switchover operation.

Table 5–4 Description of Disaster Recovery Production and Standby Environments Before and After Performing These OracleAS Guard Operations

OracleAS Guard Operation	DR Site Environment Before Operation	DR Site Environment After Operation
clone	The production site has one or more instances that need to be installed on the standby site and instantiated. The cloning operations perform this task.	The standby site has one or more new standby instances that are a logical mirror of the production site instances.
instantiate	The standby site with its Oracle homes exists, but the OracleAS Disaster Recovery relationship across sites does not exist yet for an OracleAS Disaster Recovery operation to be performed.	A logical mirror of the production site is set up and maintained at the standby site.
sync	The standby site is not consistent with the production site. OracleAS Disaster Recovery is not possible to a consistent point in time without some manual intervention.	Database redo logs are applied to OracleAS Infrastructures in combination with synchronizing external configuration files across the topology. The sync operation is performed in the event that a failover or switchover operation is necessary, then the standby site can be restored to a consistent point in time. No manual intervention is necessary to synchronize the sites after the asgctl sync operation is performed.

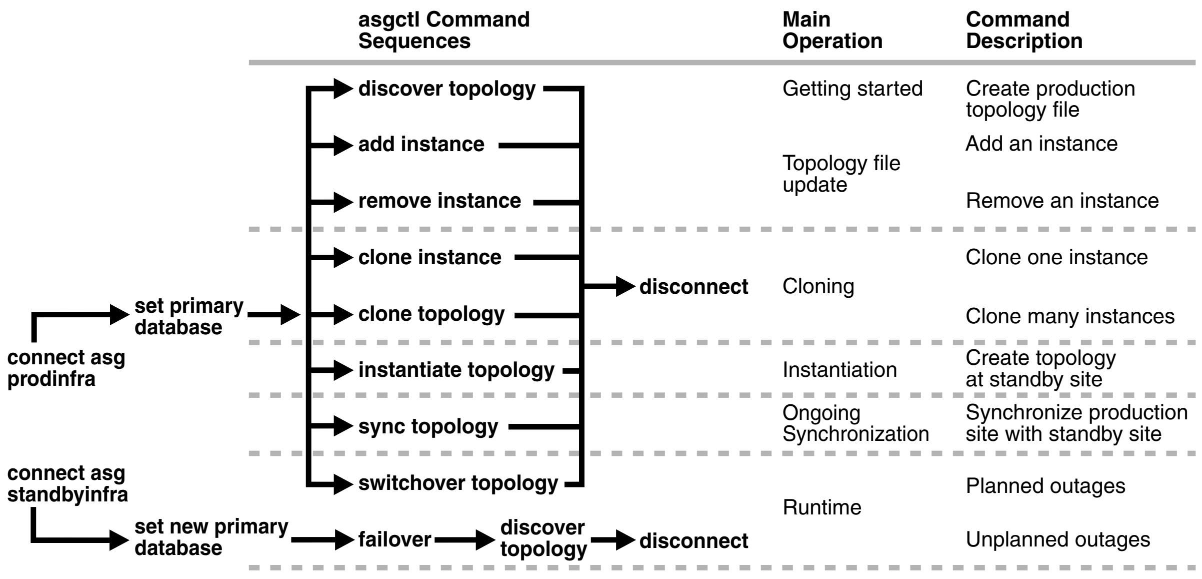
Table 5–4 (Cont.) Description of Disaster Recovery Production and Standby Environments Before and After Performing These OracleAS Guard Operations

OracleAS Guard Operation	DR Site Environment Before Operation	DR Site Environment After Operation
switchover	A planned outage at the production site will make the standby site the production site for a period of time; that is the roles of each site will be switched.	The standby site has become the production site. All OPMN services are started. The production site may become available again after the planned outage, at which time, another switchover operation could be performed to return activity back to the original production site from the standby site.
failover	An unscheduled outage at the production site has left the production site down or unavailable for an unknown period of time. The production site is lost due to some unforeseen circumstance.	The standby site has permanently become the production site. Configuration and Infrastructure data are restored to a consistent point in time on the standby site. Site services are brought up in a consistent fashion to the point of the last sync operation. All OPMN services are started.

Figure 5–10 shows a summary of the main OracleAS Disaster Recovery operations and the command sequences used to perform these operations. For example, to get started with a new DR environment once it is set up and operational, you must always create a topology file on the production site. To do that, you perform a connect command to connect the OracleAS Guard client to the OracleAS Guard server followed by identifying the production Infrastructure database using a set primary database command, then perform the discover topology command, finally followed by a disconnect command to disconnect the OracleAS client from the OracleAS server. In essence, to perform any OracleAS Disaster Recovery operation using asgctl commands, you must always connect the OracleAS Guard client to the OracleAS Guard server, identify the location of the Infrastructure database, then perform the OracleAS Disaster Recovery operation or operations of interest, and finally disconnect the OracleAS client from the OracleAS server. The only exception to this general sequence is for a failover operation, in which the production site is permanently unavailable due to some unplanned problem. In this case, you connect the OracleAS Guard client to OracleAS Guard server at the standby site, identify the standby site Infrastructure as the new Infrastructure database, then perform the failover operation. Because there is no topology file created on the standby site following a failover operation, you must then perform a discover topology command to create it for the first time. Then you can disconnect the OracleAS Guard client from OracleAS Guard server.

The asgctl command sequences shown in Figure 5–10 assume the simplest topology configuration. For example, in a more complex case, if your production or standby site has multiple MR instances installed, you must identify each instance with a set primary database command prior to performing the main Disaster Recovery operation, such as an instantiate, sync, switchover, or failover operation. Similarly, OracleAS Guard requires that you set the credentials for any OracleAS Guard server system in the topology that has different credentials from the OracleAS Guard server to which you are connected before you perform any of these main operations, such as instantiate, sync, switchover, or failover. So for both of these cases, additional asgctl commands are required in the command sequence before you can perform the main Disaster Recovery operation. See the usage notes for the set asg credentials command for more information.

Figure 5–10 Main Disaster Recovery Operations Performed Using the Following OracleAS Guard asgctl Command Sequences



5.4.5 OracleAS Guard Integration with OPMN

A typical Oracle Application Server site has multiple farms. OracleAS Guard server and its ias-component ASG process is not started by default by OPMN because it is only necessary in the context of disaster recovery sites. You must start this ias-component ASG process in all Oracle homes as described later in this section. To check the status of this component and determine if the component is running, run the following opmnctl command on each system in your topology:

```
On UNIX systems
> <ORACLE HOME>/opmn/bin/opmnctl status

On windows systems
> <ORACLE HOME>\opmn\bin\opmnctl status
```

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 topology. Connect to a node and run the following OPMN command on UNIX systems:

```
> <ORACLE HOME>/opmn/bin/opmnctl startproc ias-component=ASG
```

On Windows systems, 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=ASG
```

After the OracleAS Guard server is started it is non transient, while the remaining OracleAS Guard servers in the standby topology are transient servers. This configuration allows cross-topology 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 ASG. This is the OracleAS component name and server process name for the OracleAS Guard server.

5.4.6 Supported OracleAS Disaster Recovery Configurations

For OracleAS 10g release (10.1.2), OracleAS Guard supports not only the default OracleAS Infrastructure configuration supported on Oracle Application Server Cold Failover Clusters and single instance, but also the topologies described in Section 5.1.3, "Supported Topologies".

5.4.7 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:

- The following OracleAS Guard parameters are configurable. The value is described and the default value is indicated. The OracleAS Guard `readme.txt` file in the `<ORACLE_HOME>\dsa\doc` directory also lists these OracleAS Guard parameters that are configurable.
  - `port` - the TCP/IP port for OracleAS Guard server and client. OracleAS Guard uses a default port (port) number of 7890; for example, port=7890. If there is a second Oracle home installed on a system, this second Oracle home must have a different OracleAS Guard port number, usually incremented by one, for example, port=7891, and so on.

    Value: integer, any valid TCP/IP port number. Default is 7890.
  - `exec_timeout_secs` - timeout value for executing operating system command.

    Value: integer, number of seconds. Default is 60 seconds.
  - `trace_flags` - trace flags to be turned on.

    Value: string list, separated by ",". Default is none.
  - `backup_mode` - indicates whether to perform a full or incremental backup.

    Value: string, "full" or "incremental". Default is "incremental".
  - `backup_path` - the backup directory path to be used by OracleAS Guard server.

    Value: string, a directory path. Default is `<ORACLE_HOME>/dsa/backup`.
  - `ha_path` - the High Availability directory path where the backup scripts are located.

    Value: string, a directory path. Default is `<ORACLE_HOME>/backup_restore`.
  - `port.<host>` - the TCP/IP port for a given host.

    Value: integer, any valid TCP/IP port number.

Note: If the port number must be changed for some reason (it must be unique for each OracleAS Guard server in each Oracle home 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=ASG`) and start the OracleAS Guard server (`<ORACLE_HOME>/opmn/bin/opmnctl startproc ias-component=ASG`) to activate the change. See Section 5.4.5, "OracleAS Guard Integration with OPMN") for more information.

- `copyfile_buffersize` - the buffer size for copy file operation, in kilobytes.

  Value: integer, maximum buffer size is 500K.
- `server_inactive_timeout` - the number of seconds server will wait before shutting down due to inactivity.

  Value: integer, number of seconds. Default value is 600 seconds (10 minutes).
- `inventory_location` - the alternative Oracle Inventory location

  Value: string, a directory path.

- 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 topology production and standby topologies.
- OracleAS Guard starts up the OracleAS component services across the production topology.
- The OracleAS Guard operation status information for a topology (from either an asgctl show operation full or show operation history command) remains available for the life of the current OracleAS Guard client asgctl connect session only. When the OracleAS Guard client disconnects from the OracleAS Guard server, this topology's operation history information becomes unavailable.
- After you start an asgctl operation, you cannot run another asgctl command 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.
- 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 topology production and standby topologies.
- OracleAS Guard starts up the OracleAS component services across the production topology.
- The OracleAS Guard operation status information for a topology (from either an asgctl show operation full or show operation history command) remains available for the life of the current OracleAS Guard client asgctl connect session only. When

the OracleAS Guard client disconnects from the OracleAS Guard server, this topology's operation history information becomes unavailable.

- After you start an asgctl operation, you cannot run another asgctl command 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.

5.5 Authentication of Databases

Several levels of authentication are required when an OracleAS Guard client connects to an OracleAS guard server and begins a session to perform administrative operations within the production topology or across both production and standby topologies:

- Infrastructure authentication
- OracleAS Guard client authentication to OracleAS Guard servers
- Oracle Internet Directory authentication

Infrastructure Authentication

When initiating an OracleAS Guard administrative session, after establishing the connection between the OracleAS Guard client and OracleAS Guard server, you must identify all the OracleAS Infrastructure databases on the primary topology using the set primary database command. Infrastructure authentication must be performed before you initiate any operation that involves either the production topology or both the production and standby topologies.

Another form of Infrastructure authentication occurs as part of a failover operation. In this scenario, the production site is down and you must failover to the standby site and make this site the new production site. First, identify the new OracleAS Infrastructure database on the standby topology using the set new primary database command before performing the failover operation. See Section 5.12.1.2, "Unplanned Outages" for more information.

OracleAs Guard Client Authentication to OracleAS Guard Servers

By default, these are the same authentication credentials used for instance level authentication with the Oracle Application Server account (`ias_admin`/password) that was created during the Oracle Application Server installation and used in the connect asg command.

Note: If this is an OracleAS 10.1.3 installation, the user name must be `oc4jadmin` and the password for the `oc4jadmin` account created during the Oracle Application Server 10.1.3 installation.

These same credentials are used when the OracleAS Guard client connects to any OracleAS Guard server in the production and standby topology when executing administrative operations.

There may be cases where you want to use different credentials for a specific OracleAS Guard server or set a common set of credentials in the standby topology that differs from the credentials used in the primary topology. To set credentials for an OracleAS Guard server, use the set asg credentials command and one or more of its parameter options by either specifying the host name to which the credentials apply or the topology along with the new set of credentials (username/password).

If you set the credentials for a topology, these credentials are inherited for the entire topology. If you set the credentials for an individual host on the topology, the credentials for this host override the default credentials set for the topology. After you set the credentials so that they are different from the default connection credentials for a host system or an entire topology, whenever you initiate an OracleAS Guard administrative session, you must specify all credentials that are different from the default connection credentials for any host system or topology before you perform an operation involving all the OracleAS Guard servers within a production topology or across both production and standby topologies. Otherwise, the operation will fail with an authentication error. See the connect asg command for an example.

Oracle Internet Directory Authentication

The discover topology command requires you provide Oracle Internet Directory authentication credentials (Oracle Internet Directory password) in order to query Oracle Internet Directory to obtain instance information for the production site. See the section that follows for more information and the discover topology command.

5.6 Discovering, Dumping, and Verifying the Topology

The discover topology command discovers by querying Oracle Internet Directory all instances within the topology that share the same Oracle Internet Directory for a production site. A topology XML file is created and distributed to all Oracle homes within the topology that describes all instances for the topology. This topology file is used by all OracleAS Guard operations.

You must perform a discover topology command when you first set up your OracleAS Disaster Recovery environment in order to initially create the topology XML file. Thereafter, you should perform a discover topology operation whenever you procure another Oracle home in a production site or change roles from a production to a standby site through a switchover or failover operation. See the discover topology command for more information.

You should perform a dump topology command to inspect the information that describes your topology. See the dump topology command for more information. See Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information about topology files.

You should perform a verify topology command to validate that the primary topology is running and that the configuration is valid. In addition, if you specify the `with host` parameter, the verify operation compares the primary topology of which the local host system is a member with the standby topology to validate that they are consistent with one another and conform to the requirements for OracleAS Disaster Recovery. See Section 5.13.1, "Verifying the Topology" and the verify topology command for more information.

With both the dump topology and verify topology commands, if you want to use a policy file, edit and use the respective dump and verify policy files (`dump_policy.xml` and `verify_policy.xml`). Specify this file in the `using policy <file>` parameter of each command to dump or verify only those instances specified accordingly. See Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information.

5.7 Dumping Policy Files and Using Policy Files With Some asgctl Commands

OracleAS Disaster Recovery provides support for a variety of application server topologies as described in Section 5.1.3, "Supported Topologies". As part of this support, a set of XML formatted policy files are maintained, local to the OracleAS Guard client that performs the dump policies command, to record by instance the domain of execution operations that are permitted for each of the following asgctl commands: dump topology, verify topology, clone topology, failover, instantiate topology, switchover topology, and sync topology.

To understand the default policies in use for any these asgctl commands, enter the following command at the asgctl prompt:

```
ASGCTL> dump policies
Generating default policy for this operation
Creating policy files on local host in directory
"/private1/OraHome2/asr1012/dsa/conf/"
ASGCTL>
```

Each instance list entry in each of the XML policy files logically tags by default a production-standby peer combination with a particular attribute that defines the success requirement for the successful operation of each command. This approach provides greater flexibility in regulating how each of these OracleAS Guard operations are to be successfully used among the supported topologies, see Section 5.1.3, "Supported Topologies" for more information.

After inspecting each of the XML formatted policy files, you can subsequently edit the respective policy file and use it with the particular asgctl command using the parameter syntax `using policy <file>` and indicate the name of the policy file to be used. In this way, you can employ a particular disaster recovery policy that defines the success requirement attribute value by instance for each of these OracleAS Guard operations mentioned earlier in this chapter.

Note: If you want to maintain a set of custom policy files, you must copy, edit, and maintain them in a location other than the default location; otherwise, your custom set of policy files will be overwritten whenever you perform a discover topology command followed subsequently by a dump policies command.

The success requirement attribute value can be one of the following: [`optional | mandatory | ignore | group <MinSucceeded=<number>>`], where:

- `Optional` -- means if there is a failure for that instance continue processing other instances.
- `Mandatory` -- means if an error occurs for this instance, the entire operation fails.
- `Ignore` -- means the instance is not part of the operation.
- `Group <MinSucceeded=<number>` -- means to combine groups of Oracle instances, and if the specified number of group members is successful, then the operation is successful; otherwise, if less than the number of group members that is specified is successful, the operation fails.

Each attribute value determines the success requirement for that peer group and will be referenced during failure cases of asgctl operations to determine whether or not to continue with the OracleAS Guard operation. For example, when the success requirement is specified as mandatory, the particular OracleAS Guard operation must be successful for the specified instance for that production-standby peer combination; otherwise, the OracleAS Guard operation ceases, execution is rolled back to its starting point of execution, and an error message is returned.

For example, the following XML policy file in use for an asymmetric topology for the failover operation specifies that this asgctl operation is mandatory for the infra instance, optional for the portal_1 and portal_2 instances, can be ignored for the portal_3 instance, and must be successful for a minimum of any two of the group of three instances, BI_1, BI_2, and BI_3.

```
<policy>
  <instanceList successRequirement="Mandatory">
     <instance>infra</instance>
  </instanceList >
  <instanceList successRequirement="Optional">
     <instance>portal_1</instance>
     <instance>portal_2</instance>
  </instanceList >
  <instanceList successRequirement="Ignore">
     <instance>portal_3</instance>
  </instanceList >
  <instanceList successRequirement="Group" minSucceed="2">
     <instance>BI_1</instance>
     <instance>BI_2</instance>
     <instance>BI_3</instance>
  </instanceList >
</policy>
```

5.8 Discovering OracleAS 10.1.3 Instances in Redundant Multiple OracleAS 10.1.3 Homes J2EE Topology

As described in Section 5.1.3.5, "Redundant Multiple OracleAS 10.1.3 Homes J2EE Topology", you can discover OracleAS 10.1.3 instance changes in a redundant multiple OracleAS 10.1.3 Homes J2EE topology. In a typical scenario, OracleAS 10.1.3 J2EE is installed on a node with an OC4J instance and you want to add this node and instance to the existing redundant multiple OracleAS 10.1.3 Homes J2EE topology. The following steps take you through a typical scenario:

1. Connect to the OracleAS Guard server.

   ```
   ASGCTL > connect asg prodinfra ias_admin/<adminpwd>
   Successfully connected to prodinfra:7890
   ASGCTL>
   ```

2. Assume a Disaster Recovery Administrator has installed OracleAS 10.1.3 J2EE on a new node with an OC4J instance. Through dynamic node discovery, the cluster manages itself and automatically updates the information in each OPMN configuration file, `opmn.xml`on each node within the cluster.

3. Next, perform a discover topology within farm command. This operation determines all instances within the OracleAS 10.1.3 cluster for this production site, generates the Disaster Recovery topology XML file that describes the production topology, and then propagates this topology XML file to all instances in the Disaster Recovery production environment. Note that any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, and switchover will automatically propagate the production topology file across the standby environment.

```
ASGCTL> discover topology within farm
```

4. Now, having recently installed another oc4j instance named oc4j3 on the existing midtier host system named prodmid2, add to the local topology file an instance named oc4j3 specifying the host system named prodmid2. To propagate this updated local topology file to all instances in the Disaster Recovery production environment for this OracleAS 10.1.3 cluster, specify the `to topology` keyword. Note that any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, and switchover will automatically propagate the production topology file across the standby topology.

   ```
   ASGCTL> add instance oc4j3 on prodmid2 to topology
   ```

5.9 Adding or Removing OracleAS 10.1.3 Instances to Redundant Single OracleAS 10.1.3 Home J2EE Topology Integrated with an Existing Oracle Identity Management 10.1.2.0.2 Topology

As described in Section 5.1.3.6, "Redundant Single OracleAS 10.1.3 Oracle Home J2EE Topology Integrated with an Existing Oracle Identity Management 10.1.2.0.2 Topology", you can add or remove OracleAS 10.1.3 single home J2EE instances to or from an existing OID 10.1.2.0.2 topology, thus creating or modifying a mixed version Disaster Recovery environment. The following steps take you through a typical scenario:

1. Connect to the OracleAS Guard server.

   ```
   ASGCTL > connect asg prodinfra ias_admin/<adminpwd>
   Successfully connected to prodinfra:7890
   ASGCTL>
   ```

2. Assume that a discover topology command was performed when you first set up the Disaster Recovery environment on the existing OID 10.1.2.0.2 topology, thus creating a topology XML file that was propagated to all instances in the Disaster Recovery environment. Also assume that no other changes have been made to the topology, so the topology XML files are current or up-to-date.

3. Assume that you have completed OracleAS 10.1.3 installations using the Integrated Web server, J2EE Server, and OPMN installation option in a single Oracle Home on four individual systems. This creates the redundant single Oracle Home J2EE topology. Assume that the OracleAS Disaster Recovery solution is configured for each new node. Now you want to add the OracleAS 10.1.3 instances named OC4J1 and OC4J2 to the existing OID 10.1.2.0.2 topology. To do this, perform the following asgctl commands:

   ```
   ASGCTL> add instance oc4j1 on prodinfra to topology
   ASGCTL> add instance oc4j2 on prodinfra to topology
   ```

   Performing each add instance command updates the local topology file on the OracleAS Guard server to which the OracleAS Guard client is connected. To propagate the updated topology file to all instances in the Disaster Recovery production environment, specify the `to topology` key word. This operation then results in a redundant single OracleAS 10.1.3 Oracle Home J2EE topology with instances oc4j1 and oc4j2 being integrated with an existing OID 10.1.2.0.2 production topology. Note that any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, switchover, and failover will automatically propagate the production topology file across the standby topology.

5.10 OracleAS Guard Operations -- Standby Site Cloning of One or More Production Instances to a Standby System

Standby site cloning is the process of cloning a single production instance to a standby system (using the clone instance command) or cloning two or more production instances to standby systems (using the clone topology command). In either case, the topology XML file for each instance is copied over with the clone operation and is consistent for each instance and can be said to be updated only locally on that standby system; however, as soon as any OracleAS Guard operation is performed that affects the standby sites, such as a verify, instantiate, sync, switchover, or failover operation, then the XML topology file on the local system in the production topology is automatically propagated to all instances on nodes throughout the standby topology.

Clone Instance

The clone instance command is used to create a new standby instance target from an existing production instance source.

One of the underlying technologies used by OracleAS Guard to perform this operation is the OracleAS Backup and Restore loss of host capability. See the section on recovering a loss of host automatically in *Oracle Application Server Administrator's Guide* for more information including a list of prerequisites. This capability assumes that the target machine is a newly procured Oracle environment because it overwrites the Oracle software registry. Additionally, some of the underlying operations require elevated privileges, `root` for the UNIX environments and `Administrator` for Windows. On Windows, the user must ensure that the client and OracleAS Guard server are started with `Administrator` privileges.

There are two phases of clone. The first phase is to create the Oracle home and register it within the system environment. The second phase is to perform the OracleAS Guard instantiate operation to link it into the OracleAS Disaster Recovery environment and logically match the Oracle home with it's corresponding production home.

A series of clone instance operations on different instances are equivalent to a clone topology operation.

Clone Topology

The clone topology command performs a clone instance operation across a group of systems. The clone operation is performed on every OracleAS home that does not contain a database or it can be filtered using a policy file. For OracleAS homes that contain a database, a clone topology operation will perform the instantiate phase of the operation, skipping the creation of the Oracle home at the standby site. The operation can be performed on a subset of a topology by utilizing a policy file.

There are three methodologies that you must be aware of when planning for an OracleAS Disaster Recovery site setup:

- Creating a pure OracleAS Disaster Recovery site
- Adding OracleAS homes to an existing site with OracleAS Disaster Recovery enabled
- Integrating OracleAS Metadata Repositories within an existing database

Each operation requires a different methodology to integrate the newly installed Oracle homes into the existing site or combine them into a standby site for a production site.

Creating a Pure OracleAS Disaster Recovery Site

Prior to OracleAS 10g release 10.1.2.0.2, this was the only type of site OracleAS Guard could support. An OracleAS Disaster Recovery configuration was supported only for the default Infrastructure and OracleAS middle-tier install types. With this type of configuration, all the OracleAS homes were created using the Oracle installer. The OracleAS Guard instantiate command creates the relationships between the production and standby Oracle homes and the underlying standby Oracle database repositories.

Adding OracleAS Homes to an Existing Site with OracleAS Disaster Recovery Enabled

After an OracleAS site is OracleAS Disaster Recovery enabled, the relationship between the production and standby Oracle homes has been created. For releases previous to OracleAS 10g release 10.1.2.0.2, the only way to add new instances to the site was to break the standby relationship, add the new instance at the production site using Oracle Installer, add the new instance to the standby site using Oracle Installer, and re-create the standby site. With OracleAS 10g release 10.1.2.0.2, you can use the clone instance command to add instances to a standby site.

For example, if you need a new middle tier to scale out the services in the middle tier the new instance is installed at the production site. This operation creates the OracleAS home for the instance and establishes the necessary relationships within the OracleAS repositories.

With OracleAS 10g release 10.1.2.0.2, OracleAS Guard asymmetrical topology support, this Oracle home can optionally be ignored in regard to the site's OracleAS Disaster Recovery solution. If you want to add this instance to the standby site, the clone topology command will create the OracleAS Oracle home at the standby target host and establish the production-standby relationship for this instance. Before issuing this command, the standalone OracleAS Guard kit must be installed and started at the target host (see the OracleAS Disaster Recovery installation information in *Oracle Application Server Installation Guide* for more information) and a site discovery topology operation should be performed to discover the new instance in the production topology.

Integrating OracleAS Metadata Repositories within an Existing Database

OracleAS supports the ability to create Metadata Repository schemas in an existing database. Although OracleAS Guard recognizes and manages these databases to synchronize the Metadata Repository configuration data with the rest of the site's distributed configuration data, OracleAS Guard does not create the standby repository nor the production to standby relationship. This environment is supported using the clone topology operation.

To utilize the clone topology command, first install and start the standalone OracleAS Guard server on each standby host. Additionally, the OracleAS Backup/Restore utility is installed in the Oracle home created by the standalone OracleAS Guard install. See the section on recovering a loss of host automatically in *Oracle Application Server Administrator's Guide* for more information including a list of prerequisites. The clone topology command creates the instance Oracle homes and configuration information at the standby site. For Infrastructure instances, an implicit instantiate operation is performed to initialize the OracleAS Disaster Recovery environment. The clone topology operation can use a profile file to filter out instances for an asymmetric topology.

Warning: Do not perform a clone operation to a standby system that contains an existing Oracle home because it will get overwritten. Perform a clone operation only to a standby system where no Oracle home is installed.

Some situations in which cloning operations are useful are:

- When you want to add one or more production instances to a standby host site.
- When you want to add a single production instance to a standby host system.

The steps to perform these cloning operations are described in the following sections.

5.10.1 Cloning Single or Multiple Production Instances to a Standby System

Whether you are cloning a single production instance or multiple production instances to a standby site, the prerequisites and steps to follow are identical; the only difference is the actual asgctl command you would use for either cloning operation. For this reason, this section combines the cloning information and indicates where there are some minor differences.

Cloning a Single Production Instance to a Standby Site

As an example, you want to add a production instance to a standby system. The clone instance operation eliminates the task of having to install the Oracle instance on the standby middle-tier system and then perform an instantiate operation.

Cloning Multiple Production Instances to a Standby Site

As an example, you want to add two or more production instances to a standby middle-tier host system. The clone topology operation eliminates the task of having to install these Oracle instances on the standby middle-tier systems and then perform an instantiate operation.

As part of the clone topology operation, the production instances are cloned and the OracleAS Metadata Repository is instantiated. However, for a OracleAS Metadata Repository configuration created using OracleAS Metadata Repository Creation Assistant, no instantiate operation is performed.

If you want to use a policy file, edit and use the clone policy file (`clone_policy.xml`). Specify this file in the `using policy <file>` parameter of the clone topology command to clone a standby topology for only those instances specified accordingly. See Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information.

Prerequisites

The production instance or instances to be cloned cannot exist on the standby system.

The following are prerequisites for performing the clone instance and clone topology operation to the standby site system:

- The OracleAS Guard standalone kit must be installed on the standby system.
- Backup and Restore must be installed in the OracleAS Guard home on the standby system.
- A Java development kit with its jar utility must be installed on the standby system.
- For Windows systems, the services kit (`sc.exe`) must be installed on the standby system.

Procedure

The basic procedure consists of the following pre-clone (UNIX systems only) and clone steps.

Note: For OracleAS Disaster Recovery release 10.1.2.0.2 on Windows systems, see the pre-clone and post-clone steps in this same section in the *Oracle Application Server High Availability Guide* in the OracleAS Release 10.1.2.0.2 documentation set.

Pre-Clone Steps (For UNIX Systems Only)

For each instance on the production and standby sites, perform the following steps:

1. Log in as su - root.
2. CD to the instance home.
3. Shut down any OracleAS Guard servers.

   ```
   > <ORACLE HOME>/opmn/bin/opmnctl stopproc ias-component=ASG
   ```

4. Make sure `dsaServer.sh` in *`<ORACLE_HOME>`*`/dsa/bin` is executable by everyone. If it is not, record the permission, then change the executable permission by issuing the following command:

   ```
   chmod +x dsaServer.sh
   chmod u+x asgexec
   ```

5. Invoke asgctl and issue the startup command.

   ```
   > asgctl.sh startup
   ```

6. Log out as root on UNIX systems.

Clone Steps

From any instance on the production site, perform the following steps:

1. Log in as user (non root user on UNIX systems).
2. CD to the production instance home.
3. Invoke asgctl and run the clone instance command to clone the instance to the standby topology host system.

   Note: In the command output, you will see a number of connect messages. This is normal as the OracleAS Guard server is recycled during these operations.

   ```
   > asgctl.sh
   Application Server Guard: Release 10.1.3.0.0
   (c) Copyright 2004, 2005 Oracle Corporation. All rights reserved
   ASGCTL> connect asg prodoc4j oc4jadmin/adminpwd
   Successfully connected to prodoc4j:7890
   ASGCTL> set primary database sys/testpwd@asdb
   Checking connection to database asdb

   CLONE INSTANCE -- CLONE AN INSTANCE EXAMPLE

   ASGCTL> clone instance portal_2 to asmid2
   ```

```
Generating default policy for this operation
.
.
.

CLONE TOPOLOGY -- CLONE MULTIPLE INSTANCES EXAMPLE

# Command to use if you are using a policy file where <file>
# is the full path and file spec of the clone policy file.
ASGCTL> clone topology to standbyinfra using policy <file>
Generating default policy for this operation
.
.
.

ASGCTL> disconnect
ASGCTL> exit
>
```

4. Log out of the system.

Note: If OracleAS Guard does not run as root on UNIX systems, the user will be prompted by the OracleAS Guard client to run the underlying operations at each of the instance homes as root (manually) in order to continue with the operation.

The last step completes the cloning instance operation and brings the systems back to where they were before you started the operation. At this point, you could invoke asgctl, connect to a production system, discover the topology, and then perform a verify operation to determine if the production and standby topologies were valid and consistent with one another as you would expect them to be.

5.11 OracleAS Guard Operations -- Standby Instantiation and Standby Synchronization

After adhering to the following conditions, you are ready to use the Oracle Application Server Guard for standby instantiation and standby synchronization.

- Meet the requirements for the implementation of the OracleAS Disaster Recovery solution as described in Section 5.1.1, "OracleAS Disaster Recovery Requirements", Section 5.1.3, "Supported Topologies", and Section 5.2, "Preparing the OracleAS Disaster Recovery Environment".
- Install the OracleAS Disaster Recovery (DR) solution as described in Section 5.3, "Overview of Installing Oracle Application Server".

The following subsections describe standby instantiation and standby synchronization.

See Chapter 6, "OracleAS Guard asgctl Command-line Reference" for OracleAS Guard command-line asgctl utility reference information.

5.11.1 Standby Instantiation

The standby instantiation operation performs a number of operations to set 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 setup operations are:

- Uses a previous topology file created by performing a discovery topology operation.
- Verifies the topology definitions to ensure they comply with the rules of the OracleAS Disaster Recovery environment.
- Configures Oracle Data Guard to maintain the OracleAS Disaster Recovery environment for the database repository.
- Mirrors the configuration information of all the Oracle homes in the OracleAS topology to the corresponding Oracle home at the standby site.
- If you want to use a policy file, edit and use the instantiate policy file (`instantiate_policy.xml`). Specify this file in the `using policy <file>` parameter of the instantiate topology command to instantiate a standby topology for only those instances specified accordingly. See Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information.
- Reports any errors found for correction.

The procedure to perform a standby instantiation operation uses the following example, which assumes that you have invoked the OracleAS Guard client and performed a discover topology command to create a topology file.

See Section 6.2.1.1, "Special Considerations for Running Instantiate and Failover Operations in CFC Environments" if you have an OracleAS Disaster Recovery configuration in a CFC environment and are about to perform an instantiate operation.

1. Connect to the OracleAS Guard server.

```
ASGCTL > connect asg prodinfra ias_admin/<adminpwd>
Successfully connected to prodinfra:7890
ASGCTL>
```

2. Specify the primary OracleAS Metadata Repository database. See Section 5.15.1.2, "Specifying the Primary Database" for more information. If you have multiple OracleAS Metadata Repositories in your topology, you must authenticate each one using the set primary database command.

```
ASGCTL> set primary database sys/testpwd@asdb
```

3. Dump the policies (dump policies command), then edit and use the verify policy file (`verify_policy.xml`) and the instantiate policy file (`instantiate_policy.xml`) to specify the success requirement attribute for each instance in the file. See Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information.

```
ASGCTL> dump policies
Generating default policy for this operation
Creating policy files on local host in directory
"/private1/OraHome2/asr1012/dsa/conf/"
```

4. Verify the topology. The network hostname `standbyinfra` is used.

```
ASGCTL> verify topology with standbyinfra
```

5. Instantiate the topology at the secondary site. The network hostname `standbyinfra` is used. This command assumes that all Oracle homes have been installed using Oracle installer software. Specify the `using policy <file>`

parameter where `<file>` represents the path and file specification for the `instantiate_policy.xml` file.

```
ASGCTL> instantiate topology to standbyinfra using policy <file>
```

Whenever a standby instantiation is performed using the asgctl instantiate topology command a synchronization operation is also performed. Thus, you do not need to perform another synchronization operation immediately following the instantiation operation. If a period of time had passed following an instantiate operation, ensure that both the primary and standby sites are consistent. Then, perform a sync topology operation to ensure any changes that occurred on the primary site are applied to the secondary site.

5.11.2 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 exist:

- Deploy a new application or redeploy an existing application - Both the deployment of a new application and the redeployment 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 topology. This information has to be uniformly deployed at the standby site.
- Configuration changes - Specific changes, small to large, to a configuration, must be reflected at the standby site.
- User Provisioning - The default Infrastructure installation maintains the database for Oracle Internet Directory. As new users are added to the database, they should be synchronized to the standby site on a schedule that fulfills the business availability requirements.
- Periodic full synchronization - By default, the synchronization operations synchronizes only the pieces of configuration that have changed since the last synchronization operation. During test cycles or occasional complex configuration changes, administrators may want to fully refresh of their configuration information to the standby site to ensure mirroring of these changes.

You can specify a full or incremental synchronization. By default, an incremental synchronization is performed, which offers the best performance. However, in the following three circumstances a full synchronization 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 is a large accumulation of transactional changes over a long period of time on the primary site that must be synchronized with the secondary site.

As part of the synchronization operation, a verify operation is performed to ensure the required OracleAS Disaster Recovery environment is maintained. Additionally, if new OracleAS instances are installed into the OracleAS topology, OracleAS Guard will discover these installations.

If you want to use a policy file, edit and use the synchronization policy file (`sync_policy.xml`). Specify this file in the `using policy <file>` parameter of the sync topology command for synchronizing a standby topology for only those instances specified accordingly. See Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information.

The following example assumes that you have invoked the OracleAS Guard client and performed a discover topology command to create a topology file.

The procedure to perform standby synchronization is as follows:

1. Connect to the OracleAS Guard server.

   ```
   ASGCTL > connect asg prodinfra ias_admin/<adminpwd>
   Successfully connected to prodinfra:7890
   ASGCTL>
   ```

2. Specify the primary database. See Section 5.15.1.2, "Specifying the Primary Database" for more information.

   ```
   ASGCTL> set primary database sys/testpwd@asdb
   ```

3. Synchronize the secondary site with the primary site.

   ```
   ASGCTL> sync topology to standbyinfra
   ```

5.12 Runtime Operations -- OracleAS Guard Switchover and Failover Operations

Runtime operations include dealing with outages, whether they are scheduled or unscheduled (see Section 5.12.1, "Outages"), and monitoring ongoing OracleAS Guard operations using the asgctl command-line utility and troubleshooting (see Section 5.13, "Monitoring OracleAS Guard Operations and Troubleshooting").

5.12.1 Outages

Outages fall into two categories scheduled and unplanned.

The following subsections describe these outages.

5.12.1.1 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 the 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 operation has to be performed, which is explained in the section that follows.

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.

Note: During a switchover operation, the `opmn.xml` file is copied from the primary site to the standby site. For this reason, the value of the TMP variable must be defined the same in the `opmn.xml` file on both the primary and standby sites, otherwise this switchover operation will fail with a message that it could not find a directory. Therefore, make sure the TMP variable is defined identically and resolves to the same directory structure on both sites before attempting a switchover operation.

During site switchover, considerations must be made to avoid long periods of cached DNS information. Modifications to the site's DNS information, specifically time-to-live (TTL), must be performed. See Section 5.14.2, "Manually Changing DNS Names" for instructions.

If you want to use a policy file, edit and use the switchover policy file (`switchover_policy.xml`). Specify this file in the `using policy <file>` parameter of the switchover topology command for switching over to the standby topology only those instances specified accordingly. See Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information. This example does not show the use of a policy file.

See Section 6.2.1.3, "Special Considerations for Running a Switchover Operations in CFC Environments" if you have an OracleAS Disaster Recovery configuration in a CFC environment and are planning a switchover operation.

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 5.14.2, "Manually Changing DNS Names" for more information.
2. On the primary Infrastructure system, make sure the emagent process is stopped. Otherwise, the following error may occur 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, stop the Application Server Control (iasconsole) and stop the emagent process, as follows:

   ```
   > <ORACLE_HOME>/bin/emctl stop iasconsole
   ```

On UNIX systems, to check to see if there is an emagent process running, enter the following command:

```
> ps -ef | grep emagent
```

On UNIX systems, if after performing the stop iasconsole operation, the emagent process is still running, obtain the process ID (PID) as shown in the previous ps command, and stop the emagent process 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 10.1.2.0.2
(c) Copyright 2004, 2005 Oracle Corporation. All rights reserved
ASGCTL> connect asg prodinfra ias_admin/<adminpwd>
```

4. Switchover the topology to the secondary site. If you want to use a policy file, specify the `using policy <file>` parameter where `<file>` represents the path and file specification for the `switchover_policy.xml` file.

```
ASGCTL> switchover topology to standbyinfra
```

Note: As part of the OracleAS Guard switchover operation, an implicit sync topology operation is performed to make sure the topologies are identical. In addition, all OPMN services are stopped and then restarted on the production site.

5. Disconnect from the *old* primary site OracleAS Guard server.

```
ASGCTL> disconnect
ASGCTL>
```

6. Perform a wide area DNS switchover to direct requests to the new production site based on one of the options presented in Section 5.14, "Wide Area DNS Operations".
7. Adjust the wide area DNS TTL to an appropriate value.

Special Switchover Operation Considerations

This section describes the following special considerations relating to the switchover operation.

- When performing a switchover operation from a primary site with two Oracle Identity Management instances running to a standby site representing an asymmetric topology with only one Oracle Identity Management instance running, which means that the other node is to be ignored on the switchover site, the system administrator must not only edit the `switchover_policy.xml` policy file to indicate that this other node is to be set to Ignore, but must also shutdown all processes running on that node in order for the switchover operation to be successful. For example, if the two Oracle Identity Management instances running on the primary site are im.machineA.us.oracle.com and im.machineB.us.oracle.com, and the other node (im.machineB.us.oracle.com) is to be ignored on the switchover site, the system administrator must also shutdown all processes running on that node (im.machineB.us.oracle.com) in order for the switchover operation to succeed.

- When the discover topology command is issued following a switchover operation and the asymmetric standby site topology originally had one or more fewer middle tiers (for example, instA and instB) than there were in the original production site topology (instA, instB, and instC), a warning error message displays for each missing instance of a middle tier (instC, in this case). This warning error message is expected and can be ignored. When a discover topology command is issued following a switchover operation, OracleAS Server Guard reads the Oracle Internet Directory information, which is an exact copy of the original primary site Oracle Internet Directory information on this new primary site (former standby site). Because this Oracle Internet Directory information is identical to the original primary site Oracle Internet Directory information, when OracleAS Server Guard visits the host or home of each instance of these middle tiers to verify their existence, it discovers that some of the middle tiers do not exist, and issues warnings.

5.12.1.2 Unplanned Outages

An unplanned outage that impacts a production site occurs when it becomes unavailable and there is no possibility of restoring the production site to service within a reasonable period of time. This includes site-wide outages at the production site such as fire, flood, earthquake, or power outages.

Unplanned outages warrant performing a failover operation of the production site to the standby site.

Site Failover Operations

A site failover operation 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. A failover operation restores to the last synchronization point.

If you want to use a policy file, edit and use the failover policy file (`failover_policy.xml`). Specify this file in the `using policy <file>` parameter of the failover command for failing over to the standby topology only those instances specified accordingly. See Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information.

See Section 6.2.1.1, "Special Considerations for Running Instantiate and Failover Operations in CFC Environments" if you have an OracleAS Disaster Recovery configuration in a CFC environment and are about to perform a failover operation.

To fail over the production site to the standby site, follow these steps:

1. Connect to the OracleAS Guard server on the standby site. The network name is standbyinfra.

   ```
   ASGCTL> connect asg standbyinfra ias_admin/<adminpwd>
   Successfully connected to stanfbyinfra:7890
   ```

2. Specify that the primary OracleAS Metadata Repository 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. If you have multiple OracleAS Metadata Repositories in your topology, you must authenticate each one using the set new primary database command.

```
ASGCTL> set new primary database sys/testpwd@asdb
```

3. Perform an asgctl failover operation.

```
ASGCTL> failover
```

4. Discover the topology. You must perform this operation to create a new topology file for this production site.

```
ASGCTL> discover topology oidpassword=oidpwd
```

5.13 Monitoring OracleAS Guard Operations and Troubleshooting

After setting up your OracleAS Disaster Recovery solution, and instantiating the standby topology, and synchronizing the standby topology, 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 troubleshooting tasks. A typical OracleAS Guard monitoring or troubleshooting session may involve the following tasks:

1. Section 5.13.1, "Verifying the Topology"
2. Section 5.13.2, "Displaying the Current Operation"
3. Section 5.13.3, "Displaying a List of Completed Operations"
4. Section 5.13.4, "Stopping an Operation"
5. Section 5.13.5, "Tracing Tasks"
6. Section 5.13.6, "Writing Information About the Topology to a File"

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 communicates these requests to the other OracleAS Guard servers in the production and standby topologies, and status messages are returned to the OracleAS Guard client as well as any error messages should a particular task encounter a problem. Section 5.13.7, "Error Messages" describes where you can obtain more information about these error messages.

5.13.1 Verifying the Topology

To validate that the primary topology is running and the configuration is valid, enter the following asgctl command at the asgctl prompt.

```
ASGCTL> connect asg ias_admin/iastest2
Successfully connected to prodinfra:7890
ASGCTL> discover topology oidpassword=oidpwd
ASGCTL> verify topology
Generating default policy for this operation
prodinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
ASGCTL>
```

If you want to use a policy file, edit and use the verify policy file (`verify_policy.xml`) to specify the success requirement attribute for each instance in the file. Then specify the `using policy <file>` parameter in the verify command where `<file>` represents the path and file specification for the `verify_policy.xml` file. See Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information.

To compare a primary topology to which the local host is a member with a standby topology and ensure that they are consistent with one another and that both topologies conform to OracleAS Disaster Recovery requirements, enter the following asgctl command at the asgctl prompt and specify the name of the standby host system.

```
ASGCTL> dump policies
Generating default policy for this operation
Creating policy files on local host in directory
"/private1/OraHome2/asr1012/dsa/conf/"

ASGCTL> verify topology with standbyinfra
Generating default policy for this operation
prodinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
standbyinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
prodinfra:7890
    Verifying that the topology is symmetrical in both primary and standby configuration
ASGCTL>

# Command to use if you want to use a policy file
# verify topology with standbyinfra using policy <file>
```

5.13.2 Displaying the Current Operation

To display the status of all the current operations running on all nodes of the topology to which the OracleAS Guard client is connected, enter the following asgctl command at the asgctl prompt:

```
ASGCTL> show operation
*************************************
OPERATION: 19
  Status: running
  Elapsed Time: 0 days, 0 hours, 0 minutes, 28 secs
  TASK: syncFarm
    TASK: backupFarm
      TASK: fileCopyRemote
      TASK: fileCopyRemote
    TASK: restoreFarm
      TASK: fileCopyLocal
```

5.13.3 Displaying a List of Completed Operations

To display only operations that have completed (are *not* running on any nodes of the topology to which the OracleAS Guard client is connected for the current session), enter the following asgctl command at the asgctl prompt:

```
ASGCTL> show operation history
*************************************
OPERATION: 7
  Status: success
  Elapsed Time: 0 days, 0 hours, 0 minutes, 0 secs
  TASK: getTopology
    TASK: getInstance
*************************************
OPERATION: 16
  Status: success
  Elapsed Time: 0 days, 0 hours, 0 minutes, 0 secs
  TASK: getTopology
    TASK: getInstance
*************************************
OPERATION: 19
  Status: success
  Elapsed Time: 0 days, 0 hours, 1 minutes, 55 secs
  TASK: syncFarm
    TASK: backupFarm
      TASK: fileCopyRemote
      TASK: fileCopyRemote
    TASK: restoreFarm
      TASK: fileCopyLocall
```

5.13.4 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: 19
  Status: running
  Elapsed Time: 0 days, 0 hours, 0 minutes, 28 secs
  Status: running
.
.
.
ASGCTL> stop operation 19
```

5.13.5 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 trace flags to be enabled. In this case, the trace flag `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 trace flags that can be enabled. See the set trace command for a complete list of the trace flags that can be set.

```
ASGCTL> set trace on db
```

5.13.6 Writing Information About the Topology to a File

To write detailed information about the topology 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 the display shown in the dump topology command, except it is written to a file that you can save for future reference.

```
ASGCTL> dump topology to c:\dump_mid_1.txt
```

5.13.7 Error Messages

Appendix B, "OracleAS Guard Error Messages" categorizes and describes the error messages that may appear while using the OracleAS Disaster Recovery solution.

5.14 Wide Area DNS Operations

To direct client requests to the entry point of a production site, use DNS resolution. When a site switchover or failover is performed, client requests have to be redirected transparently to the new site that is 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 by either using a wide area load balancer or manually changing DNS names.

Note: A hardware load balancer is assumed to be front-ending each site. Check `http://metalink.oracle.com` for supported load balancers.

The following subsections describe the DNS switchover operation.

5.14.1 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 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.

5.14.2 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 switchover:

1. Make a note the current time-to-live (TTL) value of the production site's load balancer mapping. This mapping is in the DNS cache and it will remain there until the TTL expires. As an example, let's assume that the TTL is 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 from 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.

5.15 Using OracleAS Guard Command-Line Utility (asgctl)

This section includes the following subsections:

- Section 5.15.1, "Typical OracleAS Guard Session Using asgctl"
- Section 5.15.2, "Periodic Scheduling of OracleAS Guard asgctl Scripts"
- Section 5.15.3, "Submitting OracleAS Guard Jobs to the Enterprise Manager Job System"
- Section 5.16.1, "Special Considerations for Multiple OracleAS Metadata Repository Configurations"
- Chapter 6, "OracleAS Guard asgctl Command-line Reference"

5.15.1 Typical OracleAS Guard Session Using asgctl

A typical OracleAS Guard session using asgctl involves the following tasks, which are described in the following subsections:

- Section 5.15.1.1, "Getting Help"
- Section 5.15.1.2, "Specifying the Primary Database"
- Section 5.15.1.3, "Discovering the Topology"

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 5.15.1.4, "Creating and Executing an asgctl Script" and then execute the script as described in Section 5.15.2, "Periodic Scheduling of OracleAS Guard asgctl Scripts" and Section 5.15.3, "Submitting OracleAS Guard Jobs to the Enterprise Manager Job System".

5.15.1.1 Getting Help

To get help on a particular command, enter the asgctl command at the asgctl prompt and specify 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_administrator_account>/<password>]
    disconnect
    exit
    quit
    add instance <instance_name> on <instance_host> [to topology]
    clone topology to <standby_topology_host> [using policy <file>] [no standby]
    clone instance <instance> to <standby_topology_host> [no standby]
    discover topology [oidhost=<host>] [oidsslport=<sslport>] [oiduser=<user>]
oidpassword=<pass>
    discover topology within farm
    dump farm [to <file>]  (Deprecated)
    dump topology  [to <file>] [using policy <file>]
    dump policies
    failover [using policy <file>]
    help [<command>]
    instantiate farm to <standby_farm_host> (Deprecated)
    instantiate topology to <standby_topology_host> [using policy <file>]
    remove instance <instance_name> [from topology]
    set asg credentials <host> <ias_administrator_account>/<password> [for topology]
    set asg credentials <host> ias_admin/<password> [for farm] (Deprecated)
    set primary database <username>/<password>@<servicename> [pfile <filename> | spfile
<filename>]
    set new primary database <username>/<password>@<servicename> [pfile <filename> | spfile
<filename>]
    set noprompt
    set trace on|off <traceflags>
    sync farm to <standby_farm_host> [full | incr[emental]] (Deprecated)
    sync topology to <standby_topology_host> [full | incr[emental]] [using policy <file>]
    startup [asg]
    startup farm (Deprecated)
    startup topology
    shutdown [local]
    shutdown farm (Deprecated)
    shutdown topology
    show op[eration] [full] [[his]tory]
    show env
    stop op[eration] <op#>
    switchover farm to <standby_farm_host> (Deprecated)
    switchover topology to <standby_topology_host> [using policy <file>]
    verify farm [with <host>](Deprecated)
    verify topology [with <host>] [using policy <file>]
ASGCTL>
```

5.15.1.2 Specifying the Primary Database

To identify the OracleAS Infrastructure database on the primary topology, enter the following asgctl command at the asgctl prompt and specify 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
Checking connection to database 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. You must always set the primary database before performing an instantiate, sync, switchover, or failover operation.

If you have multiple OracleAS Metadata Repositories in your topology, you must authenticate each one using the set primary database command.

5.15.1.3 Discovering the Topology

You must perform a discover topology command when you first set up your OracleAS Disaster Recovery environment in order to initially create the topology XML file. There after, you should perform a discover topology operation whenever you procure another Oracle home in a production site or change roles from a production to a standby site through a switchover or failover operation. The discover topology command queries Oracle Internet Directory for all instances within the topology that share the same Oracle Internet Directory for the production site. See Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information about topology files. Enter the following asgctl command at the asgctl prompt to discover the topology:

```
ASGCTL> discover topology oidpassword=oidpwd
Discovering topology on host "infra" with IP address "123.1.2.111" prodinfra:7890
    Connecting to the OID server on host "infra.us.oracle.com" using SSL port
"636" and username "orcladmin"
    Getting the list of databases from OID
    Gathering database information for SID "asdb" from host "infra.us.oracle.com"
    Getting the list of instances from OID
    Gathering instance information for "asr1012.infra.us.oracle.com" from host
"infra.us.oracle.com"
    Gathering instance information for "asmid1.asmid1.us.oracle.com" from host
"asmid1.us.oracle.com"
    Gathering instance information for "asmid2.asmid2.us.oracle.com" from host
"asmid2.us.oracle.com"
The topology has been discovered. A topology.xml file has been written to each
home in the topology.

ASGCTL>
```

After the production topology is known by OracleAS Guard for a production site, you can execute any one of the subsequent commands to perform a subsequent asgctl operation that involves the standby site. See discover topology for more information.

5.15.1.4 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 the 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.

You can also set the noprompt state for use in executing commands in an asgctl script in which all interactive prompts are later ignored. See the asgctl set noprompt command for more information.

5.15.2 Periodic Scheduling of OracleAS Guard asgctl Scripts

For OracleAS Guard operations that you want to run periodically, such as a periodic sync topology operation to keep the standby topology synchronized with the primary topology, 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 one or two manpages 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, monthly, or at specific times. You can also purchase additional scheduler software with more options from a third party and then set the time and frequency to run the asgctl script. See the Windows operating system help for more information.

5.15.3 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) will invoke asgctl to 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.

5.16 Special Considerations for Some OracleAS Metadata Repository Configurations

This section describes special considerations for multiple OracleAS Metadata Repositories and OracleAS Metadata Repositories created using the OracleAS Metadata Repository Creation Assistant.

5.16.1 Special Considerations for Multiple OracleAS Metadata Repository Configurations

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 5.16.1.1, "Setting asgctl Credentials".
- Use a common set of credentials in the standby topology that are the same as the credentials used in the primary topology, see Section 5.16.1.2, "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 OracleAS Guard credentials so that the OracleAS Guard server can connect to each host system in the configuration.

5.16.1.1 Setting asgctl Credentials

To set different credentials for all the host systems belonging to the same topology, 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 topology**.

Note: If this is an OracleAS 10.1.3 installation, the user name must be `oc4jadmin` and the password for the `oc4jadmin` account created during the Oracle Application Server 10.1.3 installation.

These settings are good for the current session.

```
ASGCTL> set asg credentials standbyinfra ias_admin/<iasadminpwd> for topology
```

When you specify the key words, **for topology**, you set the credentials for all the host systems that belong to the same topology as the specified system; otherwise, the credentials will apply only 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 topology. In the previous example, the same credentials were set for all nodes on the standby topology, so that these credentials differ from the credentials used in the primary topology. The following command sets the credentials for a specific node, the standbyinfra node, on the standby topology.

```
ASGCTL> set asg credentials standbyinfra ias_admin/<iasadminpwd>
```

To summarize, if you set the credentials for a topology, these credentials are inherited for the entire topology. If you set the credentials for an individual host on the topology, the credentials (for this host) override the default credentials set for the topology.

In addition, for topologies that have more than one Infrastructure, such as a collocated Oracle Internet Directory+OracleAS Metadata Repository and a separate Portal OracleAS Metadata Repository, OracleAS Guard requires that you set the credentials for each system on which an Infrastructure resides before performing any important OracleAS Guard operations, such as instantiate, sync, switchover, and failover. See set asg credentials for an example.

5.16.1.2 Specifying the Primary Database

To identify the OracleAS Infrastructure database on the primary topology, 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 topology and the TNS service name of the OracleAS Infrastructure database:

```
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database 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.

If a production or standby site has multiple OracleAS Metadata Repository instances installed and you are performing an instantiate, sync, switchover, or failover operation, you must identify all of the OracleAS Metadata Repository instances by performing a set primary database command for each OracleAS Metadata Repository instance before performing either an instantiate, sync, switchover, or failover operation. See set asg credentials for an example.

5.16.1.3 Setting OracleAS Guard Port Numbers

OracleAS Guard uses a default port (port) number of 7890; for example, `port=7890`. If there are any additional Oracle homes installed on a system, each additional Oracle home must have a unique OracleAS Guard port number, that is usually incremented by the value one, for example, `port=7891`, and so forth. See Section 5.4.6, "Supported OracleAS Disaster Recovery Configurations" for more information.

5.16.2 Special Considerations for OracleAS Metadata Repository Configurations Created Using OracleAS Metadata Repository Creation Assistant

The following items are special considerations for an OracleAS Metadata Repository configuration created using OracleAS Metadata Repository Creation Assistant. These Metadata Repository databases are installed in Oracle homes with schemas containing user data. For this reason, there are some special considerations regarding OracleAS Disaster Recovery.

- On the standby site, no Metadata Repository is created by OracleAS Disaster Recovery. The System Administrator must use the OracleAS Metadata Repository Creation Assistant on the standby site and create this Metadata Repository.
- During a clone topology operation to the standby site no instantiate operation is performed on the Metadata Repository.
- **Warning:** Do not perform a clone operation to a standby system containing an existing Oracle home because it will get overwritten. Only perform a clone operation to a standby system where there is no Oracle home installed.
- The OracleAS Disaster Recovery solution assumes that user schemas are already configured for Oracle Data Guard.
- The OracleAS Disaster Recovery solution assumes that when using Oracle Data Guard, that the Metadata Repository is not in managed recovery mode.
- OracleAS Disaster Recovery will not change the recovery mode of Oracle Data Guard for the Metadata Repository if it is found to be in managed recovery mode; instead, OracleAS Guard will issue a warning indicating that the database is in managed recovery mode and this feature must be set differently.
- OracleAS Guard must be installed in every Oracle home on every system that is part of your production and standby topology configured for the OracleAS Disaster Recovery solution. OracleAS Guard can be installed as a standalone install a kit located on OracleAS Utility media #2. See the OracleAS Disaster Recovery installation information in *Oracle Application Server Installation Guide* for more information.

5.17 Special Considerations for OracleAS Disaster Recovery Environments

The following sections describe some additional special considerations for OracleAS Disaster Recovery environments.

5.17.1 Some Special Considerations That Must Be Taken When Setting Up Some OracleAS Disaster Recovery Sites

Some special considerations must be taken when setting up OracleAS Disaster Recovery for sites that include:

- Middle-tier CFC configurations
- OracleAS Guard release 10g (9.0.4) cloning

In both cases, the instance name stored in Oracle Internet Directory is comprised of the original host name on which the production site installation was performed. In the case of an OracleAS Disaster Recovery site having a symmetric topology, the standby OracleAS Disaster Recovery peer must be installed identically to the production site and for an OracleAS Guard Release 10.1.2.0.2 or higher clone instance or clone topology operation, the operation must be performed to mirror the configuration.

In an asymmetric standby topology, where the production site physical host does not exist at the standby site, the instance name should be filtered out of the topology using the policy file capabilities (see Section 5.7, "Dumping Policy Files and Using Policy Files With Some asgctl Commands" for more information). The hosts file of the host on which a discover topology operation is performed must map the original host name to the corresponding IP of the new host system on which it was cloned.

5.17.2 Handling ons.conf and dsa.conf Configuration Files for Asymmetric Topologies

The OracleAS Guard operation synchronizes the configuration files of the standby site with those of the production site through a backup operation on the primary site and restores them to the standby site.

For asymmetric topologies the standby site has fewer nodes, thus node name list in the `ons.conf` configuration file is different from the one on the production site. Therefore, the `ons.conf` configuration file should be excluded from the backup list of files so it is not restored on the standby site. If not excluded, the nodes listed in the `ons.conf` configuration file will reflect the node list of the production site and not the actual node list of the standby site. This will cause inefficiencies as OPMN will continue to ping non existing nodes.

Additionally, for asymmetric topologies the `dsa.conf` configuration file for an Oracle home may contain special settings on the production site that are different from the standby site. For example, the `inventory_location` parameter setting may be different on the standby site than it is on the primary site. In this case, you should also exclude the `dsa.conf` configuration file from the backup list of files so it is not restored on the standby site. Otherwise, in this example, the location of the OraInventory will not be correct on the standby site following a switchover or failover operation.

In both these cases, you should modify the Backup and Restore exclusion file as follows to exclude both of these configuration files from the backup list of files so neither is then restored to the standby site:

```
# Exclude Files
# - Add additional files to this list that you want to be ignored
# - during the configuration file backup/restore
c:\oracle\ias1012\opmn\conf\ons.conf
c:\oracle\ias1012\dsa\dsa.conf
```

If the directives set in the `dsa.conf` file are necessary at the site that currently functions as the production site, it may be desirable to include the `dsa.conf` file for synchronization and add a post switchover or failover step to edit physical site specific directives.

5.17.3 Other Special Considerations for OracleAS Disaster Recovery Environments

See Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for information describing some additional special considerations.

6 OracleAS Guard asgctl Command-line Reference

This chapter contains reference information describing the asgctl commands. Table 6–1 summarizes all the asgctl commands. Table 6–2 summarizes all the asgctl commands that were deprecated beginning with OracleAS release 10.1.2.0.2. Subsequent sections provide detailed reference information common to many commands and about each command.

Table 6–1 Summary of asgctl Commands

Command	Description
add instance	Adds to the local topology file, the specified instance name and name of the host system on which this instance is installed, and if specified, propagates this updated topology file to all instances in the Disaster Recovery production environment.
asgctl	Invokes 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`.
clone instance	Clones a single production instance to a standby system.
clone topology	Clones two or more production middle tier instances to standby middle tier systems.
connect asg	Connects the OracleAS Guard client to the OracleAS Guard server.
disconnect	Disconnects the OracleAS Guard client from the OracleAS Guard server.
discover topology	Discovers by querying Oracle Internet Directory all instances within the topology that share the same Oracle Internet Directory for a production site and generates a topology XML file that describes the topology.
discover topology within farm	Discovers the topology within the farm for a site when Oracle Internet Directory is not available; in this case, OracleAS Guard server uses OPMN to discover the topology within the farm.
dump policies	Directs OracleAS Guard server to write detailed, default policy information to respective XML formatted files for a set of asgctl commands. Each policy file can then be edited and later specified to define the topology's disaster recovery policy to be used with the respective administrative command.
dump topology	Directs the OracleAS Guard server to write detailed information about the topology to the screen or if specified, to a file.

Table 6–1 (Cont.) Summary of asgctl Commands

Command	Description
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 topology	Creates a topology 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 so that the primary and standby sites are 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.
remove instance	Removes from the local topology file, the specified instance name, and if specified, propagates this updated topology file to all instances in the Disaster Recovery production environment.
set asg credentials	Sets the credentials used to authenticate the OracleAS Guard client connections to OracleAS Guard servers and connections between OracleAS Guard servers to a specific host.
set echo	Sets command-echoing on or off in an asgctl script.
set new primary database	Identifies the OracleAS Infrastructure database on the standby topology as the new primary OracleAS Infrastructure database.
set noprompt	Sets the noprompt state in an asgctl script in which all interactive prompts are thereafter ignored.
set primary database	Identifies the OracleAS Infrastructure database on the primary topology.
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 env	Shows the current environment of the OracleAS Guard server to which the OracleAS Guard clients is connected.
show operation	Shows the current operation.
shutdown	Shuts down the OracleAS Guard server at the operating system command-line prompt on a system on which OPMN is not running. This command is only used with cloning an instance or cloning a topology.
shutdown topology	Shuts down a running topology.
startup	Starts up the OracleAS Guard server at the operating system command-line prompt on a system on which OPMN is not running. This command is only used with cloning an instance or cloning a topology.
startup topology	Starts up a shutdown topology.
stop operation	Stops the specified operation.
switchover topology	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.
sync topology	Synchronizes the standby site with the primary site so that the primary and standby sites are consistent.

Table 6–1 (Cont.) Summary of asgctl Commands

Command	Description
verify topology	Verifies that the topology is running and the configuration is valid. If a standby topology is specified, this command compares the primary and standby topologies to verify that they conform to the requirements for OracleAS Disaster Recovery.

Table 6–2 Summary of Deprecated asgctl Commands

Command	Description
dump farm (Deprecated)	Directs the OracleAS Guard server to write detailed information about the farm to the screen or if specified, to a file.
instantiate farm (Deprecated)	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 so that the primary and standby sites are consistent.
shutdown farm (Deprecated)	Shuts down a running farm.
startup farm (Deprecated)	Starts up a shutdown farm.
switchover farm (Deprecated)	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.
sync farm (Deprecated)	Synchronizes the standby site with the primary site so that the primary and standby sites are consistent.
verify farm (Deprecated)	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.

6.1 Information Common to OracleAS Guard asgctl Commands

This section describes information that is common to OracleAS Guard asgctl commands.

General Information

The OracleAS Guard client must be connected to an OracleAS Guard server when you issue any asgctl command with the exception of startup and shutdown commands.

The 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 system must be a member of the production site topology.

OracleAS Guard Server Information

The OracleAS Guard server must be started on the standby host system (`<standby_topology_host>`. The OracleAS Guard server can be stopped and started using the opmnctl command-line Utility as follows:

```
On UNIX systems:
<ORACLE_HOME>/opmn/bin/opmnctl  startproc  ias-component=ASG

On Windows systems:
<ORACLE_HOME>\opmn\bin\opmnctl  stopproc  ias-component=ASG
```

6.2 Information Specific to a Small Set of OracleAS Guard Commands

This section describes information that is specific to a small set of OracleAS Guard operations, such as instantiate, sync, failover, switchover, dump topology, discover topology, clone topology, verify topology, setting the primary database, and setting asg credentials.

If a production or standby site has multiple OracleAS Metadata Repository instances installed and you are performing an instantiate, sync, switchover, or failover operation, you must identify all of the OracleAS Metadata Repository instances by performing a `set primary database` command for each and every OracleAS Metadata Repository instance prior to performing either an instantiate, sync, switchover, or failover operation.

OracleAS Guard requires that you set the credentials for any OracleAS Guard server system in the topology that has different credentials from the OracleAS Guard server to which you are connected before performing any important OracleAS Guard operations, such as instantiate, sync, switchover, and failover. See set asg credentials for an example.

You must perform a discover topology command when you first set up your OracleAS Disaster Recovery environment in order to initially create the topology XML file; there after, you should perform a discover topology operation whenever you procure another Oracle home in a production site or change roles from a production to a standby site through a switchover or failover operation.

The following information and scenarios will help to clarify the usage of topology files.

- For OracleAS release 10.1.2 and earlier, use the discover topology command. If Oracle Internet Directory is not in the topology, then use the discover topology within farm command.
- For OracleAS releases 10.1.2 and 10.1.3, connect to the OracleAS Guard 10.1.2 server (from either an OracleAS 10.1.2 or 10.1.3 home) and perform a discover topology command. OracleAS Guard will write a `topology.xml` file to each OracleAS home in the discovered topology. Next, if you want to add an instance (add instance command), then you must perform this operation from the OracleAS 10.1.3 home and connect to any OracleAS 10.1.2 system in the existing topology and perform the add instance command.
- For OracleAS release 10.1.3, connect to an OracleAS Guard 10.1.3 server and add that instance to the topology (which does not yet exist). Then, add any additional instances using the same OracleAS Guard connection.

An important point to emphasize in these last two scenarios is that if the `topology.xml` file does not exist in the OracleAS home of the instance you are adding to the topology, OracleAS Guard creates a new `topology.xml` file, which essentially defines a new topology.

If you want to use a policy file, edit the contents of the XML policy file to define by instance the domain of execution operations that are permitted for any one of these asgctl commands (clone topology, dump topology, failover, instantiate topology, switchover topology, sync topology, and verify topology). Each instance list entry in this XML policy file (`clone_policy.xml`, `dump_policy.xml`, `failover_policy.xml`, `instantiate_policy.xml`, `switchover_policy.xml`, `sync_policy.xml`, and `verify_policy.xml`) logically tags a production-standby peer combination with a particular attribute that defines the success requirement for the commands successful operation. See Section 5.7, "Dumping Policy Files and Using

Policy Files With Some asgctl Commands" for more information and an example of an XML policy file.

6.2.1 Special Considerations for OracleAS Disaster Recovery Configurations in CFC Environments

In an OracleAS Disaster Recovery configuration that uses CFC on the primary topology or standby topology, or both, the following information must be considered before performing an asgctl clone, instantiate topology, switchover topology, or failover command. Before taking a cold backup or restoring the metadata repository database, the OracleAS Recovery Manager shuts down the database first.

For example, 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 a clone, 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 clone, instantiate, switchover, or failover operation completes). The steps to perform this sequence of operations are described in a note in Section 6.2.1.1, "Special Considerations for Running Instantiate and Failover Operations in CFC Environments" and Section 6.2.1.3, "Special Considerations for Running a Switchover Operations in CFC Environments".

6.2.1.1 Special Considerations for Running Instantiate and Failover Operations in CFC Environments

In an OracleAS Disaster Recovery configuration that uses CFC on the primary topology or standby topology, or both, the following information must be considered before performing an asgctl clone, instantiate, switchover, or failover operation.

Before taking a cold backup or restoring the metadata repository database, the OracleAS Recovery Manager shuts down the database first.

For example, 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 clone, 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 the sqlplus command-line Utility to startup the database.
4. Using Windows Service Control Manager, start the Oracle Process Manager.
5. Perform the asgctl commands, including the clone, instantiate, switchover, or failover operation.
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.

6.2.1.2 A Special Consideration and Workaround for Performing an Instantiate Operation in CFC Environments

When performing an instantiate operation, OracleAS Guard puts an entry for the remote database in the `tnsnames.ora` file on both the production and standby site. The service name of this entry is constructed by concatenating _REMOTE1 to the database service name (for example, ORCL_REMOTE1). The entry contains the IP address of the target host where the database is running. On the production site, the IP will refer to the standby system and on the standby site, the IP refers to the production system.

In a CFC environment, the database is accessed using a virtual IP rather than a physical IP. When OracleAS Guard creates the `tnsnames.ora` entry it should use the virtual IP, but it uses the physical IP instead. This problem will be fixed in a future release of OracleAS Guard. As a workaround, when performing an instantiate operation in this environment, edit the `tnsnames.ora` file after an instantiation operation and replace the physical IP in the entry with the virtual IP used to access the database.

6.2.1.3 Special Considerations for Running a Switchover Operations in CFC Environments

In an OracleAS Disaster Recovery configuration that uses CFC on the primary topology or standby topology or both, the following information must be considered before performing an asgctl instantiate topology, switchover topology, or failover command.

Before taking a cold backup or restoring the metadata repository database, the OracleAS Recovery Manager shuts down the database first.

For example, 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 start up the database.
4. Perform the asgctl commands, including the instantiate topology, switchover topology, or failover command.
5. 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.

6.2.2 Other Special Considerations for OracleAS Disaster Recovery Environments

See Section 5.16, "Special Considerations for Some OracleAS Metadata Repository Configurations" and Section 5.17, "Special Considerations for OracleAS Disaster Recovery Environments" for information describing some additional special considerations for OracleAS Disaster Recovery environments.

add instance

Adds to the local topology file, the specified instance name and name of the host system name on which this instance is installed, and if specified, propagates the updated topology file to all instances in the Disaster Recovery production environment.

Format

```
add instance <instance_name> on <instance_host> [to topology]
```

Parameters

instance_name
The name of the instance to be added to the topology file.

instance_host
The name of the host on which this instance is installed. In the DR environment, if this host has an alias name or has a virtual hostname, then the virtual hostname must be used because it is this value that is placed in the topology file.

to topology
A keyword, that if present in the command line, directs OracleAS Guard to propagate the updated topology file to all instances in the Disaster Recovery production environment. Any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, and switchover will automatically propagate the production topology file across the standby environment.

Usage Notes

This command is useful for managing an instance on an OracleAS Guard server to which the OracleAS Guard client is connected. For example, you may have used the remove instance command to remove from the local topology file or from all topology files within the topology, an instance on this local host system because of a problem with it. Now you want to add to the local topology file or to all topology files in the topology, the good instance. In this case, you may not have wanted to manage the bad instance through the policy file where you could have set the success requirement attribute to `Ignore` for this instance when invoking asgctl commands to run across the entire topology.

This command is particularly useful for managing Disaster Recovery farms in which OID is not available, in other words, an OracleAS Release 10.1.3 only topology. You must use the discover topology within farm command to initially create the topology file for each instance within this farm. Then you can manage instances by adding or removing individual instances from the local topology file using the add instance and remove instance commands. If you specify the `to topology` or `from topology` keywords, the updated local topology file changes are propagated to all instances in the Disaster Recovery environment. See Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information about topology files.

This command is useful for adding an OracleAS 10.1.3 J2EE instance to an OID based 10.1.2.0.2 topology to support a mixed version Disaster Recovery environment. For example, you can use the add instance command to add an OracleAS 10.1.3 J2EE instance to your OID based 10.1.2.0.2 topology. See Section 5.9, "Adding or Removing OracleAS 10.1.3 Instances to Redundant Single OracleAS 10.1.3 Home J2EE Topology

Integrated with an Existing Oracle Identity Management 10.1.2.0.2 Topology" for a use case. See Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information about topology files.

This command is also useful for supporting mixed topologies for any OracleAS release from Release 9.0.4 upwards through Release 10.1.3. The only requirement is that the Release 10.1.3 OracleAS Guard standalone kit must be installed on all systems in your Release 9.0.4 Disaster Recovery environment. See Chapter 8, "OracleAS Disaster Recovery Site Upgrade Procedure" for more information.

Example

The following command in the example adds to the local topology file only, an instance named oc4j1 that is installed on the local host system named prodinfra.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> discover topology within farm
ASGCTL> add instance oc4j1 on prodinfra
```

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`.

Example

```
> asgctl.sh
Application Server Guard: Release 10.1.3.0.0

(c) Copyright 2004, 2005 Oracle Corporation. All rights reserved
ASGCTL>
```

clone instance

Clones a single production instance to a standby system.

Format

```
clone instance <instance> to <standby_topology_host> [no standby]
```

Parameters

instance
The name of the instance.

standby_topology_host
The name of the standby topology host to which the instance is to be cloned.

no standby
A keyword, that if present in the command-line, directs OracleAS Guard to clone the instance without setting up the instances in Disaster Recovery (no Data Guard). In a MR home only, an instantiate operation would normally be done, but in this case it is not performed.

Usage Notes

This command is useful for cloning a production instance on a middle tier to a standby middle tier host system. The clone instance operation eliminates the task of having to install the Oracle instance on the standby middle tier system and perform an instantiate operation.

The production instance to be cloned cannot exist on the standby system.

The following are prerequisites for performing the clone instance operation to the standby site system

- The OracleAS Guard standalone kit must be installed on the standby system.
- Backup and Restore must be installed in the OracleAS Guard home on the standby system
- A Java development kit with its jar utility must be installed on the standby system
- For Windows systems, the services kit (`sc.exe`) must be installed on the standby system

When the `no standby` keyword is specified in the command-line, the topology entry (<topology> </topology>) containing the entries <nodes list = "nodes"/>, <discover list = "nodes"/>, and <gateway list = "nodes"/> is removed from the `opmn.xml` file so as not to conflict with the primary configuration. Normally, a switchover or failover operation rewrites this topology entry back into the `opmn.xml` file; but because neither operation occurred, the `opmn.xml` file on the standby site will be missing this information.

See Section 5.10, "OracleAS Guard Operations -- Standby Site Cloning of One or More Production Instances to a Standby System" for more information.

The basic procedure consists of the following pre-clone (UNIX systems only) and clone steps.

Note: For OracleAS Disaster Recovery release 10.1.2.0.2 on Windows systems, see the pre-clone and post-clone steps in this same section in the *Oracle Application Server High Availability Guide* in the OracleAS Release 10.1.2.0.2 documentation set.

Pre-Clone Steps (For UNIX Systems Only)

For each instance on the production and standby sites, perform the following steps:

1. Log in as su - root.
2. CD to the instance home.
3. Shut down any OracleAS Guard servers.

   ```
   > <ORACLE HOME>/opmn/bin/opmnctl stopproc ias-component=ASG
   ```

4. Make sure `dsaServer.sh` in *`<ORACLE_HOME>`*`/dsa/bin` is executable by everyone. If it is not, record the permission, then change the executable permission by issuing the following command:

   ```
   chmod +x dsaServer.sh
   chmod u+x asgexec
   ```

5. Invoke asgctl and issue the startup command.

   ```
   > asgctl.sh startup
   ```

6. Log out as root on UNIX systems.

Clone Steps

From any instance on the production site, perform the following steps:

1. Log in as user (non root user on UNIX systems).
2. CD to the production instance home.
3. Invoke asgctl and run the clone instance command to clone the instance to the standby topology host system.

   Note: In the command output, you will see a number of connect messages. This is normal as the OracleAS Guard server is recycled during these operations.

4. Log out of the system.

   Note: If OracleAS Guard does not run as root on UNIX systems, the user will be prompted by the OracleAS Guard client to run the underlying operations at each of the instance homes as root (manually) in order to continue with the operation.

This last step completes the cloning instance operation and brings the systems back to where they were before you started the clone instance operation. At this point you could invoke asgctl, connect to a production system, discover the topology, and then perform a verify operation to determine whether the production and standby topologies were valid and consistent with one another as you would expect them to be.

Example

The following command in the example clones an instance named portal_2 to the standby topology host system named asmid2.

```
1. Check the prerequisites as described in the Usage Notes.
2. Perform the Pre-Clone steps as described in the Usage Notes.
3. Perform the Clone steps as described in the Usage Notes.
   a. Log in as user to any production system.
   b. CD to any production instance home.
   c. Invoke asgctl and perform the clone instance command.
> asgctl.sh
Application Server Guard: Release 10.1.3.0.0

(c) Copyright 2004, 2005 Oracle Corporation. All rights reserved
ASGCTL> connect asg prodoc4j oc4jadmin/adminpwd
Successfully connected to prodoc4j:7890
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database asdb
ASGCTL> clone instance prodoc4j2 to asmid2
Generating default policy for this operation
.
.
.
ASGCTL> disconnect
ASGCTL> exit
>
   d. Log off the system
```

clone topology

Clones two or more production middle tier instances to standby middle tier systems.

Format

```
clone topology to <standby_topology_host> [using policy <file>] [no standby]
```

Parameters

standby_topology_host
The name of the standby topology host system.

using policy <file>
Full path and file specification for the XML policy file.

no standby
A keyword, that if present in the command-line, directs OracleAS Guard to clone the instance without setting up the instances in Disaster Recovery (no Data Guard). In a MR home only, an instantiate operation would normally be done, but in this case it is not performed.

Usage Notes

This command is useful for cloning two or more production instances on middle tier systems to a standby middle tier host system. The clone topology operation eliminates the task of having to install these Oracle instances on the standby middle tier systems and perform an instantiate operation.

As part of the clone topology operation, the instances are cloned and the OracleAS Metadata Repository is instantiated; however for a OracleAS Metadata Repository configuration created using OracleAS Metadata Repository Creation Assistant, no instantiate operation is performed.

The production instances to be cloned cannot exist on the standby systems.

The following are prerequisites for performing the clone topology operation to standby site systems.

- The OracleAS Guard standalone kit must be installed on each standby system
- Backup and Restore must be installed on each OracleAS Guard home on each standby system
- A Java development kit with its jar utility must be installed on each standby system
- For Windows systems only, the services kit (`sc.exe`) must be installed on each standby system

When the `no standby` keyword is specified in the command-line, the topology entry (<topology> </topology>) containing the entries <nodes list = "nodes"/>, <discover list = "nodes"/>, and <gateway list = "nodes"/> is removed from the `opmn.xml` file so as not to conflict with the primary configuration. Normally, a switchover or failover operation rewrites this topology entry back into the `opmn.xml` file; but because neither operation occurred, the `opmn.xml` file on the standby site will be missing this information.

See Section 5.10, "OracleAS Guard Operations -- Standby Site Cloning of One or More Production Instances to a Standby System" for more information.

The basic procedure consists of the following pre-clone (UNIX systems only) and clone steps.

Note: For OracleAS Disaster Recovery release 10.1.2.0.2 on Windows systems, see the pre-clone and post-clone steps in this same section in the *Oracle Application Server High Availability Guide* in the OracleAS Release 10.1.2.0.2 documentation set.

Pre-Clone Steps (For UNIX Systems Only)

For each instance on the production and standby sites, perform the following steps:

1. Log in as su - root.
2. CD to the instance home.
3. Shut down any OracleAS Guard servers.

   ```
   > <ORACLE HOME>/opmn/bin/opmnctl stopproc ias-component=ASG
   ```

4. **On UNIX systems only:** make sure `dsaServer.sh` in `<ORACLE_HOME>/dsa/bin` is executable by everyone. If it is not, record the permission, then change the executable permission by issuing the following command:

   ```
   chmod +x dsaServer.sh
   chmod u+x asgexec
   ```

5. Invoke asgctl and issue the startup command.

   ```
   > asgctl.sh startup
   ```

6. Log out as root on UNIX systems.

Clone Steps

From any instance on the production site, perform the following steps:

1. Log in as user (non root user on UNIX systems).
2. CD to any production instance home.
3. Invoke asgctl and run the clone topology command to clone the topology to the standby topology host system.

   Note: In the command output, you will see a number of connect messages. This is normal as the OracleAS Guard server is recycled during these operations.

   ---

4. Log out of the system.

Note: If OracleAS Guard does not run as root on UNIX systems, the user will be prompted by the OracleAS Guard client to run the underlying operations at each of the instance homes as root (manually) in order to continue with the operation.

This last step completes the cloning topology operation and brings the systems back to where they were before you started the clone topology operation. At this point you could invoke asgctl, connect to a production system, discover the topology, and then perform a verify operation to determine whether the production and standby topologies were valid and consistent with one another as you would expect them to be.

See Section 6.1, "Information Common to OracleAS Guard asgctl Commands" and Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information.

Example

The command in the following example results in the OracleAS Guard client cloning the topology to the standby topology host system standbyinfra.

```
1. Check the prerequisites as described in the Usage Notes.
2. Perform the Pre-Clone steps as described in the Usage Notes.
3. Perform the Clone steps as described in the Usage Notes.
   a. Log in as user to any production system.
   b. CD to any production instance Oracle home.
   c. Invoke asgctl and perform the clone instance command.
> asgctl.sh
Application Server Guard: Release 10.1.3.0.0

(c) Copyright 2004, 2005 Oracle Corporation. All rights reserved
ASGCTL> connect asg prodoc4j oc4jadmin/adminpwd
Successfully connected to prodoc4j:7890
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database asdb
ASGCTL> clone topology to standbyinfra
Generating default policy for this operation
.
.
.

# Command to use if you are using a policy file
# clone topology to standbyinfra using policy <file>
.
.
.
ASGCTL> disconnect
ASGCTL> exit
>
   d. Log off the system
```

connect asg

Connects the OracleAS Guard client to the OracleAS Guard server on a system on which Oracle Application Server services are running.

Format

connect asg [<host-name>[:<port>]] <ias_administrative_account>/<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 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.

port
The port number of the OracleAS Guard server in its Oracle home.

<ias_administrative_account>/password
If this is an OracleAS 10.1.3 installation, the user name must be `oc4jadmin` and the password for the `oc4jadmin` account created during the Oracle Application Server installation. If this is an OracleAS 10.1.2.0.2 or lower installation, 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.

Note: If this is an OracleAS 10.1.3 installation, the user name must be `oc4jadmin` and the password for the `oc4jadmin` account created during the Oracle Application Server 10.1.3 installation.

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 OracleAS Disaster Recovery configuration.
- The specified `ias_admin` or `oc4jadmin` account name must be configured with the necessary rights and privileges to permit OracleAS Disaster Recovery site operations (read and write access to all required files and directories, and so forth)
- An IP address can be used in place of a host name.
- If a password for the `ias_admin` or `oc4jadmin` account is not specified in the connect command, you will be prompted to enter a password.

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>
```

discover topology

Directs asgctl to query Oracle Internet Directory and determine all instances within the topology that share the same Oracle Internet Directory for a production site and generates a topology XML file that describes the topology.

Format

discover topology [oidhost=<host>] [oidsslport=<sslport>] [oiduser=<user>] oidpassword=<pass>

Parameters

host
Name of the host system where Oracle Internet Directory is installed.

sslport
The port number of the host system where Oracle Internet Directory and Secure Sockets Layer (SSL) is installed.

user
The Oracle Internet Directory user name.

pass
The password for the specified Oracle Internet Directory user name.

Usage Notes

You should perform a discover topology operation whenever you procure another Oracle home in a production site or change roles from a production to a standby site through a switchover or failover operation.

Discover topology creates the topology (stored in `topology.xml`) on which to perform all OracleAS Guard operations. This command utilizes the information in Oracle Internet Directory to define the instances included in the topology. Additionally, it gathers local information about each instance. For this reason, it requires all production site instances to have OPMN running. For instances not managed using a DCM farm, the OracleAS Guard service on the Oracle home has to be started. If the services are not started locally, a warning will be produced and the `topology.xml` file will contain only the instances discovered.

See Section 6.1, "Information Common to OracleAS Guard asgctl Commands" and Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information.

Example

The command in the following example discovers all the instances within the topology that share the same Oracle Internet Directory for a production site, and generates a topology XML file that describes the topology.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> discover topology oidpassword=oidpwd
Discovering topology on host "infra" with IP address "123.1.2.111" prodinfra:7890
    Connecting to the OID server on host "infra.us.oracle.com" using SSL port
"636" and username "orcladmin"
    Getting the list of databases from OID
    Gathering database information for SID "asdb" from host "infra.us.oracle.com"
    Getting the list of instances from OID
    Gathering instance information for "asr1012.infra.us.oracle.com" from host
"infra.us.oracle.com"
    Gathering instance information for "asmid1.asmid1.us.oracle.com" from host
"asmid1.us.oracle.com"
    Gathering instance information for "asmid2.asmid2.us.oracle.com" from host
"asmid2.us.oracle.com"
The topology has been discovered. A topology.xml file has been written to each
home in the topology.
ASGCTL>
```

discover topology within farm

Directs asgctl to discover the topology within a farm at a production site for those special cases where a farm does not have Oracle Internet Directory available.

Note: You should always use the discover topology command for discovering the topology for a site because this command uses Oracle Internet Directory to discover all instances in the topology. The discover topology within farm command is useful only in those special cases where Oracle Internet Directory is not available; in this special case OracleAS Guard uses OPMN to discover the topology within a farm.

Format

discover topology within farm

Parameters

None.

Usage Notes

The OracleAS Guard client must be connected to a OracleAS Guard server when you issue this command.

See Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information about topology files.

Example

The command in the following example for a special case in which Oracle Internet Directory is not available, uses OPMN to discover the application server topology within a farm of the OracleAS Guard server to which the OracleAS Guard client is currently connected.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database asdb
ASGCTL> discover topology within farm
Warning: If OID is part of your environment, you should use it for discovery
Discovering topology on host "infra" with IP address "123.1.2.111"
prodinfra:7890
    Discovering instances within the topology using OPMN
    Gathering instance information for "asr1012.infra.us.oracle.com" from host
"infra.us.oracle.com"
The topology has been discovered. A topology.xml file has been written to each
home in the topology.
ASGCTL>
```

dump policies

Directs OracleAS Guard Server to write detailed, default policy information in XML formatted output for the different asgctl commands to a set of policy files located on the local host at the `<ORACLE_HOME>/dsa/conf` directory on UNIX systems or `<ORACLE_HOME>\dsa\conf` directory on Windows systems.

Format

dump policies

Parameters

None.

Usage Notes

A set of XML formatted policy files are written for each of the following asgctl commands: clone topology, dump topology, failover, instantiate topology, sync topology, switchover topology, and verify topology. You can edit the respective command's policy file, then specify it in the `using policy <file>` clause for the appropriate command. This parameter lets you define the topology's disaster recovery policy for each of these OracleAS Guard operations.

For the dump policy file, by default the success requirement attribute is set to optional for all instances (middle tier and OracleAS Metadata Repository).

For the failover policy file, by default the success requirement attribute is set to optional for all instances (middle tier and OracleAS Metadata Repository) and mandatory for the Oracle Internet Directory home.

For the instantiate policy file, by default the success requirement attribute is set to mandatory for all instances.

For the switchover policy file, by default the success requirement attribute is set to optional for all instances (middle tier and OracleAS Metadata Repository) and mandatory for the Oracle Internet Directory home.

For the sync policy file, by default the success requirement attribute is set to mandatory for all instances.

For the verify policy file, by default the success requirement attribute is set to optional for all instances (middle tier and OracleAS Metadata Repository) and mandatory for the Oracle Internet Directory home.

Example

The following example writes detailed, default policy information in XML formatted output for the different asgctl commands to a set of respective policy files located on the local host.

```
ASGCTL> dump policies
Generating default policy for this operation
Creating policy files on local host in directory
"/private1/OraHome2/asr1012/dsa/conf/"
ASGCTL>
```

dump topology

Directs asgctl to write detailed information about the topology to the specified file.

Format

dump topology [to <file>] [using policy <file>]

Parameters

to <file>
Name of file on the OracleAS Guard client node where the detailed output is to be written.

using policy <file>
Full path and file specification for the XML policy file.

Usage Notes

For the dump policy file, by default the success requirement attribute is set to optional for all OracleAS homes (middle tier and OracleAS Metadata Repository).

Example

The following example writes detailed information about the topology to a local file.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database asdb
ASGCTL> dump topology to c:\dump_mid_1.txt

Contents of file c:\dump_mid_1.txt are:

Generating default policy for this operation

 Instance: asr1012.infra.us.oracle.com
    Type: Infrastructure
    Oracle Home Name: asr1012
    Oracle Home Path: /private1/OraHome
    Version: 10.1.2.0.2
    OidHost: infra.us.oracle.com
    OidPort: 389
    VirtualHost: infra.us.oracle.com
    Host: prodinfra
    Ip: 123.1.2.111
    Operation System Arch: sparc
    Operation System Version: 5.8
    Operation System Name: SunOS

 Instance: asmid2.asmid2.us.oracle.com
    Type: Core
    Oracle Home Name: asmid2
    Oracle Home Path: /private1/OraHome2
    Version: 10.1.2.0.2
    OidHost: infra.us.oracle.com
    OidPort: 389
    VirtualHost: asmid2.us.oracle.com
    Host: asmid2
    Ip: 123.1.2.333
    Operation System Arch: sparc
    Operation System Version: 5.8
    Operation System Name: SunOS

 Instance: asmid1.asmid1.us.oracle.com
    Type: Core
    Oracle Home Name: asmid1
    Oracle Home Path: /private1/OraHome
    Version: 10.1.2.0.2
    OidHost: infra.us.oracle.com
    OidPort: 389
    VirtualHost: asmid1.us.oracle.com
    Host: asmid1
    Ip: 123.1.2.334
    Operation System Arch: sparc
    Operation System Version: 5.8
    Operation System Name: SunOS
ASGCTL>
```

The following example writes detailed information about the topology to a local file. Any instances that you want left out of the output can be specified in the policy file.

```
# Command to use if you are using a policy file
ASGCTL> dump topology to c:\dump_mid_1.txt using policy <file>
```

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 on the standby site to make it the primary site.

Format

failover [using policy <file>]

Parameters

using policy <file>
Full path and file specification for the XML policy file.

Usage Notes

Make sure OracleAS Infrastructure database is running on the standby topology 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 OracleAS Disaster Recovery environment. The discovery mechanism automatically maps the topology to the corresponding peer, based off local name resolution.

For the failover policy file, by default the success requirement attribute is set to optional for all OracleAS homes (middle tier and OracleAS Metadata Repository) and mandatory for the Oracle Internet Directory home.

See Section 6.1, "Information Common to OracleAS Guard asgctl Commands" and Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information.

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 new primary database sys/testpwd@asdb
ASGCTL> failover
Generating default policy for this operation
standbyinfra:7890
    Failover each instance in the topology from standby to primary topology
standbyinfra:7890 (home /private1/OraHome2/asr1012)
    Shutting down each instance in the topology
.
.
.
    Executing opmnctl startall command
standbyinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
ASGCTL>
```

```
# Command to use if you are using a policy file
# failover using policy <file>
```

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 commands.

```
ASGCTL> help
    connect asg [<host>] [<ias_administrator_account>/<password>]
    disconnect
    exit
    quit
    add instance <instance_name> on <instance_host> [to topology]
    clone topology to <standby_topology_host> [using policy <file>] [no standby]
    clone instance <instance> to <standby_topology_host> [no standby]
    discover topology [oidhost=<host>] [oidsslport=<sslport>] [oiduser=<user>] oidpassword=<pass>
    discover topology within farm
    dump farm [to <file>]  (Deprecated)
    dump topology  [to <file>] [using policy <file>]
    dump policies
    failover [using policy <file>]
    help [<command>]
    instantiate farm to <standby_farm_host> (Deprecated)
    instantiate topology to <standby_topology_host> [using policy <file>]
    remove instance <instance_name> [from topology]
    set asg credentials <host> <ias_administrator_account>/<password> [for topology]
    set asg credentials <host> ias_admin/<password> [for farm] (Deprecated)
    set primary database <username>/<password>@<servicename> [pfile <filename> | spfile <filename>]
    set new primary database <username>/<password>@<servicename> [pfile <filename> | spfile <filename>]
    set noprompt
    set trace on|off <traceflags>
    sync farm to <standby_farm_host> [full | incr[emental]] (Deprecated)
    sync topology to <standby_topology_host> [full | incr[emental]] [using policy <file>]
    startup [asg]
    startup farm (Deprecated)
    startup topology
    shutdown [local]
    shutdown farm (Deprecated)
    shutdown topology
    show op[eration] [full] [[his]tory]
    show env
    stop op[eration] <op#>
    switchover farm to <standby_farm_host> (Deprecated)
    switchover topology to <standby_topology_host> [using policy <file>]
    verify farm [with <host>](Deprecated)
    verify topology [with <host>] [using policy <file>]
ASGCTL>
```

instantiate topology

Instantiates a topology to a standby site by establishing the relationship between standby and production instances, mirroring the configuration, creating the standby Infrastructure, and then synchronizing the standby site with the primary site.

Format

instantiate topology to <standby_topology_host>[:<port>] [with cloning] [using policy <file>]

Parameters

standby_topology_host
Name of the standby host system. 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 topology.

port
The port number of the OracleAS Guard server in its Oracle home.

with cloning
A directive to perform an instantiation operation using cloning.

using policy <file>
Full path and file specification for the XML policy file.

Usage Notes

Make sure OracleAS Infrastructure database is running on the primary topology before performing an instantiate topology 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 OracleAS Disaster Recovery environment. The discovery mechanism automatically maps the topology to the corresponding peer, based off local name resolution.

The instantiate operation performs an implicit verify operation.

For the instantiate policy file, by default the success requirement attribute is set to mandatory for all instances.

See Section 6.1, "Information Common to OracleAS Guard asgctl Commands" and Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information.

Example

The following example instantiates a standby topology by attaching the coordinating OracleAS Guard server and discovering the topology of the production and standby sites, performing site verification, and establishing a OracleAS Disaster Recovery environment with the topology containing the standby topology 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> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
```

```
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database asdb
ASGCTL> instantiate topology to standbyinfra
Generating default policy for this operation
prodinfra:7890
    Instantiating each instance in the topology to standby topology
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
standbyinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
asmid2:7890
    Verifying that the topology is symmetrical in both primary and standby configuration
.
.
.
This operation requires the database to be shutdown. Do you want to continue? Yes or No
y
.
.
.
asmid2:7890 (home /private1/oracle/asr1012)
    Starting backup/synchronization of database "orcl.us.oracle.com"
    Starting restore/synchronization of database "orcl.us.oracle.com"
    Synchronizing topology completed successfully
asmid2:7890
    Synchronizing topology completed successfully

ASGCTL>

# Command to use if you are using a policy file
# instantiate topology to standbyinfra using policy <file>
```

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
>
```

remove instance

Removes from the local topology file, the specified instance name, and if specified, propagates this updated topology file to all instances in the Disaster Recovery production environment.

Format

```
remove instance <instance_name> [from topology]
```

Parameters

instance_name
The name of the instance to be removed from the topology file.

from topology
A keyword, that if present in the command line, directs OracleAS Guard to propagate the updated topology file to all instances in the Disaster Recovery production environment. Any OracleAS Guard operation that affects the standby site, such as verify, instantiate, sync, and switchover will automatically propagate the production topology file across the standby environment.

Usage Notes

This command is useful for managing an instance on an OracleAS Guard server to which the OracleAS Guard client is connected. For example, you may have used the remove instance command to remove from the local topology file or from all topology files within the topology, an instance on this local host system because of a problem with it. Now you want to add to the local topology file or to all topology files in the topology, the good instance. In this case, you may not have wanted to manage the bad instance through the policy file where you could have set the success requirement attribute to `Ignore` for this instance when invoking asgctl commands to run across the entire topology.

This command is particularly useful for managing Disaster Recovery farms in which OID is not available, in other words, an OracleAS Release 10.1.3 only topology. You must use the discover topology within farm command to initially create the topology file for each instance within this farm. Then you can manage instances by adding or removing individual instances from the local topology file using the add instance and remove instance commands. If you specify the `to topology` or `from topology` keywords, the updated local topology file changes are propagated to all instances in the Disaster Recovery environment. See Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information about topology files.

This command is useful for adding an OracleAS 10.1.3 J2EE instance to an OID based 10.1.2.0.2 topology to support a mixed version Disaster Recovery environment. For example, you can use the add instance command to add an OracleAS 10.1.3 J2EE instance to your OID based 10.1.2.0.2 topology. See Section 5.9, "Adding or Removing OracleAS 10.1.3 Instances to Redundant Single OracleAS 10.1.3 Home J2EE Topology Integrated with an Existing Oracle Identity Management 10.1.2.0.2 Topology" for a use case. See Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information about topology files.

This command is also useful for supporting mixed topologies for any OracleAS release from Release 9.0.4 upwards through Release 10.1.3. The only requirement is that the Release 10.1.3 OracleAS Guard standalone kit must be installed on all systems in your

Release 9.0.4 Disaster Recovery environment. See Chapter 8, "OracleAS Disaster Recovery Site Upgrade Procedure" for more information.

Example

The following command in the example removes from the local topology file an instance named oc4j1.

```
ASGCTL> remove instance oc4j1
```

set asg credentials

Sets the credentials used to authenticate the OracleAS Guard connections to OracleAS Guard servers.

Format

```
set asg credentials <host>[:<port>] <ias_administrative_account>/<password> [for farm] [for topology]
```

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 OracleAS Guard server in its Oracle home.

<ias_administrative_account>/password
If this is an OracleAS 10.1.3 installation, the user name must be `oc4jadmin` and the password for the `oc4jadmin` account created during the Oracle Application Server 10.1.3 installation. If this is an OracleAS 10.1.2.0.2 or lower installation, 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 (deprecated)
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.

for topology
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 topology 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 topology. For example, you may want to use a common set of credentials in the standby topology that is different from the credentials used in the primary topology.

If you set the credentials for a topology, these credentials are inherited for the entire topology. If you set the credentials for an individual host on the topology, the credentials (for this host) override the default credentials set for the topology.

For topologies that have more than one Infrastructure, such as a collocated Oracle Internet Directory+OracleAS Metadata Repository and a separate Portal OracleAS Metadata Repository, OracleAS Guard requires that you set the credentials for each system on which an Infrastructure resides before performing any important OracleAS Guard operations, such as instantiate, sync, switchover, and failover. This is actually a two step process in which you must first identify all OracleAS Infrastructure databases on the topology using the set the primary database command for each Infrastructure, then you must set the credentials used to authenticate the OracleAS Guard connections to OracleAS Guard servers on which these Infrastructures reside. The following example illustrates this concept. Assume your production topology and standby topology consists of the following systems with installed Infrastructure and middle tier software applications.

Production topology:

host01 (Identity Management+OracleAS Metadata Repository), host04 (OracleAS Metadata Repository only), host06 (J2EE), host06 (Portal & Wireless)

Standby Topology:

host02 (Identity Management+OracleAS Metadata Repository), host05 (OracleAS Metadata Repository only), host07 (J2EE), host07 (Portal & Wireless)

The following OracleAS Guard set primary database and set asg credentials commands would be required to properly identify the Infrastructures and authenticate OracleAS Guard connections to OracleAS Guard servers prior to performing an instantiate, sync, switchover, or failover operation. Assuming that the Oracle Identity Management+OracleAS Metadata Repository Infrastructure has a service name of `orcl` and the separate Portal OracleAS Metadata Repository has a service name of `asdb`.

```
ASGCTL> set primary database sys/<password>@orcl.us.oracle.com
ASGCTL> set primary database sys/<password>@asdb.us.oracle.com
ASGCTL> set asg credentials host01.us.oracle.com ias_admin/<password>
ASGCTL> set asg credentials host04.us.oracle.com ias_admin/<password>
```

Note that for a failover operation, these steps would be carried out on the standby topology and are as follows with a change in the host system names:

```
ASGCTL> set primary database sys/<password>@orcl.us.oracle.com
ASGCTL> set primary database sys/<password>@asdb.us.oracle.com
ASGCTL> set asg credentials host02.us.oracle.com ias_admin/<password>
ASGCTL> set asg credentials host05.us.oracle.com ias_admin/<password>
```

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.

See Section 6.1, "Information Common to OracleAS Guard asgctl Commands" and Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information.

Example

The following example sets the OracleAS Guard credentials of host system standbyinfra to all host systems that belong to this topology.

```
ASGCTL> set asg credentials standbyinfra ias_admin/<password> for topology
```

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 topology, 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 are not connected
disconnect

# not connected, should get an error message
dump topology

# connect to an ASG server
connect asg prodinfra ias_admin/adminpwd

#display detailed info about the topology
dump topology

#disconnect
disconnect

# turn off echo
echo off
exit
```

set new primary database

Identifies the OracleAS Infrastructure database on the standby topology 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

Before performing a failover operation, you are required to connect to the Infrastructure node of the standby topology 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 topology as the new primary/production topology 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 noprompt

Sets the noprompt state for user interaction for use in executing commands in an asgctl script.

Format

set noprompt

Parameters

None

Usage Notes

The default value, if supplied, is taken for all interactive prompts. A prompt for a user name and password returns an error message in the noprompt state.

Example

The following example is an asgctl script containing an asgctl set noprompt command part way through the script that thereafter ignores all subsequent interactive prompting.

```
> ASGCTL @myasgctltestscript.txt

# myasgctltestscript.txt

# connect to an ASG server
connect asg prodinfra ias_admin/adminpwd

# set the primary database
set primary database sys/testpwd@asdb

# discover the production topology
discover topology oidpassword=oidpwd

# set the noprompt state
set noprompt

#display detailed info about the topology
dump topology

#disconnect
disconnect

exit
```

set primary database

Identifies the OracleAS Infrastructure database on the primary topology.

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

You must always set the primary database before performing an instantiate, sync, or switchover operation.

When you set the primary database, OracleAS Guard server logs into and validates the connection to the database.

If a production or standby site has multiple OracleAS Metadata Repository instances installed and you are performing an instantiate, sync, switchover, or failover operation, you must identify all of the OracleAS Metadata Repository instances by performing a set primary database command for each and every OracleAS Metadata Repository instance prior to performing either an instantiate, sync, switchover, or failover operation. In addition, for topologies that have more than one Infrastructure, such as a collocated Oracle Internet Directory+OracleAS Metadata Repository and a separate Portal OracleAS Metadata Repository, OracleAS Guard requires that you set the credentials for each system on which an Infrastructure resides before performing any important OracleAS Guard operations, such as instantiate, sync, switchover, and failover. See set asg credentials for an example.

OracleAS Guard requires the database to have password file authentication. If the database does not have a password file, you must use the `orapwd` utility to create a password file. Also, set the `REMOTE_LOGIN_PASSWORDFILE` initialization parameter to `EXCLUSIVE`.

See Section 6.1, "Information Common to OracleAS Guard asgctl Commands" and Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information.

Example

The following example sets the OracleAS Infrastructure database information for the primary or production topology.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database asdb
ASGCTL>
```

The following example sets OracleAS Infrastructure database information for each OracleAS Metadata Repository installed for the primary/production topology prior to a switchover operation.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> set primary database sys/testpwd@portal_1
Checking connection to database portal_1
ASGCTL> set primary database sys/testpwd@portal_2
Checking connection to database portal_2
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database asdb
ASGCTL> discover topology oidpassword=oidpwd
ASGCTL> switchover topology to standbyinfra
.
.
.
```

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
- CLIPBOARD -- trace information regarding clipboard processing
- COPY -- trace information regarding file copy processing
- FLOW -- trace information regarding work flow processing
- NET -- trace information regarding network processing
- RUNCMD -- trace information regarding the running of external commands
- SESSION -- trace information regarding session management
- TOPOLOGY -- trace information regarding processing of topology information

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 env

Shows the current environment for the OracleAS Guard server to which the OracleAS Guard client is connected.

Format

show env

Parameters

None.

Usage Notes

None.

Example

The following examples show the environment of the OracleAS Guard server to which the OracleAS Guard client is connected. In the first example, the primary database and new primary database are not yet set on host prodinfra and in the second example, the primary database has already been set on host standbyinfra.

Example 1.

```
ASGCTL> show env

     ASG Server Connection:
        Host: prodinfra
        Port: 7890

     Primary database: <not set>
     New primary database:  <not set>
```

Example 2.

```
ASGCTL> ASGCTL> show env

     ASG Server Connection:
        Host: standbyinfra
        Port: 7890

Gathering information from the database orcl

     Primary database: :
        User: sys
        Service: orcl
        Role: The database role is
               PHYSICAL STANDBY


     New primary database:  <not set>
```

show operation

Shows all operations on all nodes of the topology to which the OracleAS Guard client is connected for the current session.

Format

show op[eration] [full] [[his]tory]

Parameters

full
For all operations, shows the operation number, the job name, the job owner's user name, the job ID, the time the operation began, the time the operation ended, the elapsed time for the operation, and 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
*************************************
OPERATION: 19
  Status: running
  Elapsed Time: 0 days, 0 hours, 0 minutes, 28 secs
  TASK: syncFarm
    TASK: backupFarm
      TASK: fileCopyRemote
      TASK: fileCopyRemote
    TASK: restoreFarm
      TASK: fileCopyLocal
```

The following example shows the history of all operations.

```
ASGCTL> show op his
*************************************
OPERATION: 7
  Status: success
  Elapsed Time: 0 days, 0 hours, 0 minutes, 0 secs
  TASK: getTopology
    TASK: getInstance
*************************************
OPERATION: 16
  Status: success
  Elapsed Time: 0
 days, 0 hours, 0 minutes, 0 secs
  TASK: getTopology
    TASK: getInstance
*************************************
OPERATION: 19
```

```
Status: success
Elapsed Time: 0 days, 0 hours, 1 minutes, 55 secs
TASK: syncFarm
  TASK: backupFarm
    TASK: fileCopyRemote
    TASK: fileCopyRemote
  TASK: restoreFarm
    TASK: fileCopyLocall
```

shutdown

Shuts down a running OracleAS Guard server to which the OracleAS Guard client is connected. Use this command only on a host system where OPMN is not running and you are following the procedure to clone an instance or clone a topology.

Format

shutdown [local]

Parameters

local
When specified shuts down the OracleAS Guard server of the local Oracle home of asgctl.

Usage Notes

The OracleAS Guard server must have been started using the asgctl startup command and not the OPMN opmnctl command startproc.

Example

The following example shuts down the OracleAS Guard server on a host system in which OPMN is not running.

```
> asgctl.sh shutdown
```

shutdown topology

Shuts down the OracleAS component services across the topology, while OracleAS Guard server and OPMN will continue to run.

Format

shutdown topology

Parameters

None.

Usage Notes

This is a convenient command for shutting down the entire topology. Use the startup topology command to start it up again.

This command will shutdown OracleAS services such as OID, OC4J, WebCache, and so forth.

Example

The following example shuts down the prodinfra production topology.

```
ASGCTL> shutdown topology
Generating default policy for this operation

prodinfra:7890
    Shutting down each instance in the topology

asmid2:7890 (home /private1/OraHome2/asmid2)
    Shutting down component HTTP_Server
    Shutting down component OC4J
    Shutting down component dcm-daemon
    Shutting down component LogLoader

asmid1:7890 (home /private1/OraHome/asmid1)
    Shutting down component HTTP_Server
    Shutting down component OC4J
    Shutting down component dcm-daemon
    Shutting down component LogLoader

prodinfra:7890 (home /private1/OraHome2/asr1012)
    Shutting down component OID
    Shutting down component HTTP_Server
    Shutting down component OC4J
    Shutting down component dcm-daemon
    Shutting down component LogLoader
ASGCTL>
```

startup

Starts up an OracleAS Guard server from the asgctl prompt. Use this command only on a host system where OPMN is not running and you are following the procedure to clone an instance or clone a topology.

Format

startup [asg]

Parameters

asg
Optional keyword acronym for application server guard. This parameter has no other meaning other than to show a format similar to the connect asg and set asg credentials commands.

Usage Notes

None.

Example

The following example shuts down the OracleAS Guard server on a host system in which OPMN is not running.

```
> asgctl.sh startup
```

startup topology

Starts up a shutdown topology by starting up the OracleAS component services across the topology.

Format

startup topology

Parameters

none

Usage Notes

This is a convenient command for starting up the entire topology after it was shut down using the shutdown topology command.

This command will start up OracleAS services such as OID, OC4J, WebCache, and so forth. The startup topology command will perform the equivalent of an opmnctl startup command across each instance of the topology.

Example

The following example starts up the production topology.

```
ASGCTL> startup topology
Generating default policy for this operation

profinfra:7890
    Starting each instance in the topology

prodinfra:7890 (home /private1/OraHome2/asr1012)
    Executing opmnctl startall command

asmid1:7890 (home /private1/OraHome/asmid1)
    Executing opmnctl startall command

asmid2:7890 (home /private1/OraHome2/asmid2)
    Executing opmnctl startall command
ASGCTL>
```

stop operation

Stops a specific operation that is running on the server.

Format

stop op[eration] <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 topology

During a scheduled outage of the production site, performs the switchover operation from the production site to the standby site.

Format

switchover topology to <standby_topology_host>[:<port>] [using policy <file>]

Parameters

standby_topology_host
Name of the standby host system. 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 topology.

port
The port number of the standby host system for the OracleAS Guard server in its Oracle home.

using policy <file>
Full path and file specification for the XML policy file.

Usage Notes

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 process 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.

Make sure OracleAS Infrastructure database is running on the primary topology 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 OracleAS Disaster Recovery 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 topology operation is performed to make sure the topologies 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 topology and each of these is a transient server.

For the switchover policy file, by default the success requirement attribute is set to optional for all instances (middle tier and OracleAS Metadata Repository) and mandatory for the Oracle Internet Directory home.

During a switchover operation, the `opmn.xml` file is copied from the primary site to the standby site. For this reason, the value of the TMP variable must be defined the same in the `opmn.xml` file on both the primary and standby sites, otherwise this switchover operation will fail with a message that it could not find a directory. Therefore, make sure the TMP variable is defined identically and resolves to the same directory structure on both sites before attempting a switchover operation.

When performing a switchover operation from a primary site with two Oracle Identity Management instances running (im.machineA.us.oracle.com and im.machineB.us.oracle.com) to a standby site representing an asymmetric topology with only one Oracle Identity Management instance running (im.machineA.us.oracle.com), meaning that the other node (im.machineB.us.oracle.com) is to be ignored on the switchover site, the system administrator must not only edit the `switchover_policy.xml` policy file to indicate that this other node is to be set to Ignore, but the system administrator must also shut down all processes running on that node (im.machineB.us.oracle.com) in order for the switchover operation to be successful.

When performing a switchover operation from a primary site with two middle tiers, for example core1 and core2 instances registered in the Oracle Internet Directory, to a standby site representing an asymmetric topology with only one middle tier core1, the standby site actually has both core1 and core1 middle tiers registered in the Oracle Internet Directory. The `switchover_policy.xml` policy file is edited to ignore the core2 middle tier that does not exist on the standby site during the switchover operation. However, it should be noted that the Oracle Internet Directory, which is stored in an Oracle database, is identical for both the production site topology and the standby site topology and therefore a core2 middle tier is also shown to be registered in the Oracle Internet Directory on the standby site topology. For this reason, you cannot install to that standby site topology the same core2 middle tier with the hope of making this into a symmetric topology again. This is a strict limitation for switchover operations using asymmetric standby topologies.

When the discover topology command is issued following a switchover operation and the asymmetric standby site topology originally had one or more fewer middle tiers (for example, instA and instB) than there were in the original production site topology (instA, instB, and instC), a warning error message displays for each missing instance of a middle tier (instC, in this case). This warning error message is expected and can be ignored. When a discover topology to command is issued following a switchover operation, OracleAS Server Guard reads the Oracle Internet Directory information, which is an exact copy of the original primary site Oracle Internet Directory information on this new primary site (former standby site). Because this Oracle Internet Directory information is identical to the original primary site Oracle Internet Directory information, when OracleAS Server Guard visits the host/home of each instance of these middle tiers to verify their existence, it finds that some do not exist, and issues the warning.

See Section 6.1, "Information Common to OracleAS Guard asgctl Commands" and Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information.

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 topology to standbyinfra
Generating default policy for this operation
prodinfra:7890
    Switchover each instance in the topology to standby topology
prodinfra:7890 (home /private1/OraHome2/asr1012)
    Connecting to the primary database asdb.us.oracle.com
    Gathering information from the primary database asdb.us.oracle.com
    Shutting down each instance in the topology
.
.
.
prodinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
standbyinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
prodinfra:7890
    Verifying that the topology is symmetrical in both primary and standby configuration
ASGCTL>

# Command to use if you are using a policy file
# switchover topology to standbyinfra using policy <file>
```

sync topology

Synchronizes the standby site with the primary site to ensure that the two sites are consistent. The sync topology operation applies database redo logs for OracleAS Infrastructures to the standby site in conjunction with synchronizing external configuration files across the topology.

Format

```
sync topology to <standby_topology_host>[:<port>] [full | incr[emental]] [using policy <file>]
```

Parameters

standby_topology_host
Name of the standby site host system. 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 topology.

port
The port number of the standby host system for the OracleAS Guard server in its Oracle home.

full | incremental
The synchronization of the standby site with the primary site to make the standby site consistent can be either "full" or "incremental". The default is "incremental". By default, if a full backup has not been performed, an incremental backup operation will not be performed. Instead, a full backup operation will be performed.

using policy <file>
Full path and file specification for the XML policy file.

Usage Notes

By default an incremental synchronization is performed to make the standby site consistent with the primary site, which offers the best performance. However, there may be three circumstances when specifying a full synchronization 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.

The sync operation performs an implicit verify operation.

For the sync policy file, by default the success requirement attribute is set to mandatory for all instances.

See Section 6.1, "Information Common to OracleAS Guard asgctl Commands" and Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information.

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> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database asdb
ASGCTL> sync topology to standbyinfra
Generating default policy for this operation
prodinfra:7890
    Synchronizing each instance in the topology to standby topology
prodinfra:7890 (home /private1/OraHome2/asr1012)
    Starting backup of topology ""
       Backing up and copying data to the standby topology
    Backing up each instance in the topology
    Starting backup of instance "asr1012.infra.us.oracle.com"
    Configuring the backup script
asmid1:7890 (home /private1/OraHome/asmid1)
    Starting backup of instance "asmid1.asmid1.us.oracle.com"
asmid2:7891 (home /private1/OraHome/asmid2)
    Starting backup of instance "asmid2.asmid2.us.oracle.com"
.
.
.
asmid2:7890 (home /private1/OraHome2/asr1012)
    Starting backup/synchronization of database "asdb.us.oracle.com"
    Starting restore/synchronization of database "asdb.us.oracle.com"
    Synchronizing topology completed successfully
ASGCTL>

# Command to use if you are using a policy file
# sync topology to standbyinfra using policy <file>
```

verify topology

Validates that the primary topology is running and the configuration is valid. If a standby topology is specified, compares the primary topology to which the local host system is a member with the standby topology to validate that they are consistent with one another and conform to the requirements for OracleAS Disaster Recovery.

Format

verify topology [with <host>[:<port>]] [using policy <file>]

Parameters

host
Name of the standby host system. This host system must be a member of the standby topology.

port
The port number of the host system for the OracleAS Guard server in its Oracle home.

using policy <file>
Full path and file specification for the XML policy file.

Usage Notes

If the host system name is not specified, the topology 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 topology at the standby site will be verified along with the production topology for both local rules and distributed OracleAS Disaster Recovery rules, and the symmetry between the primary and standby sites is also checked.

For the verify policy file, by default the success requirement attribute is set to optional for all OracleAS homes (middle tier and OracleAS Metadata Repository) and mandatory for the Oracle Internet Directory home.

See Section 6.1, "Information Common to OracleAS Guard asgctl Commands" and Section 6.2, "Information Specific to a Small Set of OracleAS Guard Commands" for more information.

Example

The following example validates that the primary topology is running and the configuration is valid.

```
ASGCTL> connect asg ias_admin/iastest2
Successfully connected to prodinfra:7890
ASGCTL> verify topology
Generating default policy for this operation
prodinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
ASGCTL>
```

The following example validates that the topology to which the local host system is a member is consistent with the standby topology to which the host system standbyinfra is a member.

```
ASGCTL> connect asg prodinfra ias_admin/adminpwd
Successfully connected to prodinfra:7890
ASGCTL> set primary database sys/testpwd@asdb
Checking connection to database asdb
ASGCTL> verify topology with standbyinfra
Generating default policy for this operation
prodinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
standbyinfra:7890
     HA directory exists for instance asr1012.infra.us.oracle.com
asmid2:7890
     HA directory exists for instance asmid2.asmid2.us.oracle.com
asmid1:7890
     HA directory exists for instance asmid1.asmid1.us.oracle.com
prodinfra:7890
    Verifying that the topology is symmetrical in both primary and standby configuration
ASGCTL>

# Command to use if you are using a policy file
# verify topology using policy <file>
```

dump farm (Deprecated)

Directs asgctl to write detailed information about the farm to the specified file.

Note: The dump farm command is deprecated beginning with OracleAS release 10.1.2.0.2. Use the dump topology command, which supports the OracleAS Disaster Recovery topology concept in current and future OracleAS releases.

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

None.

Example

See the dump topology command for an example.

instantiate farm (Deprecated)

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 so that the primary and standby sites are consistent.

Note: The instantiate farm to command is deprecated beginning with OracleAS release 10.1.2.0.2. Use the instantiate topology command, which supports the OracleAS Disaster Recovery topology concept in current and future OracleAS releases.

Format

```
instantiate farm to <standby_farm_host>[:<port>]
```

Parameters

standby_farm_host
Name of the standby host system. 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 OracleAS Guard server in its Oracle home.

Usage Notes

The production local system must be part of an Oracle Notification Server (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 farm 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 OracleAS Disaster Recovery environment. The discovery mechanism automatically maps the farm to the corresponding peer, based off local name resolution.

Example

See the instantiate topology command for an example.

shutdown farm (Deprecated)

Shuts down a running farm.

Note: The shutdown farm command is deprecated beginning with OracleAS release 10.1.2.0.2. Use the shutdown topology command, which supports the OracleAS Disaster Recovery topology concept in current and future OracleAS releases.

Format

```
shutdown farm
```

Parameters

None.

Usage Notes

This is a convenient command for shutting down the entire farm. Use the startup farm command to start it up again.

Example

See the shutdown topology command for an example.

startup farm (Deprecated)

Starts up a shutdown farm.

Note: The startup farm command is deprecated beginning with OracleAS release 10.1.2.0.2. Use the startup topology command, which supports the OracleAS Disaster Recovery topology concept in current and future OracleAS releases.

Format

```
startup farm
```

Parameters

None

Usage Notes

This is a convenient command for starting up the entire farm after it was shut down using the shutdown farm command.

Example

See the startup topology command for an example.

switchover farm (Deprecated)

During a scheduled outage of the production site, performs the switchover operation from the production site to the standby site.

Note: The switchover farm to command is deprecated beginning with OracleAS release 10.1.2.0.2. Use the switchover topology command, which supports the OracleAS Disaster Recovery topology concept in current and future OracleAS releases.

Format

switchover farm to <standby_farm_host>[:<port>]

Parameters

standby_farm_host
Name of the farm host system. 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 for the OracleAS Guard server in its Oracle home.

Usage Notes

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 process 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 production local system must be part of an Oracle Notification Server (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 farm 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 OracleAS Disaster Recovery environment. The discovery mechanism automatically maps the farm 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. In turn, it starts the OracleAS Guard server on the other nodes in the "new" standby farm and each of these is a transient server.

Example

See the switchover topology command for an example.

sync farm (Deprecated)

Synchronizes the standby site with the primary site to ensure that the two sites are consistent. The sync topology operation applies database redo logs for OracleAS Infrastructures to the standby site in conjunction with synchronizing external configuration files across the topology.

Note: The sync farm to command is deprecated beginning with OracleAS release 10.1.2.0.2. Use the sync topology command, which supports the OracleAS Disaster Recovery topology concept in current and future OracleAS releases.

Format

```
sync farm to <standby_farm_host>[:<port>] [full | incr[emental]]
```

Parameters

standby_farm_host
Name of the standby site host system. 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 for the OracleAS Guard server in its Oracle home.

full | incremental
The synchronization of the standby site with the primary site to make the standby site consistent can be either "full" or "incremental". The default is "incremental". By default, if a full backup has not been performed, an incremental backup operation will not be performed. Instead, a full backup operation will be performed.

Usage Notes

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 is 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

See the sync topology command for an example.

verify farm (Deprecated)

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.

Note: The verify farm command is deprecated beginning with OracleAS release 10.1.2.0.2. Use the verify topology command, which supports the OracleAS Disaster Recovery topology concept in current and future OracleAS releases.

Format

```
verify farm [with <host>[:<port>]]
```

Parameters

host
Name of the standby host system. This host system must be a member of the standby farm.

port
The port number of the OracleAS Guard server in its Oracle home.

Usage Notes

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, and the symmetry between the primary and standby sites is also checked.

Example

See the verify topology command for an examples.

7 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 7.1.1, "Manually Backing Up the Production Site". Then you will need to restore the backup configuration files as described in Section 7.1.2, "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 "Site Failover Operations" on page 5-47.

7.1 Manually Synchronizing Baseline Installation with Standby Site Without Using OracleAS Guard asgctl Command-line Utility

Note: This section and Section 7.1.1, "Manually Backing Up the Production Site" and Section 7.1.2, "Manually Restoring to Standby Site" are retained here for the special case as described in Step 1b in "Site Failover Operations" on page 5-47 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 7.1.2, "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 "Site Failover Operations" on page 5-47 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 Recovery Manager. 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.

7.1.1 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.

7.1.1.1 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.

7.1.1.2 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 OracleAS Recovery Manager. They assume you have installed and configured OracleAS Recovery Manager 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 OracleAS Recovery Manager, 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 Recovery Manager, the variables `oracle_home`, `log_path`, and `config_ backup_path` in the OracleAS Recovery manager's configuration file, `config.inp`, should have the appropriate values. Also, the following command for the OracleAS Recovery Manager 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 Recovery Manager'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.

7.1.2 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

7.1.2.1 Restoring Configuration Files (OracleAS Infrastructure and Middle Tier)

Restoring the backed up files from the production site requires the OracleAS Recovery Manager that was used for the backup. The instructions in this section assume you have installed and configured the OracleAS Recovery Manager 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 OracleAS Recovery Manager.

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 Recovery Manager'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 Recovery Manager 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 Recovery Manager 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).

7.1.2.2 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.

8 OracleAS Disaster Recovery Site Upgrade Procedure

This chapter describes how to complete a full site Oracle Application Server Disaster Recovery (OracleAS Disaster Recovery) upgrade from OracleAS 10g (9.0.4) to OracleAS 10g (10.1.2.0.2). This procedure assumes that a successful release 9.0.4 to release 10.1.2 upgrade is possible for all the Oracle home types within the topology that define the site and extends these procedures in the OracleAS Disaster Recovery (DR) solution.

This site upgrades an existing supported DR implementation as documented in the Oracle Application Server Disaster Recovery chapter in *Oracle Application Server 10g High Availability Guide* for OracleAS 10g (9.0.4). This process will not upgrade a non-supported DR environment into an upgraded DR environment. Additionally, this procedure will utilize the standalone OracleAS Guard install within the existing release 9.0.4 Oracle homes and is worded using OracleAS Guard operational steps. If this environment is not possible, the equivalent manual steps can be performed, that is, the OracleAS Guard `sync topology` command equates to the site synchronization steps documented in the Oracle Application Server Disaster Recovery chapter in *Oracle Application Server 10g High Availability Guide* for OracleAS 10g (9.0.4).

8.1 Prerequisites

The following are prerequisites for performing a full DR site upgrade from OracleAS 10g (9.0.4) to OracleAS 10g (10.1.2.0.2):

- You must have a DR site configured according to the guidelines in Chapter 5, "OracleAS Disaster Recovery".
- The OracleAS Recovery Manager (formerly called OracleAS Backup/Restore utility) is installed in all Oracle homes of the both the production and standby sites. The Backup/Restore utility version to be used is the version that supports that release. For example, if you are performing a full Disaster Recovery site upgrade from Oracle 10g (9.0.4) to OracleAS 10g (10.1.2.0.2), then the Backup/Restore utility version must be OracleAS 10g (9.0.4).
- The OracleAS 10g (10.1.2.0.2) standalone install of OracleAS Guard, located on CDROM Disk 2, is installed in all OracleAS 10g (9.0.4) Oracle homes. See the OracleAS Disaster Recovery installation information in *Oracle Application Server Installation Guide* for more information.

8.2 Disaster Recovery Topology

The systems involved in this DR environment are contained in two sites, site A and site B. The initial roles of each are:

- Site A is the production site.
- Site B is the standby site.

Due to geographical separation of the sites, it is assumed that the current roles of each of these sites will be the final roles of these same sites at the end of this procedure. However, during the course of the procedure these roles do change. Thus, all references will be to the sites named A and B. Some of the terminology used may be confusing, depending on the role the site is maintaining at a particular point in time.

8.3 High-Level OracleAS Disaster Recovery Upgrade Steps

The following steps describe the OracleAS 10g (9.0.4) to OracleAS 10g (10.1.2.0.2) Disaster Recovery upgrade scenario. These steps refer to Infrastructure systems `infra1` and `infra2` on site A and site B, respectively.

1. Install the OracleAS 10g (10.1.2.0.2) standalone install of OracleAS Guard into each Oracle home on the production and standby sites.

   If multiple Oracle homes exist on the same system, ensure that different ports are configured for each of the OracleAS Guard servers in this configuration file. The default port number is 7890.

   ```
   <ORACLE_HOME>/dsa/dsa.conf
   ```

2. At the standby site [site B], start the OracleAS Guard server:

   ```
   <ORACLE_HOME>/opmn/bin/opmnctl startproc ias-component=DSA
   ```

3. At the production site [site A], connect to OracleAS Guard Infrastructure system `infra1` and perform a sync operation.

   This operation is used to ensure that the Oracle homes across the topology are logically synchronized.

   a. Invoke the asgctl client.

      ```
      On Unix systems
      <ORACLE_HOME>/dsa/bin/asgctl.sh

      On Windows systems
      <ORACLE_HOME>\dsa\bin\asgctl
      ```

   b. Perform a connect operation to site A Infrastructure system `infra1`.

      ```
      ASGCTL> connect asg infra1 ias_admin/<password>
      ```

   c. Set the primary database to the OracleAS Metadata Repository at site A.

      ```
      ASGCTL> set primary database sys/<password>@<site A's servicename>
      ```

   d. Discover the topology.

      ```
      ASGCTL> discover topology oidpassword=<oidpwd>
      ```

   e. Dump the topology.

      ```
      ASGCTL> dump topology to c:\policy_file_no_904_instances.txt
      ```

f. Edit the topology file, in this example named `policy_file_no_904_instances.txt` and set all the 9.0.4 instances to Ignore, for example:

```
<policy>
.
.
.
<instanceList successRequirement="Ignore">
   <instance>904Portal_3</instance>
</instanceList>
.
.
.
</policy>
```

   Then use this policy file for all Disaster Recovery related operations.

g. Synchronize the standby site B Infrastructure system `infra2` with the production site using the edited policy file from Step 3f.

```
ASGCTL> sync topology to infra2 using policy c:\policy_file_no_904_
instances.txt
```

h. Ensure there are no changes to the environment through the duration of the upgrade procedure. Note that this does not mean changes to customer data, as this will be in a different database than the Identity Management (IM)/Metadata Repository (MR) data. However, in this model, the IM/MR data will not be able to be synchronized again during the upgrade procedure.

4. Connect to OracleAS Guard at the standby [site B] Infrastructure and failover.

   The purpose of this step is to break the DR environment into two independent sites. This allows site B to be upgraded first. Once site B is upgraded, application level tests can be performed to ensure that the update was completed and that this site is operational. If you use this approach, then site A, production, is not really DR tolerant for the time period of the upgrade. Theoretically, another standby site could be established at this time as site B was upgraded.

   The steps to follow to perform the OracleAS Guard failover operation are:

   a. Perform a connect operation to site B Infrastructure system `infra2`.

   ```
   ASGCTL> connect asg infra2 ias_admin/<password>
   ```

   b. Set the new primary database to the OracleAS Metadata Repository at site B.

   ```
   ASGCTL> set new primary database sys/<password>@<site B's servicename>
   ```

   c. Perform the failover operation to this standby site, site B. The failover operation will start all the OPMN managed services across the topology equivalent of an opmnctl `startall` command.

   ```
   ASGCTL> failover using policy c:\policy_file_no_904_instances.txt
   ```

5. Start the other services for the site at site B.

   Any additional services needed for testing must be handled manually, such as applications, database jobs, Enterprise Manager, and so forth.

6. Perform an OracleAS upgrade to the site B systems [see *Oracle Application Server Upgrade and Compatibility Guide* for more information].

7. Test applications or note problems for resolution for the production site. Perform tests until you are satisfied the upgrade has been properly completed.

8. Redirect site access to site B, if desirable.

   a. During the next operation, site A will be upgraded, and Site B can provide some level of service during this upgrade procedure. Theoretically, all access can be given at this time. Once site B is upgraded, requests are serviced there, making this the production role of the DR environment. Once site A is upgraded, the software versions at both sites will be the same and a DR instantiate/sync operation will be possible (as performed in Step 12). If this approach is utilized, any updates made at the original production site [site A] will be lost.

   b. If Step 8a is implemented and site B becomes the production site, then ignore the restrictions in Step 3h because site A is about to be upgraded.

9. Perform an OracleAS upgrade to the site A systems [see *Oracle Application Server Upgrade and Compatibility Guide* for more information].

10. Test applications or note problems for resolution. Perform tests until you are satisfied the upgrade has been properly completed.

    At the end of this step, the two site upgrades are functionally equivalent and have been upgraded to OracleAS Disaster Recovery 10.1.2.0.2 Full site functionality has been enabled at site B and it is time to reestablish the production/standby relationship.

11. Stop the OracleAS Guard server in all the old OracleAS 9.0.4 Oracle homes, remove the `dsa.conf` file in the *<ORACLE_HOME>*`/dsa` directory on Unix systems or *<ORACLE_HOME>*`\dsa` directory on Windows systems, then restart the DSA process as well as the OPMN server on all the systems in the new OracleAS 10.1.2 Oracle homes.

    On UNIX systems:

    ```
    <ORACLE_HOME>/opmn/bin/opmnctl stopall
    <ORACLE_HOME>/opmn/bin/opmnctl startall
    <ORACLE_HOME>/opmn/bin/opmnctl startproc ias-component=DSA
    ```

    On Windows systems:

    ```
    <ORACLE_HOME>\opmn\bin\opmnctl stopall
    <ORACLE_HOME>\opmn\bin\opmnctl startall
    <ORACLE_HOME>\opmn\bin\opmnctl startproc ias-component=DSA
    ```

12. Use OracleAS Guard and perform a site instantiation from site B to site A if Step 8a is utilized.

    This step reestablishes the OracleAS Disaster Recovery environment between site B and Site A. In this sequence, site B is the production site, and site A is updated to mirror site B.

    Perform the following asgctl steps to complete this operation:

    a. Invoke the asgctl client.

    ```
    On Unix systems
    <ORACLE_HOME>/dsa/bin/asgctl.sh

    On Windows systems
    <ORACLE_HOME>\dsa\bin\asgctl
    ```

b. Perform a connect operation to site B's Infrastructure system `infra2`.

```
ASGCTL> connect asg infra2 ias_admin/<password>
```

c. Set the primary database to the OracleAS Metadata Repository at site B.

```
ASGCTL> set primary database sys/<password>@<site B's servicename>
```

d. Discover the topology.

```
ASGCTL> discover topology oidpassword=<oidpwd>
```

e. Dump the topology.

```
ASGCTL> dump topology to d:\policy_file_no_904_instances.txt
```

f. Edit the topology file, in this example named `policy_file_no_904_instances.txt` and set all the 9.0.4 instances to Ignore, for example:

```
<policy>
.
.
.
<instanceList successRequirement="Ignore">
   <instance>904Portal_3</instance>
</instanceList>
.
.
.
</policy>
```

Then use this policy file for all Disaster Recovery related operations.

g. Instantiate the topology to site A's standby Infrastructure system `infra1` using the edited policy file from Step 12f.

```
ASGCTL> instantiate topology to infra1 using policy d:\policy_file_no_904_
instances.txt
```

13. Perform a domain name system (DNS) switchover operation.

You would probably perform this step here to absorb the DNS timeout during the time period of the switchover operation. There will be end user access errors (service unavailable) until DNS, the site services, and the application have all been switched over and are running.

14. Use OracleAS Guard to perform a switchover operation from site B to site A.

The end goal is to have the same access at the end of upgrade as at the start of the process. Thus the roles have to be switched between the sites. Connect to the Infrastructure for site B, set the primary database, perform a discover topology, then perform a switchover to site A Infrastructure system `infra1` using the edited policy file from Step 12f.

```
ASGCTL> connect asg infra2 ias_admin/<password>
ASGCTL> set primary database sys/<password>@<site B's servicename>
ASGCTL> switchover topology to infra1 using policy d:\policy_file_no_904_
instances.txt
```

15. Note that an alternative to Steps 9 through 14 would be as follows:

a. Take down the production site [site A].

   b. Perform an OracleAS upgrade to the site A systems [see *Oracle Application Server Upgrade and Compatibility Guide* for more information].

   c. Perform site A to site B instantiation using OracleAS Guard.

16. Start or open up services at production site A for the application.

This completes the steps required for the OracleAS Disaster Recovery site upgrade procedure from OracleAS 10g (9.0.4) to OracleAS 10g (10.1.2.0.2).

8.4 Patching an Existing OracleAS Disaster Recovery Environment

For information about how to patch your OracleAS Disaster Recovery environment (patching OracleAS Guard 10.1.2.n.n (where n.n represents 0.0 and 0.2) with Release 10.1.3.0.0) to take advantage of the features in this latest release of OracleAS Guard, see the platform specific *Oracle Application Server Installation Guide* and specifically the chapter entitled "Installing in High Availability Environments: OracleAS Disaster Recovery." This chapter contains a section entitled "Patching OracleAS Guard Release 10.1.2.n.n with Release 10.1.3.0.0" that describes this patching process.

9 Setting Up a DNS Server

This chapter 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 5–9, "DNS Resolution Topology Overview".

Note: The DNS setup information provided in this chapter 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 5–9):

```
;
;  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 the zone DNS server's IP address):

```
domain    oracleas
nameserver 123.1.2.117
```

10 Secure Shell (SSH) Port Forwarding

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

10.1 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`

Part IV

Appendices

The information in this part is supplementary to the previous chapters of the book and is organized into the following appendixes:

A 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:

- Section A.1, "Troubleshooting OracleAS Disaster Recovery Topologies"
- Section A.2, "Troubleshooting Middle-Tier Components"
- Section A.3, "Need More Help?"

A.1 Troubleshooting OracleAS Disaster Recovery Topologies

This section describes common problems and solutions in OracleAS Disaster Recovery configurations. It contains the following topics:

- Section A.1.1, "Standby Site Not Synchronized"
- Section A.1.2, "Failure to Bring Up Standby Instances After Failover or Switchover"
- Section A.1.3, "Switchover Operation Fails At the Step dcmctl resyncInstance -force -script"
- Section A.1.4, "Unable to Start Standalone OracleAS Web Cache Installations at the Standby Site"
- Section A.1.5, "Standby Site Middle-tier Installation Uses Wrong Hostname"
- Section A.1.6, "Failure of Farm Verification Operation with Standby Farm"
- Section A.1.7, "Sync Farm Operation Returns Error Message"
- Section A.1.8, "On Windows Systems Use of asgctl startup Command May Fail If the PATH Environment Variable Has Exceeded 1024 Characters"

A.1.1 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 5, "OracleAS Disaster Recovery".

A.1.2 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.1.3 Switchover Operation Fails At the Step dcmctl resyncInstance -force -script

The OracleAS Disaster Recovery asgctl switchover operation requires that the value of the TMP variable be defined the same in the `opmn.xml` file on both the primary and standby sites.

Problem

OracleAS Disaster Recovery switchover fails at the step dmctl resyncInstance -force -script and displays a message that a directory could not be found.

Solution

During a switchover operation, the opmn.xml file is copied from the primary site to the standby site. For this reason, the value of the TMP variable must be defined the same in the `opmn.xml` file on both primary and standby sites; otherwise, the switchover operation will fail. Make sure the TMP variable is defined identically in the opmn.xml files and resolves to the same directory structure on both sites before attempting to perform an asgctl switchover operation.

For example, the following code snippets for a Windows and UNIX environment show a sample definition of the TMP variable.

```
Example in Windows Environment:
-------------------------------
.
.
.
<ias-instance id="infraprod.iasha28.us.oracle.com">
 <environment>
 <variable id="TMP" value="C:\DOCUME~1\ntregres\LOCALS~1\Temp"/>
 </environment>
.
.
.
Example in Unix Environment:
----------------------------
```

```
.
.
.
<ias-instance id="infraprod.iasha28.us.oracle.com">
 <environment>
 <variable id="TMP" value="/tmp"/>
 </environment>
.
.
.
```

A workaround to this problem is to change the value of the TMP variable in the `opmn.xml` file on the primary site, perform a dcmctl update config operation, then perform the asgctl switchover operation. This approach saves you having to reinstall the mid-tiers to make use of an altered TMP variable.

A.1.4 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.1.5 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 5.2.2, "Configuring Hostname Resolution" on page 5-18 for more information.

A.1.6 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 examp1 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.1.7 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): Syncronizing 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 Syncronizing 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.1.8 On Windows Systems Use of asgctl startup Command May Fail If the PATH Environment Variable Has Exceeded 1024 Characters

On Windows systems, if your system PATH environment variable has exceeded the 1024 character limit because you have many OracleAS instances installed or many third party software installations, or both on your system, the asgctl startup command may fail because you are starting the OracleAS Guard server outside of OPMN and the system cannot resolve the directory path.

Problem

Occasionally, on Windows systems with many installations, OracleAS instances or third party software, or both, the asgctl startup command, which is run outside of OPMN, may return a popup error stating it could not find a dynamic link library for a particular file, `orawsec9.dll`, followed by a DufException. For example:

```
C:\product\10.1.3\OC4J_1\dsa\bin> asgctl startup
<<Popup Error:>>
The dynamic link library *orawsec9.dll* could not be found.
<<The exception:>>
oracle.duf.DufException
```

```
        at oracle.duf.DufOsBase.constructInstance(DufOsBase.java:1331)
        at oracle.duf.DufOsBase.getDufOs(DufOsBase.java:122)
        at
oracle.duf.DufHomeMgr.getCurrentHomePath(DufHomeMgr.java:582)
        at oracle.duf.dufclient.DufClient.main(DufClient.java:132)
stado42: -->ASG_SYSTEM-100: oracle.duf.DufException
-----------------------------------------------------------------------------
```

However, this dll does exist in the ORACLE_HOME\bin directory.

This error is not seen in OracleAS Guard standalone kit because the file `orawsec9.dll` exists in the ORACLE_HOME\dsa\bin folder.

Solution

The workaround is to either manually edit the system PATH variable with the required path information or manually override the PATH in the command prompt by specifying the relevant %PATH% variables. For example:

```
C:\set PATH=C:\product\10.1.3\OracleAS_OC4J_2\bin;
C:\product\10.1.3\OracleAS_OHS1\jre\1.4.2\bin\client;
C:\product\10.1.3\OracleAS_OHS1\jre\1.4.2\bin;
C:\product\10.1.3\OracleAS_OHS1\bin;C:\product\10.1.3\OC4J_1\bin

C:\product\10.1.3\OC4J_1\dsa\bin> asgctl startup
```

A.2 Troubleshooting Middle-Tier Components

This section describes common problems and solutions for middle-tier components in high availability configurations. It contains the following topics:

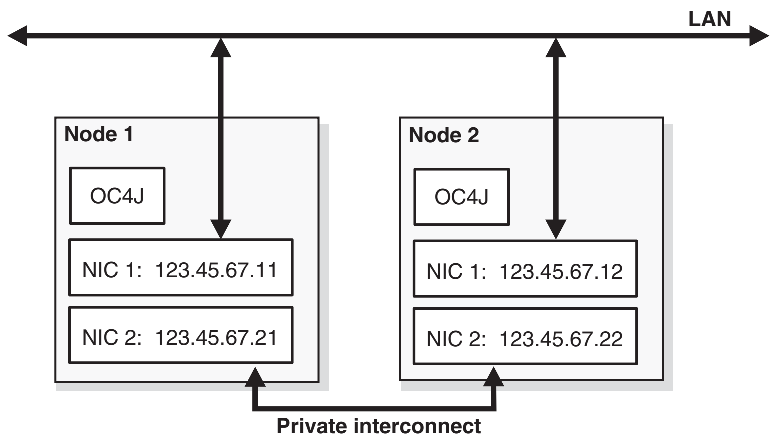
- Section A.2.1, "Using Multiple NICs with OracleAS Cluster (OC4J-EJB)"
- Section A.2.2, "Performance Is Slow When Using the "opmn:" URL Prefix"

A.2.1 Using Multiple NICs with OracleAS Cluster (OC4J-EJB)

Problem

If you are running OracleAS Cluster (OC4J-EJB) on computers with two NICs (network interface cards) and you are using one NIC for connecting to the network and the second NIC for connecting to the other node in the cluster, multicast messages may not be sent or received correctly. This means that session information does not get replicated between the nodes in the cluster.

Figure A–1 OracleAS Cluster (OC4J-EJB) Running on Computers with Two NICs



Solution

You need to start up the OC4J instances by setting the `oc4j.multicast.bindInterface` parameter to the name or IP address of the other NIC on the node.

For example, using the values shown in Figure A–1, you would start up the OC4J instances with these parameters:

On node 1, configure the OC4J instance to start with up with this parameter:

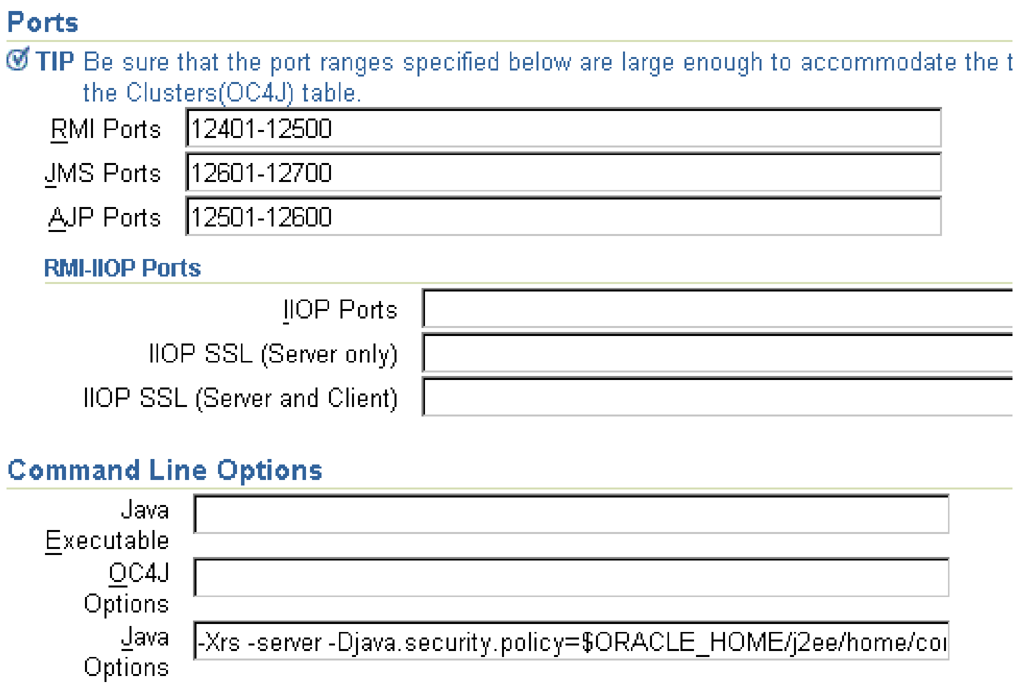
```
-Doc4j.multicast.bindInterface=123.45.67.21
```

On node 2, configure the OC4J instance to start with up with this parameter:

```
-Doc4j.multicast.bindInterface=123.45.67.22
```

You specify this parameter and its value in the "Java Options" field in the "Command Line Options" section in the Server Properties page in the Application Server Control Console (Figure A–2).

Figure A–2 Server Properties Page in Application Server Control Console



Ports

TIP Be sure that the port ranges specified below are large enough to accommodate the t
the Clusters(OC4J) table.

RMI Ports	12401-12500
JMS Ports	12601-12700
AJP Ports	12501-12600

RMI-IIOP Ports

IIOP Ports	
IIOP SSL (Server only)	
IIOP SSL (Server and Client)	

Command Line Options

Java Executable	
OC4J Options	
Java Options	-Xrs -server -Djava.security.policy=$ORACLE_HOME/j2ee/home/co

A.2.2 Performance Is Slow When Using the "opmn:" URL Prefix

Problem

If you have applications that use the "`opmn:`" prefix in their `Context.PROVIDER_URL` property, you may experience slow performance in the `InitialContext` method.

The following sample code sets the `PROVIDER_URL` to a URL with an `opmn:` prefix.

```
Hashtable env = new Hashtable();
env.put(Context.PROVIDER_URL, "opmn:ormi://hostname:port/cmpapp");
// ... set other properties ...
Context context = new InitialContext(env);
```

If the host specified in `PROVIDER_URL` is down, the application has to make a network connection to OPMN to locate another host. Going through the network to OPMN takes time.

Solution

To avoid making another network connection to OPMN to get another host, set the `oracle.j2ee.naming.cache.timeout` property so that the values returned from OPMN the first time are cached, and the application can use the values in the cache.

The following sample code sets the `oracle.j2ee.naming.cache.timeout` property.

```
Hashtable env = new Hashtable();
env.put(Context.PROVIDER_URL, "opmn:ormi://hostname:port/cmpapp");

// set the cache value
env.put("oracle.j2ee.naming.cache.timeout", "30");
```

```
// ... set other properties ...

Context context = new InitialContext(env);
```

Table A–1 shows valid values for the `oracle.j2ee.naming.cache.timeout` property:

Table A–1 Values for the oracle.j2ee.naming.cache.timeout Property

Value	**Meaning**
`-1`	No caching.
`0`	Cache only once, without any refreshing.
Greater than `0`	Number of seconds after which the cache can be refreshed. Note that this is **not automatic**; the refresh occurs only when you invoke "`new InitialContext()`" again. If the property is not set, the default value is 60.

With the property set, you will still see some delay on the first "`new InitialContext()`" call, but subsequent calls should be faster because they are retrieving data from the cache instead of making a network connection to OPMN.

Note that for optimal performance, you should also set `Dedicated.Connection` to either `YES` or `DEFAULT`, and set `Dedicated.RMIcontext` to `FALSE`.

A.3 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`

B 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 Topology Error Messages
  - OracleAS Backup and Restore Error Messages
  - OracleAS Guard Synchronize Error Messages
  - OracleAS Guard Instantiate Error Messages

B.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.

B.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.

B.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.

B.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.

B.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".

13504, Error switching the database role from primary to standby.

Cause: Failed to switchover database role from primary to standby.

Action: See secondary error.

13505, Error switching the database role from standby to primary.

Cause: Failed to switchover database role from standby to primary.

Action: See secondary error.

13061, Error failing over physical standby database.

Cause: Error occurred in performing a failover of a standby database.

Action: See secondary error.

B.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 ASG server.

Cause: Only ias_admin can connect to a ASG server.

Action: Please use ias_admin to connect to a ASG 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.

3034, Invalid command line specified: - {0}.

Cause: Invalid command line specification.

Action: Correct the command line option and retry.

3035, Failed to kill the OracleAS Guard (ASG) server process {0} with error {1}.

Cause: OracleAS Guard client is unable to kill the OracleAS Guard (ASG) server process.

Action: Use "kill -9 <pid>" command to kill the process from the command line prompt.

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.

B.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 cannnot 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 topology.

Cause: User is connected to server on a topology that is not the primary topology.

Action: Connect to primary topology node.

3544, Error processing command, must be connected to a OracleAS Guard server in the standby topology.

Cause: User is connected to server on a topology that is not the standby topology.

Action: Connect to primary topology node.

3545, Error trying to remove old passwd file %1 while creating new db.

Cause: Could not delete the old password file as part of a delete database operation. This is a problem when trying to create a new database.

Action: Delete the stale password file.

3546, Error, database SID was expected to have value but it is empty.

Cause: The database SID was suppose to have a value but it is empty.

Action: This is an internal error.

3547, 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.

3548, 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.

3549, Error cleaning up the database on the standby host.

Cause: Failed to clean up the database on the standby host.

Action: See secondary error.

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.

3554, The Oracle home that contains SID {0} cannot be found.

Cause: The Oracle home that contains a specific SID cannot be found.

Action: Make sure the Oracle home inventory is valid.

3555, Error accessing the Oracle home inventory. Make sure the inventory file exists.

Cause: The Oracle home inventory cannot be accessed.

Action: Make sure the Oracle home inventory exists

3556, Error: Unable to find the Oracle home within path {0}.

Cause: The Oracle home within the given path cannot be found.

Action: Make sure the Oracle home inventory exists.

B.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.

B.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.

B.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.

B.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 topology information from OPMN Manager on {0}.

Cause: Error trying to get topology 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.

3804, Failed trying to remove topology entry from {0}.

Cause: Failed trying to remove topology entry from opmn.xml.

Action: See secondary error.

3900, Error creating Oracle database service because the service has already been marked for deletion. Please exit the Windows Service Control Manager on node {0}. Would you like to retry?

Cause: The user has the SCM open causing a service operation to fail.

Action: User must exit SCM GUI.

B.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.

4031, Error getting environment variables using the env command.

Cause: The env command does not work.

Action: Make sure the `/bin` or `/usr/bin` directory contains the env executable.

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.

B.2.7 LDAP or OID Error Messages

The following are LDAP or OID error messages.

4101, Failed to connect to OID server on host {0}, port {1}.

Cause: Failed to the OID server on a given host and port.

Action: See secondary error.

4102, Failed to connect to OID server via SSL on host {0}, port {1}.

Cause: Failed to the OID server via SSL on a given host and port.

Action: See secondary error.

4103, User must specify host, port, user name, and password for the OID server.

Cause: User did not specify all the above parameters.

Action: User must specify all the above parameters in order to access the OID server.

4104, Failed to get the value of attribute {0} from OID server.

Cause: Failed to get the value of the given attribute.

Action: See secondary error.

4105, Failed to get the attributes for DN {0} from OID server

Cause: Failed to get the attributes of the given DN.

Action: See secondary error.

4106, Failed to get Oracle Application Server instances from OID server

Cause: Failed to get Oracle Application Server instances from the OID server.

Action: See secondary error.

4107, Failed to get infrastructure databases from OID server.

Cause: Failed to get infrastructure databases from the OID server.

Action: See secondary error.

4110, Cannot set current topology to file {0} because the file does not exist.

Cause: The topology file does not exist.

Action: Specify a filename of a file that exists.

4111, The current topology file \"{0}\" does not exist. Use the "set topology" command to specify a valid topology file.

Cause: The topology file does not exist.

Action: Specify a filename of a file that exists in dsa.conf.

B.2.8 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.

4903, The minimum succeed value of {0} was not met for the workers in group {1}.

Cause: A group of workers belonging to the same group requires that a minimum number of them succeed. That minimum succeed value was not met.

Action: See secondary error.

4950, An error occurred on host {0} with IP {1} and port {2}.

Cause: An error occurred on the server.

Action: See secondary error.

B.2.9 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 topology operation.

Action: Check secondary error.

15306, Warning during undo processing.

Cause: Error occurred during undo processing.

Action: Check secondary error.

B.2.10 OracleAS Database Error Messages

The following are OracleAS database error messages.

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 topology.

Cause: Failed to perform a sync database operation on the primary topology.

Action: See secondary error.

15607, Failed to perform a sync database operation on the standby topology.

Cause: Failed to perform a sync database operation on the standby topology.

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.

B.2.11 OracleAS Topology Error Messages

The following are OracleAS topology error messages.

15620, An Invalid Topology was specified.

Cause: Error trying to process a topology object.

Action: Retrieve a valid topology object.

15621, Error trying to verify topology {0}.

Cause: The specified topology 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, Topology {0} is not symmetrical with topology {1}.

Cause: The specified topologies are not symmetrical.

Action: See secondary error for more information.

15624, An Invalid Topology was specified. Topology {0} does not contain any valid instances.

Cause: Error trying to process a topology object. Topology object did not contain a valid instance.

Action: Retrieve a valid topology object with at least one instance.

15625, Could not find matching instance {0} in Topology {1}.

Cause: Could not get matching instances. Topologies do not appear to be symmetrical.

Action: Make the topologies symmetrical.

15626, Topologies are not symmetrical because topology name {0} is not the same as topology name {1}.

Cause: Topology names are not the same and therefore topologies are not symmetrical.

Action: Make the topologies symmetrical.

15627, Instance {0} is not symmetrical because of different Oracle Home names {1}.

Cause: Instance Home names are not symmetrical in the specified topologies.

Action: Make the topologies symmetrical.

15628, Instance {0} is not symmetrical because of different Oracle Home paths {1}.

Cause: Instance Home paths are not symmetrical in the specified topologies.

Action: Make the topologies symmetrical.

15629, Instance {0} is not symmetrical, because of different host names {1}, {2}.

Cause: Instance host names are not symmetrical in the specified topologies.

Action: Make the topologies 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 topologies appear to be identical because both have instance {0} on host {1}.

Cause: An instance can only be in a member of one topology, it appears that the primary and standby topologies are the same.

Action: Specify a primary and separate standby topology.

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 Topology was specified. Topology contains a duplicate instance named {0}.

Cause: Topology information obtained from OPMN contains a duplicate instance.

Action: Check OPMN to ensure that the topology information listed is correct.

B.2.12 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 topology.

Cause: Backup topology failed at the specified step.

Action: Check secondary error.

15695, Error executing step - {0} of Restore topology.

Cause: Restore topology failed at the specified step.

Action: Check secondary error.

15696, Error initializing backup topology operation.

Cause: Error initializing backup topology operation.

Action: Check secondary error.

15697, Error during backup topology operation - backup step.

Cause: Error during backup step processing of backup topology.

Action: Check secondary error.

15698, Error during backup topology operation - copy step.

Cause: Error during copy step processing of backup topology.

Action: Check secondary error.

15699, Error initializing restore topology operation.

Cause: Error initializing restore topology 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.

15702, Error initializing the clone instance operation.

Cause: Error initializing the clone instance operation.

Action: Check the secondary error.

15703, Error initializing the clone topology operation.

Cause: Error initializing the clone home operation.

Action: Check the secondary error.

15704, Error: Oracle home of the instance to be cloned {0} already exists.

Cause: Error cloning instance, the Oracle home already exists

Action: Clean up the Oracle home and retry.

15705, cloning instance {0}. Cloning requires OPMN to be stopped, therefore the OracleAS Guard server must be started using asgctl .

Cause: Cloning requires that OPMN be stopped, which will cause the OracleAS Guard server (ASG server process) to be stopped. This will cause the clone to fail.

Action: Use opmnctl to stop the OracleAS Guard server (ASG server process). Then use the asgctl startup topology command to restart OracleAS Guard server for this instance.

15706, Stop the backup home image operation in response to the user's request.

Cause: Stop the backup home operation because the user entered NO.

Action: None.

15707, Stop the restore home image operation in response to the user's request.

Cause: Stop the restore home operation because the user entered NO.

Action: None.

B.2.13 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 topology information is available to perform the topology operation.

Cause: No topology information is available to perform the topology operation.

Action: Check secondary error.

15723, No instances are found in the topology's backup list.

Cause: The topology'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 topology {0}.

Cause: Failed to backup OracleAS topology configuration data.

Action: Check secondary error.

15726, Failed to restore OracleAS configuration data for topology.

Cause: Failed to restore OracleAS topology configuration data.

Action: Check secondary error.

15727, Failed to backup OracleAS infrastructure database {0}.

Cause: Failed to backup OracleAS topology infrastructure database.

Action: Check secondary error.

15728, Failed to restore OracleAS infrastructure database {0}.

Cause: Failed to restore OracleAS topology infrastructure database.

Action: Check secondary error.

15729, Failed to perform the sync topology operation.

Cause: Failed to perform the sync topology operation.

Action: Check secondary error.

B.2.14 OracleAS Guard Instantiate Error Messages

The following are OracleAS Guard instantiate error messages.

15751, Error executing step {0} of instantiate topology operation.

Cause: The instantiate topology operation failed at the specified step.

Action: Check secondary error.

15752, Failed to load remote topology information.

Cause: Failed to load remote topology information.

Action: Make sure the user specified the correct host name for the topology and that the OPMN processes are running on the topologies.

15753, Error preparing to instantiate topology on host {0}.

Cause: Error preparing to instantiate topology.

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 topology operation.

Cause: Error initializing the instantiate topology operation.

Action: Check secondary error.

15757, Error initializing switchover topology operation.

Cause: Error initializing the switchover topology operation.

Action: Check secondary error.

15770, The instance {0} specified in the topology file does not match the instance {1} in home {2}.

Cause: The topology file is incorrect.

Action: Run the "discover topology command" from asgctl.

15771, The topology file {0} is the wrong version Please delete the file and rediscover the topology.

Cause: The topology file is incorrect.

Action: Run the "discover topology command" from asgctl.

15772, The topology file {0} does not contain an entry for the discovery host {1}.

Cause: The topology file is incorrect.

Action: Run the "discover topology command" from asgctl.

15773, The standby topology does not contain a entry for the mandatory primary instance {0}.

Cause: The topology file is incorrect.

Action: Run the "discover topology command" from asgctl.

15774, The host name {0} in the standby topology net descriptor for database {1} resolves to a primary host address {2}.

Cause: The topology file is incorrect.

Action: Run the "discover topology command" from asgctl.

15775, The standby topology host name {0} for the instance {1} resolves to a primary host address.

Cause: The topology file is incorrect.

Action: Run the "discover topology command" from asgctl.

15776, Error accessing the OID server.

Cause: Unable to access the OID server.

Action: Specify the correct OID information and make sure the OID server is running.

15777, Error: OID information needed to access the server was not specified.

Cause: Unable to access the OID server.

Action: Specify the correct OID information.

15778, Error getting database information for SID {0} from host {1}. This instance will be excluded from the topology.xml file.

Cause: Unable to get database information for the topology database.

Action: None.

15779, Error getting instance information for instance {0} from host {1}. This instance will be excluded from the topology.xml file.

Cause: Unable to get information for an instance.

Action: None.

15780, Instance {0} cannot be found in the topology.

Cause: The instance name does not exist in the topology file.

Action: Perform the asgctl discover topology command.

15781, Warning: Unable to update topology file on host {0} in home {1}.

Cause: The topology file cannot be written to the home.

Action: Make sure the OracleAS Guard server is running in that home.

15782, Error: Instance {0} already exists in the topolgy.

Cause: The instance already exists in the topology.

Action: Remove the instance using asgctl.

15783, Error: OID host and port information needed to access the server was not specified.

Cause: Unable to access the OID server.

Action: Specify the correct OID information.

Index