Oracle® Fusion Middleware

Disaster Recovery Guide

14*c* (14.1.2.0.0)
F85482-01
October 2025

Oracle Fusion Middleware Disaster Recovery Guide, 14*c* (14.1.2.0.0)

F85482-01

Primary Author: Oracle Corporation

Contents

5 Managing Lifecycle Operations

Preface

This document provides disaster recovery solution for Oracle Fusion Middleware components.

Audience

This document is intended for administrators, developers, and others whose role is to deploy and manage the Oracle Fusion Middleware Disaster Recovery solution using storage replication technology.

Documentation Accessibility

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

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Documents

For more information, see the following documents in the Oracle Fusion Middleware documentation set:

- *Oracle Fusion Middleware High Availability Guide*
- *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle SOA Suite*
- *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle WebCenter Portal*
- *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle WebCenter Content*
- *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Business Intelligence*
- *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Identity and Access Management*

Conventions

The following text conventions are used in this document:

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

1

Understanding Oracle Fusion Middleware Disaster Recovery

Oracle Fusion Middleware Disaster Recovery is a disaster recovery solution that provides protection to Oracle Fusion Middleware components in different Oracle product suites.

This chapter includes the following sections:

Disaster Recovery Terminology

Learn about disaster recovery terminology.

The following terms are used in disaster recovery:

- **Disaster**

  A sudden, unplanned catastrophic event that causes unacceptable damage or loss in a site or geographical area. A disaster is an event that compromises an organization's ability to provide critical functions, processes, or services for an unacceptable period and causes an organization to invoke its recovery plans.

- **Disaster Recovery**

  Ability to safeguard against natural or unplanned outages at a production site by having a recovery strategy for applications and data to a geographically separate secondary site.

- **Disaster Recovery Topology**

  The production and secondary site hardware and software components that comprise an Oracle Fusion Middleware Disaster Recovery solution.

- **Enterprise Deployment Guide**

  The Enterprise Deployment Guides provide detailed and verified instructions that help you plan, prepare, install, and configure a multi-host, secure, highly available, production topology for selected Oracle Fusion Middleware products in the scope of a single data center. The Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Fusion Middleware SOA is available *here*.

- **Alias Host Name**

  Alias host name is an alternate way to access the system besides its real network name. Typically, it resolves to the same IP address as the network name of the system. This can be defined in the name resolution system such as DNS or locally in the local hosts file on each system. Multiple alias host names can be defined for a given system.

- **Physical Host Name**

  Physical host name is the host name of the system as returned by the `gethostname()`call or the `hostname` command. Typically, the physical host name is also the network name used by clients to access the system. In this case, an IP address is associated with this name in the DNS (or the given name resolution mechanism in use) and this IP is enabled on one of the network interfaces of the system.

  A given system typically has one physical host name. It can also have one or more additional network names, that correspond to the IP addresses enabled on its network

interfaces which are used by clients to access it over the network. Each network name can be aliased with one or more alias host names.

- **Production or Primary Site**

  A production or primary site in a disaster protection topology is the system that is carrying the system's workload at a precise point in time. It is a group of hardware, network and storage resources, and processes that are actively used to carry business logic and process requests at a precise point in time.

- **Maximum Availability Architecture**

  Oracle's Maximum Availability Architecture (Oracle MAA) is the best practices blueprint for data protection and availability of Oracle products (Database, Fusion Middleware, Applications). Implementing Oracle Maximum Availability Architecture best practices is one of the key requirements for any Oracle deployment. It provides recommendations for setting up and managing an Oracle system. Oracle's Maximum Availability includes the Enterprise Deployment Guide recommendations and adds disaster protection best practices to minimize planned and unplanned downtimes for outages affecting an entire data center or region.

- **Recovery Point Objective (RPO)**

  Recovery point objective is the amount of data loss that a system can tolerate or is acceptable when an outage takes place, from a business point of view.

- **Recovery Time Objective (RTO)**

  Recovery time objective is the amount of downtime a system can tolerate or the acceptable amount of time that an application or service can remain unavailable when an outage takes place, from a business point of view.

- **Site Failover**

  Process of making the current secondary site the new production or primary site after the production site becomes unexpectedly unavailable due to a disaster at the production site. The term *failover* is also used to refer to a site failover in this document.

- **Site Switchback**

  Process of reverting the current production site and the current secondary site to their original roles. Switchbacks are planned operations done after the switchover operation is completed. The current secondary site becomes the production site and the current production site becomes the secondary site. The term *switchback* is also used to refer to a site switchback in this document.

- **Site Switchover**

  Process of reversing the roles of the production and secondary site. Switchovers are planned operations done for periodic validation or to perform planned maintenance on the current production site. During a switchover, the current secondary site becomes the new production site and the current production site becomes the new secondary site. The term *switchover* is also used to refer to a site switchover in this document.

- **Site Synchronization**

  Process of applying changes made at the production site to the secondary site. For example, when a new application is deployed at the production site, you should perform a synchronization so that the same application is also deployed at the secondary site.

- **Secondary or Standby Site**

  A secondary site is a backup location that can take over the business logic and requests that a primary site was processing. Typically, secondary sites are also named as "Standby" because they remain on "standby or inactive mode". This means that they are not

processing the production workload during normal operations. However, this does not imply that the secondary site cannot be used for other purposes. This is especially true in more modern models where the secondary site is used for reporting operations and more importantly for validating changes before applying them in the primary site.

- **Symmetric Topology**

  An Oracle Fusion Middleware Disaster Recovery configuration that is completely identical across tiers on the production and secondary site. It has identical number of hosts, load balancers, instances, and applications with the same ports being used for both sites and systems configured with identical capacity. This document describes how to set up a symmetric Oracle Fusion Middleware Disaster Recovery topology for an enterprise configuration.

- **Asymmetric Topology**

  An Oracle Fusion Middleware Disaster Recovery configuration that is different across tiers on the production and secondary site. For example, an asymmetric topology can include a secondary site with fewer hosts and instances than the production site.

  > **Note**
  >
  > Oracle does not recommend using scaled down secondary systems. Non-symmetric standbys can cause cascade falls if workloads are not handled properly and they can also produce misconfigurations and data loss.

- **System**

  A System is a set of targets (hosts, databases, application servers, and so on) that work together to host your applications. For example, to monitor an application in Enterprise Manager, you would first create a system that consists of the database, listener, application server, and host targets on which the application runs.

- **Site**

  Site is the set of different components in a datacenter needed to run a group of applications. For example, a site could consist of Oracle Fusion Middleware instances, databases, storage, and so on.

- **Virtual Host Name**

  Virtual host name is a network addressable host name that can be mapped to one or more physical systems. This can be done by enabling the associated VIP in a node through a load balancer or a hardware cluster.

  For load balancers, the term *virtual server name* is used interchangeably with *virtual host name* in this document. A load balancer can hold a virtual host name on behalf of a set of servers, and clients communicate indirectly with the systems by using the virtual host name.

  In a hardware cluster, a virtual host name is a network host name assigned to a cluster virtual IP. Because the cluster virtual IP is not permanently attached to any particular node of a cluster, the virtual host name is not permanently attached to any particular node either.

  In the context of a single host, a virtual host name is an additional host name to access the system besides the real network name. It is typically mapped to a virtual IP enabled in the node's network interfaces, or it can be mapped to an existing IP address in the system. In this last case, it becomes an alias host name of the system in the name resolution system DNS or locally in the local host file.

- **Virtual IP**

Generally, a virtual IP (VIP) is an IP that is assigned to a secondary Network Interface Controller (NIC) or to a Virtual Network Interface Controller (VNIC). The hardware nodes or virtual machines have their own physical IP address and physical host name and can use several additional VIP addresses. These VIP addresses "float" or can be migrated between different nodes. VIPs are also used in load balancers and hardware clusters. A VIP presents a single entry point IP address that abstracts accessors from the backend points and can be migrated or moved across nodes for different purposes.

Traditionally, hardware clusters use a cluster virtual IP to present to the outside world the entry point into the cluster. 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.

Currently, Virtual IPs are also managed manually or through application servers (for example, WebLogic provides virtual IP migration functionality with server migration) when a precise component needs to be failed over (transparently to consumers) to a different hardware.

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 that acts as a proxy between servers and their clients.

- **WebLogic Whole Server Migration**

  Whole server migration occurs when a WebLogic Server instance migrates to a different physical system upon failure.

- **WebLogic Service Migration**

  Service-level migration occurs when services running in a WebLogic Server move to a different WebLogic Server instance within the cluster.

Overview of Oracle Fusion Middleware Disaster Recovery

Learn about Oracle Fusion Middleware Disaster Recovery.

A disaster protection strategy must address the different phases in a system's life cycle:

- Initial Setup

  Configuring the system initially to get an initial replica of the primary system in a secondary location.

- Managing Switchover and Failover

  Moving workloads to the secondary location in an event of a planned or unplanned downtime affecting an entire data center or geographical region.

- Maintainance
  - Ongoing Synchronizations

    Maintaining secondary location up to speed with the configuration, metadata, and runtime data when it is modified in the primary system.

  - Patching

    Applying patches to the disaster recovery topology.

  - Scale Out Operations

    Scaling the system in the secondary when the primary is also modified.

Before running the initial setup and preparing the lifecycle of a disaster protection system, it is crucial to understand the critical aspects that will drive its implementation. It is also important to differentiate between features that provide protection against local failure and those that protect against a disaster.

The other two main areas that will drive decisions for different variations in the disaster protection configuration are data replication (how a system is replicated to a secondary location) and virtualization of the configuration (how to make the configuration used in the primary, valid in the secondary). While following the Oracle MAA recommendations, it is important to understand that a disaster solution needs to use a secondary system that is a replica of the primary so that it can become a first class production site itself in the event of a total loss of the primary. The following sections provide more details in these areas.

Disaster Protection vs Local Failure Protection

Providing Oracle Maximum Availability Architecture is one of the key requirements for any Oracle Fusion Middleware deployment. An Oracle Fusion Middleware Enterprise Deployment Guide provides the best practices and recommendations within the scope of a single data center (See *Enterprise Deployment Guide for Oracle SOA Suite*). Oracle Fusion Middleware includes an extensive set of high availability features such as process death detection and restart, server clustering, service migration, cluster integration, GridLink, load balancing, failover, backup and recovery, rolling upgrades, and rolling configuration changes which protect an enterprise deployment from unplanned downtime and minimizes the planned downtime.

Most of the downtime experienced by Fusion Middleware systems are caused by local failures. These are failures that affect a component or part of the resources in a data center but that can be corrected with local redundancy for that precise component. These are outages that typically do not render an entire data center as inaccessible. Therefore, disaster protection only makes sense on top of an existing strategy against these local failures. Complete-site outages and downtime affecting entire regions occur much less frequently than local storage crashes, hypervisors failures, local network issues and so on. To provide protection against this type of downtime, follow the recommendations provided in the Enterprise Deployment Guide for your Fusion Middleware component. Enterprise Deployment Guides are the foundation on top of which this disaster protection guide is built.

In addition to these local failures, enterprise deployments need protection from unforeseen disasters and natural calamities that can bring down an entire data center or geographical area. A Maximum Availability Architecture for Fusion Middleware implements all the best practices prescribed by the Fusion Middleware Enterprise Deployment Guide including disaster protection. A disaster protection solution involves setting up a secondary site at a geographically different location with equal services and resources compared to the production site. Oracle recommends configuring symmetrical topology and capacity at both the production and the secondary sites to prevent inconsistencies at the functional and performance levels. The secondary site is normally in a passive mode. This deployment model is sometimes referred to as an active-passive or active-standby model. "Passive" in this context means that the secondary site is not processing the production workload that the primary is processing at that point in time. However, it does not mean that the secondary system cannot be used during normal operation. Secondary systems in the DR configurations proposed by this guide are used to verify new applications, validate patches or to run workload tests before applying those changes to the primary system. This model is usually adopted when the two sites are connected over a WAN and network latency does not allow clustering across the two sites.

Data Replication

Most Oracle Fusion Middleware components are stateful. Different data types stored in different persistent formats need to be copied from the primary site to the secondary site.

Application data, metadata, configuration data, and security data need to be replicated periodically to the secondary site. This is done to ensure that in a switchover or failover scenario, the reply from the new active site will be perfectly consistent with the one that was offered from the original primary. Different WebLogic and Fusion Middleware components store configuration information in the file system. Additionally, artifacts such as keystores (for Identity and Trust) are critical pieces of the Oracle Fusion Middleware 14.1.2 Enterprise Deployment SSL configuration. These stores can reside in external vaults and also in the file systems.

The Oracle Fusion Middleware Disaster Recovery solution can use different replication technologies for disaster protection of Oracle Fusion Middleware middle-tier components. It can use storage level replication and is compatible with third-party storage vendor recommended solutions. It can also use other supported methods to replicate the Fusion Middleware middle-tier configurations like DBFS or rsync. Although a single replication strategy is typically used for all file systems, different data types can use different approaches according to the RTO, RPO, and consistency needs in each case.

Replication and disaster protection for the Oracle databases used by Oracle Fusion Middleware is provided through Oracle Data Guard. This is the only supported configuration to protect the Oracle Fusion Middleware against disaster with a remote mirror configuration.

The replication frequency for the different types of data (whether on the database or on a file system) should be as high as the systems recovery point objective (RPO) demands. The time consumed in transitioning from a primary system to a secondary system should be as short as the systems recovery time objective (RTO).

Access Points and Configuration Virtualization

Using configuration and metadata in the primary without any modifications in the secondary is also a key aspect to an appropriate disaster protection solution. Disaster protection solutions should elude manipulating the primary configuration to adjust to secondary. These manipulations increase RTO in failover scenarios and are difficult to maintain as applications evolve. The least amount of information that needs to be replicated, the more frequently the replication cycles can be scheduled thus reducing the system recovery point objective. To make the configuration agnostic to whether it is run from primary or standby, the following requirements must be met:

- Clients and other applications or services accessing the system should continue using the same address for their access after a switchover or failover without requiring to change the *hostname* that is used to access the failed over resources. Failures should be transparent to consumers, especially when these front-end addresses are public and used by thousands of browsers or devices.
- All the listen addresses used by different components in the Fusion Middleware system (besides the system's front-end addresses) should be hostnames that can be activated in both locations (mapping to a different IP in each location). This will avoid the need to replace the listen addresses in the configuration that the secondary receives from the primary.
- Any external dependencies (like services that are not part of the Fusion Middleware domain) should be accessible with the same configuration both from the primary and the secondary. This includes external hostnames, storage, or network resources. All of them should be equally accessible in both regions.

Symmetry Requirements

The Oracle Fusion Middleware Disaster Recovery topology uses a replica of the primary in the secondary site. Oracle does not recommend using scaled down secondary systems. Non-symmetric secondary system can cause cascade falls if workload is not handled properly and

they can also produce misconfiguration and data loss. Symmetry between the two sites is configured based on the following:

- Hardware, Nodes, and Infrastructure Resources

  The production and secondary site have identical number of hosts, load balancers, instances, and applications. The same ports are used for both sites. The systems are configured with identical capacity and should be capable to sustain the exact same workload.

- Directory Names and Paths

  Every file that exists at the production site host must exist in the same directory path at the secondary site peer host. Therefore, Oracle Home names and directory paths for WebLogic domains, deployments, and configuration must be the same at the production and secondary site.

- Port Numbers

  Port numbers are used by listeners for routing of requests. Port numbers are stored in the configuration and must be the same at the production site host and their secondary site peer hosts.

- Security

  The same user accounts must exist at both production and secondary site. The same central LDAP content and policies must be accessible from both locations. You must also configure the file system and SSL identically at the production and secondary site. For example, in the Oracle Fusion Middleware Enterprise Deployment Guide for Oracle SOA Suite the production site uses SSL so the secondary site must also use SSL configured in exactly the same way as in the primary.

- Load Balancers and Virtual Server Names

  A front-end load balancer should be set up with virtual server names for the production site and an identical front-end load balancer should be set up with the same virtual server names for the secondary site.

- Software

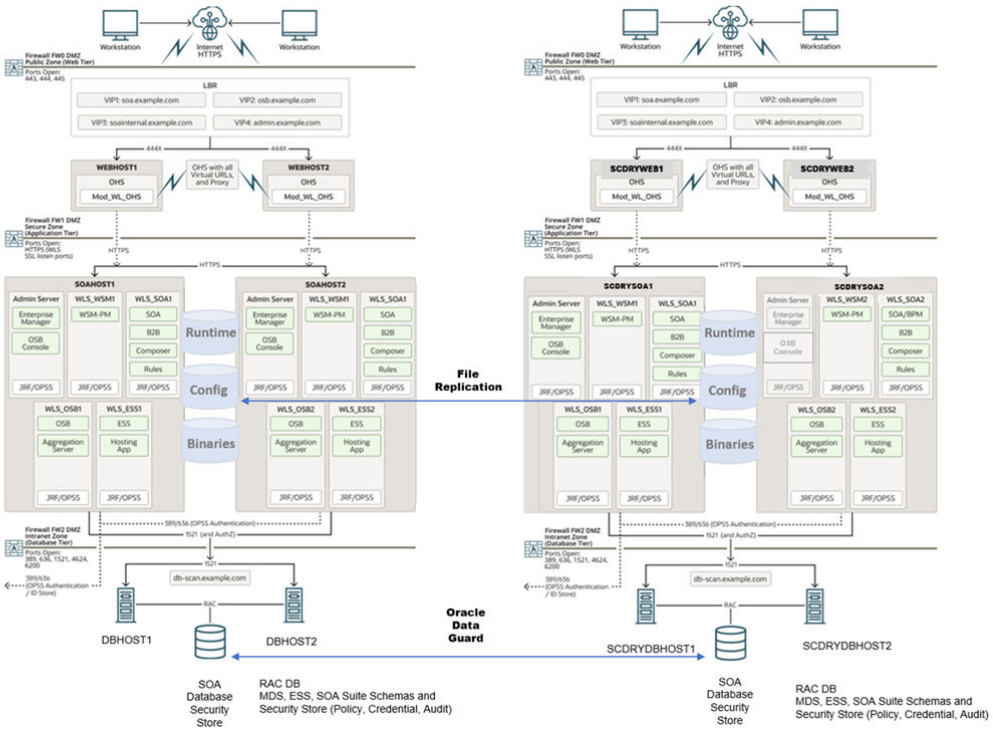
  The same versions of software must be used at the production and secondary site. The operating system patch level must also be the same at both sites and patches to Oracle or third-party software must be applied to both the production and secondary site.

Oracle Fusion Middleware Disaster Recovery Architecture Overview

Learn about the typical topology and main aspects in a disaster recovery solution for an Oracle Fusion Middleware enterprise deployment.

Figure 1-1 shows an overview of an Oracle Fusion Middleware Disaster Recovery topology for an on-premises topology.

Figure 1-1 Production and Standby Sites for Oracle Fusion Middleware Disaster Recovery Topology



The primary system is constructed following the Oracle Fusion Middleware Enterprise Deployment Guide. Some of the additional key aspects of the solution in Figure 1-1 are described below:

- The solution involves two sites. The current production site is running and active, while the second site is serving as a secondary site and is in passive mode.
- Hosts on each site have mount points that are defined for accessing shared storage system for the site as prescribed in the pertaining EDG. A replication technology (storage level replication, rsync, or DBFS) is used to copy the middle-tier file systems and other data from the shared storage of the production site to the shared storage of the standby site.
- After file replication is enabled, application deployment, configuration, metadata, data, and product binary information is replicated from the production site to the standby site.
- It is not necessary to perform any Oracle software installations at the standby site hosts. When the production site storage is replicated to the standby site storage, the equivalent Oracle Home directories and data are written to the standby site storage.
- Oracle Data Guard is used to replicate all Oracle database repositories including Oracle Fusion Middleware repositories and custom application databases. For more information about disaster protection provided by the Oracle Data Guard, see *Oracle Data Guard*.
- Middle-tiers in each region connect only to the database that is local in that region. Cross-region connection from middle-tier in primary to the database in secondary and from middle-tier in the secondary to the database in the primary should be avoided because depending on factors like firewalls and host name resolution, these connections could hang affecting the health of the Fusion Middleware system.
- During a normal operation, the user requests are initially routed to the production site.

- When there is a failure or planned outage of the production site, the following summary steps are executed so that the secondary site assumes the primary role in the topology:
  1. File replication from the production to the secondary site is stopped (when a failure occurs, replication may have already been stopped due to the failure).
  2. A failover or switchover of the Oracle databases is performed using Oracle Data Guard.
  3. The services and applications on the secondary site are started.
  4. Using a global load balancer or a DNS, change user requests are rerouted to the secondary site. At this point, the secondary site has assumed the production role.

The following chapters provide details about how to configure the disaster protection system initially and how to manage the system through its lifecycle.

Oracle Fusion Middleware Disaster Recovery in Oracle Cloud Infrastructure

Learn how Oracle Cloud Infrastructure can be used to host a secondary system for an on-premises primary deployment.

For cases where the secondary system resides in the Oracle Cloud Infrastructure (OCI), Oracle provides a framework that analyzes the primary system, creates the peer resources in the secondary, and replicates the entire Fusion Middleware system to the secondary. The framework is opensource and is available in *GitHub*. The framework creates and configures a symmetric disaster recovery system in the OCI for an existing Oracle WebLogic or Fusion Middleware domain environment based on JEE/Jakarta components (The framework does not cover or address system components such as LDAPs. It is intended for systems that are based on standard WebLogic deployments). The framework offers its greatest degree of automation for the cases where the primary environment follows the Enterprise Deployment Guide. It maintains an inventory of created resources that can be easily cleaned up or reclaimed thus allowing a quick disaster recovery deployment and verification without incurring high costs. These architectures are usually referred to as hybrid disaster protection architectures that provide many benefits as compared to "on-premises to on-premises" disaster protection systems:

- Allows a gradual and easy move of workloads to the cloud. The secondary system can be used as a bed test for moving systems to cloud. It allows getting familiar with cloud infrastructure in a quick and versatile way.
- Leverages OCI High Availability and Reliability features. Oracle Cloud Infrastructure provides many high availability features with fault domains and availability domains, continuous mirroring for storage at different levels, and redundancy for load balancers and network devices among other things.
- Reduces costs as compared to a full-blown secondary on-premises because the management and administration overhead is minimum since shared storage, network, compute, and many other infrastructure pieces are managed directly by Oracle Cloud. Oracle Cloud universal credits can be used to provision the secondary and if after some tests it is decided not to use the DR configuration, the secondary system can be cleaned up in no time and credits can be quickly reclaimed for other purposes.

Apart from these generic benefits, Oracle's WebLogic hybrid disaster recovery framework completely automates the disaster protection setup experience through a reliable process that avoids human errors and implements many MAA best practices.

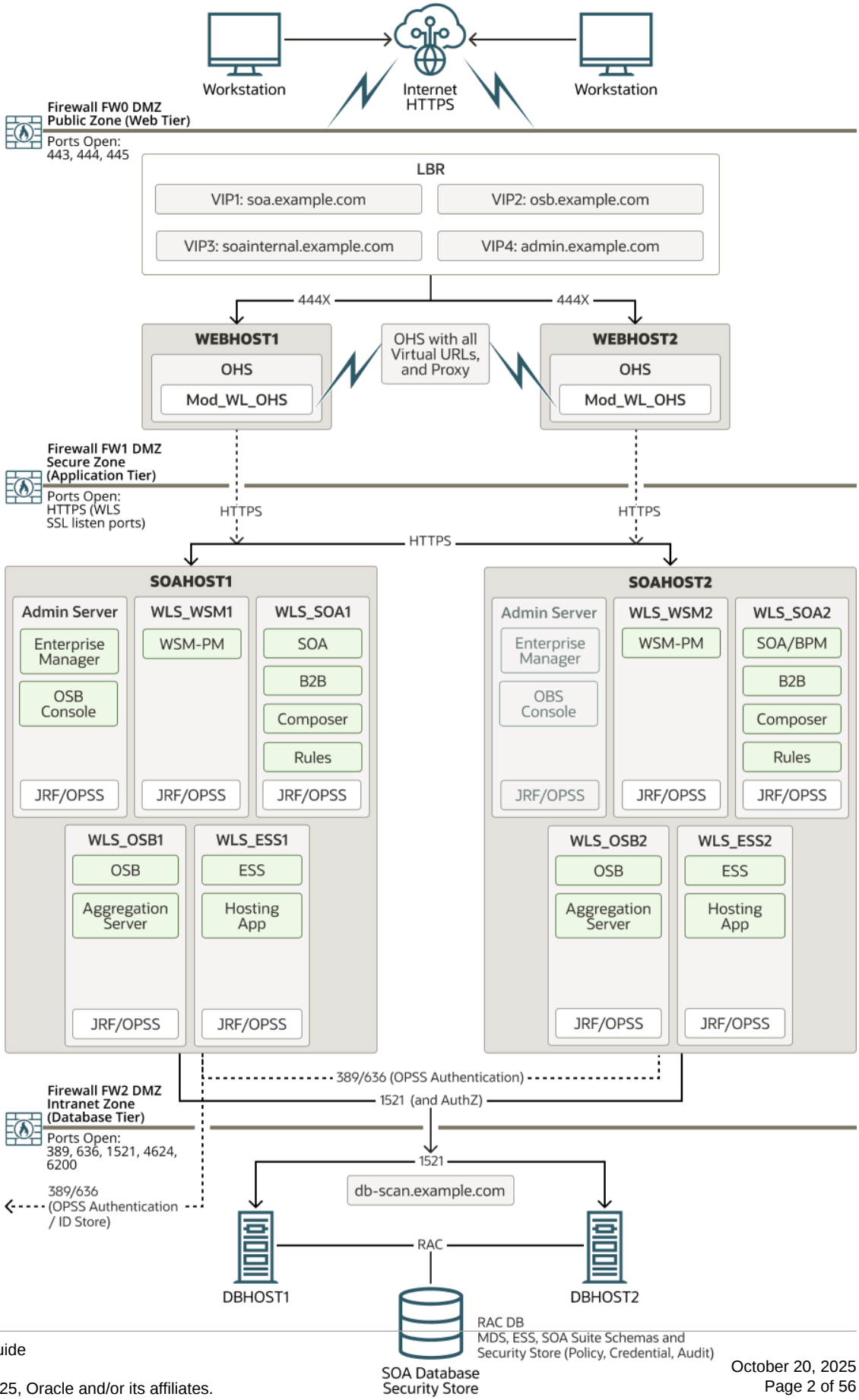
Notice that the primary system can also be on OCI. The framework can also be used to create a secondary copy in OCI even if it is created using manual procedures to install Oracle Fusion Middleware in OCI or it is using Oracle Marketplace Stacks. The setup procedures, replication technologies, and configuration in general are specific to OCI and precise implementation details are provided in these cases. For more information about the topology and how to use the different tools provided with it, see the WLS_HYDR framework page in *GitHub*.

2

Setting Up a Disaster Recovery Deployment

This chapter provides instructions about how to set up an Oracle Fusion Middleware Disaster Recovery topology for the Linux and UNIX operating systems. The procedures use the Oracle SOA Suite Enterprise Deployment (see Figure 2-1) in the examples to illustrate how to set up the Oracle Fusion Middleware Disaster Recovery solution for a specific case. After you understand how to set up the disaster recovery for the Oracle SOA Suite enterprise topology, use the same information to set up a disaster recovery system for the other enterprise deployments.

Figure 2-1 Deployment Used at Production and Secondary Sites for Oracle Fusion Middleware Disaster Recovery



Disaster Recovery Deployment Process Overview

Setting up a disaster protection system for Oracle Fusion Middleware involves the following procedures.

1. Planning Host Names

   Determines how the listen addresses used by the different components are going to be configured and virtualized so that no changes are required on the secondary when the configuration is propagated from the primary. Deciding how these addresses will be resolved may impact your system's manageability and the RTO of your disaster protection solutions.

2. Planning a File System Replication Strategy

   Determines what replication technology and approach is going to be used to meet the RTO and RPO requirements for the different artifacts that need to be copied from the primary to the secondary.

3. Preparing the Primary System for Disaster Protection

   For the optimum configuration, some changes may need to be applied in the primary in preparation for the disaster protection configuration. These changes primarily affect the storage configuration and database's address configuration.

4. Preparing the Secondary System for Disaster Protection

   The secondary system can also be set up based on the information provided in the Enterprise Deployment Guides. The infrastructure used by the Fusion Middleware system needs a specific configuration that will optimize the behavior of a disaster protection solution.

5. Setting Up the Secondary System

   Setting up the secondary system involves the following three tasks:

   a. Replicating the database by setting up the Data Guard for the database used by Fusion Middleware.

   b. Replicating the primary Fusion Middleware file systems to the secondary storage.

   c. Updating the TNS alias for the secondary's database address.

The following sections describe the procedures listed above in detail.

Planning Host Names

In a disaster recovery topology, the host name addresses used by the Fusion Middleware components must be resolvable to valid IP addresses in each site.

There are different types of host names that are used in this document:

- Physical Host Names

  These are the host names used to uniquely identify a node in a network. This is the address used to establish an SSH connection to that node. A physical host name is mapped to a fixed IP that is attached to the main Network Interface Controller (NIC) that the node uses. Physical host names are attached to a specific host and cannot be moved to a different one because they provide the main access point to the machine they belong to.

- Host Name Aliases or Virtual Hostnames

These are host names that can float from node to node. They can be assigned to a node and then disabled in that node and assigned to a different one. Virtual hostnames are used as listen address for processes and components that can move across different nodes. They enable accessors to abstract themselves from the physical host name where the process or components runs at a precise point in time.

As mentioned in the Enterprise Deployment Guides, Oracle recommends creating host name aliases along with the existing physical host names to decouple the system's configuration from those physical host names (which are different in each site and attached to specific nodes). Physical host names will typically differ between the primary and the secondary but the Fusion Middleware configuration uses virtual hostnames that are valid in both locations.

This section describes how to plan alias host names for the middle-tier hosts that use the Oracle Fusion Middleware instances at the production and secondary sites. It uses the Oracle SOA Suite Enterprise Deployment shown in Figure 2-1 for the host name examples. Each host at the production site has a peer host in the secondary site. The Fusion Middleware components of the peer host in the secondary site are configured with the same listen addresses and ports as their counterparts in the primary.

When you configure each component, use host-name-based configuration instead of IP-based configuration. For example, if you want an Oracle WebLogic Managed Server to listen on a specific IP address (for example, `172.11.2.113`), then use the host name `SOAHOST1.EXAMPLE.COM` in its configuration and then use your OS to resolve this host name to IP `172.11.2.113`. You must not include IPs directly in the configuration or in your WebLogic domain and applications or in the configuration of Fusion Middleware system components such as the Oracle HTTP Server.

The following section shows how to set up host names at the disaster recovery production and standby sites.

> **Note**
>
> In the examples listed, IP addresses for hosts at the initial production site have the format `172.11.x.x` and IP addresses for hosts at the initial standby site have the format `172.22.x.x`.

Sample Host Names for the Oracle SOA Suite Production and Standby Site Hosts

Figure 2-2 shows the Oracle Fusion Middleware Disaster Recovery topology that uses the Oracle SOA Suite enterprise deployment in the primary site.

Figure 2-2 Deployment Used at Production and Secondary Sites for Oracle Fusion Middleware Disaster Recovery

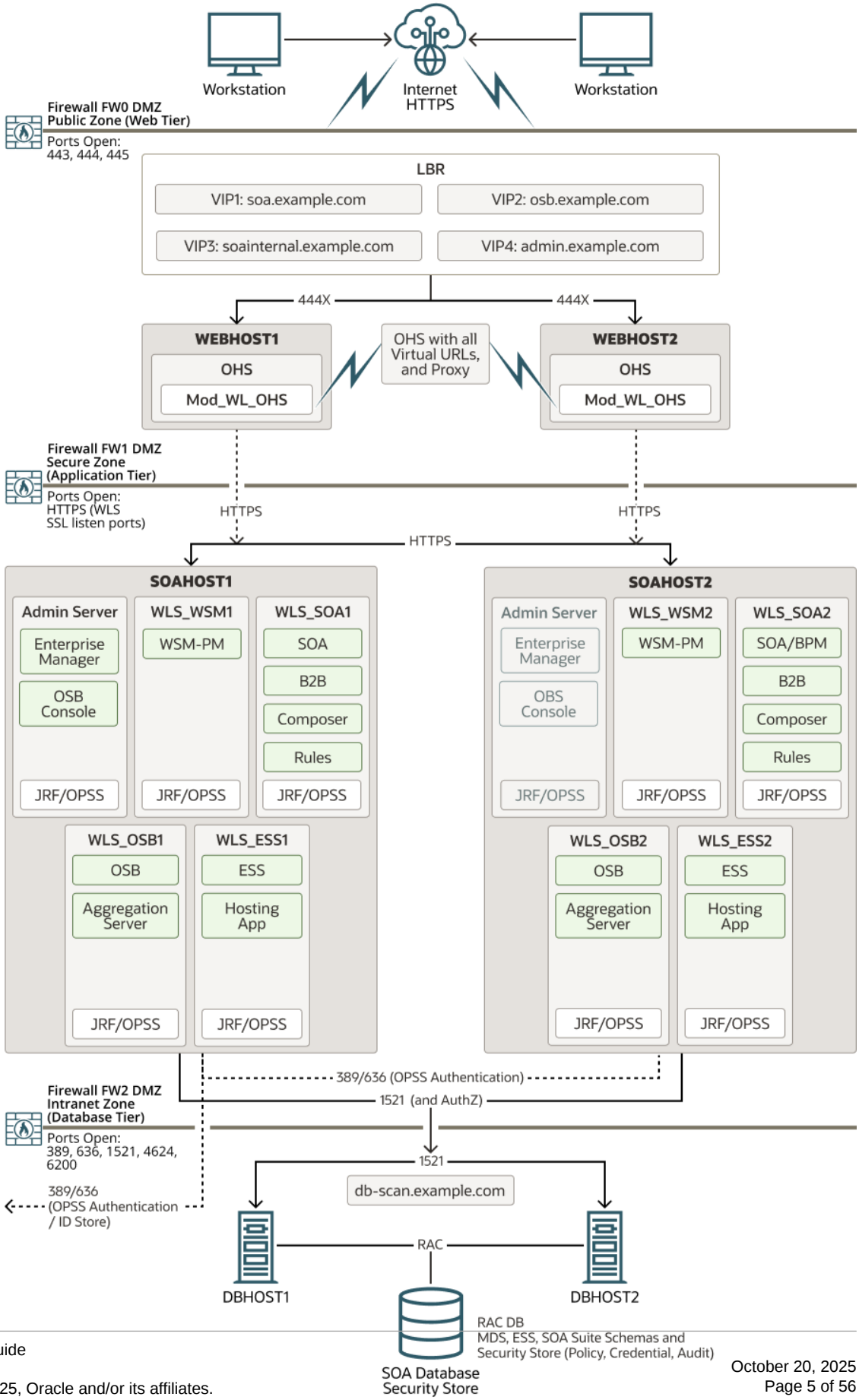


Table 2-1 lists the IP addresses, physical host names, and aliases that are used for the Oracle SOA Suite Enterprise Deployment Guide (EDG) primary site hosts. Figure 2-2 shows the configuration of the Oracle SOA Suite EDG deployment at the production site.

Table 2-1 IP Addresses and Physical Host Names for SOA Suite Production Site Hosts

IP Address	Physical Host Name	Host Name Alias
172.11.2.111	prweb1.example.com	WEBHOST1
172.11.2.112	prweb2.example.com	WEBHOST2
172.11.2.113	prsoa1.example.com	SOAHOST1
172.11.2.114	prsoa2.example.com	SOAHOST2

> **Note**
>
> This document uses WEBHOST1, WEBHOST2, SOAHOST1, SOAHOST2, and ADMINVHN as abstract placeholders for the actual alias host names. For example, in your system, SOAHOST1 may have a value of soahost1.example.com.

Figure 2-3 shows the physical host names that are used for the Oracle SOA Suite EDG deployment at the standby site.

Figure 2-3 Physical Host Names Used at Oracle SOA Suite Deployment Standby Site

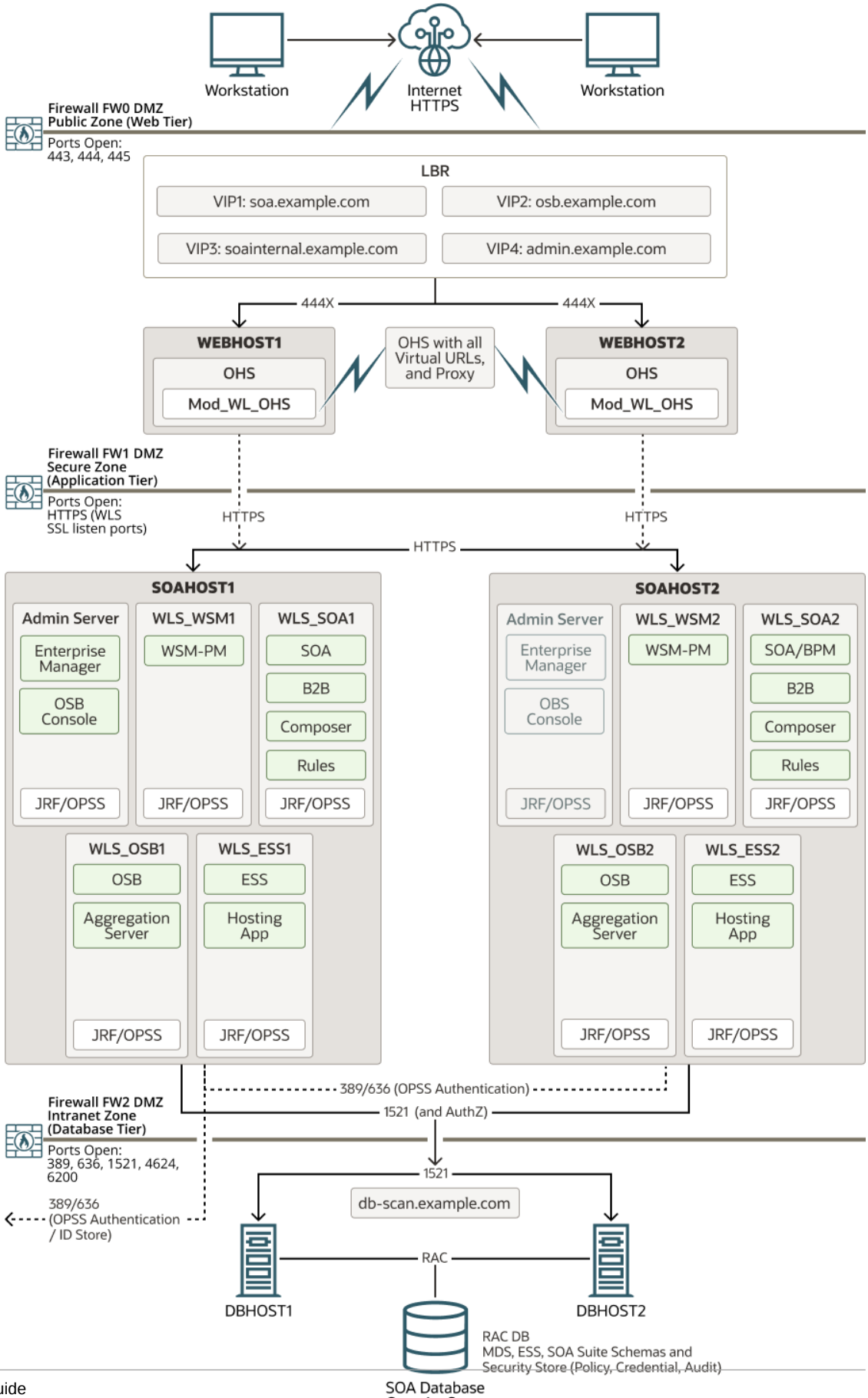


Table 2-2 lists the IP addresses, physical host names, and aliases that are used for the Oracle SOA Suite Enterprise Deployment Guide (EDG) deployment standby site hosts.

Table 2-2 IP Addresses and Physical Host Names for SOA Suite Standby Site Hosts

IP Address	Physical Host Name	Host Name Alias
172.22.2.111	stbyweb1.example.com	WEBHOST1
172.22.2.112	stbyweb2.example.com	WEBHOST2
172.22.2.113	stbysoa1.example.com	SOAHOST1
172.22.2.114	stbysoa2.example.com	SOAHOST2

> **Note**
>
> If you use separate DNS servers to resolve host names, then you can use the same physical host names for the production site hosts and standby site hosts and you do not need to define the alias host names on the standby site hosts. For more information about using separate DNS servers to resolve host names, see Resolving Host Names Using Separate DNS Servers.

Virtual IP Considerations

Besides using virtual hostnames to facilitate disaster protection configuration and in the context of local failures, some Oracle WebLogic servers may use a floating hostname as the listen address that is mapped to an IP that can be enabled in different nodes. These are called floating IPs and can be moved across different nodes inside the same region by disabling them in a network or virtual network interface in one node and enabling them in a different one. This allows failing over of the WebLogic servers and system components without reconfiguring their listen addresses which is the case for Oracle WebLogic Administration Server. In the Oracle WebLogic Cluster, VIPs and servers can also be moved across nodes by the WebLogic infrastructure and this mechanism is called WebLogic Server Migration.

> **Note**
>
> The WebLogic Administration Server cannot be clustered and therefore cannot use server migration.

In Oracle Fusion Middleware 14c, most applications support Oracle WebLogic Automatic Service Migration (ASM) where only the services running inside WebLogic (JMS, JTA) are migrated to other running WebLogic Servers. This improves recovery time because instead of entire servers being restarted, just few WebLogic subsystems are failed over. As a result, it is no longer necessary to reserve Virtual IPs for the Managed Servers in the domain (required for server migration). Those products or applications that do not support ASM, may require a VIP to configure server migration for the WebLogic Server where they run. Refer to your specific Fusion Middleware component's documentation to determine if it supports ASM. As indicated, a virtual hostname and its mapping VIP is needed for the administration server (Enterprise Deployment Guide recommendation). The WebLogic administration server cannot be clustered so it needs a mobile hostname that can be failed over to other nodes in a loss of host scenario.

Ensure that you provide the floating IP addresses for the administration server with the same virtual hostnames on the production site and the standby site.

Table 2-3 Floating IP Addresses

Virtual IP Address	Virtual Name	Alias Host Name
172.11.2.134	prsoa-vip.example.com (in production site)	ADMINVHN
172.22.2.134	stbysoa-vip.example.com (in standby site)	ADMINVHN

Host Name Resolution

Host name resolution means mapping a host name to the proper IP address required for accessing a node, a service, or an endpoint in a system. Host name resolution is a key aspect in all disaster protection systems because it allows virtualizing addresses and using primary sites' configuration in the secondary site without manipulations. A primary system may have its own host name resolution mechanics (Enterprise Deployment Guides do not prescribe a specific methodology in this domain). However, depending on the number of nodes, the number of virtual hostnames, and the components used in the Fusion Middleware topology, different host name resolution mechanisms based on a system's need is required.

In a disaster protection system, hostname aliases or virtual hostnames are used in the configuration of Fusion Middleware so that the configuration can be reused without any modifications in a different site by simply remapping the virtual hostnames to IPs that are valid in the secondary site. Host name resolution enables this dynamic mapping of hostnames to IPs and can be configured using any of the following procedures:

- Resolving Host Names Locally

  Local host name resolution uses the host name to IP address mapping that is specified by the `/etc/hosts` file on each host.

  For more information about using the `/etc/hosts` file to implement local host name file resolution, see Resolving Host Names Locally .

- Resolving Host Names Using DNS

  A DNS server is a dedicated server or a service that provides DNS name resolution in an IP network. You can use global DNS services (common to primary and secondary) or use separate DNS services in each location.

  For more information about the two methods used for implementing DNS server host name resolution, see Resolving Host Names Using Separate DNS Servers and Resolving Host Names Using a Global DNS Server.

You must determine the method of host name resolution that you will use for your Oracle Fusion Middleware Disaster Recovery topology when you plan the deployment of the topology. Most site administrators use a combination of these resolution methods in a precedence order to manage host names.The Oracle Fusion Middleware hosts and the shared storage system for each site must be able to communicate with each other.

Host Name Resolution Precedence

To determine the host name resolution method used by a particular host, search for the value of the `hosts` parameter in the `/etc/nsswitch.conf` file on the host.

If you want to resolve host names locally on the host, make the `files` entry the first entry for the `hosts` parameter, as shown in Example 2-1. When `files` is the first entry for the `hosts` parameter, entries in the host `/etc/hosts` file are first used to resolve host names.

If you want to resolve host names by using DNS on the host, make the `dns` entry the first entry for the `hosts` parameter, as shown in Example 2-2. When `dns` is the first entry for the `hosts` parameter, DNS server entries are first used to resolve host names.

For simplicity and consistency, Oracle recommends that all the hosts within a site (production or standby site) should use the same host name resolution method (resolving host names locally or resolving host names using separate DNS servers or a global DNS server).

The recommendations in the following sections are high-level that you can adapt to meet the host name resolution standards used by your enterprise.

Example 2-1 Specifying the Use of Local Host Name Resolution

```
hosts:   files   dns   nis
```

Example 2-2 Specifying the Use of DNS Host Name Resolution

```
hosts:   dns   files   nis
```

Resolving Host Names Locally

Local host name resolution uses the host name for IP mapping that is defined in the `/etc/hosts` file of a host.

When you resolve host names for your disaster recovery topology in this way, use the following procedure:

1. Ensure that the `hosts` parameter in the `/etc/nsswitch.conf` file on all the production site and standby site hosts is as shown below:

   ```
   hosts:   files   dns   nis
   ```

2. The `/etc/hosts` file entries on the hosts of the production site should have their physical host names mapped to their IP addresses along with the hostname aliases used by the Fusion Middleware components. For simplicity and ease of maintenance, Oracle recommends you to provide the same entries on all the hosts of the production site. Example 2-3 shows the `/etc/hosts` file for the production site of a SOA enterprise deployment topology.

3. The `/etc/hosts` file entries on the hosts of the standby site should have their physical host names mapped to their IP addresses along with the hostnames aliases of their corresponding peer on the production site defined as the alias host names. For simplicity and ease of maintenance, Oracle recommends that you provide the same entries on all the hosts of the standby site. Example 2-4 shows the `/etc/hosts` file for the standby site of a SOA enterprise deployment topology.

4. After you set up host name resolution by using `/etc/host` file entries, use the `ping` command to test host name resolution. For a system configured with static IP addressing and the `/etc/hosts` file entries shown in Example 2-3, a `ping webhost1` command on the production site returns the correct IP address (`172.11.2.111`) and indicates that the hostname is fully resolvable.

5. Similarly, for a system configured with static IP addressing and the `/etc/hosts` file entries shown in Example 2-4, a `ping webhost1` command on the standby site returns the correct IP address (`172.22.2.111`) and it shows that the name WEBHOST1 is associated with that IP address.

Example 2-3 Adding /etc/hosts File Entries for a Production Site Host

```
127.0.0.1        localhost.localdomain     localhost
172.11.2.134     prsoa-vip.example.com     prsoa-vip      ADMINVHN.EXAMPLE.COM
ADMINVHN
172.11.2.111     prweb1.example.com      prweb1      WEBHOST1.EXAMPLE.COM    WEBHOST1
172.11.2.112     prweb2.example.com      prweb2      WEBHOST2.EXAMPLE.COM    WEBHOST2
172.11.2.113     prsoa1.example.com      prsoa1      SOAHOST1.EXAMPLE.COM    SOAHOST1
172.11.2.114     prsoa2.example.com      prsoa2      SOAHOST2.EXAMPLE.COM    SOAHOST2
```

Example 2-4 Adding /etc/hosts File Entries for a Standby Site Host

```
127.0.0.1          localhost.localdomain       localhost
172.22.2.134       stbysoa-vip.example.com     stbysoa-vip   ADMINVHN.EXAMPLE.COM
ADMINVHN
172.22.2.111       stbyweb1.example.com     stbyweb1     WEBHOST1.EXAMPLE.COM    WEBHOST1
172.22.2.112       stbyweb2.example.com     stbyweb2     WEBHOST2.EXAMPLE.COM    WEBHOST2
172.22.2.113       stbysoa1.example.com     stbysoa1     SOAHOST1.EXAMPLE.COM    SOAHOST1
172.22.2.114       stbysoa2.example.com     stbysoa2     SOAHOST2.EXAMPLE.COM    SOAHOST2
```

> **Note**
>
> The subnets in the production site and standby site are different.

Resolving Host Names Using Separate DNS Servers

You can also use separate DNS servers to resolve host names for your disaster recovery topology (one in the primary and another in the secondary site).

When you use separate DNS servers to resolve host names for your disaster recovery topology, use the following procedure:

1. Ensure that the `hosts` parameter in the `/etc/nsswitch.conf` file on all the production site and standby site hosts is as shown below:

   ```
   hosts:   dns   files   nis
   ```

2. The DNS servers on the production and standby site must not be aware of each other and must contain entries for host names used within their own site.
3. The DNS server entries on the production site should have the physical and alias host names mapped to their IP addresses. Example 2-5 shows the DNS server entries for the production site of a SOA enterprise deployment topology.
4. The DNS server entries on the standby site should have the physical and alias host names mapped to their IP addresses. Example 2-6 shows the DNS server entries for the standby site of a SOA enterprise deployment topology.
5. Ensure that there are no entries in the `/etc/hosts` file for any host at the production or the standby site.
   a. Test the host name resolution by using the `ping` command. For a system configured with the production site DNS entries as shown in Example 2-5, a `ping webhost1` command on the production site returns the correct IP address (`172.11.2.111`) and indicates that the host name is fully qualified.
   b. Similarly, for a system configured with the standby site DNS entries shown in Example 2-6, a `ping webhost1` command on the standby site returns the correct IP address (`172.22.2.111`) and indicates that the host name is fully qualified.

Example 2-5 DNS Entries for a Production Site Host in a Separate DNS Servers Configuration

```
PRSOA-VIP.EXAMPLE.COM    IN    A    172.11.2.134
PRWEB1.EXAMPLE.COM       IN    A    172.11.2.111
PRWEB2.EXAMPLE.COM       IN    A    172.11.2.112
PRSOA1.EXAMPLE.COM       IN    A    172.11.2.113
PRSOA2.EXAMPLE.COM       IN    A    172.11.2.114

ADMINVHN.EXAMPLE.COM    IN    A    172.11.2.134
WEBHOST1.EXAMPLE.COM    IN    A    172.11.2.111
WEBHOST2.EXAMPLE.COM    IN    A    172.11.2.112
SOAHOST1.EXAMPLE.COM    IN    A    172.11.2.113
SOAHOST2.EXAMPLE.COM    IN    A    172.11.2.114
```

Example 2-6 DNS Entries for a Standby Site Host in a Separate DNS Servers Configuration

```
STBYSOA-VIP.EXAMPLE.COM   IN    A    172.22.2.134
STBYWEB1.EXAMPLE.COM      IN    A    172.22.2.111
STBYWEB2.EXAMPLE.COM      IN    A    172.22.2.112
STBYSOA1.EXAMPLE.COM      IN    A    172.22.2.113
STBYSOA2.EXAMPLE.COM      IN    A    172.22.2.114

ADMINVHN.EXAMPLE.COM       IN    A    172.22.2.134
WEBHOST1.EXAMPLE.COM       IN    A    172.22.2.111
WEBHOST2.EXAMPLE.COM       IN    A    172.22.2.112
SOAHOST1.EXAMPLE.COM       IN    A    172.22.2.113
SOAHOST2.EXAMPLE.COM       IN    A    172.22.2.114
```

> **Note**
>
> If you use separate DNS servers to resolve host names, then you can use the same host names for the production site hosts and standby site hosts. You do not need to define the alias host names.

Resolving Host Names Using a Global DNS Server

You can use a global DNS server to resolve host names for your disaster recovery topology as an alternative to the two previous options.

The term global DNS server refers to a disaster recovery topology, where a single DNS server is used for both the production and standby site.

When you use a global DNS server to resolve host names for your disaster recovery topology, use the following procedure:

1. Using a global DNS server, involves updating records on switchover. Notice that this may impact your RTO depending on how your DNS server is managed and how Time to Live (TTL) can be adjusted during switchover.
2. When you use a global DNS server, use a combination of local host name resolution and DNS host name resolution.
3. In this example, it is assumed that the production site uses DNS host name resolution and the standby site uses local host name resolution.
4. The global DNS server should have the entries for both the production and the standby site hosts. Example 2-7 shows the entries for a SOA enterprise deployment topology.

5. Ensure that the `hosts` parameter in the `/etc/nsswitch.conf` file on all the production site hosts is as shown below:

   ```
   hosts:   dns   files   nis
   ```

6. Ensure that the `hosts` parameter in the `/etc/nsswitch.conf` file on all the standby site hosts is as shown below:

   ```
   hosts:   files   dns   nis
   ```

7. The `/etc/hosts` file entries on the hosts of the standby site should have their physical host names mapped to their IP addresses along with the physical host names of their corresponding peer on the production site defined as the alias hostnames. For simplicity and ease of maintenance, Oracle recommends that you provide the same entries on all the hosts of the standby site. Example 2-9 shows the `/etc/hosts` file for the production site of a SOA enterprise deployment topology.

   a. Test the host name resolution by using the `ping` command. A `ping webhost1` command on the production site returns the correct IP address (`172.11.2.111`) and indicates that the host name is fully qualified.

   b. Similarly, a `ping webhost1` command on the standby site returns the correct IP address (`172.22.2.111`) and indicates that the host name is fully qualified.

Example 2-7 DNS entries for production site and standby site hosts when using a global DNS server configuration during normal operations (when the workload is sustained by the original primary)

```
PRSOA-VIP.EXAMPLE.COM      IN A 172.11.2.134
PRWEB1.EXAMPLE.COM         IN A 172.11.2.111
PRWEB2.EXAMPLE.COM         IN A 172.11.2.112
PRSOA1.EXAMPLE.COM         IN A 172.11.2.113
PRSOA2.EXAMPLE.COM         IN A 172.11.2.114

STBYSOA-VIP.EXAMPLE.COM    IN A 172.22.2.134
STBYWEB1.EXAMPLE.COM       IN A 172.22.2.111
STBYWEB2.EXAMPLE.COM       IN A 172.22.2.112
STBYSOA1.EXAMPLE.COM       IN A 172.22.2.113
STBYSOA2.EXAMPLE.COM       IN A 172.22.2.114

ADMINVHN.EXAMPLE.COM       IN A 172.11.2.134
WEBHOST1.EXAMPLE.COM       IN A 172.11.2.111
WEBHOST2.EXAMPLE.COM       IN A 172.11.2.112
SOAHOST1.EXAMPLE.COM       IN A 172.11.2.113
SOAHOST2.EXAMPLE.COM       IN A 172.11.2.114
```

Example 2-8 DNS entries for production site and standby site hosts when using a global DNS server configuration, after a switchover to standby site

```
PRSOA-VIP.EXAMPLE.COM      IN A 172.11.2.134
PRWEB1.EXAMPLE.COM         IN A 172.11.2.111
PRWEB2.EXAMPLE.COM         IN A 172.11.2.112
PRSOA1.EXAMPLE.COM         IN A 172.11.2.113
PRSOA2.EXAMPLE.COM         IN A 172.11.2.114

STBYSOA-VIP.EXAMPLE.COM    IN A 172.22.2.134
STBYWEB1.EXAMPLE.COM       IN A 172.22.2.111
STBYWEB2.EXAMPLE.COM       IN A 172.22.2.112
STBYSOA1.EXAMPLE.COM       IN A 172.22.2.113
STBYSOA2.EXAMPLE.COM       IN A 172.22.2.114

ADMINVHN.EXAMPLE.COM       IN A 172.22.2.134
WEBHOST1.EXAMPLE.COM       IN A 172.22.2.111
```

```
WEBHOST2.EXAMPLE.COM       IN A 172.22.2.112
SOAHOST1.EXAMPLE.COM       IN A 172.22.2.113
SOAHOST2.EXAMPLE.COM       IN A 172.22.2.114
```

Example 2-9 Production site /etc/hosts file entries when using a global DNS server configuration

```
127.0.0.1       localhost.localdomain     localhost
172.11.2.134    prsoa-vip.example.com     prsoa-vip    ADMINVHN.EXAMPLE.COM     ADMINVHN
172.11.2.111    prweb1.example.com        prweb1       WEBHOST1.EXAMPLE.COM     WEBHOST1
172.11.2.112    prweb2.example.com        prweb2       WEBHOST2.EXAMPLE.COM     WEBHOST2
172.11.2.113    prsoa1.example.com        prsoa1       SOAHOST1.EXAMPLE.COM     SOAHOST1
172.11.2.114    prsoa2.example.com        prsoa2       SOAHOST2.EXAMPLE.COM     SOAHOST2
```

Example 2-10 Standby Site /etc/hosts File Entries when using a global DNS server configuration

```
127.0.0.1      localhost.localdomain        localhost
172.22.2.134   stbysoa-vip.example.com      stbysoa-vip    ADMINVHN.EXAMPLE.COM
ADMINVHN
172.22.2.111   stbyweb1.example.com         stbyweb1       WEBHOST1.EXAMPLE.COM
WEBHOST1
172.22.2.112   stbyweb2.example.com         stbyweb2       WEBHOST2.EXAMPLE.COM
WEBHOST2
172.22.2.113   stbysoa1.example.com         stbysoa1       SOAHOST1.EXAMPLE.COM
SOAHOST1
172.22.2.114   stbysoa2.example.com         stbysoa2       SOAHOST2.EXAMPLE.COM
SOAHOST2
```

Testing the Host Name Resolution

Validate the host name assignment by connecting to each host at the production site. Use the `ping` command to ensure that the host can locate the other hosts at the production site.

External T3/T3s Clients Considerations

Systems directly accessing the WebLogic servers in a topology need to be aware of the listen address that is used by those WebLogic servers. An appropriate host name resolution needs to be provided to those clients so that the hostname alias used by the servers as listen address is correctly resolved. This is also applicable when performing deployments from Oracle JDeveloper or other development tools and scripts. Using the example addresses provided in the EDG, the client hosting Oracle Jdeveloper needs to map the `SOAHOSTx` and `ADMINVHN` aliases (corresponding to different WebLogic servers and administration server) to correct IP addresses for management and RMI operations to succeed. Depending on how controllable and how many external clients are accessing the system, a different host name resolution approach may be better for an optimum RTO and simplified management.

> **Note**
>
> This section does not apply to T3/T3s clients that reside in the same WebLogic domain as the T3/T3s servers. When connecting to a cluster in the same domain, internal clients can utilize the T3/T3s cluster syntax such as `cluster:t3://cluster_name`. You can utilize this cluster syntax, only when the clusters are in the same domain.

WebLogic uses the T3/T3s protocol for Remote Method Invocation (RMI). It is used by several WebLogic services such as JMS, EJB, JTA, JNDI, JMX and WLST. The external T3/T3s clients that do not run in the same domain as the T3/T3s servers, can use different ways to connect to a WebLogic cluster. For more information, see Using WebLogic RMI with T3 Protocol.

External T3/T3s clients can connect directly to WebLogic server channels (default or custom) that listen to the T3/T3s requests. The connection URL configured in the client's provider property contains the list of all the servers and ports of the cluster. For example,

```
t3://host1.example.com:9073,host2.example.com:9073
```

The external T3/T3s clients can also access the T3/T3s services through a TCP Load Balancer (LBR). In this scenario, the client provides URL points to the load balancer service and port. The requests are then load balanced to the WebLogic Server's T3/T3s ports. In the initial contact for the JNDI context retrieval, the external clients connect to a WebLogic Server through the load balancer and download the T3 client stubs. These stubs contain the connect information that is used for the subsequent requests.

In general, the WebLogic T3/T3s is TCP/IP based so it can support the TCP load balancing when services are homogeneous such as JMS and EJB. For example, a JMS front-end can be configured in a WebLogic cluster in which remote JMS clients can connect to any cluster member. By contrast, a JTA subsystem cannot safely use TCP load balancing in transactions that span across multiple WebLogic domains. The JTA transaction coordinator must establish a direct RMI connection to the server instance that acts as a sub coordinator of the transaction when that transaction is either committed or rolled back. Due to this, the load balancer is normally used only for the JNDI `initial context` retrieval. The WebLogic Server load balancing system controls the future T3/T3s requests, which connect to the WebLogic managed servers' addresses and ports (default or custom channels) indicated in the stubs retrieved during this initial context retrieval.

This section explains how to manage T3/T3s clients and its impact on disaster protection configuration. It also explains how to use the load balancer for all T3/T3s communications (both initial and subsequent requests) when JTA is not used.

> **Note**
>
> External clients can access the T3 services using the T3 tunneling over HTTP. This approach creates a HTTP session for each RMI session and uses standard HTTP protocols to transfer the session ID back and forth between the client and the server. This introduces some overhead and is used less frequently.

Different Approaches for Accessing T3/T3s Services

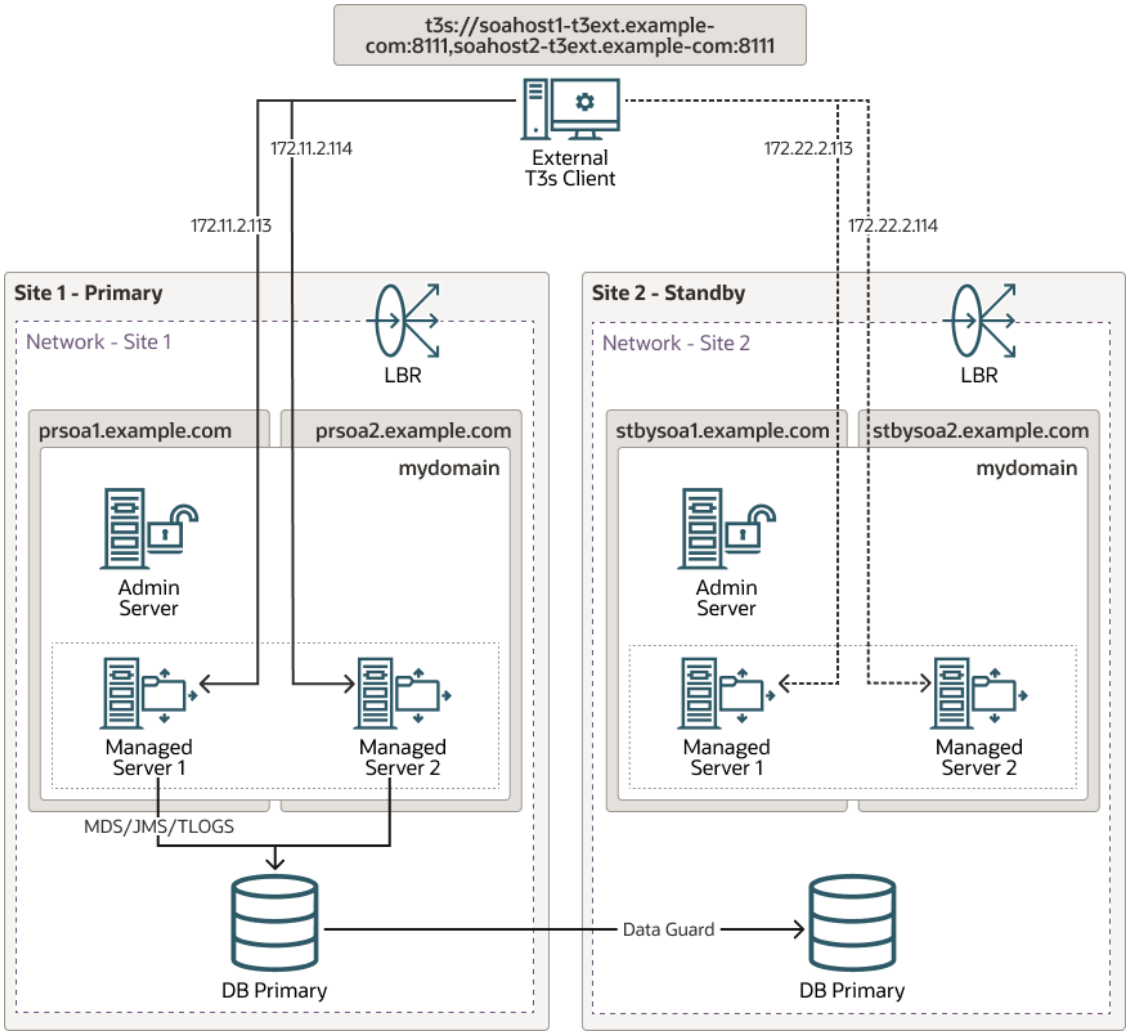
A disaster recovery scenario presents specific aspects which affect the host name resolution configuration of external T3/T3s clients. This topic explains different approaches that you can use when accessing T3/T3s services from external clients and how to manage them in a disaster recovery scenario. For each approach a configuration example is provided and the advantages and disadvantages while managing clients and switchover are explained.

> **Note**
>
> These approaches apply to a disaster recovery scenario that complies with the MAA guidelines for Fusion Middleware, where the secondary domain configuration is a replica of the primary system.

Direct T3/T3s Using Default Channels

In this approach, the external T3/T3s client connects directly to the WebLogic managed server's default channels. These default channels listen in the `Listen Address` and `Listen Port` specified in the general configuration of each WebLogic managed server.

Figure 2-4 Direct T3/T3s Using Default Channels



Configuration

- The provider URL in the T3 client uses the list of the WebLogic servers' default listen addresses and ports.

Example: In the following disaster recovery example, the listener addresses of the WebLogic servers are the primary hostnames:

```
t3://soahost1.example.com:8001,soahost2.example.com:8001
```

In case you are using the T3s protocol, the port must be set to the server's `SSL Listen Port`.

Example:

```
t3s://soahost1.example.com:8002,soahost2.example.com:8002
```

- External clients must resolve these hostnames with the primary host's IPs. These IPs must be reachable from the client. It is possible to use Network Address Translating (NAT), if there is a NAT IP address for each server. In that case, the client will resolve each server name with the appropriate NAT IP for the server.

  Example: Naming Resolution at the External Client Side

  This is achievable through `local /etc/hosts` or formal DNS server resolution.

  ```
  172.11.2.113    soahost1.example.com
  172.11.2.114    soahost2.example.com
  ```

Switchover Behavior

In a switchover scenario, there is no need to update the client's provider URL. You need to perform an update of the entries in the DNS (or client's `/etc/hosts`) of the client. After the switchover, the names used to connect to the servers must resolve with the IPs of the secondary servers.

Example: Naming Resolution at the External Client Side After a Switchover

```
172.22.2.113    soahost1.example.com
172.22.2.114    soahost2.example.com
```

Advantages

- You do not require additional configuration either at the server's side or at the front-end tier. All you require is opening the specific ports to the client.

Disadvantages

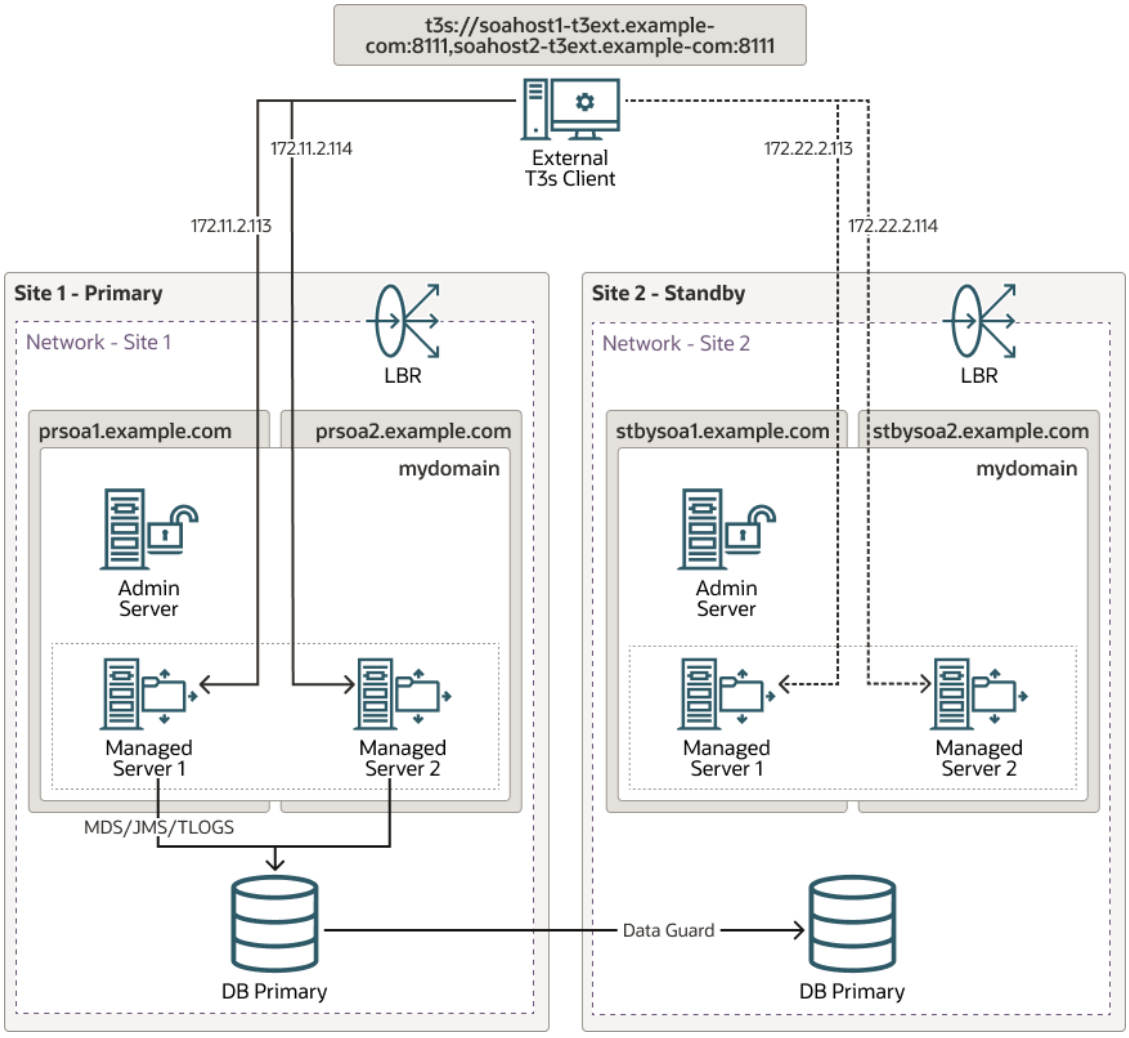
- When a switchover happens, you need to update the host resolver (DNS or the clients's `/etc/hosts`) to alter the resolution of all the hostnames that the external client uses to connect to the managed servers. For more information about implications of the client's cache, see About T3/T3s Client's DNS Cache.
- Clients must be able to resolve and reach the hostnames set in WebLogic's `Listen Address`. It is not possible to use alternate names because the default channels use the WebLogic Server's default `Listen Address`.
- The WebLogic default channels listen for T3(s) and also HTTP(s) requests. You cannot disable this setting. If you open the default port for the clients, direct HTTP(s) to the server is also allowed which can result in security concerns.
- You need to modify the client's provider URL if you have to scale out or scale in a WebLogic cluster. The first contact in a T3/T3s invocation uses only the list in the client's

provider URL. So, even without updating the list, the subsequent requests can connect to any server of the cluster as the recovered stubs list all of the members. It is recommended that the client's provider URL matches the real list of the servers for failover purposes in the first contact.

Direct T3/T3s Using Custom Channels

In this approach, the external T3/T3s client connects directly to custom channels defined in the WebLogic servers of the cluster. The `Listen Address`, the `External Listen Address`, and the `External Port` are customizable values in the custom channels. These values can differ from the WebLogic server's default listen values.

Figure 2-5 Direct T3/T3s Using Custom Channels



Once the T3/T3s external client is connected to the server during the initial context retrieval, the subsequent T3 calls connects directly to one of the listen addresses and ports configured in the custom channels as `External Listen address` and `External Port`. These requests will be load balanced according to the mechanism specified in the connection factory defined in the WebLogic and used by the client.

This approach is similar to Direct T3/T3s Using Default Channels with the exception that you can customize the addresses and ports used in T3/T3s calls when using WebLogic custom channels.

Configuration

- Each WebLogic server has the appropriate custom channels configured and these custom channel uses a unique `External Listen address` that points to that server node. Table 2-4 and Table 2-5 provides example of Custom Channel in Server 1 and Server 2.

Table 2-4 Custom Channel in Server 1

Name	Protocol	Enabled	Listen Address	Listen Port	Public Address	Public Port
t3_external_channel	t3	true	soahost1.example.com	8111	soahost1-t3ext.example.com	8111

Table 2-5 Custom Channel in Server 2

Name	Protocol	Enabled	Listen Address	Listen Port	Public Address	Public Port
t3_external_channel	t3	true	soahost2.example.com	8111	soahost2-t3ext.example.com	8111

- The external T3 client's provider URL contains the list of the external address and port of the custom channels.

  Example:

  ```
  t3://soahost1-t3ext.example.com:8111,soahost2-t3ext.example.com:8111
  ```

  If you are using the T3s protocol, you must create the custom channels with T3s protocol. Then the clients will connect using T3s protocol and appropriate port.

  Example:

  ```
  t3s://soahost1-t3ext.example.com:8112,soahost2-t3ext.example.com:8112
  ```

  In a T3s channel, you can add a specific SSL certificate for the name used as an `External Listen address`.

- The external T3/T3s client must resolve the custom channels' external hostnames with the primary host's IPs. These hostnames must be reachable from the client. It is possible to use NAT if there is a NAT address for each server. In that case, the client will resolve each server name with the appropriate NAT IP for the server.

  Example: Naming Resolution at the External Client Side

  ```
  172.11.2.113    soahost1-t3ext.example.com
  172.11.2.114    soahost2-t3ext.example.com
  ```

  With this approach, all the requests from the external client connect to `soahost1-t3ext.example.com` and `soahost2-t3ext.example.com`, both for the initial context retrieval

and for the subsequent calls. You can control these requests by using the WebLogic Server load balancing mechanism.

Switchover

In a switchover scenario, you do not have to update the client's provider URL. You must update the entries in the DNS or in the `/etc/hosts` of the client. After the switchover, the names used to connect to the servers must resolve with the IPs of the secondary servers.

Example: Naming Resolution at the External Client Side After a Switchover

```
172.22.2.113    soahost1-t3ext.example.com
172.22.2.114    soahost2-t3ext.example.com
```

Advantages

- This method allows using specific hostnames for the external T3/T3s communication different from the server's default listen address. This is useful if you do not want to expose the default server's `Listen Address` to the external clients for security reason.
- This method is useful if you are using NAT IPs and you do not want to resolve the servers' default listen addresses with different IPs internally and externally.
- This method is useful in case you want to use different names for external T3/T3s accesses just for organizational purposes. You can also use this method to isolate protocols in different interfaces or network routes.
- The protocol in the custom channel can only be limited to T3/T3s. The HTTP(s) protocol can be disabled in the custom channels.

Disadvantages

- When a switchover happens, you need to update the DNS or the clients's `/etc/hosts` at the external client side for all the hostnames used to connect to the managed servers. For more information about the implications of the client's cache, see About T3/T3s Client's DNS Cache.
- If you are using T3s, then you must create and configure specific SSL certificates for the external names in the custom channels to avoid SSL hostname verification errors in the client.
- You have to modify the client's provider URL if you have to scale out or scale in a WebLogic cluster. The first contact only uses the list in the client's provider URL. So, even without updating the list, the subsequent requests can connect to any server of the cluster as the recovered stubs list all of the members. It is always a good practice to ensure that the client's provider URL matches the real list of the servers for failover purposes in the first contact.

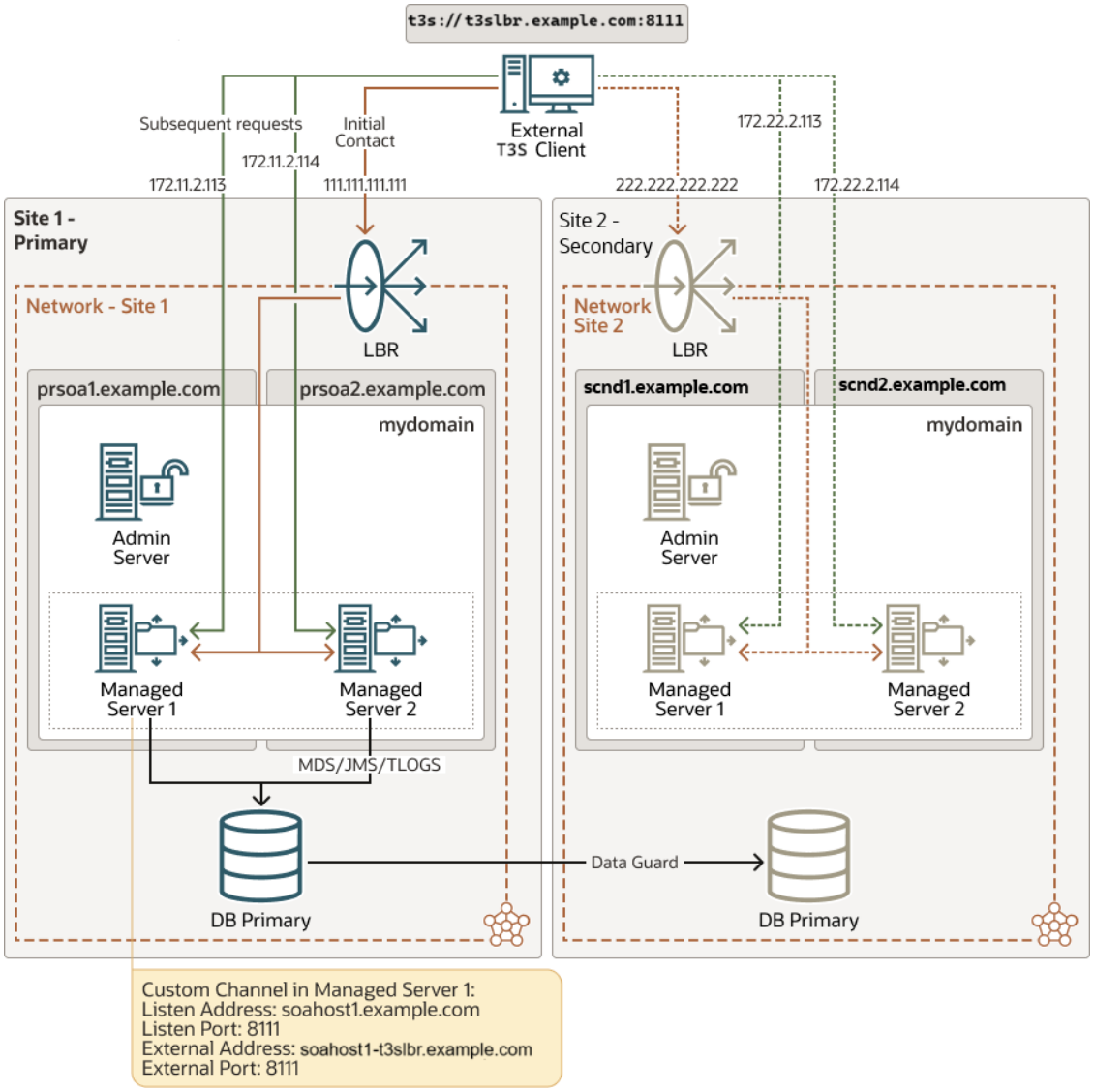
Using Load Balancer for Initial Lookup

In this scenario, the client's provider URL points to the load balancer address.

Direct T3/T3s Using Default Channels and Direct T3/T3s Using Custom Channels approaches can also use a load balancer for the initial context lookup.

However, the subsequent T3/T3s calls connects to WebLogic servers directly. If you use the default channel for these invocations, the request goes to the default channel's listen address and port. Similarly, if you use the custom channels then the subsequent request goes to the external listen address and port defined in the customs channels.

Figure 2-6 Using Load Balancer for Initial Lookup



Configuration

- You will need a TCP service in the load balancer to load balance the requests to the WebLogic Servers' T3/T3s ports (either to the default channels or to the custom channels when used).
- The external T3/T3s client's provider uses the front-end name and port of the load balancer as the point of contact.

  Example:

  ```
  t3://t3lbr.example.com:8111
  ```

- You can use default channels or custom channels as explained in Direct T3/T3s Using Default Channels and Direct T3/T3s Using Custom Channels scenarios. The external T3/T3s client must resolve the custom channels' external hostnames (or the default server's listeners hostnames, if you are using default channels) with the primary host's IPs. The client must be able to access them. It is possible to use NAT as long as there is a NAT

address for each server. In that case, the client will resolve each server name with the appropriate NAT IP for the server.

Example: Naming Resolution at the External Client Side

```
111.111.111.111    t3lbr.example.com
172.11.2.113    soahost1-t3ext.example.com
172.11.2.114    soahost2-t3ext.example.com
```

Switchover

In a switchover scenario, you do not have to update the client's provider URL. You must update the entries in the DNS or in the `/etc/hosts` used by the client so that they resolve with the IPs of the secondary site. The name used to connect to the load balancer must resolve with the IP of the secondary load balancer and the server names used in subsequent requests must point to the IPs of the secondary servers.

Example: Naming Resolution at the External Client Side After a Switchover

```
222.222.222.222    t3lbr.example.com
172.22.2.113    soahost1-t3ext.example.com
172.22.2.113    soahost2-t3ext.example.com
```

Advantages

You do not have to modify the client's provider URL if you add or remove the WebLogic Server nodes from the WebLogic cluster.

Disadvantages

- Despite using a load balancer in front, the client still needs to reach the servers directly.
- When a switchover happens, you need to update the DNS (or `/etc/hosts`) of servers' addresses at the external client side. For more information about the implications of the client's cache, see About T3/T3s Client's DNS Cache.
- The complexity of this method is higher for the T3s cases. The client connects both through the front-end LBR and directly to the server using a secure protocol. In this case, you will need to either skip the hostname verification at the client side or use an SSL certificate that is valid for *front* and *back* addresses (For example, wildcard or SAN certificates).

Using Load Balancer for All Traffic

In this approach, the initial context lookup goes through the load balancer and the subsequent connections. There are no direct connections from the external T3/T3s client to the servers.

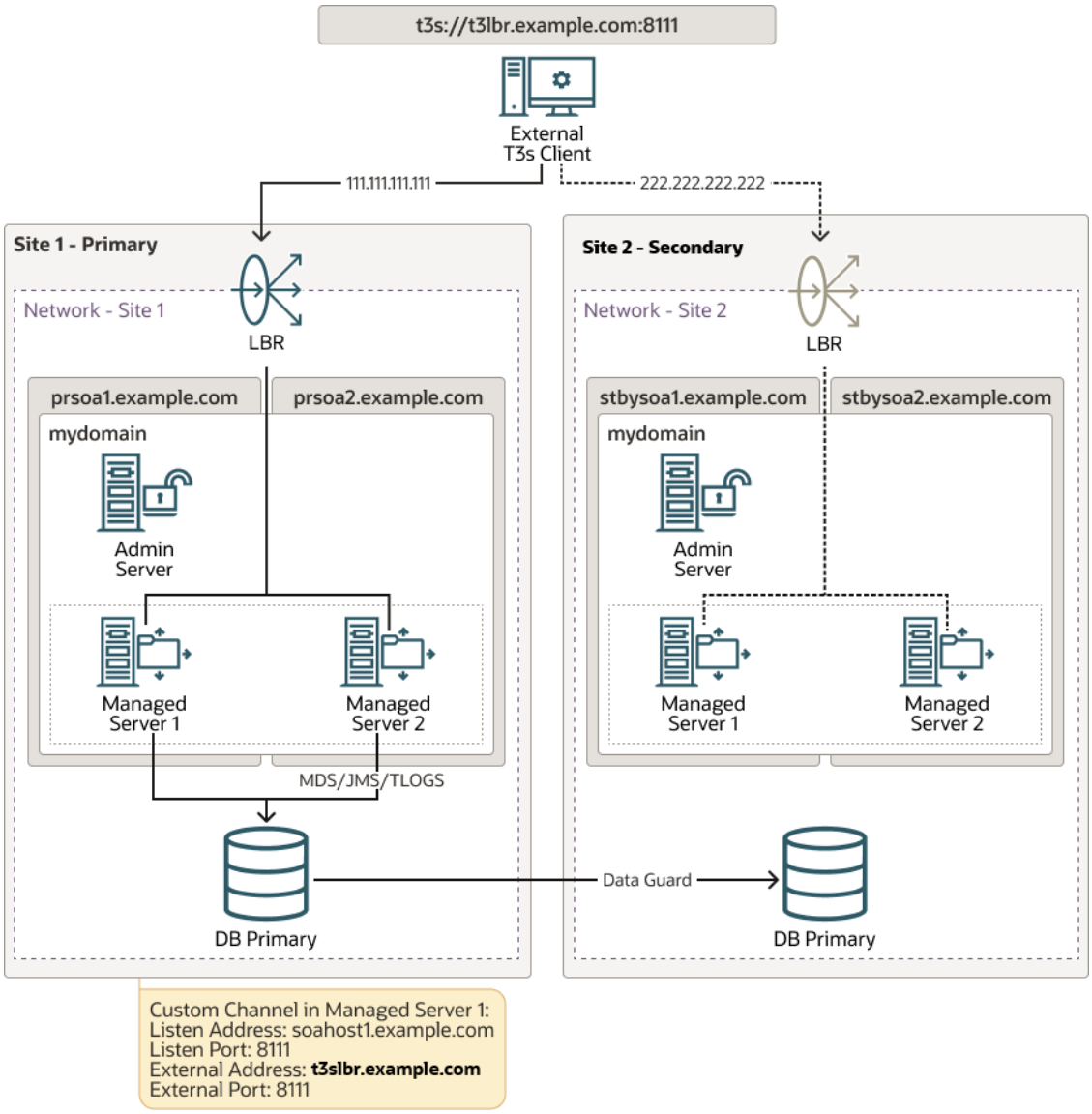
Oracle does not recommend using LBR for load balancing all types of T3/T3s communications. It is only recommended for initial context lookup. For more information, see *WebLogic RMI Integration with Load Balancers*. However, there are T3/T3s use cases, where you can use the load balancer for complete T3/T3s communication flow. WebLogic T3/T3s are TCP/IP-based protocols, so it can support TCP load balancing when services are homogeneous such as JMS and EJB. For example, you can configure an LBR front-ended JMS subsystem in a WebLogic cluster in which remote JMS clients can connect to any cluster member.

However, this approach will not work with external clients that use JTA connections. A JTA subsystem cannot safely use TCP load balancing in transactions that span across multiple WebLogic domains. When the transaction is either committed or rolled back the JTA

transaction coordinator must establish a direct RMI connection to the server instance that has been chosen as the transaction's sub coordinator.

This method is also not suitable for cases where you require direct connection to a specific server like JMX or WLST when you want to connect to a particular server only.

Figure 2-7 Using Load Balancer for All Traffic



Configuration

- The load balancer requires a TCP service to load balance the requests to the WebLogic servers' T3/T3s ports defined in the custom channels.
- The external T3/T3s client's provider URL uses the front-end name and port of the load balancer as the point of contact.

  Example:

  ```
  t3://t3lbr.example.com:8111
  ```

- The external client must resolve the load balancer address with the IP of the primary site's load balancer.

  Example:

  ```
  111.111.111.111  t3lbr.example.com
  ```

- WebLogic Server requires custom channels. Configure the external listen address and port of these custom channels with the load balancer's address and port. Table 2-6 and Table 2-7 provides examples of Custom Channel in Server 1 and Server 2.

Table 2-6 Custom Channel in Server 1

Name	Protocol	Enabled	Listen Address	Listen Port	Public Address	Public Port
t3_external_channel	t3	true	soahost1.example.com	8111	t3lbr.example.com	8111

Table 2-7 Custom Channel in Server 2

Name	Protocol	Enabled	Listen Address	Listen Port	Public Address	Public Port
t3_external_channel	t3	true	soahost2.example.com	8111	t3lbr.example.com	8111

Switchover

In a switchover scenario, you do not have to update the client's provider URL. You must update the entries in the DNS (or in the `/etc/hosts`) of the client. After the switchover, the names used to connect to the servers resolves with the IP of the secondary LBR service.

Example: Naming Resolution at the External Client Side After a Switchover

```
222.222.222.222  t3lbr.example.com
```

Advantages

- All the communication goes through the load balancer. The client only needs to know and reach the load balancer's service.
- If you have to either add or remove the WebLogic Server nodes from the WebLogic cluster, you do not have to modify the client's provider URL
- In case of a switchover, you only have to update the load balancer's front-end name in the DNS (or in the `/etc/hosts`).

> **Note**
>
> Although only one DNS name is updated, you need to refresh the client's DNS cache. For more information, see About T3/T3s Client's DNS Cache.

- In T3s cases, you can use the same SSL certificate in all the custom channels associated with the load balancer service's front-end name.

Disadvantages

- This approach is not suitable for all the T3/T3s cases. It also cannot be used in scenarios with JTA as JTA needs to explicitly connect to the server instance that acts as the sub coordinator of the transaction.

About T3/T3s Client's DNS Cache

All the approaches explained to access T3/T3s services mostly require a DNS update in disaster recovery switchover scenarios. Therefore, you must set the limit for DNS Cache TTL (Time to Live) in DNS server and client's specific cache.

The TTL limit in the DNS service is a setting that tells the DNS resolver how long to cache a query before requesting a new one. Note that the TTL value of the DNS entries will affect the effective RTO of the switchover. If the TTL value is high (for example, 20 minutes), the DNS change will take that time to be effective in the clients cache. Using lower TTL values will make this switchover faster, but this can cause overhead because the clients check the DNS more frequently. A good approach is to set the TTL to a low value temporarily (for example, one minute) before the DNS update. Once the change and switchover procedure is completed, set the TTL to the normal value again.

Besides the DNS server's TTL, `networkaddress.cache.ttl` Java property controls the Java clients' cache TTL. This Java property indicates the caching policy for successful name lookups from the name service. Specify the value as an integer to indicate the number of seconds to cache the successful lookup. Ensure you set a limit to the `networkaddress.cache.ttl` Java property so the client's Java cache does not cache the DNS entries forever or you might have to restart the client with each switchover.

Planning a File System Replication Strategy

Defining a file system replication strategy is important while designing a disaster protection system. Oracle does not recommend creating and managing the secondary system by repeating installations and configuration steps.

Applying changes first in the primary system (for example, deploying a data source) and then repeating the step in the secondary system is not a reliable approach. Even when using deployment scripts and automated pipelines, there is rarely a guarantee of transaction. This can lead to changes being applied only in the primary system and not propagated properly to the secondary system. Some configuration changes require WebLogic servers to be running and this may cause duplicated processing and undesired behavior. But more importantly, a dual change approach typically gives room to inconsistencies and management overhead that can affect the RTO and RPO of the disaster protection solution.

Instead, different file replication strategies can be used to replicate configuration changes across sites. File system replication is used in the initial setup and during the ongoing lifecycle of a system. Costs, management overhead, and RTO and RPO may have different implications during setup and ongoing lifecycle, so different replication approaches can be used in each case. Also, there are different data types involved in a Fusion Middleware topology and each one may also have different RTO and RPO needs depending on their size on the file system and how frequently they are generated and modified. You must first understand the different data types involved in a disaster protection topology to decide which replication strategy is suitable for your needs.

Data Tiers in a Fusion Middleware File System

All files involved in the Fusion Middleware system should be replicated from the primary to the secondary at the same point in time. However, different file types may need different replication frequency to simplify management costs and to reduce the total cost of ownership of a disaster protection system. This is important when you design the volumes and file systems that you want to use for replication. Some artifacts are static whereas others are dynamic.

Oracle Home and Oracle Inventory

An Oracle Home is the directory into which all the Oracle software is installed and is referenced by Fusion Middleware and WebLogic environment variables. Oracle Fusion Middleware allows you to create multiple Oracle WebLogic Server Managed Servers from one single binary file installation. The Oracle Fusion Middleware Enterprise Deployment Guides provide recommendations on how to configure storage to install and use Oracle Home so that there is no single point of failure in the Fusion Middleware system while optimizing the storage size and simplifying maintenance.

It is not necessary to install Oracle Fusion Middleware instances at the secondary site as it is installed at the production site. When the production site storage is replicated to the secondary site storage, the Oracle software installed on the production site volumes are replicated at the secondary site volumes.

A secondary system needs to behave exactly like the primary system when a failover or switchover occurs. It should admit patches and upgrades as a first-class installation. This means that when a failover or switchover occurs, the secondary system must use a standard inventory for patches and upgrades. To maintain consistency, it is required to replicate the Oracle Inventory with the same frequency as the Oracle Homes used by the different Fusion Middleware components (RTO, RPO, and consistency requirement of the Oracle Inventory must be the same as the one prescribed for Oracle Homes). The Oracle Inventory includes `oraInst.loc` and `oratab` files which are located in the `/etc` directory.

When an Oracle Home or a WebLogic home is shared by multiple servers in different nodes, Oracle recommends that you keep the Oracle Inventory and Oracle Home list in those nodes that are updated for consistency in the installations and application of patches. To update the inventory files in a node and attach an installation in a shared storage to it, use the `ORACLE_HOME/oui/bin/attachHome.sh` script.

Both Oracle Home and Oracle Inventory are static artifacts during runtime and typically require a low RTO (since they are the binaries from which Fusion Middleware domains run). It is not required to copy them across regions too frequently because they change only when patches and fixes are applied. From a consistency point of view, runtime artifacts are not usually too restrictive. For example, most patches will support previous configurations of a WebLogic Server domain. So even if you do not propagate a patch in a WebLogic Oracle Home across regions, the secondary domain configuration will work properly.

Configuration Artifacts

Configuration artifacts are files that change frequently and includes the following:

- WebLogic Domain Home: Domain directories of the Administration Server and the Managed Servers.
- Oracle instances of system components such as Oracle HTTP Server: Oracle Instance home directories.
- Application artifacts like `.ear` or `.war` files.

- Database artifacts like the MDS repository and the JDBC persistent stores definitions.
- Deployment plans used for updating technology adapters like the file and JMS adapters. They need to be saved in a location that is accessible to all nodes in the cluster where the artifacts are being deployed to.

Configuration artifacts change frequently depending on application updates. They require a low RTO and a high replication frequency. It is important to maintain consistency for configuration artifacts in different stores, else the applications may stop working after a restore. For example, WebLogic domain configuration reflecting a new JMS server must be aligned with the database table that it uses as persistent store. Replicating only the WebLogic domain configuration without replicating the pertaining table will cause a WebLogic Server failure.

Runtime Artifacts

Runtime artifacts are files generated by applications at runtime. For example, files generated by SOA's file or FTP Adapters, files managed by Oracle MFT, or any other information that applications generate through their business logic and that is stored directly on the file system. These files may change very frequently. Their RTO and RPO are purely driven by business needs. In some cases, these type of artifacts may need to be discarded after a short period of time. For example, a bid order that expires in a short period. In other cases, these files may contain transactional records of operations completed by an application that need to be preserved. How frequently they need to be replicated and how important it is to preserve this type of files in a disaster event is usually a business-driven decision.

Main Replication Options

Oracle verifies different approaches to replicate the different types of data used by Fusion Middleware deployments on the file system. Although there are other options, only the following ones are formally tested and can be utilized for Fusion Middleware deployments.

Storage Level Replication

The Oracle Fusion Middleware Disaster Recovery solution can use storage replication technology to replicate the secondary Oracle Fusion Middleware middle-tier primary system. This means using specific storage vendor technology to replicate storage volumes across regions. Typical solutions in this area involve creating an initial or baseline snapshot of the different volumes presented in the EDG and then manually, on schedule or continuously replicating changes incrementally to a secondary location.

Do not use storage replication technology to provide disaster protection for Oracle databases. Ensure that disaster protection for any database that is included in the Oracle Fusion Middleware Disaster Recovery production site is provided by Oracle Data Guard.

Ensure that your storage device's replication technology guarantees consistent replication across multiple volumes. Otherwise, problems may arise when different configuration and runtime artifacts used by the Fusion Middleware systems are replicated at different points in time. Most storage vendors guarantee consistency when multiple nodes are using different storage volumes and it is needed to take a backup of all of them (or replicate their contents) in a precise point in time. Volumes are grouped in a single logical unit frequently called *consistency volume group*, *consistency group*, or *volume group*. Volume groups provide a point in time copy of the different volumes used in an EDG so that a consistent snapshot of all of them is propagated to the secondary. Imagine a situation where a database module deployment resides in volume A while the application that uses it resides in volume B. Without consistency groups it could occur that a new version of the WAR for the application is replicated while the data source configuration it requires is not, thus causing a consistency problem in the secondary.

Storage level replication can be used as a general-purpose approach for all types of data in the EDG (Oracle Home/Oracle Inventory, configuration artifacts, and runtime artifacts). It is a valid approach both for the initial setup and ongoing replication cycles during the lifetime of a system.

Rsync

Rsync can be used as an alternative to using storage vendor-specific replication for replicating the primary Fusion Middleware system to the secondary system. Rsync is a versatile copying tool, that can copy locally or to/from another host over any remote shell. It offers a large number of options that control every aspect of its behavior and permits very flexible specification of the set of files to be copied.

Rsync implements delta-transfer algorithm which reduces the amount of data sent over the network by sending only the differences between the source files and the existing files in the destination. Due to these advantages and convenience, it is a widely used tool for backups and mirroring. To guarantee a secure copy in all cases, it is recommended using rsync over SSH.

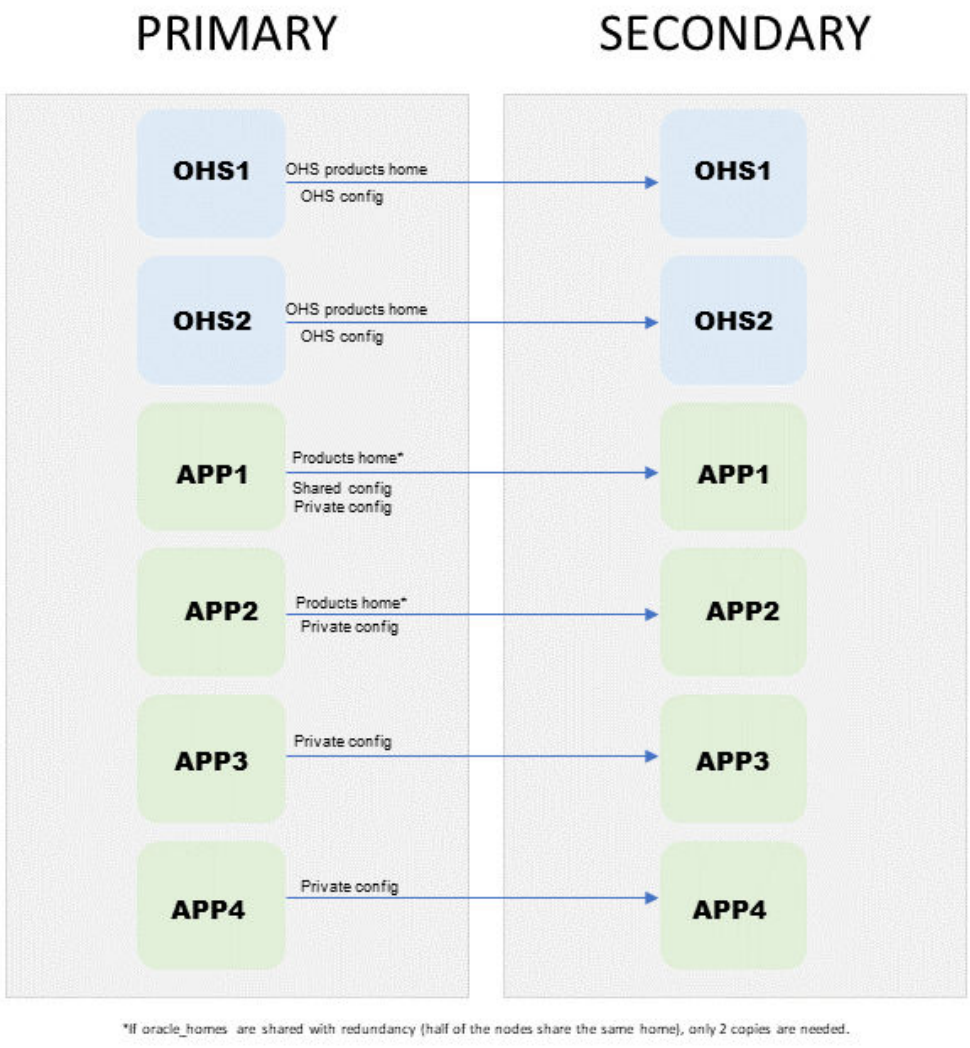
Although rsync is reliable and implements implicit retries, network outages and other connectivity issues can still cause failures in the file synchronization. Additionally, rsync does not guarantee consistency across different nodes. It does not provide a distributed point in time replication. Hence, rsync can be used as an alternative to the storage level replication under the following conditions:

- When the storage replication is not possible or feasible and/or does not meet the cost requirements.
- When primary and standby use a reliable and secure network connection for the copy.
- When checks are performed across the different nodes involved in the copy to ensure that it is valid.
- When the disaster recovery site is validated on a regular basis.

Rsync is present in most Linux distributions these days. You can install and set up `rsync` to enable replication of files from a production site host to its secondary site peer host. For instructions about how to install, set up, and use the utility, see the utility manual and *rsync*.

In a rsync replication approach, there are two main alternatives to copy from the primary to the secondary:

Figure 2-8 Peer-to-Peer Copy



In this case, the copy can be done directly from each host to its remote peer. Each node has SSH connectivity to its peer and uses rsync commands over SSH to replicate the primary system.

shows an overview of using rsync commands over SSH to replicate the primary system.

In this approach, each node uses individual scripts to rsync its Fusion Middleware installation and configuration to the remote peer node. Each node can cron the scripts. Different data types can have different schedules for the cron jobs. For example:

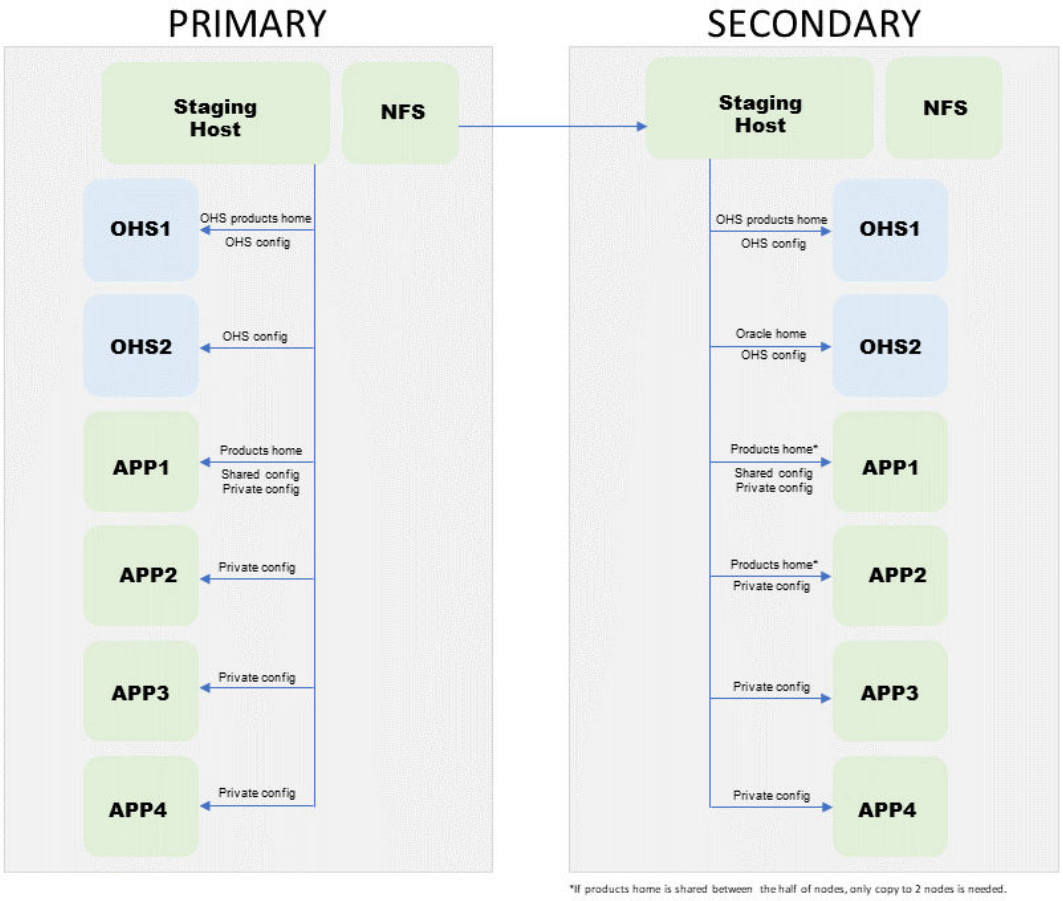
- Products once a month.
- Configuration once a day.

This is an easy setup that does not need additional hardware resources. However, it requires maintenance across many nodes since scripts are not centralized (for example, large clusters add more complexity to the solution). It is also difficult to guarantee consistency at a point in time since each node stores a separate copy. The consistency and validation of this approach can be improved using a central host to launch the scripts in each node (acts like a single scheduler) and to validate the copies.

Staging Location

In this approach, a staging location is used and typically a node acts as a coordinator connecting to each individual host that needs to be replicated. This *coordinator* copies the required configuration and binary data to a staging location. This can be on a shared storage attached to all nodes or through a staging host. This staging location can be validated without affecting the primary or the secondary nodes. This approach offloads the individual nodes from the overhead of the copies and maintains consistency for binaries by using a single install that is distributed to the rest of the nodes. It is for this reason that this approach is typically preferred to the peer-to-peer copy. When there are strong security restrictions in terms of connection allowed across data centers, it is also possible to use a node local to the primary that stages the copy from the primary nodes and distributes them to the secondary staging location. This approach limits the access across regions to just two nodes and offloads the remote copies (typically incurring in longer latency and overhead) to the coordinator box.

Figure 2-9 Staging Location



Rsync can be used as a general-purpose approach for all types of data in the EDG but needs to be tested in each specific deployment when applications are generating large amounts of data (constantly changing) during runtime. Rsync may overload the nodes running the replication and workload tests need to be executed to verify its validity. For configuration and binaries, rsync should not cause any performance issues both for the initial setup and ongoing replications. For runtime artifacts it is required to consider the required RTO and RPO to calculate data modification rates and possible overhead caused by the rysnc copy.

Oracle Database File System

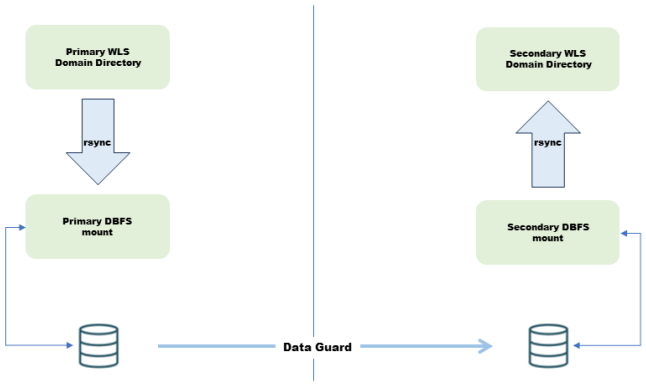
Oracle Database File System (DBFS) is an additional method that can be used for replicating primary to standby. It is mainly intended to deal with configuration or data that does not grow and changes very frequently (because of the overhead it generates in the database both at the connection and the storage level). Conceptually, a database file system is a file system interface for data that is stored in database tables.

DBFS is similar to NFS which provides a shared network file system that looks like a local file system and has both a server component and a client component. The DBFS file system can be mounted from the mid-tier hosts and accessed as a regular shared file system. Any content that is copied to the DBFS mount (as it resides in the database), is automatically replicated to the secondary site through the underlying Data Guard replication.

This method takes advantage of the robustness of the Data Guard replica. It has good availability through Oracle Driver's retry logic and provides a resilient behavior. It can be used in scenarios with medium or high latencies between the data centers. However, using DBFS for configuration replication has additional implications from the setup, database storage, and lifecycle perspectives as follows:

- It introduces some complexity due to the configuration and maintenance required by the DBFS mount. It requires the database client to be installed in the host that is going to mount it and requires an initial setup of some artifacts in the database (tablespace, user, and so on) and in the client (the wallet, the `tnsnames.ora`, and mount scripts).
- It requires additional capacity in the database because the content copied to the DBFS mount is stored in the database.
- Oracle does not recommend storing the WebLogic domain configuration or the binaries directly in the DBFS mount. This would create a very strong dependency between the Fusion Middleware files and the database. Instead, it is recommended to use the DBFS as an "assistance" file system like an intermediate staging file system to place the information that is going to be replicated to the secondary site. Any replication to standby implies two steps in this model — from the primary's origin folder to the intermediate DBFS mount and then in the secondary site from the DBFS mount to the standby's destination folder. The intermediate copies are done using rsync. As this is a low latency and local rsync copy, some of the problems that arise in a remote rsync copy operation are avoided with this model. The following diagram illustrates the replication flow

Figure 2-10 Oracle Database File System (DBFS)



- The DBFS directories can be mounted only if the database is open. When the Data Guard is not an Active Data Guard (ADG), the standby database is in mount state. Hence, to access the DBFS mount in the secondary site, the database needs to be converted to snapshot standby. When ADG is used, the file system can be mounted for reads and there is no need to transition to snapshot.

Due to the above stated reasons, it is not recommended to use DBFS as a general purpose solution to replicate all the artifacts (especially runtime files) to the standby. Using DBFS to replicate the binaries is an overkill. However, this approach is suitable to replicate (through a staging directory) few artifacts like the domain shared configuration where other methods like the storage replication or rsync do not fit the system needs. Cost or unreliable network connections between the primary and the secondary makes DBFS the best approach for file replication.

Comparing Different Replication Methods

Defining a file system replication strategy implies making a decision considering the RTO, RPO, management complexity, and total cost of ownership of each approach. Different aspects may drive the decision and the following is a list with the most relevant features. You must analyze your specific application needs and how these aspects may affect your disaster recovery design.

Management Complexity

The configuration replication procedures in the DBFS and rsync (using a staging location) methods are similar irrespective of the number of nodes in the WebLogic domain. Contrary to this, shared storage replication management gets more complicated as the number of replicated volumes increases. It requires a good lifecycle management of the different volumes and volume groups. Also, switchover and failover operations are more complex when using storage replication than in the rsync and DBFS methods. There may be additional pre and post switchover or failover steps including the activation and attachment of the replicated volumes. This complexity increases when each WebLogic managed server node uses private volumes that need to be replicated individually.

Cost Implication

In the DBFS approach, the increased costs are related to the additional storage required in the database to host the DBFS tables. Typical database tablespace and storage maintenance operations are required for an efficient recovery of allocated space. The network overhead in the cross-region copy is typically neglectable in comparison to the overall Data Guard traffic requirements.

With rsync, the storage requirement is low since file storage allocation for a WebLogic domain is typically below 100 gigabytes and the copy over rsync does not have a big network impact either.

With storage replication, once you enable replication for a volume or share, additional costs emerge in most storage vendors. Replication efficiency is also a factor affecting network and storage costs. As part of the replication process, all data being updated on the source volumes or shares is transferred to the volume replica, so volumes with continual updates incur higher network costs.

Replication Overhead

Copying through a staging directory in the DBFS and rsync scenarios does have an impact on the replication coordinator server (typically the Weblogic administration node). Additional memory and CPU resources may be required in it depending on the frequency of the copy, how big the WebLogic domain is, and whether other runtime files need to be replicated across regions in the same replication cycle.

Recovery Time Objective (RTO) for File System Data

The switchover RTO is similar in the DBFS, rsync, and storage replication approaches. For failover operations, additional steps are required by shared storage replication (activate snapshots, attach or mount volumes, and so on). These operations typically increment the downtime during a failover.

Recovery Point Objective (RPO) for File System Data

Storage replication process is continuous in most vendors with the typical Recovery Point Object (RPO) target rate being as low as a few minutes. However, depending on the change rate of data on the source volume the RPO can vary. In the DBFS and rsync methods, the user can have a finer control on the RPO for the WebLogic configuration because the information is replicated using a scheduled script and the amount of information replicated is lower than the entire storage volume. On the other hand, in the DBFS and rsync methods, it is typically the administration server’s node that acts as "manager" of the domain configuration received, so this node’s availability and capacity drives the speed of the configuration copy. When using storage replication, the replication keeps taking place regardless of the secondary nodes being up and running which can be used as an offline replication approach.

In summary, your system’s RTO, RPO, and cost needs will drive the decision for the best possible replication approach. Analyze the requirements of your applications to determine the best solution.

Preparing the Primary System

It is assumed that the primary site is created following the best practices prescribed in the Oracle Fusion Middleware Enterprise Deployment Guide. The network, hostnames, storage and database recommendations provided in the EDG need to be already present in the primary system.

This guarantees that before providing protection against disaster, local outages in the different layers (expected to happen much more frequently than disasters affecting an entire data center) will also be preempted.

The listen addresses, storage allocations, and directory considerations included in the EDG are already designed with disaster protection in mind. It is possible that some alternatives may have been adopted in the primary which can be corrected to implement disaster protection. The most common ones are using local storage for binaries and configuration information (instead of the shared storage recommended by the EDG) or not using the appropriate aliases as listen address for the different Fusion Middleware components. The following sections describe how to prepare the primary site for disaster recovery.

Preparing the Primary Storage for Storage Replication

Moving Oracle Homes To Shared Storage

When you use storage replication for Oracle Fusion Middleware Disaster Recovery, the Oracle Home and middle-tier configuration should reside on the shared storage according to the recommendations provided in the Enterprise Deployment Guide for your Fusion Middleware component. This facilitates maintenance of Oracle Homes while providing high availability since at least two Oracle Home locations are used (one for even nodes and one for odd nodes).

If the production site was initially created without disaster recovery in mind, the directories for the Oracle Fusion Middleware instances that comprise the site might be located on the local storage private to each node. In this scenario, Oracle recommends migrating the homes completely to shared storage to implement the Oracle Fusion Middleware Disaster Recovery solution. This operation may incur downtime on the primary system and needs to be planned properly to minimize its impact. This change is recommended because not only does it facilitate the DR strategy but also simplifies the Oracle Home and domain configuration management.

Follow these guidelines for migrating the production site from the local disks to the shared storage:

- Perform offline backup of the folders that are going to be moved to the shared storage. If the backup is performed using OS commands, it must be done as the root user and the permissions must be preserved.

  See *Types of Backups* and *Recommended Backup Strategy* in *Oracle Fusion Middleware Administering Oracle Fusion Middleware*
- Although you move the content to NFS, the path will be preserved. The current folder that is going to be moved to an NFS (for example, `/u01/oracle/products`) will become a mount point. In preparation for this, move or rename the current folder to another path so the mount point is empty.
- Ensure that the mount point where the shared storage will be mounted exists, is empty, and has the correct ownership.
- Mount the shared storage in the appropriate mount point. The directory structure on the shared storage must be set up as described in the Enterprise Deployment Guide.
- Once the shared storage is mounted, copy the content from the backup (or from the renamed folder) to the folder, that is now in the shared storage.

For example, moving a products folder, that is in `/u01/oracle/products`, from local disk to an NFS folder.

- To backup the content in the local folder, a copy is performed by user root, preserving the mode:

```
[root@soahost1]# cp -a /u01/oracle/products /backups/products_backup
```

- The current folder is moved because the folder that will become the mount point should be empty:

```
[root@soahost1]# mv /u01/oracle/products /u01/oracle/products_local
```

- After renaming the folder, check that the mount folder point exists (if it does not exist, it must be created) and verify that it is empty and with the correct ownership. The mount point is `/u01/oracle/products` in the following example:

```
[root@soahost1]# mkdir -p /u01/oracle/products
[root@soahost1]# ls /u01/oracle/products
[root@soahost1]# chown oracle:oinstall /u01/oracle/products
```

- The NFS volume is mounted in the mount point.

```
[root@soahost1]# mount -t nfs nasfiler:VOL1/oracle/products/ /u01/oracle/
products/
```

- To make this persistent, add the mount to the `/etc/fstab` on each host.
- Now the content from the backup is copied to the mount:

```
[root@soahost1]# cp -a /backups/products_backup /u01/oracle/products
```

You can use a similar approach to move private WebLogic domain directories to shared storage. This will facilitate the replication without requiring a staging location for the managed servers domain directory content. Refer to the EDG for details on how to use shared storage for the managed server private configuration directories.

Creating Volume Groups

Once the pertaining volumes on shared storage have been allocated according to the EDG recommendations, you need to prepare snapshots and volume groups for replication.

Most storage vendors guarantee consistency when multiple nodes are using different storage volumes and it is needed to take a backup of all of them (or replicate their contents) at a precise point in time. Volumes are grouped in a single logical unit frequently called *consistency volume group*, *consistency group* or *volume group*. Volume groups provide a point in time copy of the different volumes used in an Enterprise Deployment Guide so that a consistent snapshot of all of them is propagated to the secondary.

Imagine a situation where a database module deployment resides in volume A while the application that uses it resides in volume B. Without consistency groups it could occur that a new version of the WAR for the application is replicated while the data source configuration it requires is not, thus causing a consistency problem in the secondary. To avoid situations like this and while using the different volumes suggested in the Oracle Fusion Middleware Enterprise Deployment topology, Oracle recommends the following consistency groups. This example is using the precise case of Oracle Fusion Middleware SOA Suite but can be extrapolated to other Fusion Middleware systems:

- Create one consistency group with the volumes that contains the domain directories for the Administration Server and Managed Servers as members (`DOMAINGROUP` in Table 2-8). In

Oracle Fusion Middleware 14.1.2, the EDG places the Identity and Truststores required for configuring SSL for the different components (Node Manager, WebLogic Servers) in the same volume as the Administration Server's domain configuration. If you are placing SSL certificates and stores in a different location, ensure that it is included as part of the `DOMAINGROUP` consistency group, since the SSL configuration for the WebLogic parts has a dependency on the `KESYTORE_HOME` location (refer to the EDG for details on configuring SSL for the different components).

- Create one consistency group with the volume that contains the runtime artifacts generated by the applications running in the WebLogic Server domain (if any) (`RUNTIMEGROUP` in Table 2-8).
- Create one consistency group with the volume that contains the Oracle Home as members (`FMWHOMEGROUP` in Table 2-8).

Table 2-8 provides a summary of Oracle recommendations for consistency groups for the Oracle SOA Suite topology as shown in Figure 2-1.

Table 2-8 Consistency Groups for Oracle SOA Suite

Tier	Group Name	Members	Comments

Application	`DOMAINGROUP`	`VOLADMIN`
		`VOLSOA1`
		`VOLSOA2`
			Consistency group for the Administration Server, and the Managed Server domain directory

Application	`RUNTIMEGROUP`	`VOLRUNTIME`	Consistency group for the shared runtime content
Application	`FMWHOMEGROUP`	`VOLFMW1` `VOLFMW2`	Consistency group for the Oracle Homes

Preparing the Primary Storage for Rsync Replication

Using Peer-to-Peer

When using the rsync replication approach in a peer-to-peer model, preparing primary involves preparing the pertaining rsync scripts for the initial copy to each peer (one per node). It is recommended to offload the primary systems from performing the copy and running checksum validations. In terms of CPU, memory impact, and number of Disk I/O operations there is no significant difference whether secondary pulls information from the primary or the primary pushes the file system copy to the secondary. However, starting the rsync from secondary has other advantages as follows:

- Cron jobs' scheduling does not interfere with the primary (primary does not need to schedule anything and does not need to be manipulated to start rsync ops).
- There is a better control on hard stop for errors during rsync operations.
- Log and analysis of the operations is better offloaded to secondary.

It is for these reasons that in the peer-to-peer model, rsync scripts are typically prepared in secondary and not considered part of the primary preparation. The only aspect that needs to be accounted for is the appropriate SSH connectivity between the secondary and primary. Before running the rsync operations, ensure the SSH connectivity is allowed and working between each node in secondary and its peer in primary. It is recommended to use a SSH key for the access. Try a simple `ssh -i path_to_keyfile primary_wlsnode1_ip_primary` from the wlsnode1 peer in secondary. Repeat the verification for each OHS and WLS node

participating in the topology. The scripts and first replications will be executed as part of the secondary setup.

Using a Staging Location

When using the staging approach, you should create a central location (ideally on shared storage) to host the staging copy and scripts to transfer from the primary nodes to the staging directory and scripts to transfer from the staging directory to the target secondary nodes. When using two staging locations (one in the primary and one in the secondary to reduce the connectivity requirements across sites), three sets of scripts are required: to copy from primary nodes to the staging location in primary, to copy from the staging location in primary to the staging location in secondary, and to copy from the staging location in secondary to the different nodes in secondary. This approach is also useful as an extension to a backup strategy. Each location has a backup copy that can be distributed to the different nodes. Prepare a script in the node responsible for the staging (typically the admin server node in secondary) per node in primary. You should characterize each nodes WebLogic domain copy. For example, using a structure as follows:

Figure 2-11 Staging Location

Before running the rsync operations ensure SSH connectivity is allowed and working between each node in primary and the staging node in primary and between the staging node in primary and the staging one in secondary (if the two-staging-locations model is used). Try a simple `ssh -i path_to_keyfile primary_stagenode_ip_primary` from each WebLogic and OHS node in primary. If you are using staging locations both in primary and secondary, check SSH connectivity from the staging orchestrator in secondary also.

For each production site host on which one or more Oracle Fusion Middleware components have been installed, set up rsync to copy the following directories and files to the same directories and files on the secondary to the staging host:

1. Oracle Home directory and subdirectories.
2. Oracle Central Inventory directory which includes the Oracle Universal Installer entries for the Oracle Fusion Middleware installations.
3. Shared config folder such as the WebLogic Administration Server domain directory, deployment plans, applications, keystores, and so on. Since this folder is shared by all the mid-tier hosts in the EDG, you do not have to have a copy for each host. Do not replicate the shared WebLogic domain directory only. In context of the EDG, there are dependencies in the `/u01/oracle/config/applications` and `/u01/oracle/config/keystores` directories that need to be copied with the WebLogic domain configuration.
4. Private config folders such as the WebLogic managed server's domain and local node manager home on the host.
5. Shared runtime folder (if applicable).
6. Oracle Fusion Middleware static HTML pages directory for the Oracle HTTP Server installations on the host (if applicable).

You can use the scripts at *GitHub* as an example to prepare the primary system for the staging copy. The script `rsync_for_WLS.sh` is a wrapper that invokes `rsync_copy_and_validate.sh`. This second script contains the real logic to perform rsync copies with the recommended rsync configuration and executes a thorough validation of files after the copy is completed. If any differences are detected after several validations retries, these are logged so that they can be acted upon.

By default, the `rsync_copy_and_validate.sh` uses `oracle` as the user to perform the copies. If a different user owns the origin folders, customize the property `USER` in the script.

It also uses the environment variable `DEST_FOLDER` to determine the target location that will be used to rsync contents. If the variable is not set, the script copies contents to the node executing the script to the exact same directory path used in the source node. While using the staging approach, set the variable to the desired path (for example , for the first WLS node private configuration) `/staging_share_storage/midtier/wls_private_config/wlsnode1_private_config`.

- To rsync any folder from primary nodes, use `rsync_for_WLS.sh` in the staging node in primary as the user owning the directory in each primary node . `rsync_for_WLS.sh` needs to be executed with three parameters when using SSH key file for SSH connections: the IP of the node from which the copy will be pulled, the path to be replicated, and the ssh key file. For example:

  ```
  ./rsync_for_WLS.sh 10.1.1.1 /u01/oracle/config/domains/soaedg/  /home/oracle/keys/SSH_KEY.priv
  ```

  It can also be executed with only two parameters (the IP of the node from which the copy will be pulled and the path to be replicated). In this case, the SSH connection used by

rsync will prompt for SSH password (password-based SSH needs to be set up before running the script).

```
./rsync_for_WLS.sh 172.11.2.113 /u01/oracle/config/domains/soaedg/
```

- Replicate the entire primary administration server node first. Execute the script for the products folder, the shared configuration folder, and the private domain configuration folder as follows (using the examples provided in this book, 172.11.2.113 is the IP of the primary administration server):

```
[oracle@staging_node]$ export DEST_FOLDER=/staging_share_storage/midtier/
wls_products_home/wls_products_home1
[oracle@staging_node]$./rsync_for_WLS.sh 172.11.2.113 /u01/oracle/
products /home/oracle/keys/SSH_KEY.priv
[oracle@staging_node]$ export DEST_FOLDER=/staging_share_storage/midtier/
wls_shared_config
[oracle@staging_node]$./rsync_for_WLS.sh 172.11.2.113 /u01/oracle/config /
home/oracle/keys/SSH_KEY.priv
[oracle@staging_node]$ export DEST_FOLDER=/staging_share_storage/midtier/
wls_private_config/wlsnode1_private_config
[oracle@staging_node]$./rsync_for_WLS.sh 172.11.2.113 /u02/oracle/config/ /
home/oracle/keys/SSH_KEY.priv
```

- Now replicate the second WLS server node. Execute the script for the products folder and the private domain configuration folder as follows (using the examples provided in this book 172.11.2.114 is the IP of the primary WLS server node 2):

```
[oracle@staging_node]$ export DEST_FOLDER=/staging_share_storage/midtier/
wls_products_home/wls_products_home2
[oracle@staging_node]$./rsync_for_WLS.sh 172.11.2.114 /u01/oracle/
products /home/oracle/keys/SSH_KEY.priv
[oracle@staging_node]$ export DEST_FOLDER=/staging_share_storage/midtier/
wls_private_config/wlsnode2_private_config
[oracle@staging_node]$./rsync_for_WLS.sh 172.11.2.114 /u02/oracle/config /
home/oracle/keys/SSH_KEY.priv
```

- Replicate the two OHS nodes. Since the OHS instances is run in the standalone mode there is no shared configuration folder and only the private domain configuration needs to be replicated as follows (using the examples provided in this book, 172.11.2.111 is the primary OHS node 1):

```
[oracle@staging_node]$ export DEST_FOLDER=/staging_share_storage/webtier/
ohs_products_home/ohs_products_home1
[oracle@staging_node]$./rsync_for_WLS.sh 172.11.2.111 /u02/oracle/
products /home/oracle/keys/SSH_KEY.priv
[oracle@staging_node]$ export DEST_FOLDER=/staging_share_storage/webtier/
ohs_private_config/ohsnode1_private_config
[oracle@staging_node]$./rsync_for_WLS.sh 172.11.2.111 /u02/oracle/config /
home/oracle/keys/SSH_KEY.priv
```

After this, you should have a complete copy of all the OHS and WLS nodes in the staging location (in primary or secondary depending of the staging model used).

Preparing the Primary Storage for DBFS

In the case of DBFS, preparing the primary system requires installing the DB client in the hosts that will use the DBFS mount point. This requires creating few artifacts in the database (tablespace, user, and so on) and in the client nodes (the wallet, the `tnsnames.ora`, and so on). Download the scripts from *GitHub* and perform the following steps to configure the required DBFS mount:

1. Download the Oracle DB client from edelivery and upload it to the mid-tier host (do not install it yet). Ensure that you download the installer version, not the image-based version. Download the software and upload it to all the mid-tier hosts. For example, to `/u01/install/V982064-01.zip`.
2. Locate the `script dbfs_dr_setup_root.sh` in the scripts you downloaded from *GitHub* and copy it to the mid-tier hosts. This script performs the required tasks to get the DBFS mount ready in the host. It installs the DB client and its required operating system packages, it configures the DBFS user and schema in the database, and it mounts the DBFS file system and creates a cron so the DBFS file system is mounted on host boot.
3. Execute the script as root user using the following syntax:

   ```
   ./dbfs_dr_setup_root.sh <local_db_scan_name> <db_port> <local_PDB_service>
   <pdb_sys_password> <path_to_dbclient_installer>
   ```

   As input parameters, you need to provide the connection data used to connect to the local database used by the WLS. Provide primary PDB connection data in the primary site mid-tier nodes and provide secondary PDB connection data in the secondary site mid-tier nodes. You can get the local scan name, port, and PDB service name from the data sources of each domain.

   > **Note**
   >
   > Before running the script on the secondary hosts, you must convert the standby database to snapshot mode.

   You must provide primary PDB values. Example to run primary mid-tier hosts (it must be a single line):

   ```
   ./dbfs_dr_setup_root.sh drdba-scan.wlsdrvcnlon1ad2.wlsdrvcnlon1.oraclevcn.com
   1521 PDB1.wlsdrvcnlon1ad2.wlsdrvcnlon1.oraclevcn.com mypassword /u01/install/
   V982064-01.zip
   ```

   Provide the appropriate service name, so DBFS will connect to a CRS service rather than the default PDB administration one. Example of execution in a RAC scenario:

   ```
   ./dbfs_dr_setup_root.sh drdbrac2a-scan.subnetlon1.myvcnlon.oraclevcn.com 1521
   mypdbservice.mycompany.com mypassword /u01/install/V982064-01.zip
   ```

4. Verify if the DBFS mount is now available in the mid-tier host:

   ```
   [root@ wlsociprefix-wls-1]# df -h | grep dbfs
   dbfs-@PDB1:/ 32G 248K 32G 1% /u02/data/dbfs_root
   [root@ wlsociprefix-wls-1]# ls /u02/data/dbfs_root
   dbfsdir
   ```

Repeat the following steps in all the mid-tier hosts:

1. The db client software is installed in the mid-tier host, in the folder `/u01/app/oracle/client`. The required packages for the db client are installed using yum.
2. A user is created in the PDB database for DBFS. The username is dbfsuser and its password is set to the same `SYS` password provided as input of the script.
3. A DBFS tablespace is created in the PDB database for the DBFS mount (tablespace name is `tbsdbfs`) and a DBFS folder (name `dbfsdir`).
4. A new folder is created in the domain, `DOMAIN_HOME/dbfs`. It contains the wallet to store the dbfsuser's password and other artifacts needed (`tnsnames.ora`, `sqlnet.ora`). This wallet is used by the db client to mount the dbfs mount in the mid-tier host.
5. The script `dbfsMount.sh` is also created in `DOMAIN_HOME/dbfs`. This script is used to mount the dbfs mount in the mid-tier. It is also added to the cron on reboot, so the script is executed when the machine is rebooted.
6. The DBFS mount is mounted in `/u02/data/dbfs_root` mount point as the folder `dbfsdir`.

If there is a failure, the script can be executed again. In these re-executions, there are warnings because few artifacts may have been created already in the DB (db user, tablespace, and so on), but these messages can be ignored.

Preserving the Production Hosts Hostnames as Listener Address

When a primary site is created without considering disaster protection, it is possible that the Fusion Middleware components are not configured using host name aliases as listener addresses (as recommended in the EDG and explained in the previous sections of this document) and instead they may be using the physical host names of the nodes where they reside. In this case, you have the following two options:

- Modify the WebLogic domain configuration in the existing site to use host aliases as listener addresses for the servers. This change has additional implications as all clients accessing the WLS servers' endpoints (like JMS or RMI clients) directly need to be made aware of the new hostname.

  > **Note**
  >
  > This is not the case for HTTP access as they should already be using a front-end load balancer virtual server.

- If modifying the production configuration is not feasible, you can preserve the configuration as it is (continue using the physical host names as listener addresses for the Fusion Middleware components) and adjust the secondary to these hostnames by manipulating host name resolution.

  This can be done when using separate DNS services in the primary and the secondary or manipulating local host resolution in the secondary nodes involved (For more details, see Host Name Resolution section). In this case, the primary physical host names must be added as aliases to the `/etc/hosts` of the secondary site hosts.

  Example:

  `/etc/hosts` entries in the production site hosts that do not use aliases for the components

  ```
  127.0.0.1     localhost.localdomain     localhost
  172.11.2.134  prsoa-vip.example.com     prsoa-vip
  ```

```
172.11.2.111  prweb1.example.com        prweb1
172.11.2.112  prweb2.example.com        prweb2
172.11.2.113  prsoa1.example.com        prsoa1
172.11.2.114  prsoa2.example.com        prsoa2
```

`/etc/hosts` entries in the secondary site hosts

```
127.0.0.1    localhost.localdomain    localhost
172.22.2.134  scdrysoa-vip.example.com  scdrysoa-vip prsoa-vip.example.com
prsoa-vip
172.22.2.111  scdryweb1.example.com     scdryweb1    prweb1.example.com
prweb1
172.22.2.112  scdryweb2.example.com     scdryweb2    prweb2.example.com
prweb2
172.22.2.113  scdrysoa1.example.com     scdrysoa1    prsoa1.example.com
prsoa1
172.22.2.114  scdrysoa2.example.com     scdrysoa2    prsoa2.example.com
prsoa2
```

Preparing Data Sources in the Primary Middle-Tier

There are several approaches that can be used to configure the connection strings used in WebLogic data sources. For Disaster Recovery and Maximum Availability purposes, Oracle recommends using TNS alias in these connection strings. Other approaches are valid alternatives to address needs in special cases like local database standbys or stretched clusters.

Using TNS alias is recommended for configuration maintenance and availability purposes. Instead of using long connection strings repeated in each WebLogic data source, you can use a TNS alias in the JDBC URL. For more information, see Use a TNS Alias Instead of a DB Connection String in *Administering JDBC Data Sources for Oracle WebLogic Server*. It is also prescribed in the Enterprise Deployment Guide. In Oracle Fusion Middleware 14.1.2, database client modules are used to deploy `tnsnames.ora` files so that they are managed directly by the WebLogic infrastructure. The TNS alias is the same in the primary and the secondary, hence data sources use the same DB connect string text in both locations. The TNS alias is resolved with a `tnsnames.ora` file that is managed by the WebLogic domain configuration. This file should not be replicated from primary to standby, so each site maps the alias to its local database.

- If it is included in storage replication, it needs to be updated after each replication cycle with the appropriate local database service and address in the secondary.
- If rsync or DBFS is used, it simply needs to be in the excluded file list in the replica scripts. This will avoid string replacements and unnecessary manipulations during the initial setup and the ongoing replications for the system's lifecycle.

Each site will resolve the TNS alias with its appropriate connect string pointing to the local database only. When a new database connect string is added in primary's data sources, the `tnsnames.ora` file needs to be updated again in the secondary with the appropriate new alias information.

For example, Connect string in data sources in the primary site:

```
jdbc:oracle:thin:@soaedg
```

Where, the `tnsnames.ora` file in the primary contains the following:

```
SOAEDG =
        (DESCRIPTION=
        (ADDRESS_LIST=
            (LOAD_BALANCE=ON)
            (ADDRESS=(PROTOCOL=TCP)(HOST=prmy-scan)(PORT=1521)))
            (CONNECT_DATA=(SERVICE_NAME=soaedg.example.com))
        )
```

Connect string in data sources in the secondary site:

```
jdbc:oracle:thin:@soaedg
```

Where, the `tnsnames.ora` file in the secondary contains the following:

```
SOAEDG =
        (DESCRIPTION=
        (ADDRESS_LIST=
            (LOAD_BALANCE=ON)
            (ADDRESS=(PROTOCOL=TCP)(HOST=stby-scan)(PORT=1521)))
            (CONNECT_DATA=(SERVICE_NAME=soaedg.example.com))
        )
```

The advantage of this tnsnames approach is that since the same DB connect string is used in the WebLogic domain config, there is no need to alter the WebLogic configuration after replicating the config from primary to standby. Since each site points to the local database only, there is no risk of cross connections from the mid-tier to the remote database.

Before proceeding with the initial replication and setup of the disaster protection system, ensure that all the different data sources used by the Fusion Middlware system have been configured with a TNS alias. This simplifies configuration management and optimizies your system's RTO.

Preparing the Secondary System

This section provides information on how to set up the secondary system.

Preparing Network Components in the Secondary Site

When you plan your disaster recovery solution, you need to allocate precise hostnames in the secondary, configure a load balancer with the same virtual servers as the primary, and prepare external clients to access the system in the event of a disaster.

Preparing Host Names

In the secondary location, the host name addresses used by the Fusion Middleware components must be resolvable to valid IP addresses in the secondary site.

Use your preferred host name resolution mechanism to create this mappings as described in the Host Name Resolution section.

As prescribed in the EDG, Oracle recommends creating aliases for physical host names to isolate the real physical node names (which are different in each site) from the host names used by the Fusion Middleware components (which are the same regardless of the site).

This section describes how to plan physical host names and alias host names for the middle-tier hosts that use the Oracle Fusion Middleware instances at the secondary site for an existing primary site. It uses the Oracle SOA Suite Enterprise Deployment shown in Figure 2-1 for the host name examples. Each host at the production site has a peer host at the secondary site. The peer WebLogic servers and processes use the same ports as their counterparts on the other site.

The following section shows how to set up host names at the secondary sites to map the primary system's configuration.

> **Note**
>
> In the examples listed, IP addresses for hosts at the initial production site have the format `172.11.x.x` and IP addresses for hosts at the initial secondary site have the format `172.22.x.x`.

Sample Host Names for the Oracle SOA Suite Production and Secondary Site Listen Addresses

Learn about the Oracle SOA Suite secondary sites host name mappings.

Table 2-9 lists the IP addresses, physical host names, and aliases that are used for the Oracle SOA Suite Enterprise Deployment Guide (EDG) deployment production site hosts. The WebLogic Server's listen address and channels (if any) in the primary should use the hostname alias and not the IP address nor the physical host name.

Table 2-9 IP Addresses and Physical Host Names for SOA Suite Production Site Hosts

IP Address	Physical Host Name	Host Name Alias
172.11.2.111	prweb1.example.com	WEBHOST1
172.11.2.112	prweb2.example.com	WEBHOST2
172.11.2.113	prsoa1.example.com	SOAHOST1
172.11.2.114	prsoa2.example.com	SOAHOST2

Table 2-10 lists the IP addresses, physical host names, and aliases that are used for the Oracle SOA Suite Enterprise Deployment Guide (EDG) deployment secondary site hosts.

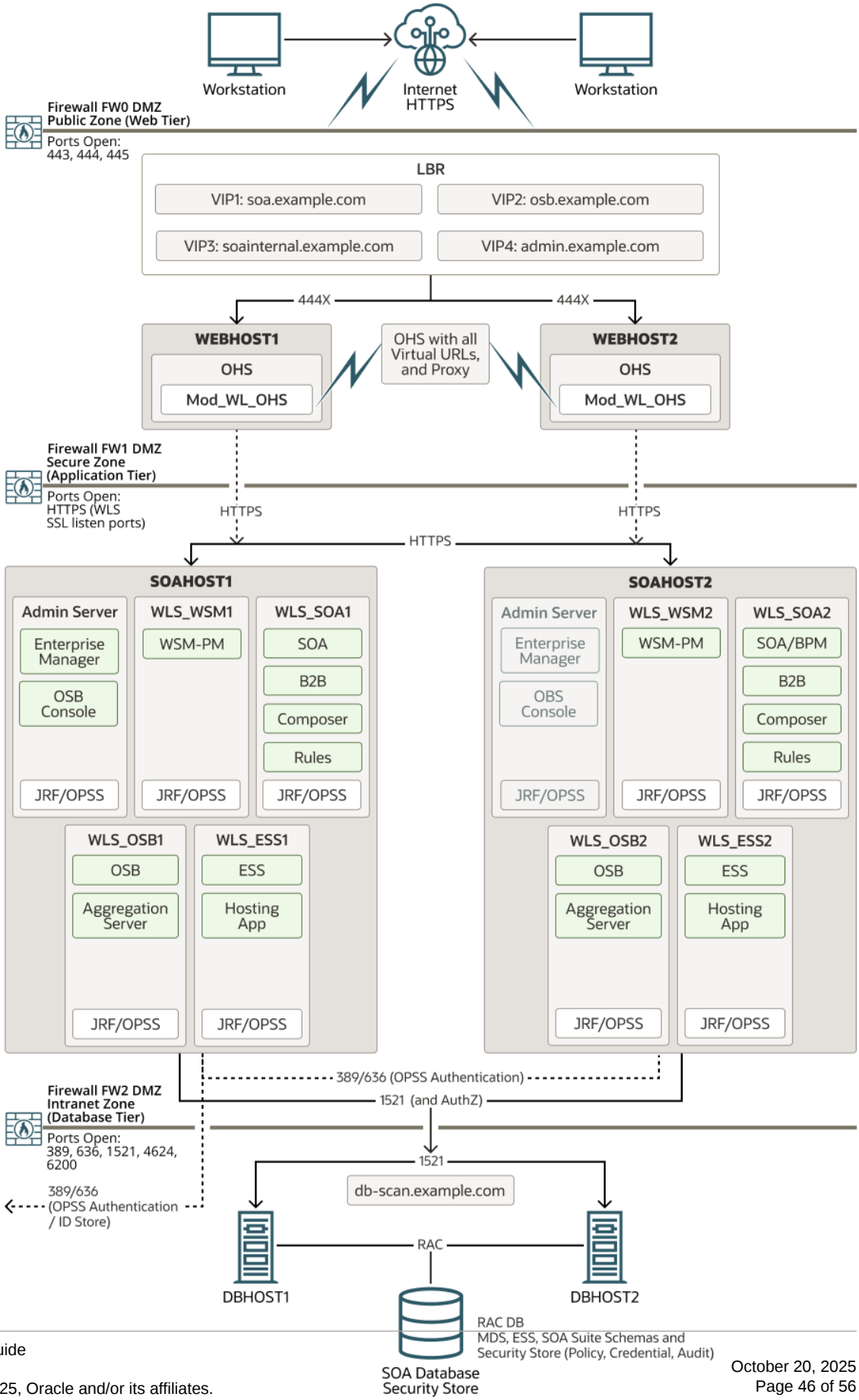
Table 2-10 IP Addresses and Physical Host Names for SOA Suite Secondary Site Hosts

IP Address	Physical Host Name	Host Name Alias
172.22.2.111	scdryweb1.example.com	WEBHOST1
172.22.2.112	scdryweb2.example.com	WEBHOST2
172.22.2.113	scdrysoa1.example.com	SOAHOST1
172.22.2.114	scdrysoa2.example.com	SOAHOST2

> **Note**
>
> If you use separate DNS servers in the primary and the secondary to resolve host names, then you can use the same physical host names for the production site hosts and the secondary site hosts, and you do not need to define the alias host names on the secondary site hosts. For more information about using separate DNS servers to resolve host names, see Resolving Host Names Using Separate DNS Servers.
>
> However, it is recommended to use different hostnames and same aliases to isolate the real physical host names (which are different in each site) from the host names used by the Fusion Middleware components (which are the same regardless the site).

Figure 2-12 shows the physical host names that are used for the Oracle SOA Suite EDG deployment at the secondary site.

Figure 2-12 Physical Host Names Used at Oracle SOA Suite Deployment Standby Secondary Site



Preparing the Required Virtual IPs

Besides the regular hostname listen addresses and as described in the Virtual IP Considerations section, additional virtual hostnames may be required for those components not supporting WebLogic Server migration. In the case of Oracle SOA Suite, only the WebLogic administration server requires this virtual hostname and mapping IP.

Table 2-11 shows the virtual IP addresses and virtual host names that were used for the Oracle SOA Suite EDG deployment production site hosts.

Table 2-11 Virtual IP Addresses and Virtual Host Names for the SOA Suite Production Site Hosts

Virtual IP Address	Virtual Host Name	Alias Host Name
172.11.2.134	prsoa-vip.example.com	ADMINVHN

Table 2-12 shows the virtual IP addresses, virtual hostnames, and alias hostnames that are used for the Oracle SOA Suite EDG deployment secondary site hosts. Figure 2-2 shows the physical host names that are used for the Oracle SOA Suite EDG deployment at the secondary site. The alias host names shown in Table 2-12 should be defined for the SOA Oracle Suite secondary site hosts, as shown in Figure 2-2.

> **Note**
>
> If you use separate DNS servers to resolve host names, then you can use the same virtual IP addresses and virtual host names for the production site hosts and the secondary site hosts, and you do not need to define the alias host names.
>
> For more information about using separate DNS servers to resolve host names, see Resolving Host Names Using Separate DNS Servers.

Table 2-12 Virtual IP Addresses, Virtual Host Names, and Alias Host Names for SOA Suite Secondary Site Hosts

Virtual IP Address	Virtual Host Name	Host Name Alias
172.22.2.134	stbysoa-vip.example.com	ADMINVHN

> **Note**
>
> The subnets in the production site and the secondary site are different.

> **Note**
>
> If you use separate DNS servers to resolve host names, then you can use the same host names for the production site hosts and the secondary site hosts, and you do not need to define the alias host names.

Preparing Load Balancer in the Secondary Site

In a disaster recovery topology, both the primary and the secondary systems use a hardware load balancer that acts as front-end for all HTTP/HTTPs requests with equivalent configuration. Each load balancer will balance the traffic between the servers of its local site.

The secondary load balancer must support the required features to act as front-end. For more information, see *Hardware Load Balancer Requirements* in *Enterprise Deployment Guide for Oracle SOA Suite*.

The virtual front-end name used by the production and the secondary load balancer must be the same. That virtual front-end name must be resolved in DNS with the IP of the load balancer of the site that has primary role in each moment.

You must configure the virtual servers and the associated ports on the load balancer for the different types of network traffic and monitoring as they were configured in the primary.

Configure the virtual servers' backend pools using the secondary's Oracle HTTP Servers listen address and using the same ports for the listeners. Also, just as in the primary, the secondary load balancer should be configured to monitor for availability so that the traffic to these is stopped as soon as possible when a service is down. This ensures that incoming traffic on a given virtual host is not directed to an unavailable service in the other tiers.

As prescribed in the EDG, Oracle recommends that you use two load balancers (or at least separate listeners) when you deal with external and internal traffic. In such a topology, one load balancer or listener is set up for external HTTP traffic and the other load balancer is set up for internal traffic. In all cases, the load balancer in secondary should be a replica of the one in the primary and it is highly recommended that it should be configured in fault tolerant mode (for more details, refer to your load balancer vendor).

Some of the virtual servers defined in the load balancer are used for inter-component communication. These virtual servers are used for internal traffic and are defined in the internal DNS of a company. When you use a single global DNS server to resolve host names, Oracle recommends you create aliases for these virtual servers. Creating aliases is not required when you use separate DNS servers to resolve host names.

The virtual servers required for the various Oracle Fusion Middleware products are described in the related EDG for each product.

> **Note**
>
> A hardware load balancer is assumed to server as a front-end for each site. For more information about supported load balancers, see *My Oracle Support*.

Preparing File Systems in the Secondary Site

File System, Directories, and Volumes

Verify if the storage in the secondary nodes are configured according to the recommendations in the EDG replicating the one used in the primary system. The EDG provides specific details about the storage volumes and mount points that need to be created for an enterprise topology. The same volumes and mount points that are used in the primary need to be configured in the secondary or else additional manipulation of the system's configuration will be needed on the secondary. This will increase the management overhead and also increase the

RTO of your disaster protection solution. Create private and shared directories that exactly reflects the same structure in the secondary as in the primary.

Setting Up the Secondary System

After the primary and secondary systems are prepared based on the information provided in the previous sections at the network (host name resolution), storage, and database connections level, the following steps are required to set up the secondary system.

Configuring Oracle Data Guard for the Fusion Middleware Database

When you plan your disaster recovery solution, you need to synchronize the databases in your system with the Oracle Data Guard. This is the only Maximum Availability approach recommended for Fusion Middleware systems.

This section provides the recommendations and considerations to set up Oracle databases that are used in an Oracle Fusion Middleware Disaster Recovery topology.

- Oracle recommends that you create Oracle Real Application Cluster (Oracle RAC) databases on both the production and secondary site as described in the Enterprise Deployment Guide.
- Oracle Data Guard is the recommended disaster protection technology for the databases running the Fusion Middleware persistent stores and Fusion Middleware metadata repositories.
- Ensure that your network is configured for low latency with sufficient bandwidth because synchronous redo transmission can affect the response time and throughput.
- Oracle Data Guard provides three protection modes: Maximum Availability, Maximum Performance (default), and Maximum Protection. Oracle recommends using the protection mode that meets your availability, performance, and data protection requirements. For more information, see *Oracle Data Guard Protection Modes* in the Oracle Data Guard Administration documentation.
- During normal operation the secondary site database should be placed in managed recovery mode. This ensures that the secondary site's database is in a constant state of media recovery. Managed recovery mode is enabled for shorter failover times.
- The `tnsnames.ora` file on the production and secondary site database nodes must have entries for databases on both the production and secondary sites.

  > **Note**
  >
  > This is not the case for the `tnsnames.ora` files in the middle-tiers.

- Oracle recommends using Automatic Storage Management (ASM) as the volume manager for Oracle database files. ASM is an Oracle recommended storage management solution that provides an alternative to conventional volume managers, file systems, and raw device. It supports single instance Oracle Database and Oracle Real Application Clusters (Oracle RAC) configurations.

  For more information about Automatic Storage Management (ASM) and Real Application Clusters (RAC), see *Oracle Automatic Storage Management* and *Real Application Clusters Installation Guide for Linux and UNIX*.

- The SERVICE_NAME used by applications in the primary should also be the same in the secondary database . However, each database can have additional services defined for other purposes. For example, specific read-only services can be created on the secondary to offload reporting and business intelligence operations to the secondary site.

Prerequisites and Assumptions

Ensure that the following prerequisites are met:

- The Oracle RAC cluster and Automatic Storage Management (ASM) instances on the secondary site have been created.
- The Oracle RAC databases on the secondary site and the production site are using a flash recovery area.
- The Oracle RAC databases are running in archivelog mode.
- The database hosts on the secondary site already have Oracle software installed.
- In a shared ORACLE_HOME configuration, the TNS_ADMIN directory must be a local, non-shared directory.

Oracle Data Guard Environment Description

The examples given in this section contain environment variables as described in Table 2-13.

Table 2-13 Variables Used by Primary and Standby Databases

Variable	Primary Database	Standby Database
Database names	soa	soa
SOA database host names	dbhost1.example.com, dbhost2.example.com	dbhost1stby.example.com, dbhost2stby.example.com
Scan address	prmy-scan.example.com	stby-scan.example.com
Database unique names	soa_pri	soa_stby
Instance names	soa1, soa2	soa1, soa2
Service names	soa_pri.example.com , soaedg.example.com	soa_stby.example.com , soaedg.example.com

Procedure for Configuring the Data Guard

The configuration of a Data Guard involves several steps: you need to prepare the primary database with the recommended parameters, prepare the TNS aliases in primary and standby environments, create the physical standby database as a duplication of the primary database, configure the Data Guard Broker, and so on. You can use one of these methods to automate most of these actions:

- Oracle provides a set of sample scripts available in the MAA repository at GitHub, in dg_setup_scripts. These scripts are designed to setup a standby database for an existing primary database, using the restore from service feature and Data Guard Broker. They allow customization (OS user names and Oracle Homes are configurable values). They support databases with and without TDE encryption, and work in environments with Read Only Oracle Home (ROOH) or with regular Oracle Homes. The scripts are validated in 12c (12.2), 18c, 19c, 21c, and 23ai RDBMS versions. Download the file and follow the instructions included in the README.md to configure a standby database.

- If both primary and standby databases are in Oracle Cloud Infrastructure, use the Data Guard configuration option in OCI Console. This skips most of the complexity in the setup operation. For more information, see Enable Oracle Data Guard on a DB System in *Oracle Base Database Service*.
- Using "Enterprise Manager Cloud Control". Setting up and managing databases using Cloud Control helps in controlling downtime and simplifies disaster recovery operations. For more information, see Provisioning Oracle Standby Databases in *Oracle Enterprise Manager Lifecycle Management Administrator's Guide*.
- If none of the above methods are feasible for your specific environment, you can manually configure the Data Guard by following one of these documents:
  - Creating a Physical Standby database using RMAN restore database from service (Doc ID 2283978.1)
  - Creating a Physical Standby using RMAN Duplicate (RAC or Non-RAC) (Doc ID 1617946.1)

Verifying the Data Guard Broker Configuration

Complete the following steps to verify that the Data Guard Broker configuration was created successfully.

1. Verify the Oracle Data Guard configuration by querying the `V$ARCHIVED_LOG` view to identify existing files in the archived redo log. For example:

```
SQL> SELECT SEQUENCE#, FIRST_TIME, NEXT_TIME
  2> FROM V$ARCHIVED_LOG ORDER BY SEQUENCE#;

SEQUENCE# FIRST_TIME NEXT_TIME
---------- ------------------ ------------------
         8 11-DEC-21 17:50:45 11-DEC-21 17:50:53
         9 11-DEC-21 17:50:53 11-DEC-21 17:50:58
         10 11-DEC-21 17:50:58 11-DEC-21 17:51:03
3 rows selected
```

2. On the primary database, issue the following SQL statement to force a log switch and archive the current online redo log file group:

```
SQL> alter system archive log current;
```

3. On the standby database, query the `V$ARCHIVED_LOG` view to verify that the redo data was received and archived on the standby database:

```
SQL> SELECT SEQUENCE#, FIRST_TIME, NEXT_TIME
  2> FROM V$ARCHIVED_LOG ORDER BY SEQUENCE#;

SEQUENCE# FIRST_TIME NEXT_TIME
---------- ------------------ ------------------
        8 11-DEC-21 17:50:45 11-DEC-21 17:50:53
        9 11-DEC-21 17:50:53 11-DEC-21 17:50:58
       10 11-DEC-21 17:50:58 11-DEC-21 17:51:03
       11 11-DEC-21 17:51:03 11-DEC-21 18:34:11

4 rows selected
```

4. Use the `show configuration` command in Data Guard Broker command line to verify that the configuration was created successfully. See Example 2-11.

Example 2-11 Verifying the Data Guard Broker Configuration

```
[oracle@dbhost1 ~]$ dgmgrl sys/'<password>'
DGMGRL for Linux: Release 19.0.0.0.0 - Production on Tue Feb 1 09:00:16 2022
```

```
Version 19.6.0.0.0

Copyright (c) 1982, 2019, Oracle and/or its affiliates. All rights reserved.
Welcome to DGMGRL, type "help" for information.
Connected to "soa_pri"
Connected as SYSDBA.

DGMGRL> show configuration
Configuration - dg_config

Protection Mode: MaxPerformance
Databases:
soa_pri - Primary database
soa_stby - Physical standby database

Fast-Start Failover: DISABLED

Configuration Status:
SUCCESS
```

Testing Database Switchover and Switchback

You can perform a database switchover and switchback.

Performing a Switchover Operation by Using Oracle Data Guard Broker

To perform a switchover operation by using Oracle Data Guard Broker, complete the following tasks:

1. Verify the Oracle Data Guard Broker configuration by running the following command:

   ```
   DGMGRL> show configuration;

   Configuration - dg_config

   Protection Mode: MaxPerformance
   Databases:
   soa_pri  - Primary database
   soa_stby - Physical standby database

   Fast-Start Failover: DISABLED

   Configuration Status:
   SUCCESS
   ```

2. Swap the roles of the primary and standby databases by running the `SWITCHOVER` command. Example 2-11 shows how Data Guard Broker automatically shuts down and restarts the old primary database as part of the switchover operation.

   ```
   DGMGRL> switchover to 'soa_stby'
   Performing switchover NOW, please wait...
   Operation requires a connection to database "soa_stby"
   Connecting ...
   Connected to "soa_stby"
   Connected as SYSDBA.
   New primary database "soa_stby" is opening...
   Oracle Clusterware is restarting database "soa_pri" ...
   Connected to "soa_pri"
   Connected to "soa_pri"
   Switchover succeeded, new primary is "soa_stby"
   ```

3. After the switchover is complete, use the `SHOW CONFIGURATION` command to verify that the switchover operation was successful:

```
DGMGRL> show configuration;
Configuration - dg_config
Protection Mode: MaxPerformance
Databases:
soa_stby - Primary database
soa_pri  - Physical standby database
Fast-Start Failover: DISABLED
Configuration Status:
SUCCESS
```

> **Note**
>
> For information about switchover and failover operation of Oracle Data Guard Broker, see *Switchover and Failover Operations* in the *Oracle Data Guard Broker*.

Replicating the Primary File Systems to the Secondary Site

Storage Replication Approach

Using storage replication process to get a copy of the primary in the secondary involves activating the snapshots that have been created from the primary volumes. Some vendors provide a multiple step process where a snapshot is cloned and then activated so that it can be mounted by the secondary nodes. Refer to your precise storage vendor details to activate, attach, and mount the pertaining shares or volume to the secondary nodes. Each node in the secondary should mount the exact same volumes using the exact same paths as its peer in the primary.

Rsync Replication Approach

In the peer-to-peer model, rsync script should be executed in each secondary node to pull the content from each respective peer in primary. In the stage rsync approach, the scripts to transfer from the staging location to each individual node should be executed to a get a copy of Oracle Homes, WebLogic Domain configuration, and runtime artifacts if required. You can also use the scripts at *GitHub* as an example to replicate to the secondary system both in peer-to-peer and using staging location.

The script `rsync_for_WLS.sh` is a wrapper that invokes `rsync_copy_and_validate.sh`. This second script contains the real logic to perform rsync copies with the recommended rsync configuration and executes a thorough validation of files after the copy is completed. If any differences are detected after several validation retries, these are logged so that they can be acted upon.

By default, the `rsync_copy_and_validate.sh` uses `oracle` as the user to perform the copies. If a different user owns the origin folders, customize the property `USER` in the script .

It also uses the environment variable `DEST_FOLDER` to determine the target location that will be used to rsync contents. If the variable is not set, the script copies contents to the node executing the script to the exact same directory path used in the source node. While using the staging approach set the variable to the desired path (for example , for the first WLS node private configuration) `/staging_share_storage/midtier/wls_private_config/wlsnode1_private_config`.

Using Peer-to-Peer

For each host on the secondary site, set up rsync to copy the following directories and files to the same directories and files from each primary peer:

1. Oracle Home directory and subdirectories.
2. Oracle Central Inventory directory which includes the Oracle Universal Installer entries for the Oracle Fusion Middleware installations.
3. Shared config folder such as the WebLogic Administration Server domain directory, deployment plans, applications, keystores, and so on. Since this folder is shared by all the mid-tier hosts in the EDG, you do not need a copy for each host. Do not replicate the shared WebLogic domain directory only. In context of the EDG, there are dependencies in the `/u01/oracle/config/applications` and `/u01/oracle/config/keystores` directories that need to be copied with the WebLogic domain configuration.
4. Private config folders such as the WebLogic managed server's domain and local node manager home on the host.
5. Shared runtime folder (if applicable).
6. Oracle Fusion Middleware static HTML pages directory for the Oracle HTTP Server installations on the host (if applicable).

- Replicate the entire primary Administration server node first. Execute the script for the products folder, the shared configuration folder, and the private domain configuration folder (do not replicate the domain directory for the shared configuration only) as follows. Using the examples provided in this book, 172.11.2.113 is the IP of the WLS administration server in primary, and 172.22.2.113 is the IP of the WLS administration server in secondary:

```
[oracle@172.22.2.113]$./rsync_for_WLS.sh 172.11.2.113 /u01/oracle/
products /home/oracle/keys/SSH_KEY.priv
[oracle@172.22.2.113]$./rsync_for_WLS.sh 172.11.2.113 /u01/oracle/config /
home/oracle/keys/SSH_KEY.priv
[oracle@172.22.2.113]$./rsync_for_WLS.sh 172.11.2.113 /u02/oracle/config /
home/oracle/keys/SSH_KEY.priv
```

- Replicate the second WLS server node. Execute the script for the products folder and the private domain configuration folder as follows. Using the examples provided in this book, 172.11.2.114 is the IP of the WLS server node 2 in primary, and 172.22.2.114 is the IP of the WLS server node 2 in secondary:

```
[oracle@172.22.2.114]$./rsync_for_WLS.sh 172.11.2.114 /u01/oracle/
products /home/oracle/keys/SSH_KEY.priv
[oracle@172.22.2.114]$./rsync_for_WLS.sh 172.11.2.114 /u02/oracle/config /
home/oracle/keys/SSH_KEY.priv
```

- Replicate the two OHS nodes. Since OHS instances run in standalone mode, there is no shared configuration folder, and only the private domain configuration needs to be replicated. Using the examples provided in this book, 172.11.2.111 is the IP of the OHS

server node 1 in primary, and 172.22.2.111 is the IP of the OHS server node 1 in secondary:

```
[oracle@172.22.2.111]$./rsync_for_WLS.sh 172.11.2.111 /u02/oracle/
products /home/oracle/keys/SSH_KEY.priv
[oracle@172.22.2.111]$./rsync_for_WLS.sh 172.11.2.111 /u02/oracle/config /
home/oracle/keys/SSH_KEY.priv
```

After this, you should have a complete copy of all the primary OHS and WLS nodes in each peer node in secondary.

Using a Staging Location

- If you are copying from a staging location on primary to a staging location in secondary, you do not need to set the `DEST_FOLDER` environment variable and you must replicate the same paths in the staging location in primary to the secondary staging location. For example:

  ```
  [oracle@staging_node_secondary]$./rsync_for_WLS.sh staging_node_primary_ip /
  staging_share_storage/midtier/wls_products_home/wls_products_home1 /home/oracle/keys/
  SSH_KEY.priv
  ```

  After you have the copy available in the secondary staging location, you must pull the information from the secondary nodes. For example (using the examples provided in this book, 172.22.2.113 is the IP of the WLS administration server in secondary):

  ```
  [oracle@172.22.2.113]$ export DEST_FOLDER=/u01/oracle/products
  [oracle@172.22.2.113]$./rsync_for_WLS.sh staging_node_secondary_ip /
  staging_share_storage/midtier/wls_products_home/wls_products_home1 /home/oracle/keys/
  SSH_KEY.priv
  ```

- If you are copying directly to the secondary nodes from a single staging location, you must pull the information directly from the single staging location. For example (using the examples provided in this book, 172.22.2.113 is the IP of the WLS administration server in secondary):

  ```
  [oracle@172.22.2.113]$ export DEST_FOLDER=/u01/oracle/products
  [oracle@172.22.2.113]$./rsync_for_WLS.sh staging_node_ip /staging_share_storage/
  midtier/wls_products_home/wls_products_home1 /home/oracle/keys/SSH_KEY.priv
  ```

DBFS Replication Approach

The secondary system needs to receive the copy from the primary that can be mounted to the pertaining DBFS directory. To mount it in the secondary, use the same scripts and steps as described in Preparing the Primary Storage for DBFS. Here is an example to run the `dbfs_dr_setup_root.sh` in the secondary mid-tier hosts. You must provide secondary database services values:

```
./dbfs_dr_setup_root.sh drdbb-scan.wlsdrvcnfra1ad2.wlsdrvcnfra1.oraclevcn.com
1521 PDB1.wlsdrvcnfra1ad2.wlsdrvcnfra1.oraclevcn.com mypassword /u01/install/
V982064-01.zip
```

Once the secondary's DBFS mount is available, replicate the contents from it to each middle-tier node replicating the directory structure and contents as available in the primary peer.

Configuring Secondary Site's Connect Strings with the Local Database

Whether you have replicated the `tnsnames.ora` from the primary or excluded it in the copy (either by placing the pertaining database client module outside the storage mounts that are replicated or explicitly excluding the file in rsync copies), you need to update the connect strings in the secondary location to point to the secondary database only. In remote disaster protection topologies, Oracle recommends configuring WebLogic data sources only with local database addresses (instead of using dual connect strings or replacing long connect strings directly in the different data sources). Update your `tnsnames.ora` file in the secondary with the service and scan the address used in the secondary database. With this configuration, the jdbc url information in the WebLogic data sources remains the same in the primary and the secondary (`jdbc:oracle:thin:@soaedg`) but the `tnsnames.ora` file in primary contains the following data:

```
SOAEDG =
        (DESCRIPTION=
        (ADDRESS_LIST=
            (LOAD_BALANCE=ON)
            (ADDRESS=(PROTOCOL=TCP)(HOST=prmy-scan)(PORT=1521)))
            (CONNECT_DATA=(SERVICE_NAME=soaedg.example.com))
        )
```

You need to update the secondary's `tnsnames.ora` file with the following data:

```
SOAEDG =
        (DESCRIPTION=
        (ADDRESS_LIST=
            (LOAD_BALANCE=ON)
            (ADDRESS=(PROTOCOL=TCP)(HOST=stby-scan)(PORT=1521)))
                  (CONNECT_DATA=(SERVICE_NAME=soaedg.example.com))
         )
```

In this example, the service remains the same between the primary and the secondary but it could be different depending on the connection needs in each case. These updates should typically be done once in the initial setup. Only if additional services and databases are used by the primary system, additional alias would need to be replaced in the secondary.

3
Validating the Setup

You must validate the secondary system on a regular basis especially after the initial setup. You can validate the standby site without performing a complete switchover by converting the Fusion Middleware standby database to snapshot standby. This allows the secondary Fusion Middleware servers to start in the standby site, so that you can verify that the secondary Fusion Middleware system has been setup properly. Testing the secondary system is a key part of a healthy and reliable disaster protection strategy.

In MAA, it is recommended to schedule periodic switchovers (every six to nine months) to run a reliable and complete verification of the secondary. This also allows preparing operations and teams for real disaster situations.

However, enterprises may want to perform periodic testing of their secondary site without shutting down the current production site and avoiding a complete switchover operation and validate behaviors for new changes, workload, and so on. Since most Fusion Middleware components (primarily the WebLogic Servers) require write access to the database (for persistent stores and service migration leasing) this is only possible by converting the standby database to snapshot standby (which makes the secondary database writable). This allows the standby middle-tier Fusion Middleware components to be started in the secondary site. Any change performed in the secondary site database while it is in the snapshot standby mode will be discarded once it is converted back to physical standby, so primary data will not be affected by secondary site validations. By using this approach with Oracle Data Guard snapshot standby, the primary system can continue processing workloads while the secondary database is actively used for verifications at the functional and performance level. Secondary systems can be used this way to verify new patches, new versions of applications, and changes in infrastructure and operational models while using an up-to-date copy of the production database.

These validation operations need to be performed with caution. If there are pending JMS messages, transactions or pending operations (like uncompleted SOA composites or business processes with faults) in the database, when it is converted into snapshot, the Fusion Middleware servers at the standby site will process them when they start. You must verify that there are no pending actions for Fusion Middleware components in the database when you convert to snapshot standby, otherwise remove records from the runtime tables in the standby database (such as JMS and JTA persistent stores) after you convert it to snapshot standby database and before you start the secondary site’s Fusion Middleware servers.

Testing the Setup

1. When using shared storage replications to synchronize middle-tier file systems, use the cloning technology provided by the shared storage vendor to create a clone of the secondary site's read-only volumes on the shared storage at the secondary site. Ensure that the cloned secondary site volumes are writable. If you want to test the secondary site just once, then this can be a one-time clone operation. However, if you want to test the secondary site regularly, you can set up periodic cloning of the secondary site read-only volumes to the secondary site's cloned read or write volumes. Modify the mounts accordingly in the secondary to mount the activated copies in each node using the shared storage.
2. If rsync is used, perform a refresh of contents from the primary to the secondary site using the example scripts.

3. If DBFS is used, the latest copy of information would be propagated to the secondary by Oracle Data Guard automatically.
4. Convert the standby database into snapshot standby by performing the following steps:
   a. Use Data Guard broker in the primary database host to convert the secondary database to snapshot standby.

      Example:

      ```
      [oracle@primarydbhost~]$dgmgrl sys/ your_sys_password@primary_db_unqname
      DGMGRL> convert database “secondary_db_unqname” to snapshot standby
      ```

   b. Use `dgmgrl show configuration` command to verify that the conversion has been correctly performed.
5. Verify that there are no pending actions in the secondary environment. If there were pending actions (transactions, messages) in the primary DB when the standby is converted to snapshot, the secondary Fusion Middleware servers will try to process them when they start. For Oracle Fusion Middleware SOA, you can use the SOA truncate script to remove the records from the SOA runtime tables in the secondary database to clean the runtime data before you start the secondary servers. See Removing Records from the Runtime Tables Without Dropping the Tables.
6. Configure hostname mappings for your validation. Since this is not a switchover and the primary site is still active, the front-end address will resolve to the primary site’s LBR IP address so that any browser access will by default be redirected to the active primary site. To access the secondary site services directly, you can use local host name resolutions by updating the `/etc/hosts` files in the WLS nodes and in controlled clients (laptop and so on) so that the front-end and access points resolve to the secondary addresses. For more information, see Host Name Resolution.

   > **Note**
   >
   > Verify that the client used for validations does not access the Fusion Middleware system via an HTTP proxy because some HTTP proxies may continue to resolve the virtual front-end name with the primary site’s LBR IP address regardless of which name is in the `/etc/hosts` of the client.

   > **Note**
   >
   > Non-Linux clients may require a reset of their local DNS cache before a browser will resolve the IP address with the customized host file entry.

7. Start the WebLogic servers in the secondary site:
   a. Start the secondary admin server as shown in the following example:

      ```
      $ cd /u01/app/oracle/middleware/oracle_common/common/bin
      $ ./wlst.sh
      wlst> nmConnect ('weblogic',
      'password','sooahost1','5556','soaedgdomain','/u01/data/domains/
      ```

```
soaedgdomain','SSL')
wlst> nmStart('Adminserver')
```

b. Start secondary managed servers (use the secondary WebLogic Remote Console or scripts).

   You can use the start or stop scripts provided by MAA in GitHub. These scripts provide more granularity and improved shutdown procedures.

> **Note**
>
> For Fusion Middleware SOA, while you validate the secondary site as described (without performing a complete switchover that is just opening standby in snapshot standby mode), "ORA-01403: no data found ORA-06512" errors may show up in the logs of the standby SOA servers. These errors are related to the SOA auto purge job. These errors arise because jobs in the database may have DB role dependencies (they are defined to be enabled only when the database is in primary role). This is an expected and desired behavior that prevents jobs from being executed twice (once in primary and once in standby). The SOA auto purge job is defined with primary role, so it is not shown in `DBA_SCHEDULER_JOBS` view when the database is in snapshot standby mode. The `database_role` defined for each job can be seen in the view `DBA_SCHEDULER_JOB_ROLE`. In summary, these errors can be ignored as long as they appear in the standby system. The scheduler job for SOA auto purge will be executed on the DB only if the instance changes its role to primary.

   In summary, review each WebLogic Server to verify that all applications and subsystems come up correctly.

8. Validate your Fusion Middleware system. You must first test the WebLogic data sources and confirm that all the applications are in running state in the Weblogic Remote Console. For SOA, use the Enteprise Manager test composite utility and verify that your specific business logic works properly. Any information persisted to the database will be discarded once the DB is placed again on physical standby mode.
9. After you have finished validations on the secondary site, perform the following steps to revert it back to standby role again:
   a. Stop managed servers and admin servers in the secondary.

      You can connect to the secondary WebLogic Remote Console and shutdown managed and administration servers in the secondary site.

   b. Convert the standby DB into a physical standby again. Use DG broker in primary DB host and convert the secondary to physical standby again as an oracle user as follows:

```
[oracle@drdbA ~]$ dgmgrl sys/your_sys_password@primary_db_unqname
DGMGRL> convert database “secondary_db_unqname” to physical standby
```

      Use `show configuration` command to verify that the conversion has been correctly performed.

   c. Revert any updated client's `/etc/hosts` file.

If you updated the front-end name in the `/etc/hosts` file of a client to point to the secondary site, revert it back so that the virtual front-end name points to the primary front-end IP again.

4

Managing Switchover, Switchback, and Failover Operations

Learn how to perform switchover, switchback, and failover for your Oracle Fusion Middleware Disaster Recovery topology.

This chapter includes the following topics.

Performing a Switchover

A switchover is a planned operation that sets the secondary site as the production role.

This operation is needed when you plan to take down the production site (for example, to perform maintenance) and make the current secondary site as the production site.

To perform a switchover operation:

1. Shut down any processes running on the production site. These include the Oracle Fusion Middleware instances, and any other processes in the application tier and in the web tier.
2. Stop the replication between the production site shared storage and the secondary site shared storage. If you are using shared storage replication, pause the replications. If you have scheduled jobs with rsync to update configuration on a regular basis, ensure to either stop the jobs or schedule the replication windows so that they do not interfere with the planned switchover.
3. When using storage replication, unmount the shared storage volume with the middle-tier artifacts on the current production site and mount the corresponding volumes on the current secondary site which is the new production site.
4. Use Oracle Data Guard to switchover the databases.
5. On the secondary site host, manually start all the processes. These include the Oracle Fusion Middleware instances, and any other processes in the application tier and the web tier.
6. Ensure that all user requests are routed to the secondary site by performing a global DNS push or something similar such as updating the global load balancer. See the Wide Area DNS Operations section.
7. Use a browser client to perform post switchover testing to confirm that requests are being resolved and redirected to the secondary site.

   At this point, the former secondary site is the new production site and the former production site is the new secondary site.
8. Reestablish the replication between the two sites but configure the replication so that the snapshot or rsync copies go in the opposite direction (from the current production site to the current secondary site). See the documentation for your shared storage to learn how to configure the replication so that snapshot copies are transferred in the opposite direction.

At this point, the former secondary site becomes the new production site, and you can perform maintenance at the original production site. After you have carried out the maintenance of the original production site, you can use it either as the production site or the secondary site.

To use the original production site as the production site, perform a switchback as explained in Performing a Switchback.

Performing a Switchback

A switchback operation reverts the roles of the current production and secondary sites.

To perform a switchback operation:

1. Shut down any processes running on the current production site. These include the Oracle Fusion Middleware instances, and any other processes in the application tier and the web tier.
2. Stop the replication between the production site shared storage and the secondary site shared storage. If you are using shared storage replication, pause the replications. If you have scheduled jobs with rsync to update configuration on a regular basis, ensure to either stop the jobs or schedule the replication windows so that they do not interfere with the planned switchback.
3. When using storage replication, unmount the shared storage volume with the middle-tier artifacts on the current production site and mount the corresponding volumes on the current secondary site which is the new production site.
4. Use Oracle Data Guard to switchback the databases.
5. On the new production site hosts, manually start all the processes. These include Oracle Fusion Middleware instances and any other processes in the application tier and the web tier.
6. Ensure that all user requests are routed to the secondary site by performing a global DNS push or something similar, such as updating the global load balancer. See Wide Area DNS Operations section.
7. Use a browser client to perform post switchback testing to confirm that requests are being resolved and redirected to the new production site.

   At this point, the former secondary site is the new production site and the former production site is the new secondary site.
8. Reestablish the replication between the two sites, but configure the replication so that the snapshot copies go in the opposite direction (from the new production site to the new secondary site). See the documentation for your shared storage to learn how to configure the replication so that snapshot copies are transferred in the opposite direction.

Performing a Failover

A failover operation sets the secondary site as the production role when the production site becomes unavailable. This is an unplanned operation where the primary site may no longer be accessible hence servers and storage in the primary cannot be managed and changes are only possible through the secondary operations.

To perform a failover operation:

1. Stop the replication between the production site shared storage and the secondary site shared storage (your shared storage should have a control module also in the secondary site).
2. When using shared storage replication, mount the shared storage volume with the middle-tier artifacts on the current secondary site which is the new production site.

3. From the secondary site, use Oracle Data Guard broker (dgrmgrl) to fail over the databases.
4. On the secondary site hosts, manually start all the processes. These include the Oracle Fusion Middleware instances and any other processes in the application tier and the web tier.
5. Ensure that all user requests are routed to the secondary site by performing a global DNS push or something similar such as updating the global load balancer. See Wide Area DNS Operations section.
6. Use a browser client to perform post failover testing to confirm that requests are being resolved and redirected to the production site.

   At this point, the secondary site is the new production site. You can examine the issues that caused the former production site to become unavailable.
7. Once the primary site is accessible, you can use the original production site as the new secondary site. You must reestablish the replication between the two sites but configure the replication so that the snapshot copies go in the opposite direction (from the current production site to the current secondary site). See the documentation for your shared storage system to learn how to configure the replication so that snapshot copies are transferred in the opposite direction.
8. Depending on the type and duration of the outage in the primary, you may need to reinstate and reconfigure Oracle Data Guard with the database in the original primary system. For more information about different failover situations and how to reinstate a failed primary, see Oracle Data Guard documentation and How To Reinstate Failed Primary Database into Physical Standby (Doc ID 738642.1).

To again use the original production site as the production site, perform a switchback as explained in Performing a Switchback.

Wide Area DNS Operations

When a site switchover or failover is performed, client requests must be redirected transparently to the new site that is now playing the primary role.

To accomplish this redirection, use either a global load balancer or manually changing DNS names.

Using a Global Load Balancer

A global load balancer deployed in front of the production and secondary sites provides fault detection services and performance-based routing redirection for the two sites.

In addition, the load balancer can provide authoritative DNS name server equivalent capabilities.

During normal operations, you can configure the global load balancer with the production site load balancer name-to-IP mapping. When a DNS switchover is required, this mapping in the global load balancer is changed to map to the secondary site's load balancer IP. This allows requests to be directed to the secondary site, which now has the production role.

This method of DNS switchover works for both site switchover and failover. One advantage of using a global 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 must be made for the global load balancer.

Manually Changing DNS Names

The DNS switchover involves manual change of the name-to-IP mapping of the production site's load balancer.

The mapping is changed to map to the IP address of the secondary site's load balancer. Follow these instructions to perform the switchover:

1. Note the current Time to Live (TTL) value of the production site's load balancer mapping. This mapping is in the DNS cache and it remains there until the TTL expires. As an example, assume that the TTL is 3600 seconds.
2. Modify the TTL value to a short interval. For example, 60 seconds.
3. Wait for one interval of the original TTL. This is the original TTL of 3600 seconds from *Step 1*.
4. Ensure that the secondary site is switched over to receive requests.
5. Modify the DNS mapping to resolve the secondary site's load balancer. It gives the appropriate TTL value for normal operation. For example, 3600 seconds.

This method of DNS switchover works for switchover or failover operations. The TTL value set in *Step 2* should be a reasonable time period where client requests cannot be fulfilled. The modification of the TTL effectively alters 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.

If the clients that point to FMW endpoints are running on Java, another TTL property can be taken into account. Configure the DNS cache in Java for caching the successful DNS resolutions. In that case, the change in the DNS server is not refreshed until Java is restarted. This can be modified by setting the property `networkaddress.cache.ttl` to a low value.

- You can do it globally, for all the applications that are running on the JVM, by modifying the property in `JAVA_HOME/jre/lib/security/java.security file: networkaddress.cache.ttl=60`
- You can define it for a specific application only, by setting the property in the application's initialization code:
  `java.security.Security.setProperty("networkaddress.cache.ttl" , "60")`

Alternatively, for global load balancers and DNS provider record updates, few cloud Web Application Firewall services like Oracle Cloud Infrastructure's Web Application Firewall provide a way to map a single front end DNS name (a CNAME) to multiple backend IPs (in a DR topology, these would be the IPs of the load balancers in the primary and the secondary). With the appropriate Edge Policies pointing to each region's load balancer IP, this can act as an effective Global Load Balancer that fails over requests from the primary to the secondary when there is a switchover. To use this alternative, refer to the WAF product specific documentation.

Expected RTO and RPO

This section provides information about the expected RTO and RPO during an outage.

Expected RTO

The Recovery Time Objective (RTO) describes the maximum acceptable downtime for a particular system when an outage occurs. The downtime caused by a failover depends on multiple "uncontrollable" factors, because it is normally an unplanned event caused by a critical issue that affects the system. But it is possible to measure the required downtime for a planned switchover event.

The following table shows the typical time taken by each switchover step in a sample Oracle FMW SOA 14.1.2 EDG system containing SOA, OSB, B2B, WSM and ESS clusters. These particular systems taken as example, use hosts with 4 CPU and 48 GB memory with 8GB maximum heap in the SOA servers. The WLS servers use out-of-the-box configuration in the connection pools of the different SOA Suite components. Additionally, this sample SOA system includes dozen of composites for different type of components (BPEL, MEDIATOR, EDN and so on).

Step No.	Switchover Step	Sample Times in FMW SOA EDG
1	Pre-switchover tasks	This does not cause downtime.
Downtime starts…		
2	Stop servers in primary site	
	2.1 Stop managed servers	~ 30 seconds (Force) / ~2 minutes (Graceful)
	2.2 Stop admin server	~ 8 seconds (Force) / ~2 minutes (Graceful)
3	Switchover DNS name	This is customer specific. For example, if you use OCI DNS it can be as low as 30 seconds, but it could take hours depending on the DNS provider used. This can be done in parallel with the rest of the steps.
4	Switchover Database	~3 minutes
5	Start the servers in secondary site	
	5.1 Start admin	~6 minutes (domain on shared storage)
	5.2 Start managed servers (in parallel)	~2 minutes
Total Downtown ~15 minutes		

Natural delays between steps or any other additional validation are not included in the above times, because they depend on how those switchover steps are executed (for example, manually, automated with custom scripts, with orchestration custom tools, and so on). So obviously, some additional time must be considered for the total time, not just the arithmetic sum of the times. The time for DNS switchover is also excluded because it is customer specific.

Normally, the total switchover time is expected to be in the range of 15-30 minutes. Here is a list of tips to minimize the downtime during the switchover operation:

- Perform any switchover related activity that does not require downtime before you stop the primary servers. For example, the WebLogic configuration replication based on rsync replication does not require downtime, you can perform it while the primary system is up

and running. Another example is to start any shutdown hosts or dependent resources in the secondary site.

- If possible, stop the managed servers and admin server in parallel.
- If applications and business allow it, use force shutdown to stop the WebLogic servers.
- The maximum time taken by the WebLogic servers to shutdown is limited by the parameters "server lifecycle timeout" (normally set to 30 seconds) and "graceful shutdown" (normally set to 120 seconds). Make sure that these parameters are configured to limit the maximum shutdown time.
- The front-end update in DNS is customer dependant. Use a low TTL value in the appropriate DNS entry (at least during the switchover operation) to reduce the time for update. Once the switchover is finished, the TTL can be reverted to its original value.
- Using Data Guard Broker commands (`dgmgrl`) to switchover the database is faster than using Enterprise Manager or other agent-based orchestrators.
- Load Balancers also take some time to realize that OHS and WebLogic servers are up before they start sending requests to them. It is usually some seconds, depending on the frequency of the LBR health checks. Lower the interval used for the checks in advance and revert it after switchover. You must be cautious when using very low intervals for the healthcheck as it could overload the backend.

Expected RPO

The Recovery Point Objective (RPO) describes the maximum amount of data loss that can be tolerated. For example, in Oracle FMW SOA's case, this is especially related to transaction logs, JMS messages, and SOA instance information all of which resides in the same database. Given that the database and the WebLogic configuration are replicated with different mechanisms, we can differentiate between the RPO for the runtime data and the RPO for the WebLogic configuration.

The actual achievable RPO for the runtime data relies upon the RPO of the database, because the runtime data (composite instances, JMS messages, TLogs, customer data, and so on) is stored in the database. In some cases, there can be runtime artifacts stored in the file systems too (like files consumed by file or ftp adapters). Therefore, the RPO for the runtime data depends upon the following:

1. The available network bandwidth and network reliability between primary and standby. Use connections between primary and secondary that provide a consistent performance for bandwidth, latency, and jitter. For a sample system like the one presented above, you can expect an RPO of approximately five minutes. For an optimum behavior, manual configuration of Fast-Start Failover Observer die, the Database may be required. Refer to the Oracle DB documentation for details about Fast-Start Failover.
2. The Data Guard protection mode used. There are three different modes: Maximum Availability, Maximum Protection or Maximum Performance (default).
    - **Maximum Availability** mode ensures zero data loss except in the case of certain double faults, such as failure of a primary database after failure of the standby database.
    - **Maximum Performance** mode offers slightly less data protection than maximum availability mode and has minimal impact on primary database performance.
    - **Maximum Protection** mode ensures that no data loss occurs if the primary database fails. To ensure that data loss cannot occur, the primary database shuts down instead of continuing processing transactions, if it cannot write its redo stream to at least one synchronized standby database.

The choice of one data guard protection mode or another is driven by business requirements. In some situations, a business cannot afford to lose data regardless of the circumstances. In other situations, the availability of the database may be more important than any potential data loss in the unlikely event of a multiple failure. Finally, some applications always require maximum database performance, and can therefore tolerate a small amount of data loss if any component fails. For more information, see Oracle Data Guard Protection Modes in the Oracle Data Guard Administration documentation.

3. Additionally, if there are runtime artifacts stored in file systems that are not located in the database (for example, files stored in custom File Storage Services which are consumed or generated by MFT or by File/FTP adapters), the RPO for this data depends on how frequently they are synchronized to the secondary location. What, how, and when this content should be synchronized is determined by the business needs. For example, if these runtime files are very volatile (created/consumed fast), syncing it may be unnecessary and an overkill. But if the content is more static, and it is required to have it in secondary in case of a DR event, the frequency to copy it should be according to the expected RPO of the system. The RPO will be the amount of data generated between the replications of this content.

   Alternatively, these runtime files can be located in a DBFS file system. In that case, they are replicated to standby via the underlying Data Guard replica, so the RPO is provided by the Data Guard protection mode.

The actual achievable RPO for the WebLogic configuration depends upon:

1. How frequently the WebLogic configuration is modified.

   The WebLogic configuration does not change as dynamically as the runtime data. Despite the initial stages of a system, it is not common to have configuration changes continuously. The more frequently the configuration is modified, the higher amount of config changes could be lost in a disaster event.

2. How frequently the WebLogic configuration is synchronized to the standby.

   When using shared storage and DBFS replication methods, the WebLogic configuration can be replicated manually or with cron jobs. One approach is to replicate the configuration after every configuration change that is performed in primary. This ensures that secondary WebLogic configuration is always up to date with primary but requires the replication process to be included in every change performed to primary. Another approach is to schedule the replication on a regular basis (for example, every night). In this case, when an outage event takes place, any configuration changes that were applied in primary after the last replication will be lost.

3. Reliability of the procedure used for the WebLogic configuration replication.

   All the replication methods are reliable, but any failure in the underlying infrastructure (for example, unavailability of the staging folder, connectivity outages, and so on) can impact the RPO. Thus, it is recommended to verify the proper functioning of the replication procedure and to perform regular validations of the secondary site.

5

Managing Lifecycle Operations

This chapter describes the additional maintainance tasks with specific implications in a disaster protection system such as ongoing synchronizations, patching, and scale out operations.

Scheduling Ongoing Replication Between the Primary and the Secondary Sites

Changes in the primary site need to be propagated to the secondary site to maintain a consistent behavior in case of a switchover or failover scenario. This is needed to guarantee that the validations and verifications are using realistic data without any modifications in the production site.

- For more information about different approaches and how to determine the replication frequency for each type of data used by the Fusion Middleware System, see Planning a File System Replication Strategy.
- During normal operations, the secondary site receives copies transferred periodically from the production site storage. As explained, this can be done through storage snapshots, rsync operations or using DBFS. After the copies are available, the secondary site storage includes all the data up to the point contained in the last transfer from the production site before the failover or switchover.
- When storage replication is used in asynchronous replication mode, then at the requested frequency (most vendors provide manual, on schedule, or continuous options) the changed data blocks at the production site shared storage (based on comparison to the previous snapshot copy) become the new snapshot copy. The snapshot copy is transferred to the secondary site shared storage.
- When using rsync or DBFS, ensure that either croned jobs or scheduler software is used to trigger the copy periodically so that it is used by the secondary WebLogic domain. The same scripts that were used in the initial setup should be scheduled to repeat synchronizations on an ongoing basis. Both DBFS and rsync with staging location require a dual scheduling: the copy from the primary nodes to the staging location and from the staging location to the secondary nodes. In the rsync peer-to-peer approach, the copy is done in a single step so only one operation needs to be scheduled.
- There will be a trade-off between the frequency of the copy and the overhead caused. The precise system's RTO should drive how to schedule the synchronization operations.
- Ensure to force a synchronization when you introduce a change to the middle-tier at the production site. For example, when you deploy a new application or module in the production site or change the configuration of data sources and JMS destinations. Follow the vendor-specific instructions to force a synchronization when using a storage replication technology or trigger the pertaining rsync copy to the secondary. When using DBFS ensure that the DBFS staging location is updated with the update from the primary also.
- After a switchover or failover, you need to revert the direction in which the replication is taking place. When the secondary location becomes the active site, the ongoing replication of configuration changes need to be reverted so that the copy should occur from this secondary region to the original primary or else you may overwrite the active system with obsolete changes from primary.

Scheduling Ongoing Replication With Rsync Scripts

Once you have determined the frequency of replication for each data type (see Planning a File System Replication Strategy), you can use scheduler software to program the replication operations. In Linux, you can use cron jobs for this replication.

If you are using a peer-to-peer model, you can use the following steps as a guideline:

1. As the root user in the secondary's WebLogic Administration Server node, execute the following:

   ```
   crontab -e
   ```

2. Edit the cron file and using the example scripts at https://github.com/oracle-samples/maa/tree/main/1412EDG, add the following lines: (notice that *)

   > **Note**
   >
   > `172.11.2.111` is the IP of the primary node running the WebLogic Administration Server in the examples provided in this document. Replace it with the precise IP in your case.

   ```
   0 0 * * * su oracle -c "/home/oracle/maa/1412EDG/rsync_for_WLS.sh
   172.11.2.111 /u01/oracle/config/ /home/oracle/my_keys/SSH_KEY.priv"
   0 0 * * * su oracle -c /home/oracle/maa/1412EDG/rsync_for_WLS.sh
   172.11.2.111 /u02/oracle/config/ /home/oracle/my_keys/SSH_KEY.priv"
   0 0 */7 * * su oracle -c "/home/oracle/maa/1412EDG/rsync_for_WLS.sh
   172.11.2.111  /u01/oracle/products/ /home/oracle/my_keys/SSH_KEY.priv"
   ```

3. As the root user in **each** secondary's WebLogic Managed Server's node, execute the following:

   ```
   crontab -e
   ```

4. Edit the cron file and using the example scripts at https://github.com/oracle-samples/maa/tree/main/1412EDG, add the following lines:

   > **Note**
   >
   > `172.11.2.112` is the IP of the Primary node running the second WebLogic Managed Server in the examples provided in this document. Replace it with the precise IP in your case)

   ```
   0 0 * * * su oracle -c /home/oracle/maa/1412EDG/rsync_for_WLS.sh
   172.11.2.112 /u02/oracle/config/ /home/oracle/my_keys/SSH_KEY.priv"
   0 0 * * *  su oracle -c "/home/oracle/maa/1412EDG/rsync_for_WLS.sh
   172.11.2.112  /u01/oracle/products/ /home/oracle/my_keys/SSH_KEY.priv"
   ```

> **Note**
>
> Since the EDG uses only two physical locations for the Oracle Homes (on shared storage), if you have more than two nodes, it is sufficient to schedule the `/u01/oracle/products` copy from the first two.

With this you have scheduled the Admin Server's domain directory and the Managed servers' domain directory to be pulled from primary every day at midnight and the Oracle Homes and JDK installation to be copied every week. Adjust the frequency according to your RTO, RPO, and change size needs. Similarly, create cron jobs in your OHS nodes (as explained in previous sections, OHS configuration changes much less frequently that WebLogic domains so you may adjust to less frequent copies).

If you are using a staging model, you can use the following steps as a guideline:

1. As the root user in the staging node and using the directory structure explained in the Preparing the Primary Storage for Rsync Replication, execute the following:

   ```
   crontab -e
   ```

2. Edit the cron file and using the example scripts at https://github.com/oracle-samples/maa/tree/main/1412EDG, add the following lines:

   > **Note**
   >
   > `172.11.2.111` is the IP of the Primary node running the WebLogic Administration Server and `172.11.2.112` is the IP of the Primary node running the second WebLogic managed server in the examples provided in this document. Replace it with the precise IP in your case.

```
0 0 * * * su oracle -c "export DEST_FOLDER=/staging_location/midtier/
wls_shared_config/;/home/oracle/maa/1412EDG/rsync_for_WLS.sh
172.11.2.111 /u01/oracle/config/ /home/oracle/my_keys/SSH_KEY.priv "
0 0 * * * su oracle -c "export DEST_FOLDER=/staging_location/midtier/
wls_private_config/wlsnode1_private_config;/home/oracle/maa/1412EDG/
rsync_for_WLS.sh 172.11.2.111 /u02/oracle/config/ /home/oracle/my_keys/
SSH_KEY.priv"
0 0 * * * su oracle -c "export DEST_FOLDER=/staging_location/midtier/
wls_private_config/wlsnode2_private_config;/home/oracle/maa/1412EDG/
rsync_for_WLS.sh 172.11.2.112 /u02/oracle/config/ /home/oracle/my_keys/
SSH_KEY.priv"

0 0 */7 * * su oracle -c "export DEST_FOLDER=/staging_location/midtier/
wls_products_home/wls_products_home1;/home/oracle/maa/1412EDG/
rsync_for_WLS.sh 172.11.2.111  /u01/oracle/products/ /home/oracle/my_keys/
SSH_KEY.priv"
0 0 */7 * * su oracle -c "export DEST_FOLDER=/staging_location/midtier/
wls_products_home/wls_products_home2;/home/oracle/maa/1412EDG/
rsync_for_WLS.sh 172.11.2.112  /u01/oracle/products/ /home/oracle/my_keys/
SSH_KEY.priv"
```

> **Note**
>
> It is recommended to have two copies of Oracle Homes from two different nodes in case one of them gets corrupted.

With this you have a *backup* in the staging node/location with the copies from primary. You will need to schedule its transfer from the staging node to the WebLogic nodes in secondary. Since there is a lag between the transfer from primary to the staging node, it is recommended to delay this second transfer. For example, at 2am everyday.

1. As the root user in the secondary's WebLogic Administration Server node, execute the following:

   ```
   crontab -e
   ```

2. Edit the cron file and using the example scripts at https://github.com/oracle-samples/maa/tree/main/1412EDG, add the following lines:

   > **Note**
   >
   > `staging_node_ip` is the IP of the staging node.

   ```
   0 2 * * * su oracle -c "export DEST_FOLDER=/u01/oracle/config/;/home/
   oracle/maa/1412EDG/rsync_for_WLS.sh staging_node_ip /staging_location/
   midtier/wls_shared_config/ /home/oracle/my_keys/SSH_KEY.priv"
   0 2 * * * su oracle -c "export DEST_FOLDER=/u02/oracle/config/;/home/
   oracle/maa/1412EDG/rsync_for_WLS.sh staging_node_ip /staging_location/
   midtier/wls_private_config/wlsnode1_private_config /home/oracle/my_keys/
   SSH_KEY.priv"
   0 2 * * * su oracle -c "export DEST_FOLDER=/u01/oracle/products  ;/home/
   oracle/maa/1412EDG/rsync_for_WLS.sh staging_node_ip  /staging_location/
   midtier/wls_products_home/wls_products_home1 /home/oracle/my_keys/
   SSH_KEY.priv"
   ```

3. As the root user in *each* secondary's WebLogic Managed Server's node, execute the following:

   ```
   crontab -e
   ```

4. Edit the cron file and using the example scripts at https://github.com/oracle-samples/maa/tree/main/1412EDG, add the following lines:

   > **Note**
   >
   > `staging_node_ip` is the IP of the staging node.

   ```
   0 2 * * * su oracle -c "export DEST_FOLDER=/u02/oracle/config; /home/
   oracle/maa/1412EDG/rsync_for_WLS.sh staging_node_ip / staging_location/
   ```

```
midtier/wls_private_config/wlsnode2_private_config; /home/oracle/my_keys/
SSH_KEY.priv"
0 2 * * * su oracle -c "export DEST_FOLDER=/u01/oracle/products; /home/
oracle/maa/1412EDG/rsync_for_WLS.sh staging_node_ip / staging_location/
midtier/wls_products_home/wls_products_home2 /home/oracle/my_keys/
SSH_KEY.priv"
```

> **Note**
>
> Since the EDG uses only two physical locations for the Oracle Homes (on shared storage), if you have more than two nodes it is sufficient to schedule the `/u01/oracle/products` copy from the first two.

With this you have scheduled the Admin Server's domain directory and the Managed servers' domain directory to be pulled from primary every day at mid night (pushed at 2AM to secondary nodes) and the Oracle Homes and JDK installation to be copied every week. Adjust the frequency according to your RTO, RPO, and change size needs. Similarly, create cron jobs in your OHS nodes (as explained in previous sections, OHS configuration changes much less frequently that WebLogic domains so you may adjust to less frequent copies).

In all cases (whether using the per-to-peer or staging models) check on a regular basis your system's cron log. For example:

```
grep rsync /var/log/cron
```

If you are using the scripts provided at https://github.com/oracle-samples/maa/tree/main/1412EDG, you can also check the logs directory reported by the cron jobs to see the result of each periodic copy.

Besides the appropriate execution of the cron jobs, it is recommended to test and validate secondary on a regular basis when periodic copies are scheduled.

```
sudo grep rsync /var/log/cron | grep log
May 6 00:00:02 bastion-vcnpho80 CROND[497562]: (root) CMDOUT ((You can check
rsync command and exclude list in /home/oracle/maa/1412EDG/logs/
rsync_u02_oracle_products__06-06-2025-00-00-02.log))
May 6 00:00:02 bastion-vcnpho80 CROND[497563]: (root) CMDOUT ((You can check
rsync command and exclude list in /home/oracle/maa/1412EDG/logs/
rsync_u02_oracle_config__06-06-2025-00-00-02.log))
May 6 00:00:02 bastion-vcnpho80 CROND[497564]: (root) CMDOUT ((You can check
rsync command and exclude list in /home/oracle/maa/1412EDG/logs/
rsync_u02_oracle_config__06-06-2025-00-00-02.log))
May 6 00:00:02 bastion-vcnpho80 CROND[497565]: (root) CMDOUT ((You can check
rsync command and exclude list in /home/oracle/maa/1412EDG/logs/
rsync_u02_oracle_config__06-06-2025-00-00-02.log))
May 6 00:00:02 bastion-vcnpho80 CROND[497566]: (root) CMDOUT ((You can check
rsync command and exclude list in /home/oracle/maa/1412EDG/logs/
rsync_u02_oracle_config__06-06-2025-00-00-02.log))
May 6 00:00:02 bastion-vcnpho80 CROND[497561]: (root) CMDOUT ((You can check
rsync command and exclude list in /home/oracle/maa/1412EDG/logs/
rsync_u02_oracle_products__06-06-2025-00-00-02.log))
May 6 00:00:02 bastion-vcnpho80 CROND[497567]: (root) CMDOUT ((You can check
```

```
rsync command and exclude list in /home/oracle/maa/1412EDG/logs/
rsync_u01_oracle_config__06-06-2025-00-00-02.log))
```

To avoid undesired overwrites in the application tier (such as accidentally syncing configuration from primary to secondary when secondary is the active site), it is a good practice to base the direction of the replication (whether to copy configuration, binaries and runtime from the original primary to secondary or the other way around) on the database role in each site. The webtier's configuration changes less frequently than the application tiers one. It is also more sensitive to provide access from the webtier to the database than from the apptier. Hence, it is recommended to add logic in the application tier's rsync scripts to base the direction of the copy on the role detected in the database. If the local database is in `PHYSICAL STANDBY` or `SNAPTHOT STANDBY` role, the scripts should pull information from its remote peer, and if it is in `PRIMARY ROLE` the scripts should push information to that remote peer. Refer to the scripts at https://github.com/oracle-samples/maa/tree/main/app_dr_common for examples to implement this logic.

Patching an Oracle Fusion Middleware Disaster Recovery Site

An appropriate disaster recovery strategy needs to address how to apply Oracle Fusion Middleware patches to upgrade the Oracle Homes that participate in an Oracle Fusion Middleware Disaster Recovery site.

Oracle Central Inventory for any Oracle Fusion Middleware instance in the primary system that you are patching should be located on the production site shared storage or rsync-covered location so that the Oracle Central Inventory for the patched instance can be replicated to the secondary site.

Perform the following steps to apply an Oracle Fusion Middleware patch:

1. Perform a backup of the production site to ensure that the starting state is secured.
2. Apply the patch set to upgrade the production site instances.
3. After you apply the patch set, manually force a synchronization of the production site shared storage and secondary site shared storage. This replicates the production site's patched instance and Oracle Central Inventory in the secondary site's shared storage.
4. After you apply the patch set, ensure your secondary database is a physical standby (no snapshot standby) so that the Oracle Data Guard will synchronize the Oracle primary and secondary databases. When few Oracle Fusion Middleware patch sets make updates to repositories, this step ensures that any changes made to the production site databases are synchronized to the secondary site databases.

   The upgrade is now complete. Your disaster recovery topology is ready to resume processing.

> **Note**
>
> Patches must be applied only at the production site for an Oracle Fusion Middleware Disaster Recovery topology. If a patch is for an Oracle Fusion Middleware instance or for the Oracle Central Inventory, the patch is copied when the production site shared storage is replicated to the secondary site shared storage. A synchronization operation should be performed when a patch is installed at the production site.

When patching the database, check the documentation for that specific patch for information on how to apply the patch in a Data Guard topology.

A disaster recovery topology helps (in some cases) to reduce the patching downtime for the primary system. There are differences depending on the components affected by the patch.

- Database Patches

  Oracle Fusion Middleware Disaster Recovery uses Data Guard. The advantage of using Data Guard instead of having only a primaryDB system is that you can first patch one site and then the other. However, not all the database patches allow this approach. The downtime and procedure to patch the database depends on the type of patch. Database patches are of the following types:

  – Data Guard Standby-First

    These can be applied first in standby and then in primary. There are various options available for applying this type of patches. See Oracle Patch Assurance - Data Guard Standby-First Patch Apply (Doc ID 1265700.1).

  – Non Data Guard Standby-First

    These kind of patches must be applied on both primary and standby databases at the same time and require a shutdown.

    If the database patch is standby first applicable, the downtime can be minimized or reduced to a switchover. If not, it requires shutdown of primary and standby and must be applied in both.

- Mid-tier-only Patches (patches modifying only mid-tier binaries)

  – A few Fusion Middleware patches are marked as `FMW_ROLLING_ORACLE_HOME` in their readme. This type of patches does not incur in any downtime regardless of using disaster recovery or not.

  – Other patches are not `FMW_ROLLING_ORACLE_HOME` enabled and require a mid-tier shutdown. In these cases, a disaster recovery topology helps minimizing downtime by using the following procedure:

    1. Convert the secondary database to snapshot standby.
    2. Patch the secondary mid-tier domain first.
    3. Test the secondary domain with the patch.
    4. After everything is validated on the secondary, convert the secondary database back to physical standby.
    5. Switchover to secondary (at this point secondary region becomes your primary and runs the business).
    6. Convert the old primary database to snapshot.
    7. Patch the old primary mid-tier and test it.
    8. Convert the database back to physical standby.
    9. Switchback to the original site.

       In these cases, the downtime is only the time spent on the switchover operation. Without a standby system the downtime would include the patching time and the time to stop and start the system.

- Midtier Patches that Include DB Schema Changes

  If the patch is not `FMW_ROLLING_ORACLE_HOME` enabled, the approach is a bit different to avoid losing the database changes (db schema changes require to patch mid-tier and db at the same time). Perform the following steps:

  1. Convert the secondary database to snapshot standby.

2. Patch the secondary mid-tier domain first.
3. Test the secondary domain with the patch.
4. After everything is validated on the secondary, convert the secondary database back to physical standby. At this point, the secondary WebLogic domain is misaligned: the mid-tier has one version but the schemas are in the older version.
5. Patch Primary

   The downtime with this approach is the same as without a secondary, but the procedure above has an advantage that it allows you to verify the patch's behavior in the secondary before applying it in the primary.

Scale Operations in a Fusion Middleware Disaster Recovery System

<<Awaiting inputs for this section/topic>>