Oracle® Application Server Portal

Configuration Guide

10*g* (9.0.4)

Part No. B10356-01

November 2003

ORACLE®

Oracle Application Server Portal Configuration Guide, 10g (9.0.4)

Part No. B10356-01

Primary Author: Peter Lubbers

Contributing Authors: Pravin Prabhakar, Frank Rovitto, Rosie Harvey, Darren McBurney, and Rod Ward.

Contributors: Arun Arat Tharakkal, Balaravikumar.Shanmugasundaram, Barry Hiern, Binodkumar Gupta, Chris van Es, Chung-Ho Chen, Dawn Tyler, Demetris Christou, Dmitry Nonkin, Eddy Chee, Eric Lee, Greg Cook, Harry Wong, Helen Barnes, Jason Pepper, Joan Carter, John Bellemore, Madhu Muppagowni, Marcie Caccamo, Mark Clark, Mark Loper, Matthew Davidchuk, Michele Cyran, Mick Andrew, Nick Pounder, P.V. Dharan, Pascal Gibert, Paul Encarnacion, Paul Spencer, Peter Moskovits, Pushkar Kapasi, Ramana Adusumilli, Ratna Bhavsar, Rob Giljum, Ross Clewley, Sachin.Parashar, Senthil Arunagirinathan, Sergiy Pecherskyy, Sunil Marya, Susan Highmoor, Tim Willard, Todd Vender, Venu Surakanti, and Viswanath Dhulipala.

Contents

4 Performing Basic Configuration and Administration

5   Performing Advanced Configuration

6   Securing OracleAS Portal

7   Monitoring and Administering OracleAS Portal

8 Configuring the Search Features in OracleAS Portal

9   Tuning Performance in OracleAS Portal

10   Exporting and Importing Content

11 Syndicating Content Into OracleAS Portal

12 Using the Federated Portal Adapter

13 Troubleshooting OracleAS Portal

B   Using the OracleAS Portal Configuration Assistant Command Line Utility

C   Using OracleAS Portal Installation and Configuration Scripts

D   Configuring the Parallel Page Engine

E   Using Oracle Application Server Configuration Files

F   Integrating JavaServer Pages with OracleAS Portal

G Using the wwv_context APIs

H Using TEXTTEST to Check Oracle Text Installation

I   Administering Web Clipping

Index

Send Us Your Comments

Oracle Application Server Portal Configuration Guide, 10*g* (9.0.4)

Part No. B10356-01

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

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

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

- Electronic mail: appserverdocs_us@oracle.com
- FAX: 650-506-7375 Attn: Oracle Application Server Documentation Manager
- Postal service:
  Oracle Corporation
  Oracle Application Server Documentation
  500 Oracle Parkway, M/S 1op6
  Redwood Shores, CA 94065
  USA

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

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

Preface

This manual describes how to configure Oracle Application Server Portal. This includes how to plan, upgrade, check pre-installation requirements, and perform post-installation tasks. This guide further explains some more advanced Portal deployments, and explains how to perform the advanced configuration required for these deployments. Finally, there is information about monitoring and troubleshooting.

A word about the OracleAS Portal documentation set...

If you've used previous releases of OracleAS Portal, you'll most likely notice that the documentation set is quite different in 10*g* (9.0.4). In the past, most task-related information was available in the online help. Beginning with 10*g* (9.0.4), this information is presented in book form only. Field-level and conceptual help are still available through the online help system and, in future releases, will provide links to step-by-step information in the appropriate manuals.

Intended Audience

This guide is intended for two kinds of users:

- **OracleAS Portal Administrators**, who are responsible for configuring and maintaining OracleAS Portal.

- **Oracle Application Server administrators**, who must configure OracleAS Portal to work with other Oracle Application Server components.

Documentation Accessibility

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

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

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

Organization

This manual contains five parts, thirteen chapters, nine appendixes, and an index.

Part I, "Concepts"

This part contains the chapters that explain the OracleAS Portal architecture, and how to plan a Portal.

Chapter 1, "Understanding the OracleAS Portal Architecture"

This chapter introduces OracleAS Portal, and explains how it fits in the Oracle Application Server architecture.

Chapter 2, "Planning Your Portal"

This chapter provides conceptual information about planning Portals.

Part II, "Installation and Basic Configuration"

This part contains the chapters that discuss installation and basic configuration.

Chapter 3, "Installing OracleAS Portal"

This chapter guides the administrator through the installation process.

Chapter 4, "Performing Basic Configuration and Administration"

This chapter assumes that OracleAS Portal has been installed as part of the Oracle Application Server and addresses the basic tasks that you can perform once installation is complete.

Part III, "Advanced Configuration Topics"

This part contains the chapters that discuss special configurations.

Chapter 5, "Performing Advanced Configuration"

This chapter provides instructions on how to perform more advanced OracleAS Portal configuration and integration configuration, including middle-tier, proxy server, Oracle Application Server Web Cache, and Oracle Application Server Single Sign-On configuration.

Chapter 6, "Securing OracleAS Portal"

This chapter provides instructions on how to secure OracleAS Portal.

Chapter 7, "Monitoring and Administering OracleAS Portal"

This chapter provides information about monitoring tools, and how to use them to successfully monitor OracleAS Portal.

Chapter 8, "Configuring the Search Features in OracleAS Portal"

This chapter provides instructions on configuring Oracle Text to perform text searching in page groups created with OracleAS Portal and information on how to set up and start using Oracle Ultra Search.

Chapter 9, "Tuning Performance in OracleAS Portal"

This chapter discusses how you can tune the performance of your OracleAS Portal configuration.

Chapter 10, "Exporting and Importing Content"

This chapter discusses how to use the import and export functionality that enables you to migrate portal content between portal installations.

Chapter 11, "Syndicating Content Into OracleAS Portal"

This chapter provides instructions on how to use OracleAS Syndication Services to syndicate content into OracleAS Portal.

Chapter 12, "Using the Federated Portal Adapter"

This chapter discusses how to configure the Federated Portal Adapter.

Chapter 13, "Troubleshooting OracleAS Portal"

This chapter provides solutions to problems you may encounter while installing, or using OracleAS Portal.

Part IV, "Appendices"

This part contains the appendixes.

Appendix A, "Using the Portal Dependency Settings File"

This appendix provides information on using the *Portal Dependency Settings File*. This file contains all the settings required to configure the integration of OracleAS Portal with its dependent components.

Appendix B, "Using the OracleAS Portal Configuration Assistant Command Line Utility"

This appendix describes how to use the OracleAS Portal Configuration Assistant, and provides explanations of all the options for each mode that OPCA can be run in.

Appendix C, "Using OracleAS Portal Installation and Configuration Scripts"

This appendix provides information about various scripts that are used for customizing the configuration.

Appendix D, "Configuring the Parallel Page Engine"

This appendix provides information on configuring the Parallel Page Engine (PPE), a part of the OracleAS Portal middle-tier. The PPE reads page metadata, calls providers for portlet content, accepts provider responses, and assembles the requested page in the specified page layout.

Appendix E, "Using Oracle Application Server Configuration Files"

This appendix provides information about the configuration files that can affect the connection to and the behavior of the Oracle Application Server and its middle-tier components, as well as the other machines to which it is connecting.

Appendix F, "Integrating JavaServer Pages with OracleAS Portal"

This appendix describes how you can secure OracleAS Portal to allow access to only approved JSPs, and prevent unauthorized access by JSPs to portlet content. It also describes the steps required to allow access for protected external JSPs that require login.

Appendix G, "Using the wwv_context APIs"

This appendix describes the wwv_context API.

Appendix H, "Using TEXTTEST to Check Oracle Text Installation"

This appendix describes the use of the TEXTTEST utility to check the functionality of Oracle Text.

Appendix I, "Administering Web Clipping"

This appendix describes the steps involved in configuring Web Clipping.

Part V, "Index"

Index

The index provides a way to quickly find information within this manual.

Related Documents

For more information, see the following manuals in the OracleAS Portal documentation set:

- Oracle Application Server Portal Release Notes
- *Oracle Application Server Portal User's Guide*

You'll find a wealth of information about OracleAS Portal on Portal Center, http://portalcenter.oracle.com.

> **Note:** A complete glossary of OracleAS Portal-related terminology can be found in the *Oracle Application Server Portal User's Guide*

You may also find the following manuals in the Oracle Application Server documentation set useful:

- *Oracle Application Server 10g Concepts*

- *Oracle Application Server 10g Security Guide*

- *Oracle HTTP Server Administrator's Guide*

- *Oracle Application Server Web Cache Administrator's Guide*

- *Oracle Application Server Wireless Administrator's Guide*

- *Oracle Application Server Single Sign-On Administrator's Guide*

- *Oracle Internet Directory Administrator's Guide*

- *Oracle Application Server Syndication Services Developer's and Administrator's Guide*

- *Oracle Application Server 10g Migrating from Oracle Application Server*

Conventions

The following conventions are used in this manual:

Convention	Meaning
Italicized text	Italicized type introduces important terms used for the first time.
Boldface text	Boldface type is used for emphasis and to represent the names of items as they appear on your screen.
CAPITALIZED text	Capitalized text indicates procedure names.
< >	Angle brackets enclose user-supplied information.
[ ]	Brackets enclose optional clauses from which you can choose one or none.
. . .	Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.

Part I

Concepts

Part one contains the following chapters:

- Chapter 1, "Understanding the OracleAS Portal Architecture"
- Chapter 2, "Planning Your Portal"

1

Understanding the OracleAS Portal Architecture

This chapter introduces Oracle Application Server Portal and explains how it fits in the Oracle Application Server architecture.

This chapter contains the following sections:

- What Is the Oracle Application Server?, which provides you with a basic understanding of the solutions and components comprising Oracle Application Server so you can better understand how they work in concert with OracleAS Portal.

- Understanding the OracleAS Portal Architecture, which describes how OracleAS Portal, and relevant pieces of Oracle Application Server work together.

- Understanding Caching in OracleAS Portal, which describes the caching configurations you can implement to increase the availability and scalability of medium to large deployments.

> **Note:** OracleAS Portal cannot be installed standalone, but must be installed as part of Oracle Application Server.

1.1 What Is the Oracle Application Server?

Oracle Application Server is a completely standards-based application server that provides a comprehensive and fully integrated platform for running Web sites, J2EE applications, and Web services. It addresses all the challenges that you face as you refine your business processes to become an e-business.

Oracle Application Server provides full support for the J2EE platform, XML, and emerging Web services standards. With Oracle Application Server, you can simplify information access for your customers and trading partners by delivering *enterprise portals* that can be customized and accessed from a network browser or from wireless devices. It enables you to redefine your business processes and integrate your applications and data sources with those from your customers or partners. You can deliver tailored customer experiences through real-time personalization, and assess and correlate customer navigation, purchasing, ratings, and demographic data.

You can also implement a centralized management, security, and directory framework to manage and monitor all of your distributed systems and diverse user communities. Oracle Application Server maximizes your Web site infrastructure by deploying your fast, scalable Internet applications through built-in Web caching, load balancing, and clustering capabilities.

1.1.1 What are the Oracle Application Server Solutions and Components?

Oracle Application Server is actually a set of Oracle Application Server *solutions*. Each solution contains one or more *components*. A component can be a service, an API, or an application. The solutions provided by Oracle Application Server are:

- J2EE and Internet Applications
- Portals
- Wireless
- Business Intelligence
- E-Businesses Integration
- Availability and Scalability
- Caching
- Management and Security

All of these solutions are built upon a scalable and highly available infrastructure, as illustrated in Figure 1–1.

Figure 1–1   Oracle Application Server Solutions
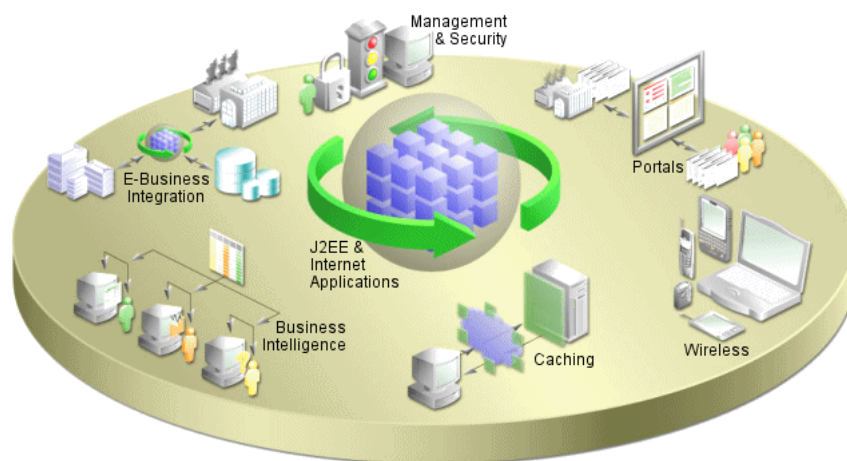
The next few sections explain a bit about each solution and the components they contain.

J2EE and Internet Applications

Oracle Application Server is built entirely on a J2EE framework that supports the latest industry standard technologies and programming languages, including J2EE API specifications, XML, and Web services. This comprehensive and flexible framework enables you to design, develop, and deploy dynamic Web sites, portals, and transactional applications using familiar programming languages and technologies. Oracle Application Server also provides comprehensive Web services to expose business functions to authorized parties over the Internet from any Web device.

The following Oracle Application Server components are configured to use the J2EE and Internet Applications solution:

- Oracle HTTP Server

- Oracle Application Server Containers for J2EE

- Oracle Application Server TopLink

- Oracle Business Components for Java

- Oracle Application Server Web Services

- Oracle Application Server Forms Services

- Oracle XML Developer Kit

- Oracle PL/SQL

- Oracle Content Management Software Development Kit

- Oracle Application Server MapViewer

Portals

Oracle Application Server provides an out-of-the-box portal that requires little programming and maintenance effort, if any. You can use Oracle Application Server to build, deploy, and maintain self-service and integrated enterprise portals. Oracle Application Server enables wizard-based development, as well as deploying, publishing, and consuming Web services on an extensible framework.

The following Oracle Application Server components are configured to use the Portals solution:

- Oracle Application Server Portal

- Oracle Application Server Portal Developer Kit

- Oracle Ultra Search

- Oracle Application Server Syndication Services

Wireless

Oracle Application Server Wireless simplifies wireless development and deployment by providing the ability to deliver content to any device, to use any protocol, and to work across any wireless network. In addition, OracleAS Wireless includes wireless services, such as e-mail and location-based services that simplify wireless-enabling applications and portals. Oracle Application Server provides application developers independence from the underlying wireless infrastructure. OracleAS Wireless is built on the core Oracle Application Server infrastructure, leveraging open standards support in XML and J2EE to deliver a high-performance and scalable wireless infrastructure.

The following Oracle Application Server component is configured to use the Wireless solution:

- Oracle Application Server Wireless

Business Intelligence

Oracle Application Server provides comprehensive personalization and business intelligence services. Using Oracle Application Server business intelligence features,

you can dynamically serve personalized content recommendations to both registered and anonymous visitors as they browse your site; perform dynamic, ad-hoc query reporting and analysis using a standard Web browser; and publish high quality, dynamically generated reports on a scalable, secure platform.

The following Oracle Application Server components are configured to use the Business Intelligence solution:

- Oracle Application Server Reports Services

- Oracle Application Server Discoverer

- Oracle Application Server Personalization

E-Businesses Integration

Oracle Application Server has a powerful set of features that provide communications and integration capabilities for e-business applications. Using Oracle Application Server, you can integrate enterprise applications, trading partners, and Web services, emphasizing scalability and manageability, and provide seamless query and transaction access to many non-Oracle data sources.

The following Oracle Application Server components are configured to use the E-Business Integration solution:

- Oracle Application Server InterConnect

- Oracle Application Server ProcessConnect

Availability and Scalability

Oracle Application Server provides a flexible deployment model that enables you to architect your system for high availability and scalability. Oracle Application Server provides a variety of options for improving availability and scalability, and provides features for implementing fault tolerance, death detection, and failover. Additionally, Oracle Application Server supports such high availability options as cold failover clusters and Real Application Cluster (RAC).

Caching

Oracle Application Server Web Cache is a Web caching solution with the unique capability of caching both static and dynamically generated Web content. OracleAS Web Cache significantly improves the performance and scalability of heavily loaded Web sites. In addition, OracleAS Web Cache provides a number of features to ensure consistent and predictable responses. These features include page fragment caching, Edge Side Includes (ESI) and Edge Side Includes for Java (JESI)

support, compression, dynamic content assembly, Web server load balancing, Web cache clustering, and failover.

The following Oracle Application Server component is configured to use the caching solution:

- Oracle Application Server Web Cache

Management and Security

Oracle Application Server provides a set of management facilities that are based on industry standards to simplify all aspects of Web site administration. Using Oracle Application Server, you can:

- Use encrypted secure sockets layer (SSL) connections, user and client certificate-based authentication, and single sign-on across all applications

- Implement an LDAP directory that provides a single repository and administration environment for user accounts

The following Oracle Application Server components are configured to use the Management and Security solution:

- Oracle Enterprise Manager

- Oracle Application Server Single Sign-On

- Oracle Application Server Certificate Authority

- Oracle Application Server Java Authentication and Authorization Service

- Oracle Internet Directory

- OracleAS Infrastructure 10*g*

1.1.2 Overview of the Oracle Application Server Architecture

The Oracle Application Server architecture consists of three basic tiers:

- Client Tier

- Middle-Tier

- Infrastructure Tier

It's important to understand a bit about the overall Oracle Application Server architecture so you can more fully understand how your OracleAS Portal configuration fits within that structure. The next few sections provide some key concepts and terms you'll need as you plan your configuration strategy.

Client Tier

From the client computer, a user can connect to the middle-tier and the infrastructure tier to access the self-service tools for publishing information, build applications, deploy content management, and administer enterprise portal environment.

Middle-Tier

The middle-tier, or *application server* tier, is a set of Oracle Application Server components installed into a single Oracle home. A single enterprise can have one or more application server installations, either residing on one host or, for more complex installations, distributed across multiple hosts.

Infrastructure Tier

The infrastructure installation consists of several components that help authenticate users, store access control information, and pass on the required content to the user based on the privileges the user has on OracleAS Portal. Like the middle-tier components, infrastructure components can be distributed across multiple hosts to enable scalability and high availability.

Figure 1–2 illustrates the three parts of the Oracle Application Server architecture.

Figure 1–2   Components of the Oracle Application Server Architecture
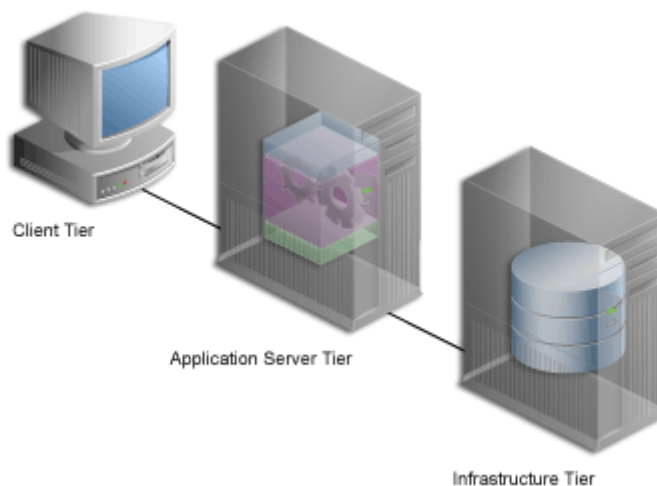
1.1.2.1 What Are the Middle-Tier Components?

The middle-tier is the part of an Oracle Application Server architecture that contains several components responsible for accepting requests from clients, validating the requests, and providing content, while using intelligent data caching for faster and reliable performance.

For OracleAS Portal, the middle-tier handles all Web requests by forwarding them to the appropriate provider. This is also where Portal pages are assembled, and where the caching of Portal content is managed. The middle-tier also provides other functions for other Oracle Application Server components.

Some of the key components for OracleAS Portal in the Oracle Application Server middle-tier are:

- **Oracle Application Server Containers for J2EE**. Oracle Application Server Containers for J2EE (OC4J) are fast, lightweight, and scalable J2EE containers that are written in Java and run on a standard Java Virtual Machine (JVM). OracleAS Portal's Parallel Page Engine (PPE), for example, is a servlet that assembles Portal pages, and runs in the Oracle Application Server Containers for J2EE. OC4J have been designed for ease of use and to support standard APIs.

- **Oracle HTTP Server**. Oracle HTTP Server (OHS) is the underlying deployment platform for all programming languages and technologies Oracle Application Server supports. Providing a Web listener for OC4J and the framework for hosting static and dynamic pages and applications over the Web, Oracle HTTP Server includes significant features that facilitate load balancing, administration, and configuration.

  For OracleAS Portal, OHS handles all incoming HTTP requests to OracleAS Portal, by forwarding them to the either the Parallel Page Engine (PPE) servlet or mod_plsql. The PPE is a servlet that assembles Portal pages, and runs in the Oracle Application Server Containers for J2EE. mod_plsql is an OHS module that is used to access Portal and page metadata by executing PL/SQL procedures residing in the Oracle Application Server database, as well as generating HTTP responses.

- **Oracle Application Server Web Cache**. Works together with OracleAS Portal's own file-based caching to cache page definitions and content in memory, to boost performance. OracleAS Portal is closely integrated with OracleAS Web Cache to improve Portal's overall availability, scalability, and performance. OracleAS Web Cache combines caching, compression, and assembly technologies to accelerate the delivery of both static and dynamically generated Portal content.

Figure 1–3 The Middle-Tier Components
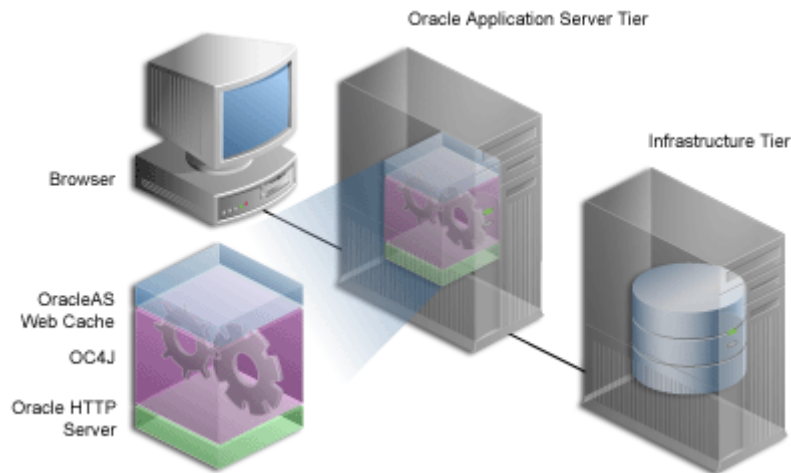
There are three types of middle-tier installations:

1. **Oracle Application Server Containers for J2EE and OracleAS Web Cache**, which is the simplest configuration and does not contain any of the Portal Solution components.

2. **OracleAS Portal and OracleAS Wireless**, which adds the Portal and Wireless solutions to those provided by Oracle Application Server Containers for J2EE and OracleAS Web Cache.

3. **Business Intelligence and Forms**, which contains all of the middle-tier components, including OracleAS Portal.

To use OracleAS Portal, you must choose Option 2, or Option 3.

Refer to the following sections for more information:

- Section 2.1.7, "How Should I Configure My Hardware and Software?"
- Section 5.3, "Configuring Multiple Middle-Tiers with a Load Balancing Router"

1.1.2.2 What Are the Infrastructure Components?

By default, the infrastructure tier handles all authentication requests and hosts the Oracle Application Server Metadata Repository, which contains schemas and business logic used by application server components (including OracleAS Portal) and other pieces of the infrastructure.

For the OracleAS Portal middle-tier installation, the infrastructure tier is a prerequisite.

The Oracle Application Server Infrastructure contains:

- **Application Server Control**. This administration console for the Oracle Application Server enables you to administer clusters, start and stop services, enable and disable components, view logs and ports, and monitor servers in real-time.

- **Oracle Internet Directory**. An LDAP version 3 compliant repository for storing user credentials and group memberships for OracleAS Portal and other Oracle products.

- **Oracle Application Server Single Sign-On (SSO)**. Authenticates user credentials against Oracle Internet Directory for OracleAS Portal and other applications, thus enabling users to log on once to the Web portal to access multiple accounts and applications with a single username and password.

- **Oracle Application Server Metadata Repository**. The repository is installed in an Oracle9*i* Database Server and consists of a collection of schemas that contain product metadata for Oracle Application Server components. Some middle-tier components, such as OracleAS Portal, store their metadata in this repository and need access to that metadata during runtime.

Figure 1–4   The Infrastructure Tier Components
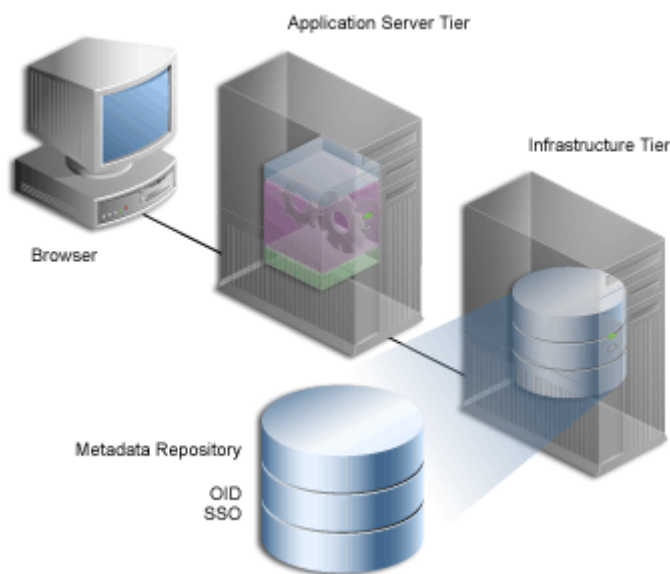
You can install multiple instances of any of these components on multiple servers, and then connect the servers to suit your needs. Deployment configuration options for OracleAS Portal range from installing everything on a single machine to multitier configurations in which the pieces comprising OracleAS Portal are located across multiple servers.

There are three types of Infrastructure installations:

1. **Oracle Identity Management**, which installs and configures Oracle Identity Management services (Oracle Internet Directory, OracleAS Single Sign-On, Oracle Delegated Administration Services, Oracle Directory Integration and Provisioning, OracleAS Certificate Authority).

2. **OracleAS Metadata Repository**, which installs a new Oracle9*i* Database Server containing the OracleAS Metadata Repository, and also stores the database objects that comprise OracleAS Portal, Oracle Internet Directory and OracleAS Single Sign-On.

3. **Oracle Identity Management components and OracleAS Metadata Repository**, which consists of all the components listed in the preceding two installation types.

> **Note:** Throughout this guide, you will see references to *ORACLE_HOME*. *ORACLE_HOME*, represents the full path of the Oracle home, and is used in cases where it is easy to determine which Oracle home is referenced. The following conventions are used in procedures where it is necessary to distinguish between the middle-tier, Infrastructure, or OracleAS Metadata Repository Oracle home:
>
> - *MID_TIER_ORACLE_HOME*, represents the full path of the middle-tier Oracle home.
>
> - *INFRA_ORACLE_HOME*, represents the full path of the Infrastructure Oracle home.
>
> - *METADATA_REP_ORACLE_HOME*, represents the full path of the Infrastructure home containing the OracleAS Metadata Repository.

1.2 Understanding the OracleAS Portal Architecture

After your development team builds your Web portal, the next step is to deploy a production version of it. Successful deployment means that end users are able to access content in a timely manner, without delays, errors, or server downtime. Because OracleAS Portal can be installed in a variety of configurations on different machines, a successful deployment ultimately depends how you configure Portal to address the requirements of your site. This section provides some background information that should be useful to you as you plan your configuration.

1.2.1 How Does OracleAS Portal Integrate with Other Components?

Some Oracle Application Server components serve as *portlet providers*[1] for OracleAS Portal, which means you can easily integrate information from various components into a single portal page. Other components provide essential services to OracleAS Portal, as described in the following list.

- **Oracle Application Server Reports Services**. OracleAS Portal includes a simple report building facility. However, as your reports become more complex, you may want to import the report into OracleAS Reports Services to take full

---

[1] Applications and information sources, represented as portlets, communicate with the portal through a provider. Each portlet only has one provider, and a provider can have one or more portlets that expose an underlying application or information source.

advantage of the functionality it offers. You can deploy any OracleAS Reports Services report as a portlet.

> **See Also:** *Oracle Application Server Reports Services Publishing Reports to the Web*

- **Oracle Application Server Discoverer**. As a portlet provider, OracleAS Discoverer offers Worksheet portlets and List of Workbooks portlets to OracleAS Portal users. A Worksheet portlet contains information from a single Discoverer worksheet. The portlet displays this information in a table, a graph, or both. The List of Workbooks portlet presents a list of available workbooks.

  > **See Also:**
  >
  > - The chapter titled "*Publishing workbooks to OracleAS Portal*" in the *Oracle Application Server Discoverer Plus User's Guide* describes how to add a discoverer portlet.
  >
  > - The chapter titled "*Using OracleAS Discoverer with OracleAS Portal*" in the *Oracle Application Server Discoverer Configuration Guide* describes how to register the OracleAS Discoverer portlet provider with OracleAS Portal.

- **Oracle Application Server Syndication Services**. Delivers any database, legacy file system, or Internet content to Internet subscribers, and automatically provides content updates using standards across any network. This simplifies the process of syndication or automated content exchange. OracleAS Syndication Services provides a comprehensive solution for content aggregation, syndication, and distribution by letting you make available any or all of your content. Refer to Chapter 11, "Syndicating Content Into OracleAS Portal" for more information on how to syndicate content into OracleAS Portal.

- **Oracle Ultra Search**. Integrated with OracleAS Portal, Oracle Ultra Search enables OracleAS Portal users to add a powerful multi repository search to their portal pages. It also has the capability to crawl OracleAS Portal's own repository and search *public* content. For more information about Oracle Ultra Search, refer to Chapter 8, "Configuring the Search Features in OracleAS Portal".

- **Oracle Application Server Wireless**. Working with OracleAS Wireless, OracleAS Portal automatically transforms the portal page structure to a format appropriate for the smaller screens of most wireless devices. Only portlets generating OracleAS Wireless XML content can display on a wireless device.

OracleAS Portal developers also have access to a set of page design tools that help in creating portal pages that optimize the wireless experience. With these tools, developers can build a distinct portal structure for their wireless users. The wireless pages and portal pages can share portlet instances, which enables clients to reuse portlets on browser and wireless clients without reconfiguring each portlet.

Refer to Section 4.6, "Configuring Mobile Support in OracleAS Portal" for more information.

- **Oracle Enterprise Manager**. Oracle Enterprise Manager provides the Application Server Control, which can be used for monitoring, diagnostics, and for the configuration of OracleAS Portal specific integration and performance settings. Refer to Chapter 7, "Monitoring and Administering OracleAS Portal" for more information about monitoring OracleAS Portal.

- **Oracle Application Server Forms Services**. Oracle Forms applications combine interactive, graphical interfaces with strong support for data validation. Forms developers can quickly create applications with powerful data manipulation features. OracleAS Forms Services deploys Forms applications to Java clients in a Web environment. OracleAS Forms Services automatically optimizes class downloads, network traffic, and interactions with the Oracle database. OracleAS Forms Services applications are secured by the OracleAS Single Sign-On, and accessed from an OracleAS Portal environment provided by Oracle Application Server.

- **Oracle Application Server Single Sign-On**. OracleAS Single Sign-On authenticates users, who are attempting to gain access to non-public areas of your portal. Refer to Section 6.1.6.1, "Relationship Between OracleAS Portal and OracleAS Single Sign-On" for more information.

- **Oracle Internet Directory**. Oracle Internet Directory is Oracle's highly scalable, LDAP version 3 service and hosts the Oracle common user identity. OracleAS Portal queries the directory to determine a user's privileges and what they are entitled to see and do in the portal. In particular, OracleAS Portal retrieves the group memberships of the user from the directory to determine what they may access and change. Refer to Section 6.1.6.2, "Relationship Between OracleAS Portal and Oracle Internet Directory" for more information.

- **Oracle Delegated Administration**. In addition to querying Oracle Internet Directory for user and group information, OracleAS Portal must provide users with a user interface to add and modify user and group information. To change information in the directory, you use the Oracle Delegated Administration Services user interface. OracleAS Portal provides links to the Oracle Delegated

Administration Services for users with sufficient privileges to add and change users and groups. Refer to Section 6.1.6.4, "Relationship Between OracleAS Portal and DAS" for more information.

Oracle Directory Integration and Provisioning. Oracle Directory Integration Platform notifies OracleAS Portal upon the occurrence of any directory events (for example, user deletions) to which OracleAS Portal subscribes. In essence, the directory integration server informs OracleAS Portal when a change occurs in the directory that requires a change in OracleAS Portal. Refer to Section 6.1.6.3, "Relationship Between OracleAS Portal and Oracle Internet Directory" for more information.

- **Oracle Application Server Metadata Repository**. The OracleAS Metadata Repository maintains information about the available instances in a cluster. This simplifies the process of creating clusters and synchronizing applications and state information across a cluster because all Oracle Application Server instances share the same repository. OracleAS Portal uses a schema within the OracleAS Metadata Repository to store and manage the content and metadata configured to use a portal instance.

You'll find additional information in the white paper titled "OracleAS Portal Architecture Overview" on the Oracle Technology Network, http://otn.oracle.com.

1.2.2 How Do the Pieces Fit Together?

A portal is comprised of groups of pages, each page divided into regions. The regions specify how space on a given page is allotted to that page's items and portlets.

1.2.2.1 How Are Pages Assembled in OracleAS Portal?

Each time a user requests an OracleAS Portal page, the page is dynamically assembled and formatted according to the portlets and layout chosen for that page. Keep in mind that the parts that comprise the page are typically drawn from a variety of sources. For example, the page's layout, look and feel, and user customizations are stored in the database as part of the overall page definition, completely separate from any page content. This information may, in turn, be cached by the middle-tier. (However, if full-page caching is used, pages are not assembled, because they are served directly out of the cache.)

The portlets that appear on the page can be written in XML, PL/SQL or Java. For PL/SQL portlets, the source is an OracleAS Metadata Repository database. This

could be the database where the current instance of OracleAS Portal is installed, or some other OracleAS Metadata Repository database located on a remote server, which is accessed through the Federated Portal Adapter. If written in Java, a Web provider provides the portlet from any location accessible from the network, either Internet or Intranet. For example, you could create a Portal page that displays both the following types of content:

■ Portlet content from an external Web provider.

■ Content from a portlet that resides in the OracleAS Metadata Repository.

Figure 1–5   Portal Page Request Flow
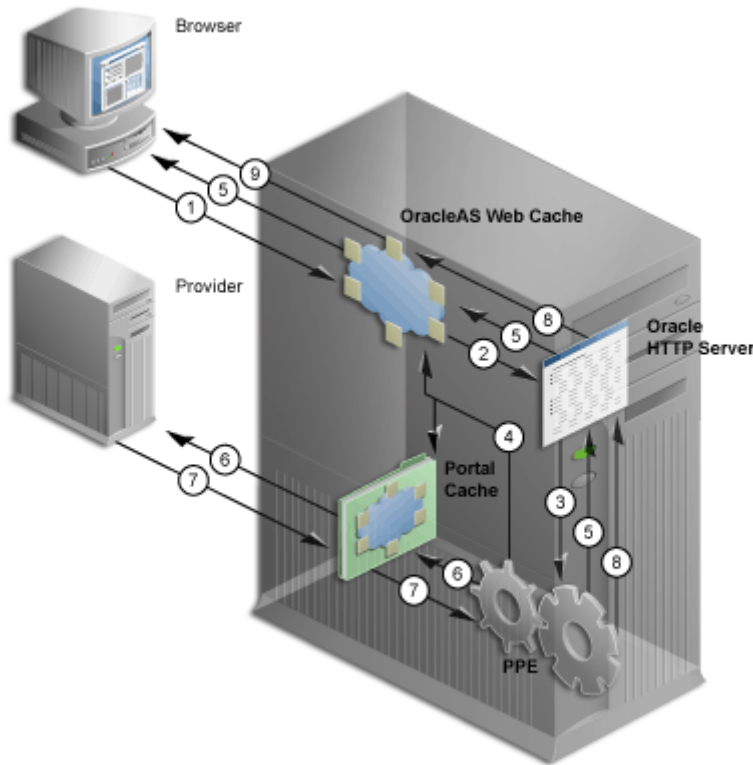
Figure 1–5 shows how a page is assembled. As you can see, when a client requests an OracleAS Portal page, many Oracle Application Server components must respond to various parts of the request:

1. The client browser requests a portal page. OracleAS Web Cache receives this request.

2. OracleAS Web Cache forwards the request to the Oracle HTTP Server (OHS)

3. OHS sends the request to the Parallel Page Engine (PPE) through mod_oc4j.

4. The PPE retrieves the portal page definition. The page definition contains information about the portlets on a page and their layout.

   a. First, it checks if OracleAS Web Cache has a valid, cached copy of the definition.

   b. Next, it checks if the portal cache has a valid, cached copy.

   c. Finally, if no cached copy of the definition exists, then the PPE generates a page definition from data in the portal repository. The portal repository is either in the OracleAS Metadata Repository or in your customer database.

5. The PPE parses the page definition. If a fully cached copy of the page exists, then the page is returned to the client browser through OracleAS Web Cache. If it does not, the PPE builds the page from cached and non-cached data with the remaining steps.

6. For each portlet on the page, the PPE checks if a cached copy of the portlet content exists in the portal cache, and then forwards a request to the appropriate provider, through OracleAS Web Cache (not shown in the image).

7. Each provider either validates the cached portlet or generates content for the portlet. Web providers return this directly to the PPE using HTTP/S. Database (DB) providers return the results to the PPE through OracleAS Web Cache, Oracle HTTP Server, and mod_plsql, using HTTP/S or SOAP.

8. The PPE aggregates the content into a single page. This page is sent to OracleAS Web Cache, and possibly stored in the cache.

9. OracleAS Web Cache returns the final page to the client browser.

1.2.2.2 How Does Communication Flow in OracleAS Portal?

The OracleAS Portal implements a distributed architecture consisting of multiple communication points and protocols. For complex configurations including the introduction of firewalls and proxies, you need to understand the communication points, and how the various components of OracleAS Portal integrate together. Likewise, to allow for the distribution of the various functions across multiple servers, it is necessary to be aware of the network protocols that are used in the internode communication.

The OracleAS Portal architecture consists of three basic tiers: the client browser (pictured at the far left) the middle-tier server (pictured on the bottom left), and the infrastructure server and repositories (pictured on the top left). Although the default installation places all servers and repositories on the same host, it is recommended that you install these functions on separate servers, for increased performance and high availability.

Figure 1–6 illustrates in great detail the communication flow between the various components of OracleAS Portal and Oracle Application Server.

Figure 1–6   Communication Flow and Protocols
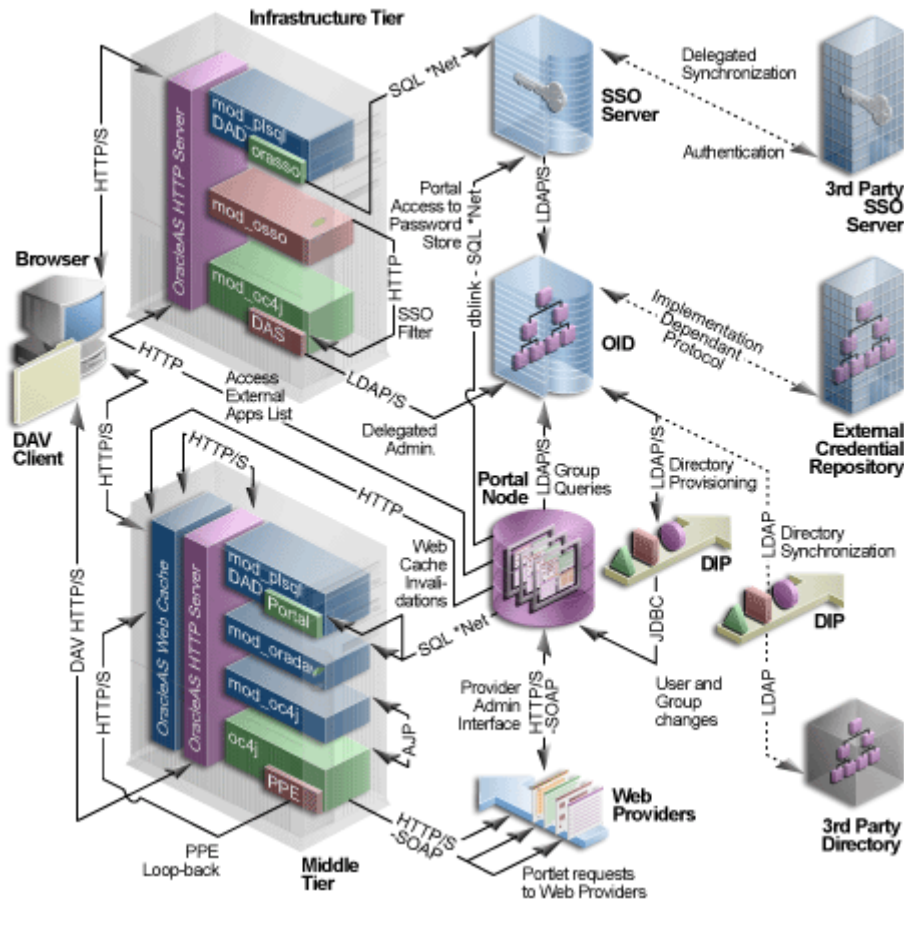
The three tiers and the communication protocols used between them is described next:

- Client

- Infrastructure Tier

- Middle-Tier

Client

- The client sends a request to OracleAS Portal that is part of the middle-tier, using the HTTP/S protocol. The use of firewalls and proxies is supported between the client and the middle-tier.

- If the user needs to be authenticated, the client browser is redirected to the Oracle HTTP Server in the infrastructure tier. This connection is through HTTP/S and supports the implementation of both firewalls and reverse proxies in the network environment.

Infrastructure Tier

The infrastructure tier consists of the Oracle HTTP Server, OracleAS Single Sign-On, Oracle Internet Directory, and OracleAS Metadata Repository.

- If the requested page requires authentication, the user is challenged for a username and password. This function is carried out by the DAD and mod_plsql combination, through a redirection to OracleAS Single Sign-On for authentication. All authentication requests are communicated using the SQL*Net protocol.

- OracleAS Single Sign-On verifies user credentials against the Oracle Internet Directory through LDAP/S. The credentials are compared to those found within the Directory (LDAP compare) and the result returned to OracleAS Single Sign-On. Upon successful authentication, OracleAS Single Sign-On creates a single sign-on cookie. Once the user is authenticated and an appropriate OracleAS Portal session created, the user may access pages and other objects.

- As the Access Control Lists for all Portal objects are held in the OracleAS Metadata Repository, the OracleAS Portal uses an LDAP/S request to communicate with the Oracle Internet Directory to query the appropriate user and group membership information defined in the Directory. When a user first logs in to OracleAS Portal, the group memberships of the user are copied to the portal node and cached on that tier. This process allows for fast lookup of object

privileges. Once the object and page privileges of the user are known, the Parallel Page Engine goes on to generate the page from the appropriate pieces.

- All user provisioning is performed against the Oracle Internet Directory. The interface between the Infrastructure tier's HTTP Server and the LDAP server is through the Delegated Administration Services (DAS) servlet. The DAS interface uses the LDAP/S protocol to communicate with the Oracle Internet Directory.

- The OracleAS Single Sign-On model includes the addition of mod_osso, which allows any URL to be protected within the OracleAS Single Sign-On environment. Calls to the Delegated Administration Services servlet are protected by the mod_osso plug-in, this verifies that the user has been properly authenticated before providing access to the Oracle Internet Directory. In effect, mod_osso filters the URL and forwards the HTTP/S-based request, only if the user has previously been authenticated.

- The Oracle Directory Integration Platform automatically keeps the locally cached information up to date with changes in the Oracle Internet Directory. Just as the Oracle Directory Integration Platform keeps the local cache synchronized with the Oracle Internet Directory, it also keeps the Oracle Internet Directory synchronized with any external repository. The Oracle Directory Integration Platform communicates with the Oracle Internet Directory through LDAP/S.

Middle-Tier

The middle-tier consists of the OracleAS Web Cache, Oracle HTTP Server, Oracle Application Server Containers for J2EE, and other Oracle Application Server components.

> **Note:** OracleAS Web Cache and Oracle HTTP Server can be installed on different hosts to allow scalability and high availability.

- OracleAS Web Cache front ends the middle-tier components and thus optimizes the throughput of OracleAS Portal. When a page request comes from the browser, OracleAS Web Cache evaluates the URL and services the request from the cache if possible. If a requested page is not previously cached, the request is forwarded to its origin server (Oracle HTTP Server in this case) for generation. As a web accelerator, OracleAS Web Cache allows the use of HTTP or HTTPS communication between itself and:

  - The client browser

- The appropriate origin server

- Both the origin server and the client browser

- The Parallel Page Engine (PPE) runs as a servlet within the Oracle Application Server Containers for J2EE. A URL request to the servlet is forwarded through the Oracle HTTP Server's plug-in, mod_oc4j. As a standards-based plug-in, mod_oc4j communicates with Oracle Application Server Containers for J2EE using the Apache Java Protocol (AJP).

- The PPE itself makes requests to both database providers and Web providers through HTTP/S-based communication. The render request to a database provider is through a URL loopback to the Oracle HTTP Server and mod_plsql, while the call to a Web provider is by use of a SOAP-based message protocol over HTTP/S.

- If any Web providers require information from the OracleAS Metadata Repository, they issue the appropriate call through the PDK using a SOAP-based message protocol over HTTP/S.

- The OracleAS Web Cache component uses an Invalidation-based cache methodology. If a requested URL can be serviced from the cache, it is assumed to be correct until the specified URL is invalidated. If a user customizes their OracleAS Portal experience, or if the privileges configured to use the user changes, the OracleAS Portal invalidates the appropriate cached objects within OracleAS Web Cache. To do this, the OracleAS Portal issues a HTTP/S-based request directly from the OracleAS Metadata Repository to the invalidation port of the OracleAS Web Cache.

1.3 Understanding Caching in OracleAS Portal

OracleAS Portal uses three methods to cache Web pages and content:

- **Invalidation-based caching** is performed using OracleAS Web Cache. An item remains in the cache until some event occurs that requires it to be refreshed. For example, a user may update some item, requiring the cache to be updated. In response to the event, the OracleAS Metadata Repository or a Provider sends an invalidation message to OracleAS Web Cache. The next time there is a request for the invalidated item, it is refreshed in the cache. You can set the expiry time for invalidation-based caching. See Section 5.7.3.3, "Setting the Expiry Time for Invalidation-based Caching" for more information.

- **Validation-based caching** is performed using the OracleAS Portal Cache. Before an item in the OracleAS Portal Cache is used, the Parallel Page Engine,

or mod_plsql, contacts the OracleAS Metadata Repository or a Provider to determine if the cached item is still valid.

- **Expiry-based caching** also uses the OracleAS Portal Cache. A retention period for the item specifies how long it is valid in the cache, before a refresh is required. Pages that use expiry-based caching may also be cached in the user's browser.

1.3.1 Understanding OracleAS Web Cache

OracleAS Web Cache is a powerful server acceleration and load balancing solution. Using OracleAS Web Cache is required for running OracleAS Portal. OracleAS Web Cache offers intelligent caching, page assembly, and compression features. OracleAS Web Cache accelerates the delivery of both static and dynamic Web content, and provides load balancing and failover features for Oracle Application Server.

To increase the availability and scalability of medium to large deployments, consider configuring multiple instances of OracleAS Web Cache to run as members of a cache cluster. A cluster is a collection of cooperating OracleAS Web Cache instances that work together to provide a single logical cache. Cache clusters provide failure detection and failover, increasing the availability of your Web site. If an OracleAS Web Cache instance fails, other members of the cache cluster detect the failure and take ownership of the cached content of the failed cluster member. This is achieved because the nodes that receive requests hold the content, after forwarding the request to the owner cache node.

By distributing the Web site's content across multiple OracleAS Web Cache servers, more content can be cached and more client connections can be supported, expanding the capacity of your Web site. You make use of the processing power of more CPUs and, because multiple requests are executed in parallel, you increase the number of requests that are served concurrently

OracleAS Portal functions as a Web Cache origin server to take advantage of the following Web Cache features:

- Caching dynamically generated, user specific page and portlet content
- Fine-grained cache control
- Invalidation-based caching
- Layer 7 load balancing and failover detection
- Performance assurance and surge protection

Portal sites can choose from the following deployment options:

- **Co-located:** Web Cache runs on the same physical server as the Portal middle-tier. This configuration is appropriate for smaller, low-volume sites where the scalability of the middle-tier is not a concern.

- **Dedicated:** Web Cache is deployed on a dedicated server that sits in front of one or more Portal middle-tier servers. Dedicated deployments are usually preferable to co-located deployments, as there is no risk of resource contention with other server processes. OracleAS Web Cache performs well on commodity hardware, so a dedicated deployment does not have to be costly in terms of hardware expenditure.

To avoid a single point of failure in very high-volume sites, two or more nodes running OracleAS Web Cache may be deployed behind a Load Balancing Router (LBR). If you have multiple deployments of OracleAS Portal, each Portal site can have its own Web Cache server. One or more sites can also share a single Web Cache server. Similarly, a provider can share a Web Cache with a Portal site, or a dedicated Web Cache can be deployed in front of the Web server that hosts the provider. Refer to Section 5.7, "Configuring OracleAS Web Cache Caching in OracleAS Portal" for more information about configuring OracleAS Web Cache.

In addition to providing failover, an OracleAS Web Cache cluster also balances the load it forwards to the middle-tier, and can also act as a reverse proxy.

Figure 1–7   Adding OracleAS Web Cache to a Medium to Large Portal Configuration
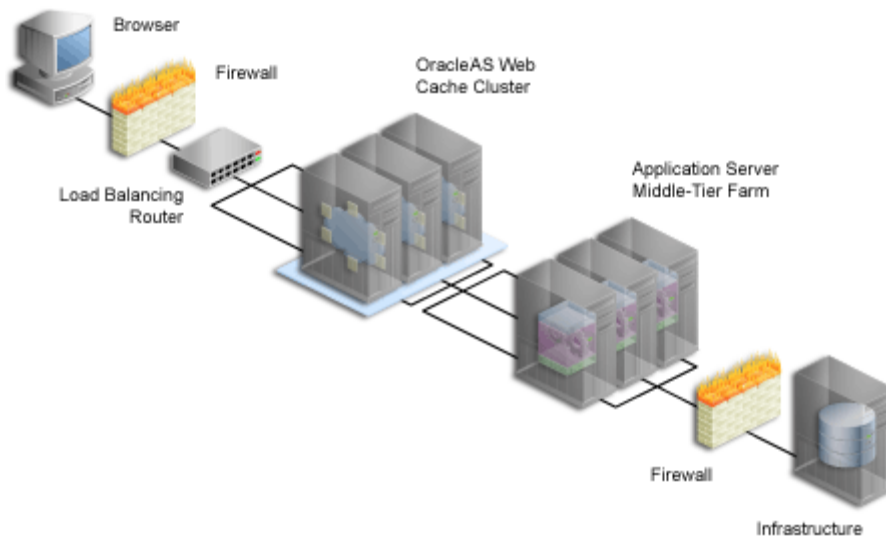
After the initial request to the owner node, the content is cached across all instances. In Figure 1–7, the LBR distributes incoming requests to the three OracleAS Web Cache instances. When the on-demand content is not available on the node receiving the request, the other instances are checked for the cached content, and the content matching the request is returned to the Browser.

To take advantage of OracleAS Web Cache's clustering capability, you must configure each instance as a member of a cache cluster. In this setup, there is no one-to-one relationship between an OracleAS Web Cache instance and a matching middle-tier instance. As shown in Figure 1–7, OracleAS Web Cache 1 provides load balancing between middle-tiers 1, 2, and 3. OracleAS Web Cache 2 and 3 do the same.

> **See Also:**   *Oracle Application Server Web Cache Administrator's Guide*

You'll find additional information about caching and performance on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

1.3.2 Understanding Portal Cache

Portal cache is a file system-based cache for OracleAS Portal pages and portlets. Portal cache supports validation-based caching and expiry-based caching.

Portal cache consists of two kinds of caches:

- **Portal Content Cache**

  The content cache contains user and system level content generated by OracleAS Portal, which includes page metadata, database portlets, Web portlets, documents, style sheets, images, and full-page caches.

- **Portal Session Cache**

  OracleAS Portal uses session cookies to maintain session details for each portal user. This session cookie is encrypted and contains important information like the database username, lightweight username and Globalization Support characteristics of the session. In order for mod_plsql to execute a portal request, it must get the database username from the session cookie. To avoid an expensive decrypt operation with each user request, mod_plsql decrypts the session cookie once and maintains the relevant details in a session cache on the local file system.

Portal content and session cache content resides on the file system, typically under *ORACLE_HOME*/Apache/modplsql/cache, and is configured in the file *ORACLE_HOME*/Apache/modplsql/conf/cache.conf.

It is possible to increase performance by moving the session cache to a more performant file system. This takes the form of a memory-based file system that is commonly available on Windows and UNIX platforms. For more information on increasing the performance of the portal cache, refer to Section 9.6, "Tuning File System Cache to Improve Caching Performance".

In multiple middle-tier configurations, you can setup the portal cache for each middle-tier on a shared file system. This ensures that each middle-tier can share cached content, rather than each drawing from its own independent cache.

For example, one middle-tier might handle a request for an item by caching it in the portal cache. Because you typically use a load balancing router for configurations having multiple middle-tiers, the next request for the item could be handled by a different middle-tier. This middle-tier could access the cached version if the portal caches for each middle-tier are shared on a common file system.

Various parameters for configuring portal cache include:

- Cache location

- Total cache size

- Maximum cacheable file size

- Maximum time a cached file can be in the cache system

- Cleanup of the cache storage

    See Also:

    - *Oracle Application Server 10g Performance Guide*

    - **cache.conf** section in the *Oracle HTTP Server Administrator's Guide*.

1.3.3 Understanding Cache Invalidation in OracleAS Portal

OracleAS Portal makes use of two caching systems - OracleAS Web Cache, and portal cache. OracleAS Web Cache supports invalidation-based caching and expiry-based caching. The portal cache supports validation-based caching and expiry-based caching.

Cache invalidations can be classified into two groups:

- **Hard Invalidations**

    Hard invalidations are queued up over the duration of a single browser request and are then processed when the OracleAS Portal UI action completes. The results will be seen immediately. Most page edits and all portlet customizations are treated as hard invalidations.

- **Soft Invalidations**

    Soft invalidations are queued up over many browser requests and are then processed later by the soft invalidation database job. Security related changes, for example, granting privileges on a page to a user or group, are treated as soft invalidations.

1.3.3.1 Cache Invalidation Resource Requirements

Large numbers of cache invalidations may slow down the system for the following reasons:

- Communication with OracleAS Web Cache

    When either hard or soft invalidations are processed, a TCP/IP connection is established with the OracleAS Web Cache invalidation port to send invalidation messages.

For hard invalidations, all the messages queued in one browser request are sent using a TCP/IP connection to OracleAS Web Cache. For soft invalidations, all the messages processed by the soft invalidation job are sent to OracleAS Web Cache using a TCP/IP connection. OracleAS Web Cache receives these invalidation messages and attempts to invalidate cached data. This load may affect OracleAS Web Cache's ability to respond to requests for data.

- Cache invalidation queue storage

  Both hard and soft invalidation messages are queued into a database table in the OracleAS Metadata Repository. As the queue grows in size, more database resources are required to maintain the queue.

- Cache invalidation queue optimization

  During the processing of hard or soft invalidation messages, queue optimization removes duplicate or unnecessary invalidation messages. For example, if a page group is being invalidated, individual invalidation messages for pages in the page group are unnecessary. If a large number of invalidation messages have been queued up, the optimization process may take a long time.

1.3.3.2 Cache Invalidation and Multiple DADs

OracleAS Portal supports invalidation of data cached in OracleAS Web Cache based on the DAD for a given Portal instance.

Invalidation messages sent to OracleAS Web Cache require the DAD information to be included. This is because data cached in OracleAS Web Cache uses the URL as one of the cache lookup keys and the URLs used to access Portal data contain the DAD name. Therefore, the DAD name must be included explicitly in the invalidation message.

> **Caution:** The use of multiple DADs to access a single Portal instance is not supported.

1.3.4 What's Next?

Now that you have a basic understanding of the Oracle Application Server architecture, how OracleAS Portal fits in, and the working of caching in OracleAS Portal, you're ready to move on to Chapter 2, "Planning Your Portal". By the end of that chapter, you should have a good idea of how you want to configure your installation.

2

Planning Your Portal

This chapter details the task flow involved in planning, installing, configuring, and administering Oracle Application Server Portal. After reading this chapter, you should understand how to plan the hardware and software you need to effectively build a portal.

This chapter contains the following sections:

- What Do I Need to Consider?
- What Do I Need to Do?

> **Note:** If you are unfamiliar with the terms used in this chapter, you may want to review Chapter 1, "Understanding the OracleAS Portal Architecture".

2.1 What Do I Need to Consider?

To develop a plan for configuring your portal, it is critical that you have a firm grasp of the goals you want your system to achieve. Take a look at the following sections to see what's involved in each of these crucial decision points:

- Which Topology Is Right for Me?
- How Much Hardware Do I Need?
- How Can I Maximize Performance?
- How Can I Make My Portal Scale?
- How Can I Make My Portal Highly Available?
- How Can I Secure My Portal?

- How Should I Configure My Hardware and Software?

- Getting the Most Out of Your Configuration

2.1.1 Which Topology Is Right for Me?

Oracle Application Server offers a variety of topology options. The Oracle Application Server recommended topologies range from small general development implementations to very large enterprise-wide implementations.

> **See Also:** Overview of the recommended topologies in *Oracle Application Server 10g Concepts* located in the Oracle Application Server 10*g* Documentation Library.

2.1.2 How Much Hardware Do I Need?

Servers, databases, and resources supporting your Web portal must handle wide variations in user traffic, especially during peak intervals.

As with any Web portal, the server and database capacity with which you'll need to deploy a portal largely depends on the number of user requests that you anticipate. Displaying a single page to a user may require many separate transactions, from verifying whether the user has permission to view the page, to loading the images that appear on the page, to calling a style sheet that contains formatting information for the page.

The upper and lower limits of what you'll need are determined by how you expect your users to use the portal. At a minimum, you'll need enough server capacity to satisfy the average load during a work day, with response times that are acceptable to your user base. If possible, you should strive to satisfy the volume of page requests you anticipate during peak intervals of high user activity. Hardware resources such as CPU, memory, I/O capacity, and network bandwidth are key to reducing response times. You must install OracleAS Portal on a server or group of servers that can handle a large number of transactions, or your users will experience slow response times.

The same is true of your database. If you have many applications competing for the same database resources, your Web portal performance may suffer. You can install multiple instances of OracleAS Portal in the same database, for example, a production instance for developing new pages and portlets, and a separate instance for deploying your finished Web Portal. You must consider whether your database can satisfy requests from both instances in a timely manner.

Adding more servers and database capacity will certainly improve your Web portal's performance, but unless you have unlimited funds at your disposal, you'll need to balance good performance against the costs configured to use each new piece of hardware and software.

> **See Also:** *Oracle Application Server 10g Administrator's Guide*

2.1.3 How Can I Maximize Performance?

Response time is the time between the receipt of a user request and the completion of the response to the request. Your Web portal should respond as quickly as possible with the least amount of software and hardware overhead. Some performance considerations are:

- **Distributing the load**

  If you anticipate a heavy volume of traffic on your Web portal, you can distribute the load across multiple servers, each with its own middle-tier instance. If one server is overloaded with too much traffic, a second server can handle the overflow. For more information, see Section 2.1.8.1, "Load Balancing".

- **Protecting against failures**

  A distributed OracleAS Portal configuration offers improved performance over a single machine configuration because you are making more software and hardware resources available to the Web portal. You can use additional servers and software to provide *failover*, thus ensuring system stability. For more information, see Section 2.1.8.2, "Failover and Redundancy".

- **Implementing cache clusters**

  To increase the availability and scalability of medium to large deployments, you can configure *cache clusters*. Cache clusters provide failure detection and failover, increasing the availability of your Web site. For more information, see Section 1.3, "Understanding Caching in OracleAS Portal".

  > **See Also:** *Oracle Application Server 10g Performance Guide*

2.1.4 How Can I Make My Portal Scale?

Clustering enables you to scale your system beyond the limitations of a single application server instance on a single host. A cluster unifies multiple application server instances spread over multiple hosts to collectively serve a single group of applications. In this way, clustering makes it possible to serve increasing numbers

of concurrent users after the capacity of a single piece of hardware is exhausted. For more information, see Section 2.1.8.3, "Scalability", and Section 1.3, "Understanding Caching in OracleAS Portal".

2.1.5 How Can I Make My Portal Highly Available?

Clustering also enables you to achieve a higher level of system availability than is possible with only a single application server instance. An application running on a single instance of an application server is dependent on the operating system and host on which the server is running. In this case, the host poses as a single point of failure because if the host goes down, the application becomes unavailable.

Application server clusters enable higher availability by providing redundancy and backup and eliminating a single point of failure. Clients access the cluster through a load balancer that can send requests to any application server instance in the cluster. In the case that an application server instance becomes unavailable, the load balancer can continue forwarding requests to the remaining application server instances, as any instance can service any request.

> **See Also:** *Oracle Application Server 10g High Availability Guide*

2.1.6 How Can I Secure My Portal?

Sensitive data should be secured without affecting content that you want to make available to all users.

To support a flexible approach to controlling access to Web content, OracleAS Portal leverages other components of Oracle Application Server and Oracle9*i* Database Server to provide strong protection for your portal. OracleAS Portal interacts with all of the following components to implement its security model:

- Oracle Application Server Single Sign-On

- mod_osso, an Oracle HTTP Server listener module, which implements SSL-based traffic.

- Oracle Application Server Web Cache

- Oracle Internet Directory

- Oracle Delegated Administration Services

- Oracle Directory Integration Platform

For more information, see Chapter 6, "Securing OracleAS Portal".

2.1.7 How Should I Configure My Hardware and Software?

This section discusses how you should configure your hardware and software installations for optimal use of OracleAS Portal and all related Oracle Application Server components. This section explains how you can configure your hardware to set up a small development environment, as well as deploy larger sites serving many users.

2.1.7.1 Using a Single Machine

In the simplest configuration, all of the component pieces (application server and infrastructure) are installed on a single machine as shown in Figure 2–1. In fact, a single database could also reside on the machine, containing separate schemas for OracleAS Portal, Oracle Internet Directory, and OracleAS Single Sign-On.

Figure 2–1   OracleAS Portal Single Machine Configuration
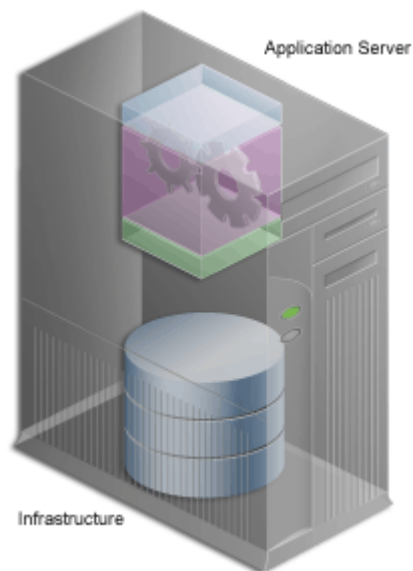
This configuration works nicely in a small development environment in which your developers are using OracleAS Portal's declarative interface to build pages, portlets and applications. It also easily supports a small deployment of the finished Web portal. If you expect to deploy a larger site that delivers more content to more users, you'll need more than a single server or the simple configuration shown in Figure 2–1.

2.1.7.2 Using Multiple Machines

If a single machine configuration does not suit your needs, consider moving the various pieces of the OracleAS Portal architecture to other machines. A rule of thumb when configuring your Web portal is: the larger the site, the more servers you'll require, each server performing more specialized work. Adding extra hardware increases performance. Adding more software instances supports *redundancy*.

Deployment options for configuring larger Web portal sites include:

- Separating the Middle-Tier from the Infrastructure
- Installing the OracleAS Metadata Repository in an Existing Database
- Installing Oracle Identity Management Separately
- Adding Middle-Tier Instances
- Installing OracleAS Web Cache Separately from the Middle-Tier
- Configuring High Availability for the Infrastructure

These tasks should be performed in the order they appear in on this list until you are satisfied that your configuration can handle the demands of your deployed Web portal. If your site must handle only a moderate workload, you could first separate the middle-tier from the database, then think about moving Oracle Identity Management to another server. You probably won't need to perform all of these configuration tasks. But as the site grows, you should expand its underlying configuration by following the sequence shown in this list.

> **Note:** Before you go online with your Web portal, it's a good idea to set up and test a small pilot system. This enables you to gather valuable configuration and tuning information based on real usage patterns, without affecting the users you plan to serve.

2.1.7.2.1 Separating the Middle-Tier from the Infrastructure
The first thing you should consider when configuring a larger system is installing the middle-tier separately, as shown in Figure 2–2.

> **See Also:** *Oracle Application Server 10g Performance Guide*

Figure 2–2 Separating the Application Server Middle-Tier from the Infrastructure
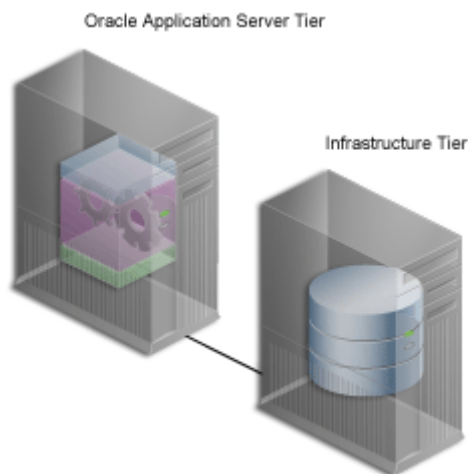
This frees the database and middle-tier from having to compete for hardware resources, such as such as I/O, memory, and disk space. Installing them on separate machines also gives you more flexibility in performance tuning. This is important for sites that plan on storing a lot of content in the Oracle Application Server Metadata Repository. Tuning parameters, such as those for an operating system, are different from those for middle-tier components such as the HTTP server. Setting a performance parameter for one may not provide optimal performance for another.

2.1.7.2.2 Installing the OracleAS Metadata Repository in an Existing Database In $10g$ (9.0.4), the Oracle Universal installer can install a new database seeded with the OracleAS Metadata Repository, or it can use an existing database (customer database). If you want to use an existing database, you need to run the new Oracle Application Server Repository Creation Assistant (REPCA) tool, available on the REPCA CD-ROM, to populate the existing database with the OracleAS Metadata Repository. You do this before running the installer to install other Oracle Application Server components.

2.1.7.2.3 Installing Oracle Identity Management Separately OracleAS Single Sign-On authenticates user credentials against Oracle Internet Directory for OracleAS Portal and other applications, thus requiring users to log on to the Web portal only once with a single username and password, to enable access to multiple accounts and applications.

Once users have logged in to a deployed OracleAS Portal site, they can access any other OracleAS Single Sign-On secured application from portlets within the portal.

As shown in Figure 2–3, Oracle Identity Management is located on a different machine from the OracleAS Metadata Repository. A single instance of Oracle Identity Management can be configured to work with multiple Oracle products, including multiple instances of the OracleAS Portal middle-tier.

Figure 2–3   Oracle Identity Management Installed Separately from the OracleAS Metadata Repository
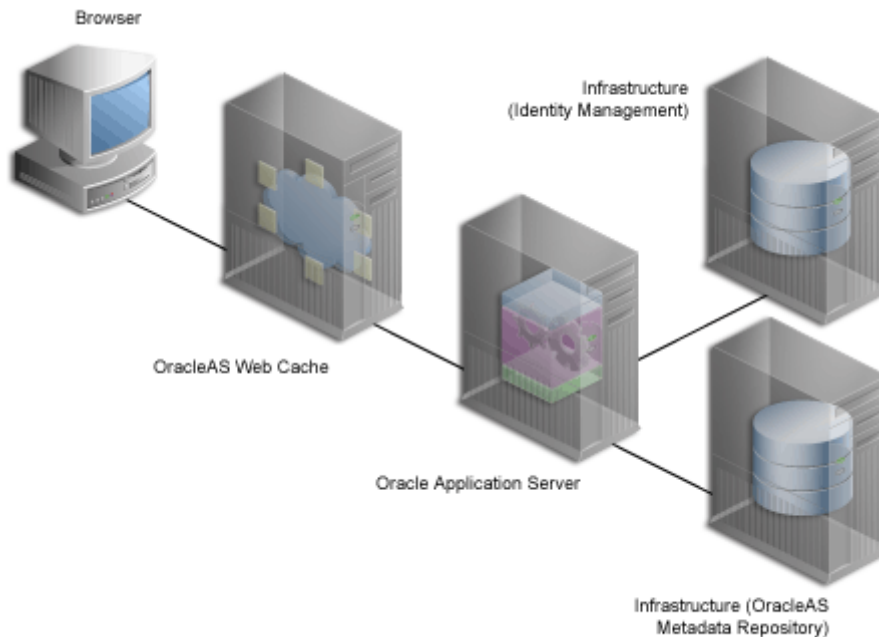
The system shown in Figure 2–3 is an example of a *distributed configuration.* The configuration includes a centralized Oracle Identity Management server that could support multiple middle-tier instances. Moving Oracle Identity Management to its own server gives you the flexibility to tune its performance independently of the database and middle-tier.

In addition, isolating Oracle Identity Management from middle-tier installations ensures greater stability for the entire distributed system. If the machine where a middle-tier is installed fails, the Single Sign-on Server and other middle-tier

instances that rely on it to validate logins are not affected. Additionally, different security policies can be used to manage the various machines in the configuration.

> **See Also:** *Oracle Application Server 10g Installation Guide*

2.1.7.2.4 Adding Middle-Tier Instances You can add redundant middle-tier instances, each with identical configuration settings, to support the largest Web portals. The added middle-tier instances are shown in Figure 2–4. It is a good idea to install each middle-tier instance on its own machine to isolate any hardware failures.

Figure 2–4   Multiple Middle-Tiers
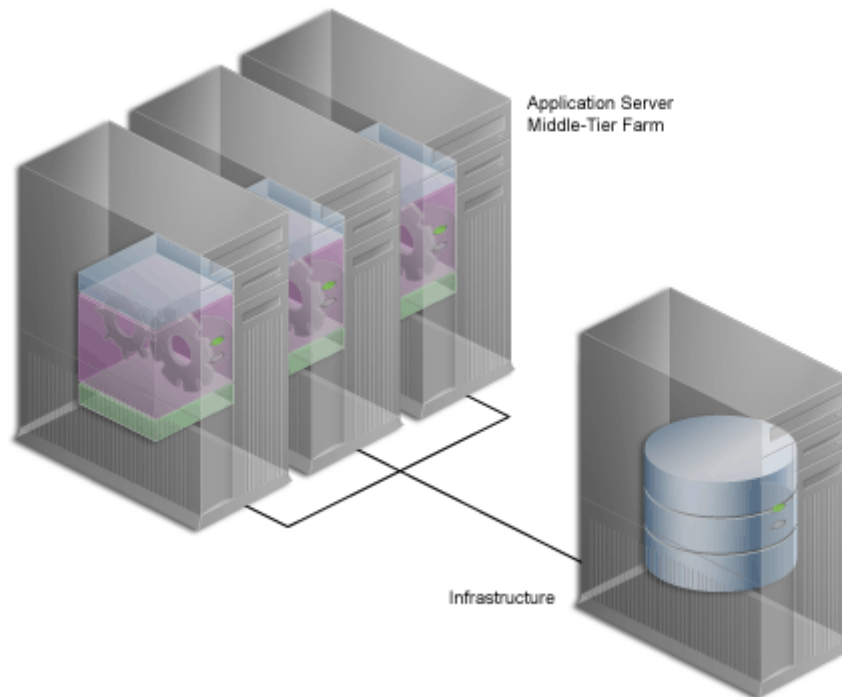
The middle-tier forwards user requests for portal pages to a provider, then assembles the pages with the returned content. As you add more middle-tier instances to your OracleAS Portal configuration, you increase the capacity for user requests and improve the overall performance of your portal. In addition, because the middle-tier performs some processing before forwarding a request, less time is

spent sending and receiving data over the network. Database and network resources are used more efficiently.

2.1.7.2.5 Installing OracleAS Web Cache Separately from the Middle-Tier You can also separate the OracleAS Web Cache server from the middle-tier to enable better caching of data, faster request times, and reduction in the load on the middle-tier. This also improves the performance of OracleAS Portal.

2.1.7.2.6 Configuring High Availability for the Infrastructure In Oracle Application Server 10*g*, all Oracle High Availability (HA) solutions, including Cold Failover Cluster, Data Guard, and RAC, are supported for the Infrastructure.

> **See Also:** *Oracle Application Server 10g High Availability Guide*

2.1.8 Getting the Most Out of Your Configuration

A distributed OracleAS Portal configuration offers improved performance over a single machine configuration because you are making more software and hardware resources available to the Web portal. But there are other benefits. You can use additional servers and software to provide *failover*, thus ensuring system stability. And you can deal with wide fluctuations in the amount of work your Web portal is expected to perform over the course of a day using *load balancing* between multiple servers. Finally, you can add more servers to a distributed configuration to support more users, thus providing *scalability*.

2.1.8.1 Load Balancing

If you anticipate a heavy volume of traffic on your Web portal, you can distribute the load across multiple servers, each with its own middle-tier instance. If one server is overloaded with too much traffic, a second server can handle the overflow.

Oracle Application Server provides its own load balancing capability by pooling server instances to service incoming requests. If one instance does not respond, then the request is forwarded to another instance. This ensures that content and applications are always available to users of your deployed site.

For very large sites, you can add a Load Balancing Router (LBR) to distribute incoming requests to the middle-tier servers, as shown in Figure 2–5. An LBR is a very fast network device that distributes network requests across a large number of servers. It provides users of your Portal with a single published address, instead of them having to send each request to a particular middle-tier server.

Figure 2–5   Multiple Server Configuration Using a Load Balancing Router
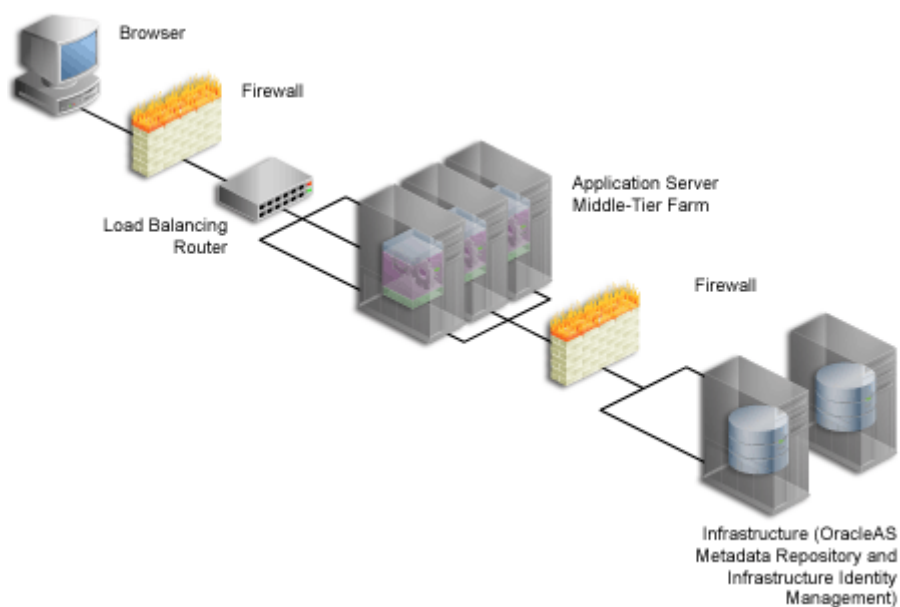
See Section 5.3, "Configuring Multiple Middle-Tiers with a Load Balancing Router" for more information on adding an LBR to distribute incoming requests to middle-tier servers.

As an example, the high traffic personal site, My.Oracle.com (MOC), uses an LBR to sort requests. Because the software logic for distributing loads is contained in the LBR itself rather than installed separately on each individual middle-tier server, an LBR lowers the overall administrative costs of your configuration. MOC is both an intranet and extranet Web site. It provides Oracle customers and employees with a single customizable entry point to all of Oracle's on-line services as well business information from external providers such as NASDAQ and Business Week.

Adding an LBR can also help your configuration deal with load variations. Users may access your site, use its applications, and request content at a much higher frequency during certain peak intervals, for example, between 9 AM and 10 AM when most users log on to begin their work day. During these periods of heavy traffic, the LBR can distribute page requests among the various middle-tier instances to ensure quick response times.

If your peak load occurs on a regular basis, consider a configuration that specifically addresses the need to handle peak load requirements. If your peak load is

infrequent, you may be willing to tolerate slower response times at peak intervals rather than spend additional money on hardware.

Note that the LBR itself can be configured to support failover. The My.Oracle.com configuration in Figure 2–6 could add a second LBR, which would be available in case the primary router fails.

2.1.8.2 Failover and Redundancy

Failover is the ability to switch to a backup when part of your system fails, such as a server or database. When an Oracle9*i* Database Server fails, for example, it restarts using any preserved state information from the backup.

Redundancy is the technique of providing duplicate machines configured identically. The redundant machines provide enough capacity to service requests, and provide backups in case of failures and errors. You implement redundancy by increasing the number of machines in your configuration. One server is typically active while the other monitors the first server's activity, ready to take over if it fails.

As shown in Figure 2–6, My Oracle.com provides for failover using an additional middle-tier server that can take over if any of the other servers encounter problems that cause them to fail.

> **Note:** The components depicted in Figure 2–6 represent only one of many possible configurations. Oracle does not expressly recommend or endorse these specific vendors, or components, or both.

Figure 2–6   My Oracle.com Middle-Tier Configuration
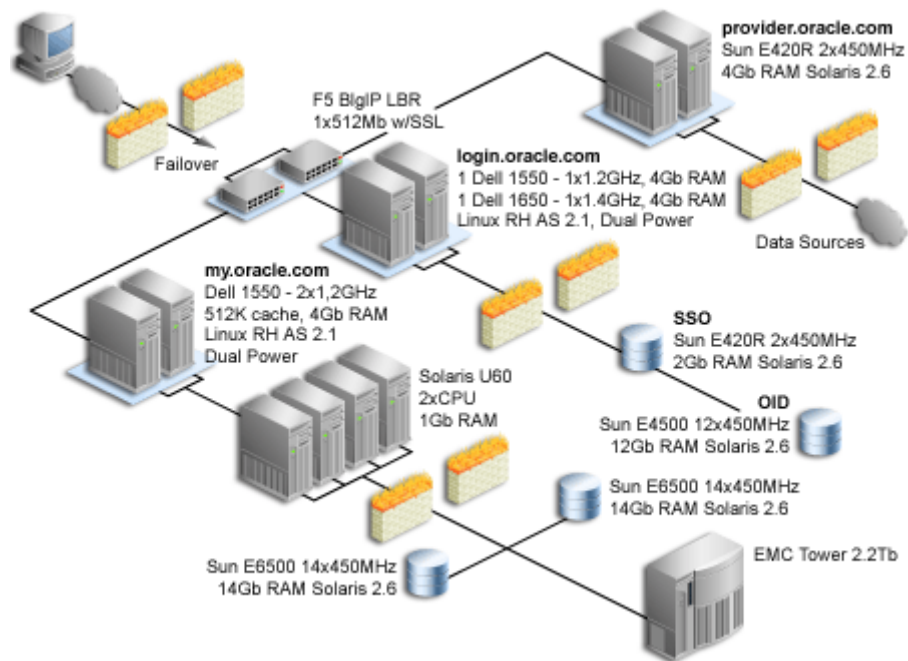
To set up redundant middle-tier instances, you configure the original and each redundant instance with identical server name and server port entries, for example, *my.oracle.com* and port *5000*.

One alternative to redundancy is to setup failover by using any excess capacity that you have in your overall configuration. For example, you might have four middle-tier servers, each running at 75% capacity. If one server fails, the other three can take over the workload of the fourth (25% X 3 = 75%, which is the capacity of the failing server).

2.1.8.3 Scalability

Scalability is the ability of a Web portal to handle more requests as the number of users and the volume of content increases over time. As the portal handles more traffic, users should not notice any change in performance, as measured by response intervals and frequency of errors. If scalability is your goal, you need a flexible configuration that will enable you to add database capacity and servers incrementally as needed without adversely affecting the rest of your configuration.

When My.Oracle.com (MOC) was set up, for example, it was initially expected to serve approximately 40,000 Oracle employees. The user base is anticipated to expand eventually to a million and a half, most of them users of the Oracle Technology Network (OTN), each automatically provided with an MOC account.

2.2 What Do I Need to Do?

This section describes the task flow involved in planning, installing, configuring, and administering OracleAS Portal.

Successfully deploying OracleAS Portal consists of the following steps:

1. Planning your portal

2. Upgrading your current portal configuration (if necessary)

3. Verifying pre-installation requirements

4. Installing Oracle Application Server

5. Performing basic configuration and administration

6. Performing advanced configuration

7. Securing OracleAS Portal

8. Monitoring OracleAS Portal

9. Troubleshooting OracleAS Portal

The following sections provide high-level descriptions of each step and point to more detailed information in various locations, including this configuration guide, other Oracle Application Server 10*g* Documentation Library books, technical white papers, and on Portal Center, http://portalcenter.oracle.com.

2.2.1 Planning Your Portal

If you are new to OracleAS Portal, you may benefit from reading Chapter 1, "Understanding the OracleAS Portal Architecture" to understand how OracleAS Portal fits into the Oracle Application Server architecture.

These white papers may also be helpful:

- OracleAS Portal Architecture Overview

- Planning Your OracleAS Portal Configuration

You can find these white papers on the Oracle Technology Network at http://otn.oracle.com.

Upgrading OracleAS Portal

More On Portal Center

You'll find the latest information on upgrading from an earlier release of OracleAS Portal on Portal Center, http://portalcenter.oracle.com/upgrades/. On the Upgrades page, you'll find:

- Instructions for downloading the upgrade scripts

- Online documentation in the form of the OracleAS Portal - Upgrading the Database Repository Release 3.0.9 to 9.0.4, and OracleAS Portal - Upgrading the Database Repository Release 9.0.2 to 9.0.4 guides.

- Links to related documentation, such as Database Migration guides and the Oracle Application Server Migration guides.

2.2.2 Verifying Pre-Installation Requirements

To ensure a smooth installation, you must verify that you have fulfilled all prerequisites and have performed all pre-installation steps. The *Oracle Application Server 10g Installation Guide* contains the general Oracle Application Server requirements, while Chapter 3, "Installing OracleAS Portal" discusses the portal-specific steps.

2.2.3 Installing OracleAS Portal

The *Oracle Application Server 10g Installation Guide* contains the steps for installing the Oracle Application Server middle-tier and infrastructure required to run OracleAS Portal. You'll find additional information in Chapter 3, "Installing OracleAS Portal".

2.2.4 Performing Post-Installation Configuration

Chapter 4, "Performing Basic Configuration and Administration" contains information about all the post-configuration tasks that can be performed by the OracleAS Portal administrator.

More On Portal Center

You'll find additional information in the paper "Strategies for Administering Privileges in OracleAS Portal," on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

2.2.5 Performing Advanced Configuration

Part III, "Advanced Configuration Topics" is targeted at the Oracle Application Server administrator. Chapter 5, "Performing Advanced Configuration" provides instructions on how to perform more advanced OracleAS Portal configuration and integration configuration, including virtual hosts, Load balancing routers, proxy server, OracleAS Web Cache and OracleAS Single Sign-On configuration. Other chapters in the part deal with setting up Search, Import and Export, Syndication, and more.

2.2.6 Securing OracleAS Portal

Chapter 6, "Securing OracleAS Portal" contains in-depth information on how to configure the security features in OracleAS Portal.

2.2.7 Monitoring OracleAS Portal

OracleAS Portal can be monitored through the Oracle Enterprise Manager Application Server Control. Additionally there are performance reports that can be generated to monitor performance.

See Chapter 7, "Monitoring and Administering OracleAS Portal" for more information on monitoring OracleAS Portal.

You'll find additional information on the following topics, on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

- The paper "*Tuning Oracle Net Services to optimize mod_plsql Database access times*"

- The paper "*Object Access Reporting from the Performance Logs in Oracle9iAS Portal*".

2.2.8 Troubleshooting OracleAS Portal

Chapter 13, "Troubleshooting OracleAS Portal" discusses various issues and ways for resolving and diagnosing errors.

Refer to the book *Oracle Application Server Portal Error Messages Guide*, for more information on error messages.

Part II

Installation and Basic Configuration

Part two contains the following chapters:

3

Installing OracleAS Portal

This chapter provides a brief overview of the installation process and describes a few things you need to know about installing OracleAS Portal that are not covered in the *Oracle Application Server 10g Installation Guide*. This chapter contains the following sections:

- How Does the Installation Process Work?
- What Is Installed By Default?
- Configuring OracleAS Portal During and After Installation

More On
Portal Center

If you are planning to upgrade OracleAS Portal from a previous release, you'll need to refer to the Upgrade documentation on Portal Center, http://portalcenter.oracle.com/upgrades/.

3.1 How Does the Installation Process Work?

To install OracleAS Portal:

1. Read the Release Notes and the Release Notes Addendum for any late breaking changes that might affect your installation.

2. Check that you meet the Oracle Application Server requirements.

3. Choose a *topology*. For the recommended topologies that help illustrate the flexibility of Oracle Application Server, see the subsequent table. For complete instructions on how to install and configure the infrastructure and the middle-tier in different topologies, refer to the *Oracle Application Server 10g Installation Guide*.

Topology	Environment	Requires Infrastructure?	Portal Included?
Java Developer Topology	Development	No	No
Portal and Wireless Developer Topology	Development	Yes	Yes
Forms, Reports, and Discoverer Developer Topology	Development	Yes	No
Integration Architect and Process Modeler Topology	Development	Yes	No
Departmental Topology	Deployment	Yes	Yes
Enterprise Data Center Topology: J2EE Applications	Deployment	Yes	No
Enterprise Data Center Topology: Portal, Wireless, and Business Intelligence Applications	Deployment	Yes	Yes
Development Life Cycle Support Topology	Development and Deployment	N/a	No
OracleAS Certificate Authority Topology	Deployment	Yes	No

4. Install the Oracle Application Server Infrastructure using the Oracle Installer. You can follow the steps to install an infrastructure with a new Oracle9*i* Database Server, which you'll need to house the Oracle Application Server Metadata Repository, or you can install the OracleAS Metadata Repository into an existing database. The procedure also installs a new Oracle Internet Directory.

> **Note:** If you plan to install the OracleAS Metadata Repository, including the OracleAS Portal schema, in an existing database, you need to run the Oracle Application Server Repository Creation Assistant tool (RepCA), available on the "Oracle Application Server Repository Creation Assistant" CD-ROM, to populate the existing database with the OracleAS Metadata Repository. You must do this before running the installer to install other Oracle Application Server components. Refer to the *Oracle Application Server 10g Installation Guide* for more information.

The *Oracle Application Server 10g Installation Guide* tells you how to start the Installer, what command-line options are available, and provides step-by-step procedures. If you need multiple Oracle Application Server instances, you must run the Installer multiple times, once for each Oracle Application Server instance.

5. Install the Portal *middle-tier*, also using the Oracle Installer. You deploy and run your applications, including OracleAS Portal, on Oracle Application Server middle-tiers. Instructions for installing the middle-tier are also in the *Oracle Application Server 10g Installation Guide*.

> **Note:** The OracleAS Portal online help system, which uses *Oracle Help for the Web*, relies on certain fonts to render the Online Help User Interface in different languages. To get the correct fonts installed, you must select all the languages in which you want to render the online help at installation time. To do this, click the **Product Languages** button, and select your languages on the **Select a Product to Install** screen, during the installation.
>
> Additionally, you must make sure that the languages that are installed on the Application Server middle-tier correspond with the languages that are installed on the Application Server Infrastructure, to avoid problems with the Set Language request issued to the OracleAS Single Sign-On server. See Section B.2.3, "LANGUAGE" for more details.
>
> Installing all languages increases the time required for the middle-tier installation.

Figure 3–1   Installation Overview
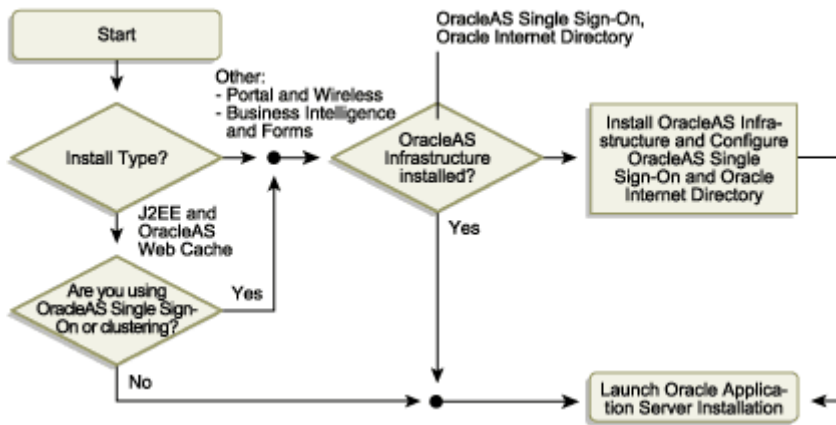
6. After you install Oracle Application Server, go to the Oracle Application Server page at `http://hostname.domain:7777` (7777 is the default port number used during installation). Here you can view the documentation library, take the Quick Tour, and run some demos. To run the demos, click the **Demonstrations** tab then select **Portal and Wireless** from the Navigation panel.

7. Access OracleAS Portal by entering the following URL in your browser:

   `http://<hostname>:<portnumber>/pls/<dad>`

   The **Portal Builder** page is displayed as shown in Figure 3–2.

Figure 3–2   Portal Builder Page
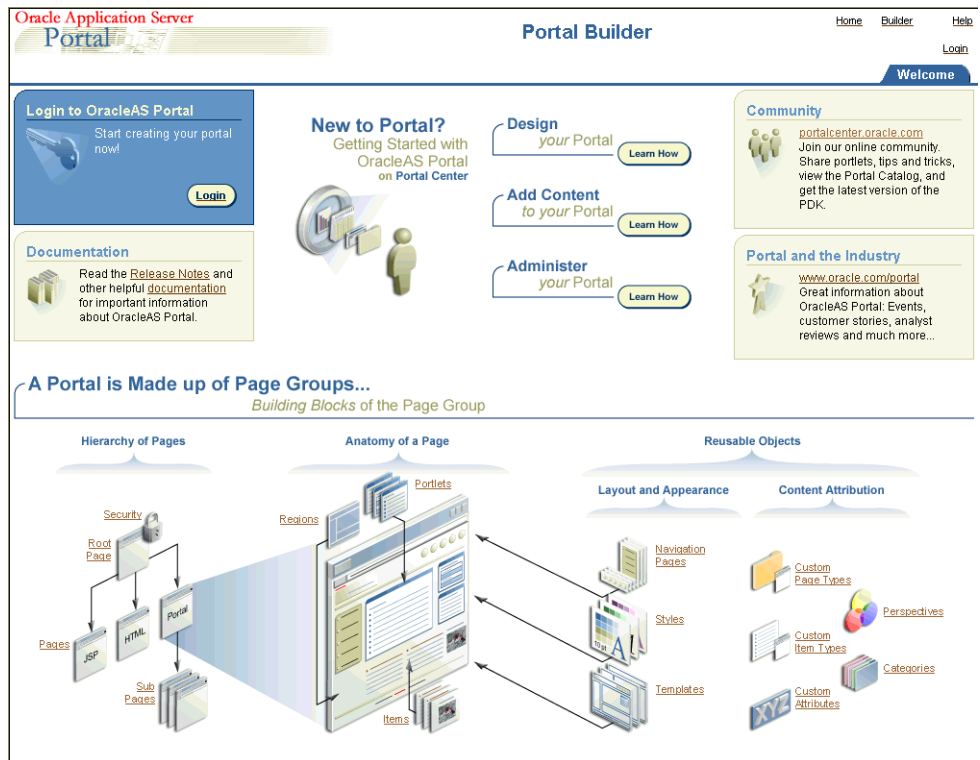
The following table explains the components that make up the URL used to access OracleAS Portal.

Table 3–1   Portal URL Descriptions

Parameter	Description
`hostname`	Defines the machine on which you installed OracleAS Portal.
	Enter both the hostname and the fully qualified domain name. For example, enter host.domain.com.
	This name must also match the `ServerName` parameter in the configuration file, `httpd.conf`, located in:
	`ORACLE_HOME`/`Apache/Apache/conf`
`port number`	Defines the port number you specified earlier to access OracleAS Portal.

Table 3–1   (Cont.) Portal URL Descriptions

Parameter	Description
`pls`	Defines the virtual path and indicates that the request is for a PL/SQL procedure which alerts the Oracle HTTP Server to reroute the request to mod_plsql.
`dad`	Defines the Database Access Descriptor (DAD) you specified earlier for your OracleAS Portal installation. The DAD contains information on how to connect to the database. By default the DAD is 'portal'.

8. Click the **Login** link, located in the top right corner:

Figure 3–3   Login Screen

9. Login as the *portal* user, using the ias_admin password:

Figure 3–4   Sign In Screen
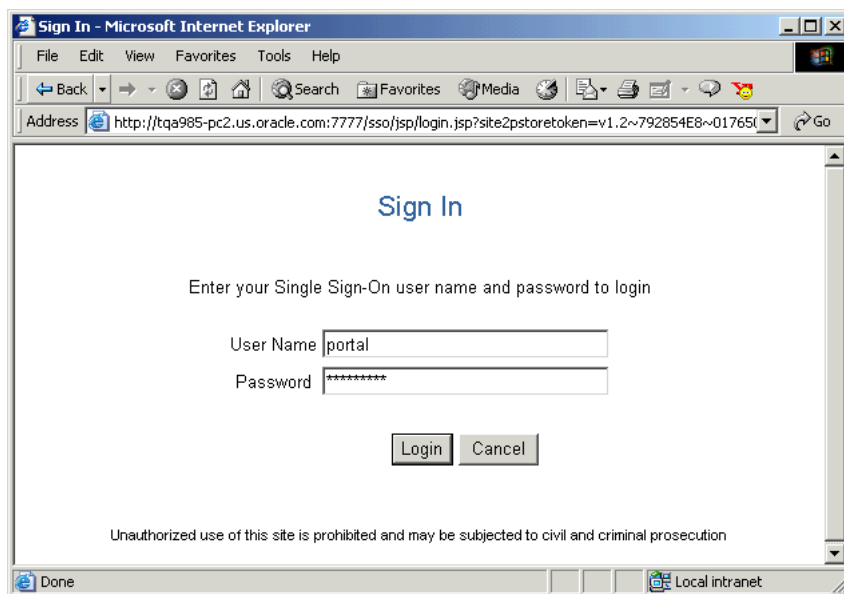
10. After you have verified that OracleAS Portal is up and running, by logging in, you can run the OracleAS Portal Diagnostic Assistant (PDA) and view the generated reports for additional verification. Refer to Section 13.5, "Using the OracleAS Portal Diagnostics Assistant" for instructions on how to run the PDA.

3.2 What Is Installed By Default?

When you install OracleAS Portal, some default database schemas and user accounts are also installed. It's a good idea to take the time to familiarize yourself with these defaults.

- The default database schemas are described in "OracleAS Portal Default Schemas" in Chapter 6, "Securing OracleAS Portal".

    See Also: *Oracle Application Server Single Sign-On Administrator's Guide*

- The OracleAS Portal default user accounts and groups are described in Section 6.1.2.1, "OracleAS Portal Default, Seeded User Accounts" and Section 6.1.2.2, "OracleAS Portal Default, Seeded Groups".

The installation of the OracleAS Metadata Repository component of Oracle Application Server 10*g* creates a new database and populates it with a collection of schemas used by Oracle Application Server components, such as the OracleAS Portal metadata schema.

Out of the box, the initialization parameters for this new database are suitable for a very small OracleAS Portal configuration with few users. If you plan to use OracleAS Portal, it is recommended that you modify the initialization parameters for the database based on the requirements for installing the OracleAS Metadata Repository in an existing database, using the settings specified in the *Oracle Application Server 10g Installation Guide*. As you make changes in your configuration, you may need to further tune the initialization parameters based on the size of your configuration, and the number of simultaneous users of OracleAS Portal. The init.ora file can be found in the database's ORACLE_HOME. If init.ora is modified, the database must be restarted for this change to take effect.

3.3 Configuring OracleAS Portal During and After Installation

During a middle-tier installation, that includes OracleAS Portal, you can specify if you want to configure, and automatically start OracleAS Portal at the end of the installation. If you select that option, Oracle Universal Installer (OUI) will go configure OracleAS Portal in two phases:

1. OracleAS Portal middle-tier deployment

2. OracleAS Portal schema configuration in the OracleAS Metadata Repository

If you choose not to configure OracleAS Portal, and want to do this later, you need to:

- Use Oracle Enterprise Manager Application Server Control to deploy OracleAS Portal on the middle-tier. Refer to Section 7.1.2, "Using Application Server Control to Configure OracleAS Portal" for more information.

- Use the Portal Dependency Settings file and tool, to perform the OracleAS Portal schema configuration in the OracleAS Metadata Repository. You need to perform this step, because OracleAS Metadata Repository configuration is not performed by default, when you use Application Server Control, so that existing configuration entries in the OracleAS Metadata Repository are not automatically overwritten. Refer to Appendix A, "Using the Portal Dependency Settings File" for more information.

You can update the OracleAS Metadata Repository with any changes made to the Portal Dependency Settings file iasconfig.xml, related to middle-tier component properties, such as OracleAS Web Cache, and Oracle Enterprise Manager.

Portal does not support serving two middle-tiers from a single repository, unless it is front-ended by a load balancing router (LBR). Refer to Section 5.3, "Configuring Multiple Middle-Tiers with a Load Balancing Router" for instructions on how to set up OracleAS Portal with an LBR.

If you want to add additional middle-tiers to a farm that is already using Infrastructure Services, you do not want to overwrite the existing configuration entries during the deployment. In this case, you would install the additional middle-tier, without configuring OracleAS Portal, and configure OracleAS Portal, using Application Server Control, and the Portal Dependency Settings file and tool.

> **Note:** By default, iasconfig.xml resides in *ORACLE_HOME*/portal/conf. If the Portal Dependency Settings file is accessible over a network file system, you can share the file across multiple hosts, avoiding the need to manually replicate it every time the file is modified. If the installation is running on an operating system that supports symbolic links, it is recommended that you use this mechanism to reference a shared file. If, however, the Portal Dependency Settings file is not accessible over the network, you must ensure that the file is kept up-to-date with changes to your site topology. Refer to Section A.1.2, "Updating the Portal Dependency Settings File" for more information.

To use Application Server Control to deploy OracleAS Portal on the middle-tier, follow the following steps outlined in Section 7.1.2, "Using Application Server Control to Configure OracleAS Portal".

At this point, your OracleAS Portal middle-tier components are deployed, and configured. The DAD has been created, and the Portal Dependency Settings file iasconfig.xml has been updated.

To update the OracleAS Metadata Repository with any changes made to the Portal Dependency Settings file iasconfig.xml, run the script ptlconfig, located in the directory *ORACLE_HOME*/portal/conf, as follows:

```
ptlconfig -all -dad portal
```

Additional middle-tiers are often added to production sites, to improve scalability. The two-phased process described in the preceding text allows the flexibility of adding additional middle-tiers, without restarting the site.

4

Performing Basic Configuration and Administration

This chapter assumes that OracleAS Portal has been installed as part of the Oracle Application Server and addresses the basic tasks that the portal administrator can perform after installation is complete.

This chapter contains the following sections:

- Getting Started with OracleAS Portal Administration

- Finding Out Information About Portal

- Performing Basic Page Administration

- Configuring Self-Registration

- Performing Basic Portal Administration

- Configuring Mobile Support in OracleAS Portal

- Managing Users, Groups, and Passwords

- Configuring Browser Settings

- Configuring Language Support

- Configuring OracleAS Portal for WebDAV

4.1 Getting Started with OracleAS Portal Administration

Basic OracleAS Portal configuration can be performed on the **Administer** tab available from OracleAS Portal. Additionally, there are other administrative tools available to configure OracleAS Portal, and its related components.

This section will introduce you to the various different administrative tools:

- Using the OracleAS Portal Administer Tab

- Using Additional Administrative Tools

4.1.1 Using the OracleAS Portal Administer Tab

The OracleAS Portal framework provides administrative services, such as access to monitoring and configuration tools, single sign-on, directory integration, caching, and security. A lot of the features needed to manage users and groups, to set up security and search features, and to administer the portal and database are incorporated into a series of dialog boxes accessed through portlets on a Portal page.

After you have installed OracleAS Portal, you need to log in as an administrator, to perform various administrative functions.

After you have logged in to OracleAS Portal, the **Portal Builder** page is displayed to you, as shown in Figure 4–1:

Figure 4–1   The Portal Builder Page

Click the **Administer** tab to view all the sub-tabs and portlets that help you administer the Portal. The **Administer** tab is shown in .

Figure 4–2   The Administer Tab on the Portal Builder Page

You will see the following sub-tabs in the **Administer** tab screen:

- Portal - This sub-tab enables you to create users and groups, administer the SSO server, and administer other services including Oracle Internet Directory, Oracle Ultra Search, Oracle Application Server Web Cache, proxy settings, and so on.

- Portlets - This sub-tab enables you to view the Portlet repository and its refresh log file contents, and register remote providers and provider groups.

- Database - This sub-tab enables you to create and edit database schemas, create and edit database roles, and monitor database information like database parameters, memory consumption, and database storage details.

Portal

This sub-tab under the **Administer** tab in the **Builder** page contains the portlets shown in Table 4–1. This sub-tab is displayed by default when you click the **Administer** tab.

Table 4–1   Portlets in the Portal Sub-Tab

Portlet Name	Enables You to
Services	■ Specify default home page, default style, and so on.
	■ Administer users and groups in the Oracle Internet Directory or configure the directory settings.
	■ Administer log registry.
	■ Establish basic and advanced search features.
	■ Specify Proxy Server settings.
	■ Administer and monitor the performance of OracleAS Portal and its dependent components such as the Oracle HTTP Server, mod_plsql Services, Parallel Page Engine Services, OracleAS Web Cache, OracleAS Metadata Repository, OracleAS Syndication Services, Oracle Ultra Search, and providers using the Oracle Enterprise Manager Application Server Control.
	See Chapter 7, "Monitoring and Administering OracleAS Portal" for more information on administering the log registry and monitoring Portal performance.
SSO Server Administration	■ Edit SSO Server configuration.
	■ Create or edit configuration information for partner applications.
	■ Create or edit configuration information for external applications.
	See Chapter 6, "Securing OracleAS Portal" for more information.
Export/Import Transport Set	■ Export a transport set.
	■ Import a transport set.
	■ Browse the status of, download scripts for, reuse, or delete transport sets.
	See Chapter 10, "Exporting and Importing Content" for more information.

Table 4–1   (Cont.) Portlets in the Portal Sub-Tab

Portlet Name	Enables You to
User	■ Create new users and specify account information. ■ Edit or delete users.
Portal User Profile	■ Establish the user's preferences and global privilege information in the Portal.
Group	■ Create groups, assign users to them, and designate group administrators. ■ Edit or delete groups.
Portal Group Profile	■ Establish the group's preferences and privilege information in the Portal.

Portlets

This sub-tab under the **Administer** tab in the **Builder** page contains the portlets shown in Table 4–2.

Table 4–2   Portlets in the Portlets Sub-Tab

Portlet Name	Enables You To
Portlet Repository	■ View and refresh all local and remote portlets. ■ Refresh information about all the portlets in the repository. ■ View Portlet Repository refresh log.
Remote Providers	■ Add a provider to the portlet repository. ■ Change configuration and access information about a provider.
Remote Provider Group	■ Register multiple providers with a single URL. ■ Edit a Provider Group registration.

Database

This sub-tab under the **Administer** tab in the **Builder** page contains the portlets shown in Table 4–3.

Table 4–3   Portlets in the Database Sub-Tab

Portlet Name	Enables You To
Schemas	■ Create new database schemas, or edit existing schemas.

Table 4–3 (Cont.) Portlets in the Database Sub-Tab

Portlet Name	Enables You To
Roles	■ Create new database roles, or edit existing roles.
Database Information	■ Monitor and view various database related information and parameters.
Database Memory Consumption, Transactions and Locks	■ Monitor database jobs. ■ View reports and charts of memory consumption and transactions. ■ Monitor session and locks. ■ Terminate undesirable user sessions.
Database Storage	■ Monitor and view various database storage related information.

4.1.2 Using Additional Administrative Tools

For some administrative tasks that cannot be performed through the OracleAS Portal **Administer** tab, you may need to use one of the following tools:

- Oracle Enterprise Manager Application Server Control

- Portal Dependency Settings File and Tool

- OracleAS Portal Configuration Assistant

- Portal Installation and Configuration Scripts

4.1.2.1 Oracle Enterprise Manager Application Server Control

The Oracle Enterprise Manager Application Server Control is included when you install Oracle Application Server. From OracleAS Portal's perspective, consider this to be the administration console for the Oracle Application Server. The Application Server Control enables you to perform the following administration and configuration operations:

- Enable and disable components

- Administer clusters

- Start and stop services

- View logs and ports

- Perform real-time monitoring

- Modify the Infrastructure services used by an Oracle Application Server middle-tier.

These functions of the Application Server Control are described in more detail in Chapter 7, "Monitoring and Administering OracleAS Portal".

4.1.2.2 Portal Dependency Settings File and Tool

OracleAS Portal is dependent on the components: Oracle Application Server Web Cache and Oracle Internet Directory. It may be necessary to fine tune, or configure these components after Oracle Application Server is installed.

To simplify configuration changes, OracleAS Portal introduces the *Portal Dependency Settings File*. This file stores configuration data from all the dependent components in a central place and the content of the file is updated when there are configuration changes.

You can use the Portal Dependency Settings file to:

- Check settings used by an OracleAS Portal instance.

- Update settings in the Oracle Application Server Metadata Repository.

The Portal Dependency Settings file is described in more detail in Appendix A, "Using the Portal Dependency Settings File".

4.1.2.3 OracleAS Portal Configuration Assistant

The OracleAS Portal Configuration Assistant (OPCA) is a Java-based configuration tool for installing and configuring the OracleAS Portal schema in the OracleAS Metadata Repository. OPCA is described in more detail in Appendix B, "Using the OracleAS Portal Configuration Assistant Command Line Utility".

> **Note:** Most actions that used to be performed using the MIDTIER mode of OPCA (`ptlasst`) in earlier versions of OracleAS Portal, are now done using the Portal Dependency Settings file and tool.

4.1.2.4 Portal Installation and Configuration Scripts

There are also various scripts, copied to your `ORACLE_HOME` during the installation of OracleAS Portal, These scripts may be needed to perform administrative actions, and are described in more detail in Appendix C, "Using OracleAS Portal Installation and Configuration Scripts".

4.2 Finding Out Information About Portal

This section covers the following topics:

- Accessing OracleAS Portal in Your Browser

- Finding Your OracleAS Portal Version Number

4.2.1 Accessing OracleAS Portal in Your Browser

After OracleAS Portal is installed, access it by entering the following URL in your browser:

```
http://<hostname>:<portnumber>/pls/<dad>
```

For an explanation of the URL components, see Table 3–1, " Portal URL Descriptions".

> **See Also:**
>
> - Section 3.2, "What Is Installed By Default?"
>
> - *Oracle Application Server 10g mod_plsql User's Guide*

4.2.2 Finding Your OracleAS Portal Version Number

To find your Portal version number:

1. In the Portal Builder, click the **Administer** tab.

2. Click the **Portal** sub-tab.

3. In the **Services** portlet, click the **Global Settings** link.

   The version number for your OracleAS Portal is shown at the bottom of the page.

4.3 Performing Basic Page Administration

This section covers the following topics:

- Setting a Default Home Page

- Setting the System Default Style

- Creating Personal Pages

- Setting the Total Space Allocated for Uploaded Files

- Setting the Maximum File Size for Uploaded Files

- Changing the Page Group Quota

- Specifying an Error Message Page

- Setting the Page Users See When They Log Out

- Removing the Context-Sensitive Help Link

4.3.1 Setting a Default Home Page

The home page is the first page that is displayed to a user after logging in to Oracle Portal. Here's how the logic works:

- If the user has specified a personal home page, that page is displayed when the user logs on.

- If the user has not selected a personal home page, but the portal administrator has set one for him or her, the default home page specified for that user is displayed.

- If the user has not selected a personal home page, but belongs to a default group, the default home page specified for that group is displayed.

- If there is no default home page for the user's default group, the system default home page is displayed.

If mobile support is enabled, you can specify a default mobile home page to display when a user accesses the portal from a mobile device.

> **Note:** You must be the portal administrator to define a default home page for the system, a group, or a user.

4.3.1.1 Setting the System Default Home Page

If there is no default home page for the user's default group, the system default home page is displayed.

To set the system default home page:

1. In the **Services** portlet, click **Global Settings**.

By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Next to the **Default Home Page** field, click the **Browse Pages** icon to see a list of pages from which to choose.

    > **Note:** You cannot enter a value in this field; you must select one from the pop-up list.

3. Click **Return Object** next to the page you want to make the system default home page.

4. Click **OK**.

    > **Note:** To check that you set the system default home page correctly, log out of the portal and log back in again. When you log back in, you should be taken the page that you specified as the system default home page.

4.3.1.2 Setting a Group's Default Home Page

If the user has not selected a personal home page, but belongs to a default group, the default home page specified for that group is displayed.

To set a group's default home page:

1. In the **Portal Group Profile** portlet, in the **Name** field, enter the name of the group for which you want to assign a default home page.

    By default, the **Portal Group Profile** portlet is on the **Administer** tab of the **Builder** page.

    > **Note:** If you are not sure of the group name, click the **Browse Groups** icon and select from the list provided.

2. Click **Edit**.

3. Next to the **Default Home Page** field, click the **Browse Pages** icon to see a list of pages from which to choose.

    > **Note:** You cannot enter a value in this field; you must select one from the pop-up list.

4. Click **Return Object** next to the page you want to make the default home page for this group.

5. Click **OK**.

> **Note:** Click **Reset** to reset the group's default home page to the system default home page.

4.3.1.3 Setting a User's Default Home Page

If the user has not selected a personal home page, but you have set one for him or her, the default home page specified for that user is displayed.

To set a user's default home page:

1. In the **Portal User Profile** portlet, in the **Name** field, enter the username of the user for whom you want to assign a default home page.

   By default, the **Portal User Profile** portlet is on the **Administer** tab of the **Builder** page.

   > **Note:** If you are not sure of the username, click the **Browse Users** icon and select from the list provided.

2. Click **Edit**.

3. Next to the **Default Home Page** field, click the **Browse Pages** icon to see a list of pages from which to choose.

   > **Note:** You cannot enter a value in this field; you must select one from the pop-up list.

4. Click **Return Object** next to the page you want to make the default home page for this user.

5. Click **OK**.

   > **Note:** Click **Reset** to reset the user's default home page to the system default home page.

4.3.2 Setting the System Default Style

If you are the portal administrator, you are responsible for selecting a style to serve as the system default.

When a style is deleted, all pages and item regions that used the style revert to the page group default style. If the page group default style is **<None>**, all pages and regions revert to the system default style.

> **Note:** To set the system default style, you must be the portal administrator.

To set the system default style:

1. In the Portal Builder, click the **Administer** tab.

2. Click the **Portal** sub-tab.

3. In the **Services** portlet, click the **Global Settings** link.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

4. In the **Default Style** section, choose a style from the **Display Name** list.

   > **Note:** The list includes all public styles in the **Shared Objects** page group.

5. Click **OK** to return to the Portal Builder.

4.3.3 Creating Personal Pages

A personal page provides an area within OracleAS Portal where authorized users can store and share their own content. Personal pages are located under the **Shared Objects** page group, and are organized alphabetically by user name.

> **Note:** To create personal pages for users, you must be the portal administrator.

This section covers the following topics:

- Automatically Creating a Personal Page for New Users

- Creating a Personal Page for an Existing User

4.3.3.1 Automatically Creating a Personal Page for New Users

To configure OracleAS Portal to automatically create a personal page for new users:

1. In the **Services** portlet, click **Global Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Ensure that you are on the **Main** tab.

3. Select **Create Personal Pages for New Users**.

4. Click **OK**.

Whenever a new user logs on for the first time, a personal page is automatically created for that user.

Notes:

- Personal pages are automatically created when new users log on for the first time (that is, when the user record is created for the user), not for users that already exist.

- If you create a new user and then edit that user's profile before he or she logs on for the first time, a personal page will not be created when that user logs on for the first time. This is because editing the user's profile creates the user record, so when the user logs in, he or she is not considered a new user.

4.3.3.2 Creating a Personal Page for an Existing User

To configure OracleAS Portal to create a personal page for an existing user:

1. In the **Portal User Profile** portlet:

   a. In the **Name** field, enter the name of the user for whom you want to create a personal page.

   Note: If you are not sure of the name of the user, click the **Browse Users** icon and select from the list provided.

> **b.** Click **Edit**.

By default, the **Portal User Profile** portlet is on the **Administer** tab of the **Builder** page.

2. Ensure that you are on the **Preferences** tab.

3. Select **Create Personal Page**.

> **Note:** If you do not see this check box, the user already has a personal page.

4. Click **OK**.

> **Notes:**
>
> - Personal pages are accessible from the Navigator in the **Shared Objects** page group. Any authorized user can drill down into the **Personal Pages** area of the **Shared Objects** page group, but they can only view their own personal page, or those personal pages to which they have been granted access.
>
> - Personal pages for users with user names that do not begin with an alphabetic character are located under the **Others** area of **Personal Pages**.
>
> - Personal pages cannot be deleted.

4.3.4 Setting the Total Space Allocated for Uploaded Files

You can limit the amount of space provided in your database to store documents uploaded to page groups. If you want to limit the amount of space provided for a single page group, see Section 4.3.6, "Changing the Page Group Quota".

You can also limit the size of individual files that content contributors can upload to page groups. See Section 4.3.5, "Setting the Maximum File Size for Uploaded Files" for more information.

When a user uploads a file to the portal, the upload is monitored in the middle-tier to detect if the total space or maximum file size are exceeded. If either of these limits is exceeded, the upload is terminated and an error message is displayed.

> **Note:** To set the total space allocated for uploaded files, you must be the portal administrator.

To set the total space allocated for uploaded files:

1. In the **Services** portlet, click **Global Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Make sure you are on the **Main** tab.

3. In the **Total Space Allocated** radio group, select **Limit To** to limit the amount of space provided to store files uploaded to the page groups in this portal.

4. In the field, enter the maximum amount of megabytes provided for uploaded files across the whole portal. When this limit is reached, users will no longer be able to upload files to page groups in the portal.

   > **Notes:**
   >
   > - Select **No Limit** if you do not want to impose a limit for uploaded files.
   >
   > - The **Used Space** field displays the amount of space currently used by documents uploaded to page groups in this portal.

5. Click **OK**.

4.3.5 Setting the Maximum File Size for Uploaded Files

You can limit the size of individual files that users can upload to the page groups in your portal.

You can also limit the total amount of space provided in your database to store documents uploaded to page groups. See Section 4.3.4, "Setting the Total Space Allocated for Uploaded Files" for more information.

When a user uploads a file to the portal, the upload is monitored in the middle-tier to detect if the maximum file size or portal file quota is exceeded. If either of these limits is exceeded, the upload is terminated and an error message is displayed.

> **Note:** To set the maximum file size for uploaded files, you must be the portal administrator.

To set the maximum file size for uploaded files:

1. In the **Services** portlet, click **Global Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Make sure you are on the **Main** tab.

3. In the **Maximum File Size** radio group, select **Limit To** to specify the maximum size allowed for individual files uploaded to the portal.

4. In the field, enter the maximum size (in MB) for each individual file uploaded to the portal. If a content contributor attempts to upload a file larger than this size, an error is displayed.

   > **Note:** Select **No Limit** if you do not want to impose a maximum file size.

5. Click **OK**.

4.3.6 Changing the Page Group Quota

You can limit the amount of space provided in your page group to store uploaded documents.

> **Note:** To change the page group quota, you must have at least one of the following privileges:
>
> - Portal administrator
> - Manage all privileges on the page group
> - Manage all global privileges on all page groups

To change the page group quota:

1. In the **Portal Navigator** page, click the **Page Groups** tab.

2. Click **Properties** next to the page group with which you want to work.

3. In the **Page Group Quota** section, select **Limit to** to limit the amount of space provided to store uploaded documents.

4. In the field provided, enter the size limit (in MB) for uploaded documents in the page group. When this limit is reached, users will no longer be able to upload documents to the page group.

> **Note:** Select **No limit** if you do not want to impose a limit for uploaded documents.

5. Click **OK**.

4.3.7 Specifying an Error Message Page

OracleAS Portal enables you to choose the error message page that you want to display to users. You can choose the default system error page, or you can specify your own customized error page.

OracleAS Portal includes an error message page (called Sample Error Page) that you can edit to match the look and feel of the other pages in your portal. The Sample Error Page is available under the **Portal Design-Time** page group and includes a portlet that displays all the diagnostic information. Alternatively, you can create your own error message page in any of your page groups. To do this, you must include the Error Message Portlet on the page and turn caching off.

> **Note:** By default, the Error Message Portlet is located under the Administration Portlets page of the Portlet Repository.

To specify an error message page:

1. In the **Services** portlet, click **Global Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. In the **Error Page** section, select one of the following:

   - **System Error Page** to use the system error page to display full-page error messages to users. The system error page automatically includes all the diagnostic information.

- **Error Page** to use your own page to display full-page error messages to users. Click the **Browse Pages** icon to select the error message page that you want to use.

3. Click **OK**.

4.3.8 Setting the Page Users See When They Log Out

You can specify the page that is displayed to users after they have logged out by setting the default home page for the PUBLIC (that is, non-authenticated) user.

> **Note:** You must be the portal administrator to define a default home page.

To set the page users see when they log out:

1. In the **Portal User Profile** portlet, in the **Name** field, enter **PUBLIC**.

   By default, the **Portal User Profile** portlet is on the **Administer** tab of the **Builder** page.

2. Click **Edit**.

3. Next to the **Default Home Page** field, click the **Browse Pages** icon to see a list of pages from which to choose.

   > **Note:** You cannot enter a value in this field; you must select one from the pop-up list.

4. Click **Return Object** next to the page you want to be displayed when users log out.

5. Click **OK**.

   > **Note:** Click **Reset** to remove this setting.

4.3.9 Removing the Context-Sensitive Help Link

If you have access to SQL*PLUS, you can suppress the **Context-sensitive Help** link that appears in the banner in OracleAS Portal wizards, dialog boxes, alerts, and so

on. Note that you cannot suppress the "**?**" icon that appears in the blue bar of wizards, dialog boxes, and alerts.

You cannot perform this task through the UI; it must be done programmatically through SQL*PLUS.

> **Note:** You must make the following API calls in both the portal schema and in the portal SSO schema.

To remove the context-sensitive help icon:

1. Access SQL*PLUS.

2. Enter:

   ```
   exec wwui_api_body.set_display_help (wwui_api_body.DISPLAY_HELP_OFF);
   ```

To re-instate the context-sensitive help icon:

1. Access SQL*PLUS.

2. Enter:

   ```
   exec wwui_api_body.set_display_help (wwui_api_body.DISPLAY_HELP_ON);
   commit;
   ```

4.4 Configuring Self-Registration

To enable users to create their own portal user accounts, you must configure the self-registration feature. After completing this process the self-registration link is exposed in the **Login** portlet.

You can set up an approval process for self-registered users so that they cannot log in until their accounts have been approved. When the account has been approved or rejected, the user is notified by e-mail.

If you do not require approval for self-registered users, the user will be able to log in to the portal immediately after registering.

> **Note:** To set up self-registration, you must be the portal administrator.

To set up self-registration:

1. In the **Services** portlet, click **Global Settings**.

    By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. In the **Self-Registration Options** section, select **Enable Self-Registration**.

3. Select **No Approval Required** if self-registered users can log on to the portal immediately after registering.

4. Select **Approval Required** if self-registered users need to be approved before they can log on to the portal.

    a. Click **Configure** to set up the approval process.

    b. In the **Recipients** field, enter the names of the users or groups that you want to approve self-registered users.

    > **Note:**   Use a semicolon (;) as the separator between multiple users or groups. Each step of the approval routing can include both users and groups.

    c. In the **Routing Method** radio group, choose:

        –   **One at a time, all must approve** if you want each user or group to be notified in turn and every user or group must approve self-registered users before they can log on.

        –   **All at the same time, all must approve** if you want all the users and groups to be notified at the same time and every user or group must approve self-register users before they can log on.

        –   **All at the same time, only one must approve** if you want all the users and groups to be notified at the same time, but only one user or group member must approve self-registered users before they can log on.

    d. Click **Add Step**.

    e. Repeat steps a to d to add more steps to the approval process.

> **Notes:**
>
> - You do not need to change any other settings on this tab, or any of the settings on the other tabs in this screen.
>
> - The final approver in the approval chain must be a member of the PORTAL_ADMINISTRATORS group, and must have an e-mail address defined in Oracle Internet Directory.

   f. Click **OK** to return to the Global Settings screen.

   g. In the **E-Mail (SMTP) Host** section, enter the **Host Name** and **Port** of your e-mail server so that self-registered users can be informed by e-mail when their accounts are accepted or rejected.

5. Click **OK**.

6. Go to the home page of your portal.

7. Switch to Edit mode.

8. If the home page of your portal does not already contain a **Login** portlet, add the **Login** portlet to the page.

By default, the **Login** portlet can be found in the **SSO/OID** page under the **Administration** page in the Portlet Repository.

9. Next to the **Login** portlet, click the **Actions** icon.

10. Click **Edit Defaults**.

11. Select **Enable Self-Registration**.

12. In the **Self-Registration Link Text** field, enter the text that you want users to click to register with the portal.

13. Leave the **Self-Registration URL** field blank to use OracleAS Portal's own self-registration screen.

If you have created your own self-registration screen, enter the URL in this field.

14. Click **OK**.

4.5 Performing Basic Portal Administration

This section covers the following topics:

- Simplifying the Full URL of an OracleAS Portal Instance
- Configuring Oracle HTTP Server to Use the OracleAS Portal Homepage
- Configuring a Portal DAD
- Clearing the Portal Cache
- Using a Custom Image Directory

4.5.1 Simplifying the Full URL of an OracleAS Portal Instance

You can simplify the full URL created by the OracleAS Portal installation to a more memorable or meaningful URL using the Redirect directive. In this way, end users can access OracleAS Portal by entering a simple URL.

By default, the URL for a new OracleAS Portal installation requires you to enter:

```
http://hostname:portnumber/pls/dad
```

You can simplify this URL to:

```
http://hostname/redirectpath
```

1. Open the Oracle HTTP Server configuration file, `httpd.conf`. This file is located in the following directory:

   ```
   ORACLE_HOME/Apache/Apache/conf/
   ```

2. Enter the redirect path as follows:

   ```
   Redirect /DADnamepath http://hostname:portnumber/pls/dad
   ```

   For example:

   ```
   Redirect /portalhome http://mysite.oracle.com/pls/portal
   ```

   In this example, end users can enter:

   ```
   http://mysite.oracle.com/portalhome
   ```

   to access the full URL, which is:

   ```
   http://mysite.oracle.com/pls/portal
   ```

> **Note:**
>
> - The example `http://mysite.oracle.com/portalhome`
>   assumes that the default port `80` is being used. If the default
>   port is not being used, then the user would have to enter the
>   URL with the port number,
>   `http://mysite.oracle.com:<port>/portalhome`.
>
> - You can also edit the `httpd.conf` file using the Oracle
>   Enterprise Manager Application Server Control.
>
> If the `httpd.conf` file is updated manually, you must synchronize
> the manual configuration changes done on the middle-tier by
> running *ORACLE_HOME*/dcm/bin/dcmctl as follows:
>
> ```
> dcmctl updateConfig -ct ohs
> ```
>
> Finally, restart Oracle HTTP Server, by running the following
> command from *ORACLE_HOME*/opmn/bin:
>
> ```
> opmnctl restartproc type=ohs
> ```

4.5.2 Configuring Oracle HTTP Server to Use the OracleAS Portal Homepage

To set the OracleAS Portal homepage as the Oracle HTTP Server's default
homepage:

1. In the directory *ORACLE_HOME*/Apache/Apache/htdocs/, make a backup
   copy of the files `index.html.html` and `index.html.<lang>`, where
   `<lang>` is the language code. For example, `index.html.en` is the index
   HTML file for English.

2. Edit `index.html.<lang>` by replacing the entire contents of the file with the
   following HTML redirection code:

   ```
   <HTML>
   <SCRIPT LANGUAGE=JavaScript>
   document.location="http://host.domain:port/pls/portal"
   </SCRIPT>
   </HTML>
   ```

Notes:

- Do not specify a port number if you are running OracleAS Portal on port 80.

- If you plan to support other languages, you need to have the language-specific index HTML files with the redirection code, for these languages.

4.5.3 Configuring a Portal DAD

mod_plsql provides support for building and deploying PL/SQL-based applications on the Web. PL/SQL stored procedures can retrieve data from database tables and generate HTTP responses containing formatted data and HTML code to display in a Web browser.

A Database Access Descriptor (DAD) is a set of values that specify how an application connects to an Oracle database to fulfill an HTTP request. The information in the DAD includes the user name (which also specifies the schema and the privileges), password, connect-string and Globalization Support language of the database.

There are two types of DADs: general DADs and Portal DADs. An OracleAS Portal middle-tier uses a Portal DAD to access the OracleAS Metadata Repository, which is discussed in this section. For information on general DADs, refer to the *Oracle HTTP Server Administrator's Guide*.

You configure DAD information from the **mod_plsql Services** page in the Oracle Enterprise Manager Application Server Control. See Figure 4–3.

Figure 4–3   Application Server Control - DAD Configuration

Create DAD: Database Connection

Database Access Descriptor Name

A unique name for your Database Access Descriptor. The name must not contain any special characters or spaces and may not exceed 64 characters.

DAD Name or Location   /pls/myPortal

The Portal DAD Name or Location should be pre-pended with a /pls/

Database Connectivity Information

Username   myPortal
Password   ********
Connect String   cn=iasdb,cn=oraclecontext
Connect String Format   NetServiceNameFormat (SQL*Net entry)
NLS Language   American_America.UTF8

NLS Language should match the backend Database

Use the format <NLS_LANGUAGE>_<NLS_TERRITORY>.<NLS_CHARACTERSET>, e.g., American_America.UTF8. You can obtain these values by querying the nls_database_parameters table.

You can access the **mod_plsql Services** page from:

- Application Server Control - As described next.

- OracleAS Portal - Select **Portal Service Monitoring** in the **Services** portlet (**Administer** tab) to access the Application Server Control. Then select **mod_plsql Services** from the **Component Status** table.

To configure a Portal DAD from the Application Server Control:

1. Navigate to the Application Server Control.

   Typically, `http://<host>.<domain.com>:1812`. For more information, see Section 7.1, "Using the Application Server Control".

2. Navigate to the Application Server instance where you would like to add the DAD.

3. Select **HTTP Server** from the System Components table.

4. Click **Administration**.

5. Click **PL/SQL Properties**.

6. In the **DADs** section, click **Create** to configure a new DAD.

From the DADs section you can also maintain existing DADs. To edit an existing DAD, click the **DAD Name**. To delete a DAD, select the DAD Name and click **Delete**.

7. For **DAD Type**, choose **Portal** and then click **Next**.

8. For **DAD Name or Location**, specify the path that points to the default DAD.

> **Caution:**   Do not enter non-ASCII (such as multibyte) characters for the DAD Name.

9. Enter Database Connection information (see Figure 4–3):

   ■ **Username** - Enter the Oracle database account username.

   ■ **Password** - Enter the Oracle database account password. The password is typically set at installation, but you can change it by typing a new password in this field.

   If you leave the **Oracle User Name** and **Oracle Password** fields blank, the user is prompted to enter a username and password when first logging in.

   ■ **Connect String** - Enter the connection string (if the database is remote) and then use the *Connection String Format* property to specify the format of the connect string you have entered here.

   Leave this field blank if the database is local.

   ■ **Connect String Format** - Specify the format used for the *Connect String* property.

   If the *Connection String Format* is not specified, mod_plsql assumes the connect string format is either SIDFormat (host:port:sid) or, resolvable as NetServiceNameFormat. The differentiation between the two is made by the presence of colon characters (:) in the connect string.

   For database installations like Real Application Clusters (RAC), it is recommended that users configure the connect string using the NetServiceNameFormat such that the lookup is through LDAP. This allows database nodes to be for added/removed without having to reconfigure each Oracle Application Server middle-tier separately to recognize added/removed nodes.

   ■ **NLS Language** - Enter the NLS (Globalization Support) language that is represented throughout the DAD, that is, the Globalization Support

language of the back-end Database. Use the format `<NLS_LANGUAGE>_<NLS_TERRITORY>.<NLS_CHARACTERSET>`, for example, American_America.UTF8.

To obtain these values, query the `nls_database_parameters` table as follows:

```
select value, parameter from nls_database_parameters where parameter in
('NLS_LANGUAGE','NLS_TERRITORY','NLS_CHARACTERSET');
```

10. Click **OK**.

11. Restart the Oracle HTTP Server.

Once restarted, the new DAD is accessible from the Oracle HTTP Server.

4.5.4 Clearing the Portal Cache

Sometimes you must clear the portal cache (the OracleAS Portal File System Cache). For example, if you change the character set of the OracleAS Metadata Repository. In such cases, the existing content in the Portal cache will be invalid, as it will continue to have content configured to use the older character set.

To clear the Portal cache:

1. Navigate to the Portal cache directory. The default path is *ORACLE_HOME*/Apache/modplsql/cache.

2. Perform a recursive delete of all the files under this directory. For example, on UNIX platforms, issue the following command:

```
rm -rf *
```

> **Caution:** Ensure that you are in the correct directory before issuing this command. Do not delete the `cache` directory.

4.5.5 Using a Custom Image Directory

To avoid losing custom images stored in the OracleAS Portal images directory (which is *ORACLE_HOME*/portal/images by default), it is recommended that you create your own images directory and set up an appropriate Oracle HTTP Server alias for this directory.

For example, add an entry, similar to the one shown next, to the file *ORACLE_ HOME*/portal/conf/portal.conf. It is recommended that you use the local Oracle Enterprise Manager Application Server Control instance to make this change. For more information, refer to the *Oracle HTTP Server Administrator's Guide* or the *Oracle Application Server Web Cache Administrator's Guide*.

```
Alias /mycompany/images/ "/opt/app/myportal/images/"
<Directory "/opt/app/myportal/images/">
    AllowOverride None
    Order allow,deny
    Allow from all
    ExpiresActive on
    ExpiresDefault A2592000
  <Files *>
    Header set Surrogate-Control 'max-age=2592000'
  </Files>
</Directory>
```

You do not need to perform any specific OracleAS Web Cache configuration as OracleAS Web Cache is already configured to globally cache .bmp, .gif, .png, .jpg, and .jpeg files.

4.6 Configuring Mobile Support in OracleAS Portal

This section discusses how OracleAS Portal and Oracle Application Server Wireless are configured to operate together. OracleAS Portal pages can be viewed from a wide variety of devices including desktop browsers, mobile phones, and PDAs. OracleAS Portal uses OracleAS Wireless to provide wireless functionality to receive requests from wireless devices, and transform content provided by the portal into an appropriate format.

This chapter contains the following sections:

- What Is Installed By Default?

- Configuring Mobile Settings in OracleAS Portal

- Manually Reconfiguring the Mobile Setup

4.6.1 What Is Installed By Default?

Performing a standard Oracle Application Server installation of OracleAS Portal and Oracle Application Server Wireless configures mobile support in Portal as follows:

- A master service that provides mobile device access to the installed Portal is created, which refers to the Portal home page URL. As mobile access to Portal is mediated by OracleAS Wireless, a mobile device must communicate with OracleAS Wireless to access content on OracleAS Portal.

- OracleAS Wireless service's URL refers to OracleAS Portal. Users that access this OracleAS Wireless service will be directed to the public home page of the Portal. When a mobile browser contacts OracleAS Portal through the home URL, the request will be redirected to the OracleAS Wireless service.

4.6.2 Configuring Mobile Settings in OracleAS Portal

You can change most of these mobile settings using a standard desktop browser.

To modify mobile settings:

1. In the **Services** portlet, click **Global Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Click the **Mobile** tab.

Figure 4–4   The Mobile Tab in the Global Settings Page
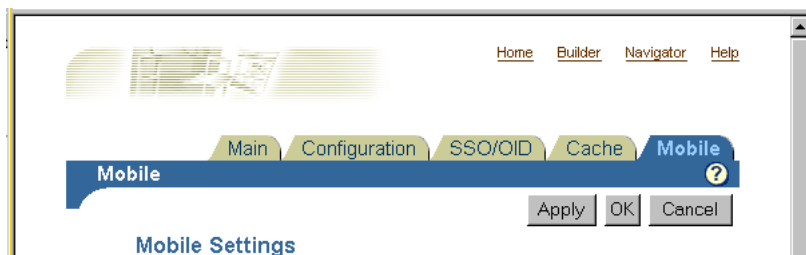
All the options for setting mobile options for OracleAS Portal are found here.

> **Note:**   In a hosted environment, you can control each subscriber individually. The exception to this is the OracleAS Wireless service URL setting. When OracleAS Portal is operating in hosted mode (with multiple subscribers), any change to the OracleAS Wireless service URL must be made by the hosting administrator, using a command line script, as it affects all subscribers.

In the **Mobile Settings** page, you can perform the following actions:

- Enabling Mobile Access

- Enabling Mobile Page Design

- Logging Mobile Responses

4.6.2.1 Enabling Mobile Access

This setting controls the response of OracleAS Portal to mobile clients that request portal pages by connecting through OracleAS Wireless.

If you want OracleAS Portal to be able to return pages and portlets in response to mobile requests, you must select the **Enable Mobile Access** option in the **Mobile** tab. If you do not select this option, OracleAS Portal responds to mobile requests with a message stating that it is not mobile enabled.

To enable mobile access:

1. In the **Services** portlet, click **Global Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Click the **Mobile** tab.

3. Select the **Enable Mobile Access** option.

Figure 4–5   Enabling Mobile Access

Mobile Settings
Select this check box to enable mobile access in the portal. For mobile access, the portal must be properly configured with OracleAS 10*g* Wireless. Clear this check box to disable mobile support in the portal, preventing users from accessing the portal from mobile devices.

☑ Enable Mobile Access

4. Click **OK**.

You'll find additional information about how OracleAS Portal responds to mobile requests, in the article "Life Cycle of a Mobile Request," on Portal Center, http://portalcenter.oracle.com.

When mobile friendly Portal pages are created, an option to preview the page as it would display in a mobile device, is available in the page editor. Selecting the **Enable Mobile Access** option in the **Mobile** tab enables the display of the **Mobile: Preview** option in the page editor.

Figure 4–6   Mobile Preview Option in the Page Editor
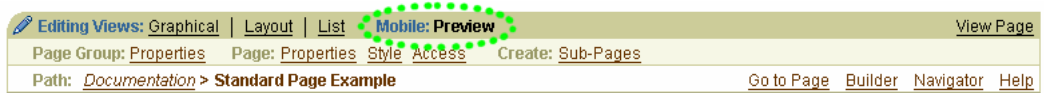
4.6.2.2 Enabling Mobile Page Design

This option enables page designers to create and edit pages using the mobile page editor, and also to specify mobile home pages. Pages of type **Mobile** are typically referred to as mobile pages.

To enable mobile page design:

1. In the **Services** portlet, click **Global Settings**.

    By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Click the **Mobile** tab.

3. Select the **Enable Mobile Page Design** option.

Figure 4–7   Enabling Mobile Page Design

4. Click **OK**.

You'll find additional information about the dedicated mobile page editor, in the article "Using the Mobile Page Editor," on Portal Center, http://portalcenter.oracle.com.

In addition, if you select this option, the following screens display an extra field that allows the selection of a home page specific to mobile access of the OracleAS Portal:

- **Main** tab in the **Global Settings** page of the **Services** portlet.

- **Preferences** tab in the **Portal User Profile** portlet

- **Preferences** tab in the **Portal Group Profile** portlet.

- **Edit Account Information** page, accessed using the **Account Info** link.

Notes:

- Once a mobile page is created, it is possible to edit the page using the mobile page editor, even if the **Enable Mobile Page Design** option is not selected.

- If the **Enable Mobile Page Design** option is not selected, page designers cannot create dedicated mobile pages. But if the option is selected, OracleAS Portal still supports mobile requests for standard pages (that is, pages that were not designed using the mobile page editor).

4.6.2.3 Logging Mobile Responses

This setting controls the logging of OracleAS Portal mobile portlet responses.

To enable logging of mobile responses:

1. In the **Services** portlet, click **Global Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Click the **Mobile** tab.

3. Select the **Log Mobile Responses** option.

Figure 4–8   Logging Mobile Responses

Developer Settings

Select this check box to enable mobile portlet content logging for portal development and testing purposes. When enabled, the portal logs the content that mobile portlets generate when they are rendered on mobile pages in response to requests from users who are logged on. It is recommended that you only select this option for portals which are under development since this option impacts performance. Changing the state of this switch causes the invalidation of cached page descriptions.

☐ Log Mobile Responses

4. Click **OK**.

Portlet responses are logged if all the following conditions are met:

- The **Log Mobile Responses** option is selected.

- The user making the request is logged on.

- The request is either from a mobile device, or it is for a mobile page.

You'll find additional information in the article "Provider Debugging Techniques: Using the Mobile Log Viewers," on Portal Center, http://portalcenter.oracle.com.

> **Note:**   Enabling or disabling the **Log Mobile Responses** option results in all the currently cached page data being invalidated. It is recommended that you do not change this option frequently after your OracleAS Portal has been deployed for general access.

4.6.3 Manually Reconfiguring the Mobile Setup

An Oracle Application Server reconfiguration that has resulted in a change to the Oracle Application Server Wireless service URL or OracleAS Portal home page URL, requires the changes to be reflected in the stored information in OracleAS Portal, and the OracleAS Wireless service definition that refers to OracleAS Portal. You should reconfigure OracleAS Wireless and OracleAS Portal to ensure that the communication between them is not affected.

You have to manually reconfigure OracleAS Wireless and OracleAS Portal to update the values of the following referenced URLs, as required:

- Updating the OracleAS Portal Home Page URL References

- Updating the OracleAS Wireless Portal Service URL Reference

4.6.3.1 Updating the OracleAS Portal Home Page URL References

The OracleAS Portal home page URL is the address that OracleAS Wireless service definition refers to. If the home page URL has changed, you need to update the following references to it:

- The Oracle Application Server Wireless Service Definition
- OracleAS Portal's Internal Reference to Itself

4.6.3.1.1 The Oracle Application Server Wireless Service Definition Use the OracleAS Wireless Webtool to update the Portal service definition. Edit the **URL** value displayed in the **Basic Info** section of the OracleAS Wireless service definition that references OracleAS Portal. This section is accessible from the **Services** tab of the OracleAS Wireless Webtool. You access the Wireless Tools at:

```
http://server:port/mobile/
```

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

4.6.3.1.2 OracleAS Portal's Internal Reference to Itself To change OracleAS Portal's own reference to its home page URL, use the script `cfgiasw.csh` (UNIX) or `cfgiasw.cmd` (Windows) to manually update the value. The script files are located here:

```
ORACLE_HOME/assistants/opca/cfgiasw.csh
```

To run the script, use the following command:

```
cfgiasw.csh -s portal -sp portal -c portal_db -h 'http://my_portal_server.com/pls/portal/portal.home'
```

The preceding example is specific to a UNIX machine. For more information on the `cfgiasw` script, see Section C.8, "Using the cfgiasw Script to Configure Mobile Settings".

4.6.3.2 Updating the OracleAS Wireless Portal Service URL Reference

Oracle Application Server Wireless is used by OracleAS Portal as an intermediary in providing access to mobile devices. To provide this access, OracleAS Portal must know the URL to the OracleAS Wireless service on which the Portal is registered. If the OracleAS Wireless service URL has changed, its reference within OracleAS Portal needs to be updated. This reference can be updated in either of the following ways:

- Specify the OracleAS Wireless Portal Service URL Using the Global Settings Page
- Use the cfgiasw Script to Update the OracleAS Wireless Service URL Reference

4.6.3.2.1   Specify the OracleAS Wireless Portal Service URL Using the Global Settings Page
To update the value of the OracleAS Wireless Portal Service URL:

1. In the **Services** portlet, click **Global Settings**.

    By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Click the **Mobile** tab.

3. Enter the URL in the **OracleAS 10*g* Wireless Portal Service URL** field.

Figure 4–9   Specifying the OracleAS Wireless Portal Service URL

4. Click **OK**.

You can change the **OracleAS 10*g* Wireless Portal Service URL** setting only when OracleAS Portal is not operating with multiple subscribers.

The **Portal home page URL** and **Portal character set** fields are for information only. If OracleAS Portal is operating with multiple subscribers, only the hosting administrator should change the value of **OracleAS 10*g* Wireless Portal Service URL**.

4.6.3.2.2   Use the cfgiasw Script to Update the OracleAS Wireless Service URL Reference   If you need to change OracleAS Portal's reference to the URL of Oracle Application Server Wireless Portal service, you can use the script `cfgiasw.csh` (UNIX) or

cfgiasw.cmd (Windows) to manually set the value. The script files are located here:

ORACLE_HOME/assistants/opca/cfgiasw.csh

To run the script, use the following command:

```
cfgiasw.csh -s portal -sp portal -c portal_db -w 'http://my_iasw_server.com/ptg/rm?PAoid=12345'
```

The preceding example is specific to a UNIX machine. For more information on the cfgiasw script, see Section C.8, "Using the cfgiasw Script to Configure Mobile Settings".

4.7 Managing Users, Groups, and Passwords

For more information on managing users, groups and passwords, refer to Chapter 6, "Securing OracleAS Portal".

4.8 Configuring Browser Settings

> **See Also:** The Browser Recommendations section in the Preface of the *Oracle Application Server Portal User's Guide*.

4.9 Configuring Language Support

OracleAS Portal is designed to allow application development and deployment in different languages. OracleAS Portal is configured with the languages that are selected in the Oracle Universal Installer (OUI) during the Oracle Application Server middle-tier installation. Languages that are configured show up in the **Set Language** portlet. You can use OracleAS Portal in the language that corresponds to the language setting in the browser, or to the language you have selected in the **Set Language** portlet. To configure additional languages after installation, the OracleAS Portal Configuration Assistant (OPCA) must be used in LANGUAGE mode.

To install languages, run OPCA in the LANGUAGE mode. Note that you must run OPCA for each language that you want OracleAS Portal to support. See Section B.2.3, "LANGUAGE" for more information.

The following example loads the Dutch language strings into the OracleAS Metadata Repository.

```
ptlasst.csh -mode LANGUAGE -lang nl -available
```

See "Usage" in Section B.2.3, "LANGUAGE" for more information on the usage of the LANGUAGE OPCA mode.

Enabling the Use of Territories

Once a language is installed into OracleAS Portal, the end user can select the language to be used from the languages displayed in the **Set Language** portlet.

OracleAS Portal's globalization support enables you to define the preferred *Locale* and *Territory* to be used for a given language. For example, Australian English, or Canadian French.

The **Set Language** portlet is not available by default, and has to be added to the **Portal Builder** page.

Adding the Set Language Portlet to the Portal Builder Page

To add the **Set Language** portlet to the Portal Builder page:

1. Click the **Administer** tab in the **Portal Builder** page.

2. Click **Edit** on top of the page.

3. Select the column where you want to add the portlet, and click the **Add Portlet** icon located above that column.

4. In the Portlet Repository, click **Portal Content Tools**.

5. Click **Set Language** in the **Available Portlets** area, and click **OK**.

   The **Set Language** portlet is now available in the **Administer** tab screen of the **Portal Builder** page.

   > **Note:** If you add the **Set Language** portlet to a page and subsequently install another language, the new language is not displayed when you view the page. As a workaround, re-register the portlet.

Enabling the Use of Territories and Locales

To enable the use of territories and locales:

1. Click the **Edit Defaults** icon for the **Set Language** portlet.

2. In the **Edit Set Language Portlet Settings** screen shown, select the **Enable Territory Selection** option. The **Edit Set Language Portlet Settings** screen is shown in Figure 4–10.

Figure 4–10   The Edit Set Language Portlet Settings screen
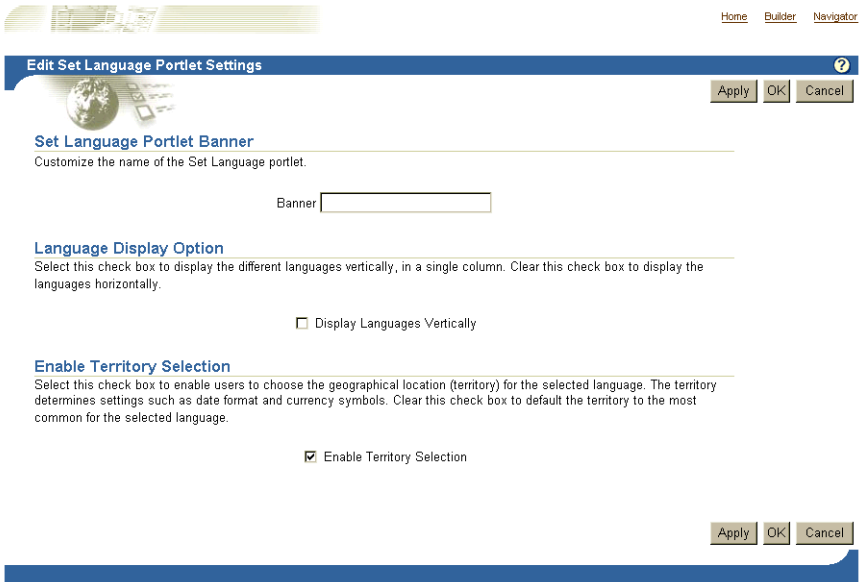
3. Click **OK**.

By selecting the **Enable Territory Selection** option, the appropriate locales for each registered language are displayed. The locales are listed after the languages in the Set Language portlet, as shown in Figure 4–11.

Figure 4–11   The Set Language Portlet

> **See Also:**   *Oracle Application Server 10g Globalization Guide*

4.10 Configuring OracleAS Portal for WebDAV

WebDAV is a protocol extension to HTTP 1.1 that supports distributed authoring and versioning. With WebDAV, the Internet becomes a transparent read and write medium, where content can be checked out, edited, and checked in to a URL address. mod_dav is an implementation of the WebDAV specification. The standard mod_dav implementation supports read and write access to files.

The term OraDAV refers to the capabilities available through the mod_oradav module. mod_oradav is the Oracle module that is an extended implementation of mod_dav, and is integrated with the Oracle HTTP Server. mod_oradav can read and write to local files, but also to an Oracle database. The Oracle database must have an OraDAV driver installed. The OraDAV driver is installed by default on installation of OracleAS Portal. mod_oradav calls this driver to map WebDAV activity to database activity. mod_oradav enables WebDAV clients to connect to an Oracle database, read and write content, and query and lock documents in various schemas.

> **See Also:**   *Oracle HTTP Server Administrator's Guide*

When Oracle Application Server is installed, all required OraDAV parameters are set with values that enable access to Oracle database content through a Web browser or a WebDAV client. If necessary, you can modify parameter values if the default values do not meet your needs.

Similar to the Portal DAD configuration file, WebDAV has it own configuration file (*ORACLE_HOME*/Apache/oradav/conf/oradav.conf) that contains the OraDAV parameters, and start with *DAV* and *DAVParam*. These parameters are specified within a *<Location>* directive. The oradav.conf file is included in the httpd.conf file in an include statement.

> **See Also:**   *Oracle Application Server Portal User's Guide*

4.10.1 Performing Basic WebDAV Configuration

After OracleAS Portal has been installed as part of the Oracle Application Server installation, the oradav.conf file should be populated with a *<Location>* directive that points to the portal schema. In the following example, the location /dav_portal/portal will be OraDAV-enabled and will (once populated with the correct values) connect to the portal schema so that users can use WebDAV clients to access portal data.

Example 4–1   Configuration Parameters for Portal Access

```
<Location /dav_portal/portal>
   DAV Oracle
   DAVParam ORACONNECT dbhost:dbport:dbsid
   DAVParam ORAUSER portal_schema
   DAVParam ORAPASSWORD portal_schema_password
   DAVParam ORAPACKAGENAME portal_schema.wwdav_api_driver
</Location>
```

By default, the OracleAS Portal DAV URL is:

```
http://hostname:port/dav_portal/portal/
```

For example:

```
http://mysite.oracle.com:7777/dav_portal/portal
```

The dav_portal part of the URL is the default name of a virtual directory used to differentiate between portal access through a WebDAV client and portal access that uses the pls virtual directory. portal is the DAD of the portal installation. You can

also configure virtual hosts to provide a different, simpler, or easier to remember URL for WebDAV access, if need be.

Users connect to a portal in WebDAV clients using the same user name and password that they use to log in to the portal itself. If the portal is in a hosted environment, users also need to add their company information to their user name, as follows:

```
<username>@<company>
```

If you are using the SQL*Net Advanced Security Option (ASO), the ORACONNECT parameter in the `oradav.conf` file must be replaced with `ORASERVICE dbhost` as shown next:

```
<Location /dav_portal/portal>
  DAV Oracle
  DAVParam ORASERVICE dbhost
  DAVParam ORAUSER portal_schema
  DAVParam ORAPASSWORD portal_schema_password
  DAVParam ORAPACKAGENAME portal_schema.wwdav_api_driver
  Options Indexes
</Location>
```

This allows the database alias to be resolved by the `tnsnames.ora` file.

Notes:

- When you add a new DAD without specifying the username and password, or if you change the Portal database schema username or password, using SQL*Plus, you will need to update the `dads.conf` and `oradav.conf` files manually.

- Whenever you make changes to the `oradav.conf` file, the HTTP Server must be restarted before the new settings will take effect.

4.10.2 Setting Up a WebDAV Client

The steps required to set up a WebDAV client to connect to OracleAS Portal varies depending on the client. All clients will eventually request a URL. The Portal DAV URL is very similar to the URL you use to access the portal itself in your Web browser, and uses the following format:

```
http://<hostname>:<port>/<dav_location>
```

If you experience problems while connecting to OracleAS Portal from a WebDAV client, refer to the WebDAV troubleshooting section in the *Oracle Application Server Portal Error Messages Guide*.

4.10.3 Checking the Version of OraDAV Drivers

You can check the version of the OraDAV drivers from any Web browser, as shown in the following example:

```
http://<machine>:<port>/<dav location>/~OraDAV-Version
```

The output will be like the following example:

```
Version 1.0.3.2.3-0030
Using Container Version 1.5
```

4.10.4 Checking the Version of mod_oradav.so

You can check the version of mod_oradav.so by running the oversioncheck binary and specifying mod_oradav.so as its argument, as shown subsequently:

```
ORACLE_HOME/Apache/Apache/bin/oversioncheck ORACLE_HOME/Apache/oradav/lib/mod_oradav.so
```

Part III

Advanced Configuration Topics

Part 3 contains the following chapters:

5

Performing Advanced Configuration

This chapter discusses the configuration that must be performed to achieve some of the more advanced configurations. To perform these configurations, you must be familiar with the available administrative tools, described in Section 4.1, "Getting Started with OracleAS Portal Administration".

This chapter contains the following sections:

- Changing the OracleAS Portal Port
- Configuring SSL
- Configuring Multiple Middle-Tiers with a Load Balancing Router
- Configuring Virtual Hosts
- Configuring OracleAS Portal to Use a Proxy Server
- Configuring Reverse Proxy Servers
- Configuring OracleAS Web Cache Caching in OracleAS Portal
- Changing the Infrastructure Services Used By a Middle-Tier
- Configuring OracleAS Wireless
- Changing the OracleAS Portal Schema Password

5.1 Changing the OracleAS Portal Port

Refer to the chapter "Port Change Procedures" in the *Oracle Application Server 10g Administrator's Guide* for information about procedures involved in changing ports in Oracle Application Server. If you change the OracleAS Web Cache port you must specify the OracleAS Web Cache settings that Portal should use on the **Portal Web**

Cache Settings screen, as described in Section 7.2.3, "Portal Web Cache Settings Page".

> **Note:** To view a list of all the ports currently in use by the components of a particular Oracle Application Server instance, refer to the steps outlined in Section 7.4, "Viewing Oracle Application Server Port Information".

5.2 Configuring SSL

OracleAS Portal uses a number of different components (such as the Parallel Page Engine, Oracle HTTP Server, and OracleAS Web Cache) each of which may act as a client or server in an HTTP communication. As a result, each component in OracleAS Portal's middle-tier must be configured individually to use the HTTPS protocol.

Configuring SSL is described in Chapter 6, "Securing OracleAS Portal". The following sections describe the various configuration options you have available for SSL with OracleAS Portal:

- Section 6.3.2.1.2, "SSL to OracleAS Single Sign-On"

- Section 6.3.2.1.3, "SSL to OracleAS Web Cache"

- Section 6.3.2.1.4, "SSL Throughout OracleAS Portal"

- Section 6.3.2.1.5, "External SSL with Non-SSL Within Oracle Application Server"

5.3 Configuring Multiple Middle-Tiers with a Load Balancing Router

This section describes how you can set up OracleAS Portal in a multiple middle-tier environment, front-ended by a load balancing router (LBR) to access the same Oracle Application Server Metadata Repository.

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

Let us assume that we want to configure the multiple middle-tier configuration, shown in Figure 5–1. In the example, we show OracleAS Web Cache on the same

machine as the Portal and Wireless middle-tier, although they can theoretically be on different machines.

Figure 5–1 Multiple Middle-Tier Configuration with Load Balancer
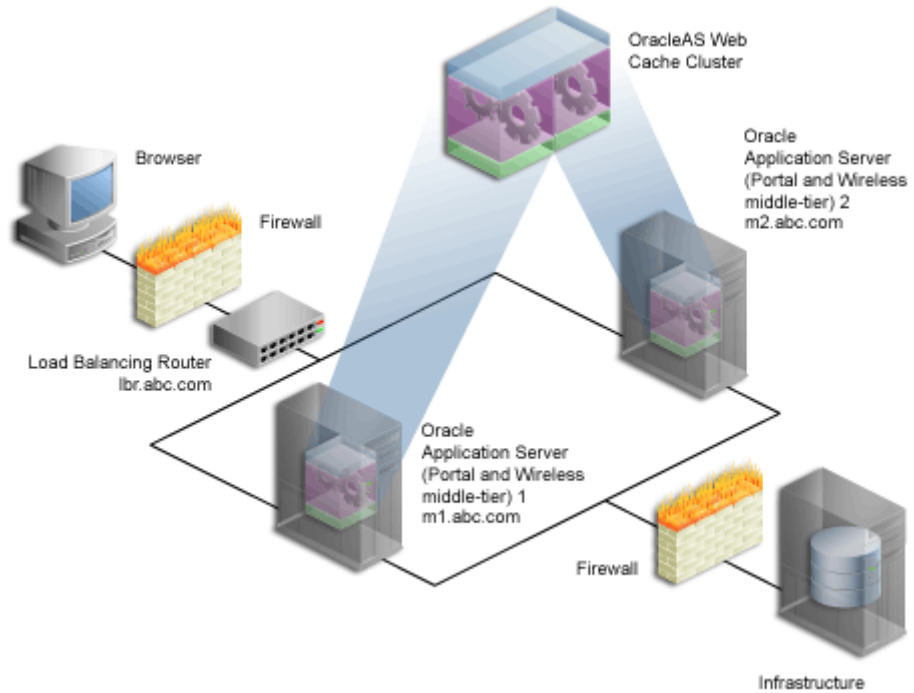
Table 5–1 Additional Information

Machine	Details
Load Balancing Router (LBR)	Machine Name: `lbr.abc.com`
	IP Address: `L1.L1.L1.L1`
	Listening Port: `80`
	Invalidation Port: `4001` (accessible only from inside)

Table 5–1   (Cont.) Additional Information

Machine	Details
Oracle Application Server (Portal and Wireless middle-tier) 1 (**M1**)	Machine Name: `m1.abc.com` IP Address: `M1.M1.M1.M1` Oracle HTTP Server Listening Port: `7778` OracleAS Web Cache Listening Port: `7777` OracleAS Web Cache Invalidation Port: `4001` OracleAS Web Cache Administration Port: `4002`
Oracle Application Server (Portal and Wireless middle-tier) 2 (**M2**)	Machine Name: `m2.abc.com` IP Address: `M2.M2.M2.M2` Oracle HTTP Server Listening Port: `7778` OracleAS Web Cache Listening Port: `7777` OracleAS Web Cache Invalidation Port: `4001` OracleAS Web Cache Administration Port: `4002`

Notes:

- The name and port values used in this section are for illustration purposes only, and you will need to substitute these with your own.

- To view a list of all the ports currently in use by the components of a particular Oracle Application Server instance, refer to the steps outlined in Section 7.4, "Viewing Oracle Application Server Port Information".

To understand how to configure OracleAS Portal with an LBR, it is important to understand a bit more about the internal architecture of Portal.

- The Parallel Page Engine (PPE) in Portal makes loopback connections to Oracle Application Server Web Cache to request page metadata information. In a default configuration, OracleAS Web Cache and the OracleAS Portal middle-tier are on the same machine and the loopback is local. When Oracle Application Server is front-ended by an LBR, all loopback requests from the PPE will start contacting OracleAS Web Cache through the LBR. Assume that the OracleAS Portal middle-tier and OracleAS Web Cache are on the same machine, or on the same subnet. Then, without additional configuration,

loopback requests result in network handshake issues during the socket connection calls.

In order for loopbacks to happen successfully, you must set up a Network Address Translation (NAT) bounce back rule in the LBR, which essentially configures the LBR as a proxy for requests coming to it from inside the firewall. This causes the response to be sent back to the source address on the network, and then forwarded back to the client.

■   OracleAS Portal leverages OracleAS Web Cache to cache a lot of its content. When cached content in OracleAS Web Cache changes, OracleAS Portal sends invalidation messages from the database to OracleAS Web Cache. OracleAS Portal can only send invalidation messages to one Web Cache node. In an OracleAS Web Cache cluster. Portal relies on that OracleAS Web Cache member to invalidate content in the other members of the cluster. When Oracle Application Server is front-ended by an LBR, the LBR must be configured to accept invalidation requests from the database and balance the load among the cluster members.

> **Note:**   You will notice in Figure 5–1 that the infrastructure is behind the LBR. The infrastructure can be one host, or distributed over multiple hosts. In order to configure the infrastructure properly, refer to the Chapter titled "Advanced Configurations" in the *Oracle Application Server Single Sign-On Administrator's Guide*.

To configure OracleAS Portal in a multiple middle-tier environment, front-ended by an LBR, you must perform the following steps:

■   Step 1: Install a Single Portal and Wireless Middle-Tier (M1)

■   Step 2: Configure OracleAS Portal on M1 to Be Accessed Through the LBR

■   Step 3: Confirm That OracleAS Portal is Up and Running

■   Step 4: Install a New Portal and Wireless Middle-Tier (M2)

■   Step 5: Configure the New Middle-Tier (M2) to Run Your Existing Portal

■   Step 6: Configure Portal Tools and Web Providers (Optional)

■   Step 7: Enable Session Binding on OracleAS Web Cache

■   Step 8: Confirm the Completed Configuration

5.3.1 Step 1: Install a Single Portal and Wireless Middle-Tier (M1)

Install a single Portal and Wireless application server middle-tier, and verify the installation. To do this perform the following steps:

1. Follow the steps described in Chapter 3, "Installing OracleAS Portal", to install a Portal and Wireless Oracle Application Server 10*g* middle-tier on the first machine (**M1**). It is assumed that you use the services of an existing Oracle Application Server Infrastructure.

    > **See Also:** *Oracle Application Server 10g Installation Guide*.

2. Verify that you have installed the middle-tier successfully by ensuring that you can access the OracleAS Portal home page at:

    ```
    http://m1.abc.com:7777/pls/portal
    ```

    Your configuration now looks like Figure 5–2, with the details described in Table 5–1.

Figure 5–2   Installation of OracleAS Portal Middle-Tier
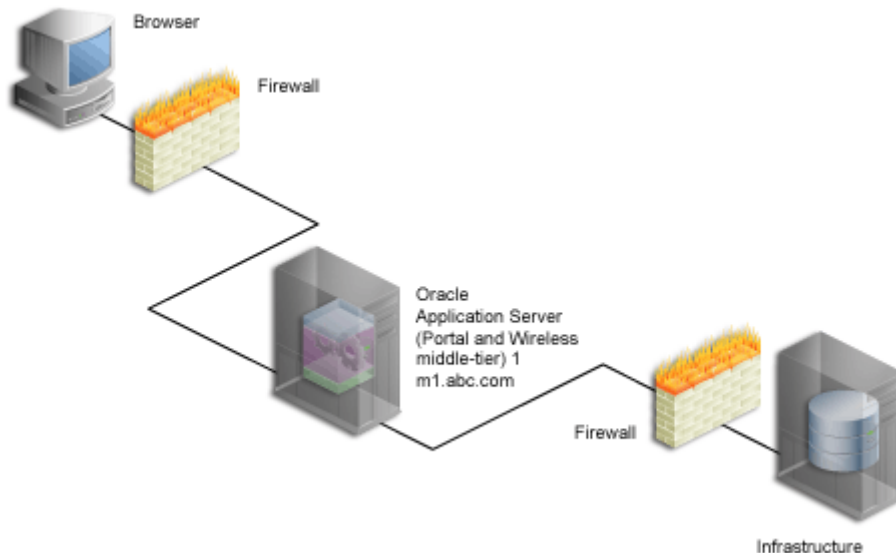
3. Access your `iasconfig.xml` file, located in *ORACLE_HOME*/portal/conf, and verify that it looks something like Example 5–1:

Example 5–1 iasconfig.xml After the First Middle-Tier Installation

```
<IASConfig XSDVersion="1.0">
   <IASInstance Name="iAS-1.m1.abc.com" Host="m1.abc.com" Version="9.0.4">
      <WebCacheComponent ListenPort="7777" AdminPort="4002" InvalidationPort="4001"
InvalidationUsername="invalidator" InvalidationPassword="@Bd4D+TnaUYFTJebppI
EqRc3/kleybcc70A==" SSLEnabled="false"/>
      <EMComponent ConsoleHTTPPort="1814" SSLEnabled="false"/>
   </IASInstance>
   <IASInstance Name="iAS.infra.abc.com" Host="infra.abc.com" Version="9.0.4">
      <OIDComponent AdminPassword="@BVs2KPJEWC5a0l4n8lbTxUY=" PortSSLEnabled="true"
LDAPPort="3060" AdminDN="cn=orcladmin"/>
   </IASInstance>
   <PortalInstance DADLocation="/pls/portal" SchemaUsername="portal"
SchemaPassword="@Beyh8p2bOWELQCsA5zRtuYc=" ConnectString="cn=iasdb,cn=oraclecontext">
      <WebCacheDependency ContainerType="IASInstance" Name="iAS-1.m1.abc.com"/>
      <OIDDependency ContainerType="IASInstance" Name="iAS-1.m1.abc.com"/>
      <EMDependency ContainerType="IASInstance" Name="iAS-1.m1.abc.com"/>
   </PortalInstance>
</IASConfig>
```

You now proceed with the next step where you configure OracleAS Portal to be accessed through an LBR.

5.3.2 Step 2: Configure OracleAS Portal on M1 to Be Accessed Through the LBR

To configure OracleAS Portal so it can be accessed through the load balancing router, perform the following steps:

1. Configure the LBR (`lbr.abc.com`) to accept requests on port 80 and forward those to the OracleAS Web Cache port (7777) running on machine `m1.abc.com.` To do this, you need to:

   a. Set up a group, or *pool* on the LBR, to which individual servers can be added.

   b. Add the desired servers' IP addresses, and port numbers to the group.

   c. Create a *virtual server* that listens on port 80, and balances load between the members of the group.

   d. Make sure the LBR translates the port that it is listening on to forward requests to the port that OracleAS Web Cache is listening on.

> **Note:** Consult the LBR documentation to set up the groups, and a virtual server.

2. Configure the OracleAS Portal middle-tier on **M1** to allow underlying components to construct URLs based on the LBR hostname (`lbr.abc.com`) and LBR port number (80), so that self-referential URLs rendered on OracleAS Portal pages are valid for the browser. To do this, perform the following steps:

   a. Define a virtual host, using the **Create Virtual Host** wizard, as explained in Section 5.4.1.1, "Create the Virtual Host for www.xyz.com", with the following exceptions:

      – On the **Addresses** page (step 9), specify the hostname of the LBR (`lbr.abc.com`) in the **Server Name** field for your virtual host.

      – In step 23, specify `80` for the Port directive in the VirtualHost container.

   b. Define a second virtual host, using the **Create Virtual Host** wizard, as explained in Section 5.4.1.1, "Create the Virtual Host for www.xyz.com", with the following exceptions:

      – On the **Addresses** page (step 9), specify the hostname of **M1** (`m1.abc.com`) in the **Server Name** field for your virtual host.

      – In step 23, specify `7777` for the Port directive in the VirtualHost container.

      – When prompted to restart the Oracle HTTP Server (step 26), click **Yes**.

3. Define a site that matches the virtual host entry, created in the previous step (`lbr.abc.com`), using OracleAS Web Cache Manager on **M1**, as follows:

   a. Access the OracleAS Web Cache Manager, using Oracle Enterprise Manager Application Server Control on **M1** as described in Section 5.7.1, "Accessing OracleAS Web Cache Manager".

   b. Click **Site Definitions** under **Origin Servers, Sites, and Load Balancing**.

   c. Click **Add Site**.

   d. On the **Add Site** page, specify `lbr.abc.com` for the **Host Name** and `80` for **Port Number**. Keep the default values for all other fields.

   e. Click **Submit**. You will now see `lbr.abc.com` in the **Site Definitions** table.

4. Use OracleAS Web Cache Manager on **M1**, to map the site `lbr.abc.com` to middle-tier `m1.abc.com`.

    a. In the navigation frame, select **Site-to-Server Mapping** under **Origin Servers, Sites, and Load Balancing**.

    b. In the **Site-to-Server Mapping** page, select the first mapping in the table and click **Insert Above**.

    c. In the **Create Site-to-Server Mapping** page, select the **Select from Site definitions** option and then select a site definition created in the previous step (`lbr.abc.com`).

    d. In the **Select Application Web Servers** section, select the application server **M1** (`m1.abc.com`) specified in the **Origin Servers** page.

    e. Click **Submit**.

    f. Click **Apply Changes** on the top of the page.

    g. In the **Cache Operations** page, click **Restart** to restart Web Cache on **M1**.

To verify that the site has been mapped correctly, navigate to the **Site-to-Server Mapping** page, and ensure that **M1** is mapped to the site `lbr.abc.com`.

5. Configure machine `m1.abc.com` so that it can resolve the LBR hostname to have the correct IP address. You can either rely on DNS resolution, or make entries in the `/etc/hosts` file as follows:

```
L1.L1.L1.L1  lbr.abc.com
```

Where `L1.L1.L1.L1` is the IP address for the LBR. There is no need to restart the system after making these changes.

Caution: Ensure that the `/etc/hosts` file does not have an entry that points the local hostname to `127.0.0.1`. For example:

```
127.0.0.1 m1.abc.com
```

---

6. Configure the LBR to perform Network Address Translation (NAT) bounce back for loopback requests coming from the PPE running on `m1.abc.com`. This ensures that when the PPE makes a loopback request to OracleAS Web Cache, there are no errors.

> **Notes:**
>
> - NAT bounce back is set up differently on individual LBRs. Consult your LBR's configuration guide on how to set this up.
>
> - The log files contain the NAT bounce back addresses for all loopback requests from the Parallel Page Engine (PPE), that get forwarded to OracleAS Web Cache or Oracle HTTP Server through the LBR.

7. Configure the LBR (`lbr.abc.com`) to accept invalidation requests from the OracleAS Metadata Repository on a separate port (4001 in this example), so that it is forwarded to the OracleAS Web Cache running on machine `m1.abc.com` on port 4001.

> **Note:** The LBR does not have to listen on the OracleAS Web Cache invalidation port. On LBRs that do not have *Port Mapping* ability the port number must match the OracleAS Web Cache invalidation port.

   a. Set up a group, or *pool* on the LBR, to which individual servers can be added.

   b. Add the desired servers' IP addresses, and port numbers to the group.

   c. Create a virtual server that listens on port 4001, and balances load between the members of the group.

   d. In the case where the LBR's port, that is listening for the invalidation requests, and the OracleAS Web Cache's invalidation port are different, you must ensure that the LBR translates the port that it is listening on to forward requests to the port that OracleAS Web Cache is listening on.

> **Notes:**
>
> - Consult the LBR documentation to set up the groups, and virtual server.
>
> - If the Oracle Application Server Infrastructure is behind another firewall, you need to make sure that it can send invalidation messages to the LBR.

> **Caution:** For security reasons, the invalidation port on the LBR (port 4001) should only be accessible from within the network.

8. You must manually edit the `iasconfig.xml` file, typically located in *ORACLE_HOME*/`portal/conf`. Before editing the file, it is recommended that you make a backup copy of it. The file must be updated to have the correct *farmname*, *hostname*, and *port* information used to access OracleAS Portal, and to perform the OracleAS Web Cache invalidation, as shown in Example 5–2 (all changes are shown in bold):

Example 5–2   iasconfig.xml File Edited to Include Farm Element

```
<IASConfig XSDVersion="1.0">

   <IASFarm Name="Farm-1.lbr.abc.com" Host="lbr.abc.com">
      <WebCacheComponent ListenPort="80" AdminPort="4002" InvalidationPort="4001"
InvalidationUsername="invalidator" InvalidationPassword="welcome1" SSLEnabled="false"/>
   </IASFarm>

   <IASInstance Name="iAS-1.m1.abc.com" Host="m1.abc.com" Version="9.0.4">
      <OIDComponent AdminPassword="@BVs2KPJEWC5a0l4n8lbTxUY=" PortSSLEnabled="true"
LDAPPort="3060" AdminDN="cn=orcladmin"/>
      <EMComponent ConsoleHTTPPort="1814" SSLEnabled="false"/>
   </IASInstance>

   <PortalInstance DADLocation="/pls/portal" SchemaUsername="portal"
SchemaPassword="@Beyh8p2bOWELQCsA5zRtuYc=" ConnectString="cn=iasdb,cn=oraclecontext">
      <WebCacheDependency ContainerType="IASFarm" Name="Farm-1.lbr.abc.com"/>
      <OIDDependency ContainerType="IASInstance" Name="iAS-1.m1.abc.com"/>
      <EMDependency ContainerType="IASInstance" Name="iAS-1.m1.abc.com"/>
   </PortalInstance>

</IASConfig>
```

9. Encrypt any plain text passwords in the `iasconfig.xml` configuration file. To do this, navigate to *ORACLE_HOME*/`portal/conf`, and run the following command:

```
ptlconfig -encrypt
```

> **Note:** To use `ptlconfig`, the `ORACLE_HOME` environment
> variable must be set.

10. Register the URL changes with OracleAS Portal. Make sure that the new URLs
    used for accessing OracleAS Portal use the LBR hostname and port, and that the
    OracleAS Web Cache invalidation URLs (OracleAS Web Cache hostname and
    invalidation ports) are that of the LBR. To do this, navigate to *ORACLE_
    HOME*/`portal/conf`, and run the following command:

    ```
    ptlconfig -dad <portal_dadname> -wc -site
    ```

    For example,

    ```
    ptlconfig -dad portal -wc -site
    ```

11. Register the secured request with OracleAS Single Sign-On by configuring it as
    a partner application. The script `ossoreg` performs this registration. `ossoreg`
    is located on the middle-tier in *MID_TIER_ORACLE_HOME*/`sso/lib`.

    a. Ensure that you have your environment configured properly to run
       `ossoreg`:

       ```
       ORACLE_HOME=MID_TIER_ORACLE_HOME
       LD_LIBRARY_PATH=ORACLE_HOME/lib
       ```

    b. Run `ossoreg`. The following example illustrates the usage of `ossoreg`.

       ```
       MID_TIER_ORACLE_HOME/jdk/bin/java -jar
         MID_TIER_ORACLE_HOME/sso/lib/ossoreg.jar -site_name lbr.abc.com
         -mod_osso_url http://lbr.abc.com -config_mod_osso TRUE
         -oracle_home_path MID_TIER_ORACLE_HOME -u install_user -config_file
         MID_TIER_ORACLE_HOME/Apache/Apache/conf/osso/osso.conf
         -admin_info cn=orcladmin -virtualhost
       ```

    Refer to the section titled "Registering mod_osso" in Chapter 4 of the *Oracle
    Application Server Single Sign-On Administrator's Guide* for more information.

After these steps, your configuration looks like Figure 5–3 with the details described
in Table 5–1.

Figure 5–3   OracleAS Portal Being Accessed Through the LBR
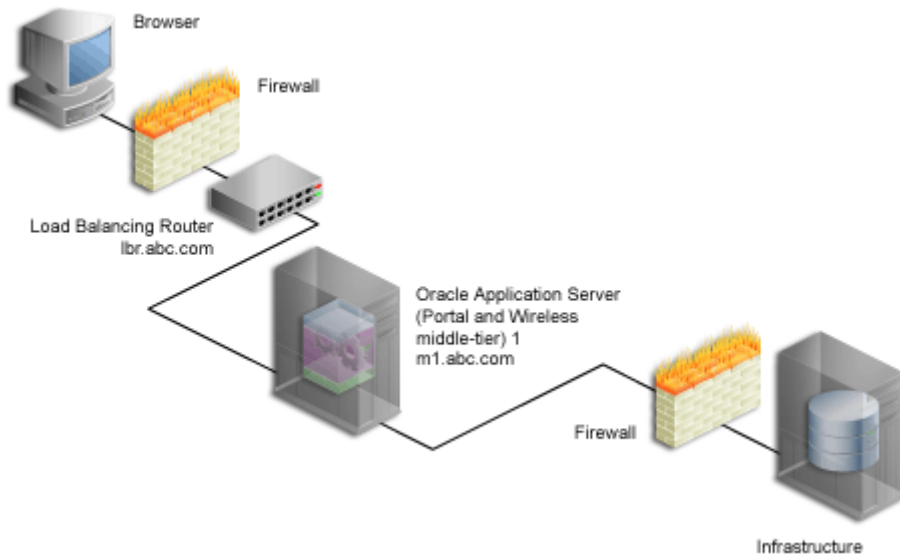
5.3.3 Step 3: Confirm That OracleAS Portal is Up and Running

Confirm that OracleAS Portal is up and running by performing the following tests in the specified order. If a test fails, address the issues, before proceeding with the next test. To diagnose the OracleAS Web Cache, Oracle HTTP Server, and LBR configuration and logs, refer to the chapter "Managing Diagnostic Log Files" in the *Oracle Application Server 10g Administrator's Guide*.

1. Test access to OracleAS Web Cache and Oracle HTTP Server through the LBR, by accessing a static file that is cached in OracleAS Web Cache, and make sure it works. For example, you can access the following URL:

   ```
   http://lbr.abc.com/relnotes.htm
   ```

2. Test the connection to Oracle Application Server Metadata Repository through the LBR, by accessing the following URL:

   ```
   http://lbr.abc.com/pls/portal/htp.p?cbuf=test
   ```

   The response should be "**test**". If this succeeds, it means that the Oracle Application Server middle-tier can connect to the OracleAS Metadata Repository. If this fails, scan the Oracle HTTP Server error_log file for details about the request failure, and take appropriate actions.

3. Test access to OracleAS Portal, by completing the following steps:

a. Access the OracleAS Portal homepage at
`http://lbr.abc.com/pls/portal`. If this does not work, scan the
`application.log` file for the OC4J_Portal instance, and look for errors.
The most common reason for this error is because the PPE cannot make
loopback connections. For this to work:

– Ensure that Network Address Translation (NAT) is enabled in the LBR.

– Ensure that the middle-tier on `m1.abc.com` can resolve the address of
`lbr.abc.com`. To do this, run the following command from
`m1.abc.com`:

`ping lbr.abc.com`

b. Click the portal login link. If this does not work, it could be due to one of
the following reasons:

– The Infrastructure middle-tier is down or is not working. Check the
OHS `error_log` file in the *INFRA_ORACLE_HOME* for more details.

– The partner application URL registration for OracleAS Portal is
incorrect, or OracleAS Single Sign-On is down.

c. Click some links in the portal.

d. Confirm that content is getting cached in OracleAS Web Cache. To do this,
access the OracleAS Web Cache Manager on **M1** as described in
Section 5.7.1, "Accessing OracleAS Web Cache Manager".

Note: The **Web Cache Administration** link in the **Services** portlet
will not work in the multiple middle-tier configuration.

Under **Monitoring**, click **Popular Requests**. Select **Cached** from the
Filtered Documents drop-down list, and click **Update**. If you accessed
OracleAS Portal, you will see portal content (For example, URLs that
contain `/pls/portal`). If you don't see any portal content, open another
browser and login to OracleAS Portal. Return to the **Popular Requests**
page, and click **Update**, to refresh the page content. This should provide
enough content for verification.

e. Perform some simple page edits in OracleAS Portal, like adding a portlet to
a page and make sure that the new content shows up. If the new content

does not display properly, or if you see errors, OracleAS Web Cache invalidation is misconfigured.

5.3.4 Step 4: Install a New Portal and Wireless Middle-Tier (M2)

Follow these steps to install a new Portal and Wireless middle-tier on **M2** (`m2.abc.com`):

1. Set the `IASCONFIG_LOC` environment variable to point to the same location that `IASCONFIG_LOC` is pointing to on machine `m1.abc.com`. The `iasconfig.xml` file allows Portal configuration to be performed from any of the hosts involved in a Web site topology. The environment variable should ideally point to a location that is accessible over a shared file system, so that installations done on different machines can reference and update the same file.

The environment variable should be set in the second middle-tier before starting the installation. To override the default location of the configuration file, you must set the environment variable `IASCONFIG_LOC` to a directory in which the file is stored, for example:

```
set IASCONFIG_LOC=/usr/local/ias904
```

Note: By default, `iasconfig.xml` resides in *ORACLE_HOME*`/portal/conf`. If the Portal Dependency Settings file is accessible over a network file system, you can share the file across multiple hosts, avoiding the need to manually replicate it every time the file is modified. If the installation is running on an operating system that supports symbolic links, it is recommended that you use this mechanism to reference a shared file. If, however, the Portal Dependency Settings file is not accessible over the network, you must ensure that the file is kept up-to-date with changes to your site topology. Refer to Section A.1.2, "Updating the Portal Dependency Settings File" for more information.

---

2. Run Oracle's Universal Installer to install a Portal and Wireless Oracle Application Server 10*g* middle-tier on the second machine (**M2**).

> **Note:** It is recommended that you use the same physical path for installing the second middle-tier. This helps when you make configuration changes on one machine and want to transfer the changes to another machine. If the physical path is different on other machines, you must ensure that the path elements are corrected after copying the files.

3. Clear the selection for **OracleAS Portal** in the **Select Configuration Options** screen during the installation of Oracle Application Server middle-tier, as shown in Figure 5–4.

Figure 5–4   Select Configuration Options Screen
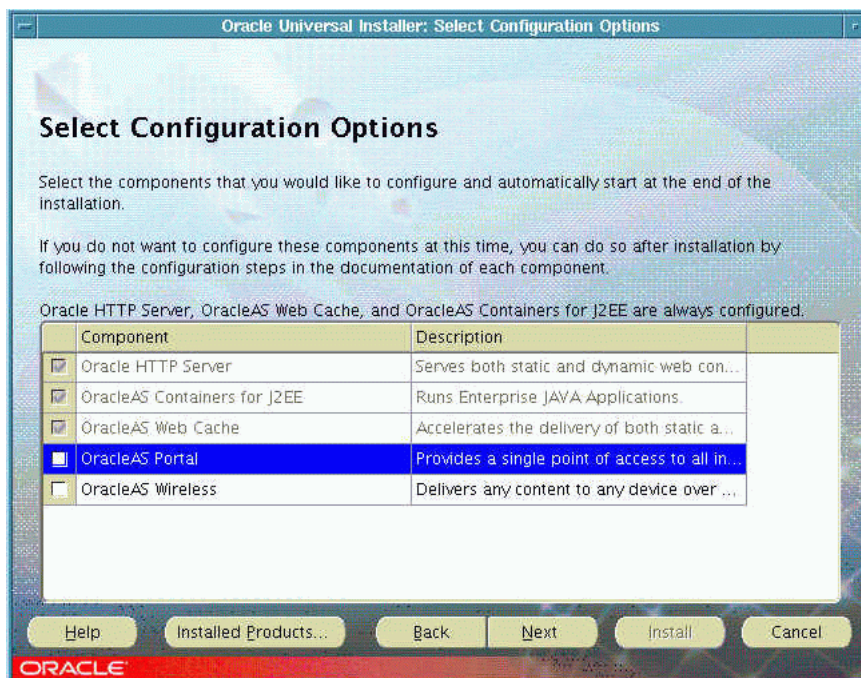
> **Caution:** Selecting OracleAS Portal in the Select Configuration Options screen overwrites your previous configuration entries in OracleAS Portal. For more information, see Section 3.3, "Configuring OracleAS Portal During and After Installation".

4. Enable **OracleAS Portal**, accessing the Application Server Control. Refer to Section 7.1.2, "Using Application Server Control to Configure OracleAS Portal", for further instructions.

> **Note:** This will deploy the OracleAS Portal middle-tier components, but will not overwrite information in the OracleAS Metadata Repository.

5. Optionally, re-register the Wireless gateway URL with the load-balancer's address. See Section 5.9, "Configuring OracleAS Wireless" for more information.

6. This new installation should not have affected your previous configuration. Confirm that OracleAS Portal is up and running on **M1**, and can be accessed through the LBR. See Section 5.3.3, "Step 3: Confirm That OracleAS Portal is Up and Running" for more information on how to check this.

5.3.5 Step 5: Configure the New Middle-Tier (M2) to Run Your Existing Portal

Follow these steps, in the order specified, to configure **M2** to run your existing portal:

1. Configure the new OracleAS Portal middle-tier to allow underlying components to construct URLs based on the LBR hostname (lbr.abc.com) and LBR port number (80). To do this, perform the following steps, using the Application Server Control on **M2**:

    a. Define a virtual host, using the **Create Virtual Host** wizard, as explained in Section 5.4.1.1, "Create the Virtual Host for www.xyz.com", with the following exceptions:

    – On the **Addresses** page (step 9), specify the hostname of the LBR (lbr.abc.com) in the **Server Name** field for your virtual host.

- – In step 23, specify `80` for the Port directive in the VirtualHost container.

  b. Define a second virtual host, using the **Create Virtual Host** wizard, as explained in Section 5.4.1.1, "Create the Virtual Host for www.xyz.com", with the following exceptions:

  - – On the **Addresses** page (step 9), specify the hostname of **M2** (`m2.abc.com`) in the **Server Name** field for your virtual host.

  - – In step 23, specify `7777` for the Port directive in the VirtualHost container.

  - – When prompted to restart the Oracle HTTP Server (step 26), click **Yes**.

2. Copy the configuration settings for OracleAS Portal from the middle-tier **M1**, to the middle-tier **M2**. It is a good idea to make backup copies of the files first. To do this, perform the following steps:

  a. Copy *ORACLE_HOME*/Apache/modplsql/conf/dads.conf from **M1** to **M2**.

  b. Copy *ORACLE_HOME*/Apache/modplsql/conf/cache.conf from **M1** to **M2**.

  c. Copy *ORACLE_HOME*/j2ee/OC4J_Portal/applications/portal/portal/WEB-INF/web.xml from **M1** to **M2**.

  d. Copy *ORACLE_HOME*/Apache/Apache/conf/osso/osso.conf from **M1** to **M2**

  e. If **M1** and **M2** are installed using different physical paths, you need to make sure that the path elements are corrected after copying the files.

  f. If you have not defined IASCONFIG_LOC in Section 5.3.4, "Step 4: Install a New Portal and Wireless Middle-Tier (M2)", you must update the iasconfig.xml file on **M2**, by following the steps described in Section A.1.2, "Updating the Portal Dependency Settings File".

  g. To synchronize the manual configuration changes done on **M2**, run the following command from *ORACLE_HOME*/dcm/bin/dcmctl:

  ```
  dcmctl updateConfig
  ```

  h. Restart OHS on **M2**, by running the following command from *ORACLE_HOME*/opmn/bin:

  ```
  opmnctl restartproc type=ohs
  ```

3. Configure the machine `m2.abc.com` to resolve the LBR hostname to have the correct IP address. You can either rely on DNS resolution for this, or make entries in the `/etc/hosts` file as follows:

```
L1.L1.L1.L1  lbr.abc.com
```

> **Caution:** Ensure that the **/etc/hosts** file does not have an entry that points the local hostname to **127.0.0.1**. For example:
>
> ```
> 127.0.0.1 m2.abc.com
> ```

4. Access the OracleAS Web Cache Manager on **M1** from the Oracle Enterprise Manager Application Server Control on the middle-tier where OracleAS Web Cache is installed, as described in Section 5.7.1, "Accessing OracleAS Web Cache Manager".

5. Use OracleAS Web Cache Manager on **M1**, to add the OracleAS Web Cache on **M2** to the OracleAS Web Cache cluster on **M1**. To do this, perform the following steps:

   a. Click **Clustering** under **Properties**.

   b. On the **Clustering** page, under the **Cluster Members** table, click **Add**.

   c. On the **Add Cache to Cluster** page, specify the following information for **M2** to be included in this Web Cache cluster:

Property	Value
Host Name	m2.abc.com
Admin Port	4002
Protocol for Admin Port	HTTP
Cache Name	m2.abc.com-WebCache
Capacity	30

> **Note:** For the value of the **Cache Name** property, you can specify any name.

   d. Click **Submit**.

e. To verify that the OracleAS Web Cache on **M2** has been added to the cluster, locate `m2.abc.com` in the **Cluster Member** table.

Refer to the section "Configuring a Cache Cluster" in the "Specialized Configurations" chapter of the *Oracle Application Server Web Cache Administrator's Guide* for more information.

6. Use OracleAS Web Cache Manager on **M1**, to add **M2** as an additional origin server to the OracleAS Web Cache cluster, created in the previous step. To do this, perform the following steps:

a. Click **Origin Server**, under **Origin Servers, Sites, and Load Balancing**.

b. In the **Origin Server** page, click **Add** under the **Application Web Servers** table.

c. In the **Add Application Web Server** page, provide the following information:

Property	Value
Hostname	m2.abc.com
Port	7778
Routing	ENABLED
Capacity	100
Failover Threshold	5
Ping URL	/
Ping Interval	10
Protocol	HTTP

> **Note:** For the **Port** value, use the **M2**'s OHS listening port.

d. Click **Submit**

e. To verify that the origin server has been added properly, locate `m2.abc.com` in the **Origin Server** table.

Refer to the section "Map Sites to Origin Servers" in the *Oracle Application Server Web Cache Administrator's Guide* for more information.

7. Use OracleAS Web Cache Manager on **M1**, to map the site `lbr.abc.com` to the two origin servers `m1.abc.com`, and `m2.abc.com`, by performing the following steps:

   a. In the navigation frame, select **Site-to-Server Mapping** under **Origin Servers, Sites, and Load Balancing**.

   b. On the **Site-to-Server Mapping** page, Select the mapping for the LBR site in the table and click **Edit Selected**.

   c. In the **Select Application Web Servers** section, select an application Web server specified in the **Origins Servers** page for **M2** (`m2.abc.com`).

   d. Click **Submit**.

   e. To verify that the site has been mapped correctly, ensure that both **M1** and **M2** are mapped to the site `lbr.abc.com` in the **Site to Server Mappings** table.

   Refer to the section "Map Sites to Origin Servers" in the *Oracle Application Server Web Cache Administrator's Guide* for more information.

8. To save your configuration changes, click **Apply Changes** on the top of the page. Perform the following steps in the **Cache Operations** page:

   a. Click **Propagate** to propagate changes to **M2**.

   b. Click **Restart** to restart Web Caches on **M1** and **M2**.

9. Configure the LBR (`lbr.abc.com`) to accept invalidation requests from the OracleAS Metadata Repository on a separate port (4001 in this example), so that it is forwarded to the OracleAS Web Cache running on machine `m1.abc.com` on port `4001`, and `m2.abc.com` on port `4001`.

> **Note:** The LBR does not have to listen on the OracleAS Web Cache invalidation port. On LBRs that do not have *Port Mapping* ability, the port number must match the OracleAS Web Cache invalidation port.

---

10. Configure the LBR (`lbr.abc.com`) to accept requests on Port 80, so that it balances the load between the OracleAS Web Cache running on machine `m2.abc.com:7777`, and the OracleAS Web Cache running on `m1.abc.com:7777`.

> **Note:** Consult the LBR documentation to complete this step.

11. Configure the LBR to perform Network Address Translation (NAT) bounce back for loopback requests coming from OHS on `m2.abc.com`. Refer to Step 6 in Section 5.3.2, "Step 2: Configure OracleAS Portal on M1 to Be Accessed Through the LBR" section for more information.

   After these steps, your configuration looks like Figure 5–1.

   > **Note:** For adding more middle-tiers, follow the procedures outlined in the sections Section 5.3.4, "Step 4: Install a New Portal and Wireless Middle-Tier (M2)" and Section 5.3.5, "Step 5: Configure the New Middle-Tier (M2) to Run Your Existing Portal", for each middle-tier.

5.3.6 Step 6: Configure Portal Tools and Web Providers (Optional)

To ensure that the Portal Tools (OmniPortlet and OracleAS Web Clipping) providers and Locally Built, as well as Custom built Web providers work properly, in the middle-tier environment, some additional configuration is required.

Configuring Portal Tools Providers in the Multiple Middle-Tier Environment

Perform the following steps for Portal Tools (OmniPortlet and OracleAS Web Clipping) providers to function properly in the multiple middle-tier environment:

1. Configure OmniPortlet to use a shared preference store. By default, the OmniPortlet provider uses the file-based preference store. However, in a multiple middle-tier environment, you must use a shared preference store, like the database preference store (`DBPreferenceStore`). To configure Portal Tools providers to use `DBPreferenceStore`, perform the following steps:

   a. Navigate to the directory *ORACLE_HOME*/j2ee/OC4J_Portal/applications/jpdk/jpdk/doc/dbPreferenceStore. For example:

```
cd ORACLE_HOME/j2ee/OC4J_Portal/applications/jpdk/jpdk/doc/dbPreferenceStore
```

b. On the database where the PORTAL schema is installed, log on to SQL*Plus as the user that owns the **DBPreferenceStore** table. Ensure that this user has the necessary privileges to create tables and indexes. For example:

```
sqlplus scott/tiger
```

c. Run the `jpdk_preference_store2.sql` script as follows in SQL*Plus:

```
@jpdk_preference_store2
```

d. Add the following entry to the file `data-sources.xml`, located in the directory *ORACLE_HOME*/`j2ee/OC4J_Portal/config`:

```
<data-source
   class="com.evermind.sql.DriverManagerDataSource"
   name="omniPortletprefStore"
   location="jdbc/UnPooledConnection"
   xa-location="jdbc/xa/XAConnection"
   ejb-location="jdbc/PooledConnection"
   connection-driver="oracle.jdbc.driver.OracleDriver"
   username="scott"
   password="tiger"
   url="jdbc:oracle:thin:@infra.host.com:1521:orcl"
   inactivity-timeout="30"
/>
```

e. Edit the file `provider.xml` located in the directory *ORACLE_HOME*/`j2ee/OC4_Portal/applications/portalTools/omniPortlet/WEB-INF/providers/omniPortlet`. Edit the `preferenceStore` tag as shown in bold:

```
<provider class="oracle.webdb.reformlet.ReformletProvider">
   <vaultId>0</vaultId>
   <session>true</session>
   <preferenceStore
class="oracle.portal.provider.v2.preference.DBPreferenceStore">
      <name>omniPortletprefStore</name>
      <connection>jdbc/PooledConnection</connection>
   </preferenceStore>
```

More information on configuring the database preference store can be found in the PDK article titled "Installing the DBPreferenceStore Sample", located at `http://portalstudio.oracle.com/pls/ops/docs/FOLDER/COMMUNITY/PDK/jpdk/v2/doc/dbPreferenceStore/installing.db.preference.store.v2.html`, on Portal Studio at http://portalstudio.oracle.com.

2. Optionally, you can change the settings for the HTTP proxy configuration, or the repository used by OmniPortlet and OracleAS Web Clipping. You can change the settings on the Portal Tools **Edit Provider** pages (OmniPortlet, and OracleAS Web Clipping), accessible from the Portal Tools providers' test pages. The test pages are located at the following URLs:

   – OmniPortlet provider test page on **M1**:
   ```
   http://m1.abc.com:7777/portalTools/omniPortlet/provid
   ers/omniPortlet
   ```

   – Web Clipping provider test page on **M1**:
   ```
   http://m1.abc.com:7777/portalTools/webClipping/provid
   ers/webClipping
   ```

   Refer to Section I.1, "Configuring the Web Clipping Repository", and Section I.2, "Configuring HTTP or HTTPS Proxy Settings" for more information.

3. Copy the *ORACLE_HOME*/j2ee/OC4J_ Portal/applications/portalTools/omniPortlet/WEB-INF/provide rs/omniPortlet/provider.xml from **M1** to **M2**.

4. Copy the *ORACLE_HOME*/j2ee/OC4J_ Portal/applications/portalTools/webClipping/WEB-INF/provide rs/webClipping/provider.xml from **M1** to **M2**.

5. Copy the *ORACLE_HOME*/j2ee/OC4J_ Portal/config/data-sources.xml from **M1** to **M2**.

6. Click **Edit Registration** for the OmniPortlet Provider on the **Providers** tab of the Navigator, under **Locally Built Providers**. Then click the **Connection** tab, and change the first part of the provider registration URL from `http://m1.abc.com:7777/` to `http://lbr.abc.com/`.

7. Click **Edit Registration** for the Web Clipping Provider on the **Providers** tab of the Navigator, under **Locally Built Providers**. Then click the **Connection** tab and change the first part of the provider registration URL from `http://m1.abc.com:7777/` to `http://lbr.abc.com/`.

8. Verify that OmniPortlet and the Web Clipping Provider work properly through the LBR, by going to the test pages at the following URLs.

   ■ OmniPortlet Provider:
   ```
   http://lbr.abc.com:80/portalTools/omniPortlet/providers/
   omniPortlet
   ```

■ Web Clipping Provider:
```
http://lbr.abc.com:80/portalTools/webClipping/providers/
webClipping
```

Note: If you want to use the OracleAS Web Clipping provider, or the Web Page Data Source for OmniPortlet, you must also enable session binding in OracleAS Web Cache, as described in Step 7: Enable Session Binding on OracleAS Web Cache.

Creating Locally Built Web Providers in the Multiple Middle-Tier Environment

Locally Built providers are providers that are defined within an instance of OracleAS Portal. Perform the following steps to create a Locally Built Web Provider that functions properly in the multiple middle-tier environment:

1. Access OracleAS Portal through the LBR at the URL
   `http://lbr.abc.com/pls/portal`.

2. In the **Services** portlet, click **Global Settings**. By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page. Click the **Configuration** tab, and point the **Default JPDK Instance** URL to the **M1** middle-tier, by specifying
   `http://m1.abc.com:7777/jpdk/servlet/soaprouter/`.

3. Create Web providers and create portlets under them. This creates a `provider.xml` file for each new provider.

4. To use the Locally Built Web provider in the multiple middle-tier environment, you need to copy the following files to **M2**:

   a. Copy the *ORACLE_HOME*/j2ee/OC4J_
      Portal/applications/portalTools/providerBuilder/WEB-INF/
      providers/<providerName> directory from **M1** to **M2**.

   b. Copy the *ORACLE_HOME*/j2ee/OC4J_
      Portal/applications/portalTools/providerBuilder/WEB-INF/
      deployment/<providerName>.properties file from **M1** to **M2**.

   c. Copy the *ORACLE_HOME*/j2ee/OC4J_
      Portal/applications/portalTools/providerBuilder/WEB-INF/
      deployment_provideruI/provideruiacls.xml file from **M1** to **M2**.

d. Copy the entry for `<providerMap>` in *ORACLE_HOME*/j2ee/OC4J_
Portal/applications/portalTools/providerBuilder/WEB-INF/
deployment_providerui/providerstore.xml from **M1** to **M2** and
change the `<warDir>` element with the appropriate value for the *ORACLE_
HOME* for **M2** (shown in bold):

```
<providerMap name="MyProvider" baseLanguage="en">
    <displayName language="en" translation="myprovider"></displayName>
    <timeout>20</timeout>
    <timeoutMessage language="en" translation="Timed Out"></timeoutMessage>
    <loginFrequency>Never</loginFrequency>
    <httpURL>http://lbr.abc.com:80/portalTools/builder/providers/MYPROVIDER</httpURL>
    <cookieDomain>abc.com</cookieDomain>
    <serviceId>MYPROVIDER</serviceId>
    <requireSessionData>false</requireSessionData>
    <httpAppType>Portal</httpAppType>
    <warDir>ORACLE_HOME/j2ee/OC4J_
Portal/applications/portalTools/providerBuilder/WEB-INF</warDir>
    <warFile>providerBuilder</warFile>
</providerMap>
```

e. Click **Edit Registration** for the provider on the **Providers** tab of the
Navigator, under **Locally Built Providers**. Then click the **Connection** tab
and change the first part of the provider registration URL from
`http://m1.abc.com:7777/` to `http://lbr.abc.com/`.

f. Verify that the Web provider works properly through the LBR, by going to
the test page at the URL
`http://lbr.abc.com:80/portalTools/builder/providers/<pro`
`viderName>`.

5. In the **Services** portlet, click **Global Settings**. By default, the **Services** portlet is
on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page. Click
the **Configuration** tab, and point the **Default JPDK Instance** URL to the LBR,
by specifying `http://lbr.abc.com:80/jpdk/servlet/soaprouter/`.

> **Note:** If you edit Locally Built Web Providers again, you will need
> to manually replicate the changes in the previously mentioned files
> to the other middle-tiers. However, since the middle-tiers are
> accessed through the LBR, you cannot specify on which middle-tier
> the changes are being made. If you want to create new Web
> providers, repeat the previous steps.

Configuring Custom Built Providers in a Multiple Middle-Tier Environment

A custom built provider is any web provider that is not seeded by the OracleAS Portal installation, or created using OracleAS Portal. To configure the custom built provider, you must deploy it on the first middle-tier, and register it to OracleAS Portal, using the **M1** URL (`http://m1.abc.com:7777/<webApp>/providers/<providerName>`. To configure it to work in the multiple middle-tier environment, you must perform the following steps:

1. Configure the custom built provider to use a shared preference store. Refer to the steps in the section, Configuring Portal Tools Providers in the Multiple Middle-Tier Environment, in this document.

   More information on configuring the database preference store can be found in the PDK article titled "Installing the DBPreferenceStore Sample", located at `http://portalstudio.oracle.com/pls/ops/docs/FOLDER/COMMUNIT Y/PDK/jpdk/v2/doc/dbPreferenceStore/installing.db.preferenc e.store.v2.html`, on Portal Studio at http://portalstudio.oracle.com.

2. Copy the *ORACLE_HOME*/j2ee/OC4J_ Portal/applications/<webApp>/WEB-INF/providers/<providerNam e>/provider.xml from **M1** to **M2**.

3. Click **Edit Registration** for the provider on the **Providers** tab of the Navigator, under **Registered Providers**. Then click the **Connection** tab, and change the first part of the provider registration URL from `http://m1.abc.com:7777/` to `http://lbr.abc/`

4. Verify that the custom built provider works properly through the LBR, by going to the test pages at the URL `http://lbr.abc.com:80/<webApp>/providers/<providerName>`

5.3.7 Step 7: Enable Session Binding on OracleAS Web Cache

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

- The Web Clipping Studio, used by both the OracleAS Web Clipping Portlet and the Web Page Data Source of OmniPortlet, uses HTTP session to maintain state, for which session binding must be enabled. Refer to Appendix I, "Administering Web Clipping" for more information about Web Clipping.

- Enabling session binding forces all the user requests to go to a given OracleAS Portal middle-tier, resulting in a better cache hit ratio for the portal cache. Refer to Section 1.3.2, "Understanding Portal Cache" for details on the portal cache.

> **Note:** Regardless of whether you have configured an LBR in your topology, you must enable session binding in OracleAS Web Cache, if you have more than one middle-tier. In this configuration OracleAS Portal does not require session binding to be set up on the LBR.

To make OracleAS Web Cache bind the portal user session to the OracleAS Portal middle-tier, perform the following steps:

1. In OracleAS Web Cache Manager on either **M1**, or **M2**, click **Session Binding** under **Origin Servers, Sites, and Load Balancing**.

2. In the **Session Binding** page, select the LBR site name (`lbr.abc.com:80`) in the table, and then click **Edit Selected**.

3. From the **Please select a session** list, change the session value to **All Sessions**. Leave **Inactivity Timeout** at the default setting.

4. Click **Submit** to apply the new settings to the site `lbr.abc.com:80`.

5. If **Session Binding Cookie** is disabled, click **Enable**.

6. To save your configuration changes, click **Apply Changes** at the top of the page.

7. On the **Cache Operations** page, click **Propagate** to propagate the changes.

8. Click **Restart** to restart OracleAS Web Cache on M1 and M2.

5.3.8 Step 8: Confirm the Completed Configuration

To verify that your complete configuration is working as expected, perform the following steps:

1. To clear the contents stored in OracleAS Web Cache, restart **M1** and **M2**, as follows:

   a. Access the Application Server Control, typically located at `http://www.abc.com:1812`.

   b. Click the **M1** instance.

   c. Click **Restart All**.

   d. Repeat the steps for **M2**.

2. Test access to OracleAS Portal through the LBR, by completing the following steps:

   a. Access the OracleAS Portal homepage at
   `http://lbr.abc.com/pls/portal`.

   b. Click the portal login link.

   c. Click some links in the portal.

   d. Confirm that content is getting cached in OracleAS Web Cache. To do this, access the OracleAS Web Cache Manager on **M1** as described in Section 5.7.1, "Accessing OracleAS Web Cache Manager".

   Under **Monitoring**, click **Popular Requests**. Select **Cached** from the **Filtered Documents** drop-down list, and click **Update**. If you accessed OracleAS Portal, you will see portal content (For example, URLs that contain `/pls/portal`).

   Perform some simple page edits in OracleAS Portal, like adding a portlet to a page and make sure that the new content shows up. If the new content does not display properly, or if you see errors, OracleAS Web Cache invalidation is misconfigured.

5.4 Configuring Virtual Hosts

The Oracle HTTP Server (OHS) supports the configuration of virtual hosts. Virtual hosts allow a single machine to appear as any number of different sites. You can, for example, configure a machine to represent both `www.abc.com`, as well as `www.xyz.com`. Another example would be configuring a machine to represent both `my.oracle.com`, as well as `oraclepartnernetwork.oracle.com`. To configure virtual hosts with Oracle Application Server Portal, you must set this up on the Oracle HTTP Server. Additional Oracle Application Server Web Cache and Oracle Application Server Single Sign-On configuration is also required.

Note: If your intent is simply to change the hostname of your middle-tier, refer to the chapter titled "Changing Network Configurations" in the *Oracle Application Server 10g Administrator's Guide*.

Let's assume that your server name is `www.abc.com`, and you connect to OracleAS Portal at `http://www.abc.com:7779/pls/portal`. The IP address of the machine that the middle-tier is installed on is `196.12.67.8`.

You want to access OracleAS Portal at `http://www.abc.com:7779/pls/portal`, using the real servername, as well as `http://www.xyz.com:7779`, using a virtual hostname, where both URLs resolve to the same IP address.

In this example, port `7779` is the OracleAS Web Cache listening port, and port `7778` is the OHS listening port.

Let's also assume that the OracleAS Single Sign-On is installed on a different machine with the IP address `123.45.67.8`, and accessed at the URL `http://www.login.com:7777/pls/orasso`.

Notes:

- The IP addresses used in this example are for illustration purposes only and may not be valid IP addresses.

- The name and port values used in this section are for illustration purposes only and you will need to substitute these with your own.

- In this section we only describe how to configure virtual hosts for the OracleAS Portal middle-tier, and this does not modify the hostname for OracleAS Single Sign-On. For more information on how to customize the OracleAS Single Sign-On hostname, refer to the section titled "Deploying OracleAS Single Sign-On with a Proxy Server" in the "Advanced Configurations" chapter of the *Oracle Application Server Single Sign-On Administrator's Guide*, and to the chapter titled "Changing Network Configurations" in the *Oracle Application Server 10g Administrator's Guide*.

Figure 5–5 illustrates the previously described configuration. OracleAS Web Cache and the Oracle Application Server are shown as residing on the same middle-tier machine, although they can exist on different machines.

Figure 5–5 Virtual Host Overview
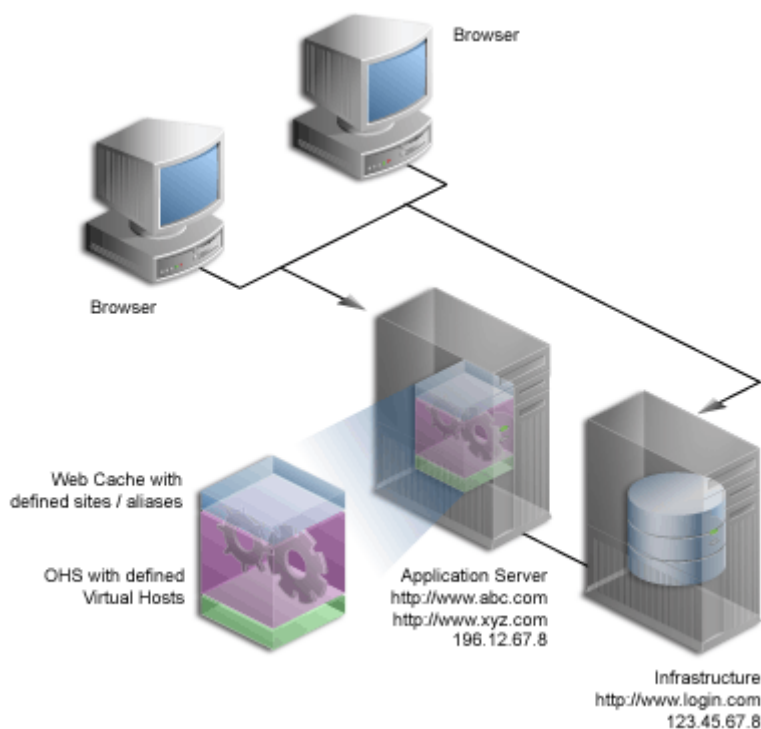
> **Note:** The domain names www.abc.com, www.xyz.com, and www.login.com must be valid domain names, and you must be able to ping them.

To configure the virtual host, perform the following steps in the specified order:

1. Create Virtual Hosts.

2. Configure OracleAS Web Cache.

3. Register OracleAS Portal with OracleAS Single Sign-On.

4. Verify the Configuration.

5.4.1 Create Virtual Hosts

You must create **virtual hosts** entries in the `httpd.conf` file for the virtual host name `www.xyz.com`, as well as for the real server name `www.abc.com`. To define the virtual hosts, use Oracle Enterprise Manager Application Server Control to perform the following steps:

- Create the Virtual Host for www.xyz.com.

- Create the Virtual Host for www.abc.com.

Once you have finished this step, do the following:

1. Verify the httpd.conf File.

2. Verify That the Virtual Hosts are Configured Correctly.

5.4.1.1 Create the Virtual Host for www.xyz.com

To create the virtual host for `www.xyz.com`:

1. Access the Oracle Enterprise Manager Application Server Control.

   Typically the Application Server Control is located at `http://www.xyz.com:1812`. Refer to Chapter 7, "Monitoring and Administering OracleAS Portal" for more information about using the Application Server Control.

2. Click the link for the middle-tier where OracleAS Portal is installed.

3. Click the **HTTP Server** link.

4. Click the **Virtual Hosts** link.

5. Click the **Create** button in the **Virtual Hosts** page.

6. On the **Introduction** page, click **Next** to create a new virtual host, using the Virtual Host Creation wizard.

7. On the **General** page, provide the information listed in Table 5–2.

Table 5–2   Virtual Host Information

Virtual Host Information	Value
Document Root Directory	*ORACLE_HOME*/Apache/Apache/htdocs
Directory Index	Can be left blank
Server Administration E-Mail	Valid e-mail address

Table 5–2 (Cont.) Virtual Host Information

Virtual Host Information	Value
Virtual Host Type	name-based

8. Click **Next**.

9. On the **Addresses** Page, provide the following information in the **Server Name** field for your virtual host:

   ```
   www.xyz.com
   ```

10. Select the option **Listen on all the main server IP addresses**.

11. Click **Next**.

12. On the **Ports** page, select **Listen on a specific port**, and select the OHS listening port, `7778` in our example, from the port dropdown list.

13. Click **Next**.

14. On the **Error Log** page, select all default values.

15. Click **Next**.

16. Review the summary on the **Summary** page.

17. Click **Finish**.

18. When prompted to restart the Oracle HTTP Server, click **No**.

19. Ensure that your server name, `www.xyz.com`, is listed in the table.

20. Click the **Administration** link.

21. Click **Advanced Server Properties**.

22. Select **httpd.conf**.

23. Add the `Port` and the `Rewrite` directives in the VirtualHost container, as follows (shown in bold text):

    ```
    NameVirtualHost *:7778

    <VirtualHost *:7778>
         ServerName www.xyz.com
         Port 7779
         ServerAdmin you@your.address
    ```

```
                   RewriteEngine On
                   RewriteOptions inherit
             </VirtualHost>
```

24. Click **Apply**.

25. When asked to restart the HTTP Server, click **No**.

5.4.1.2 Create the Virtual Host for www.abc.com

To create the virtual host for www.abc.com:

1. Follow steps 1 through 8 in Section 5.4.1.1, "Create the Virtual Host for www.xyz.com".

2. On the **Addresses** Page (step 9), provide the following information in the **Server Name** field for your virtual host:

www.abc.com

3. Follow steps 10 through 24 in Section 5.4.1.1, "Create the Virtual Host for www.xyz.com".

4. When prompted to restart the Oracle HTTP Server, click **Yes**.

5.4.1.3 Verify the httpd.conf File

After configuring virtual hosts for www.abc.com and www.xyz.com, take a look at your httpd.conf file, using Application Server Control, as follows:

1. Access the Oracle Enterprise Manager Application Server Control.

2. Click the link for the application server where OracleAS Portal is installed.

3. Click the **HTTP Server** link.

4. Click the **Administration** link.

5. Click **Advanced Server Properties**.

6. Select **httpd.conf**.

Your httpd.conf file should have the following new section:

```
NameVirtualHost *:7778

<VirtualHost *:7778>
   ServerName www.xyz.com
   Port 7779
   ServerAdmin you@your.address
```

```
   RewriteEngine On
   RewriteOptions inherit
</VirtualHost>

<VirtualHost *:7778>
   ServerName www.abc.com
   Port 7779
   ServerAdmin you@your.address
   RewriteEngine On
   RewriteOptions inherit
</VirtualHost>
```

Entries for the virtual hosts can vary depending on the existing content of the `httpd.conf` file, but you must have virtual host entries for both `www.abc.com` and `www.xyz.com`.

> **Note:** The `httpd.conf` file can also be updated manually. The file can be edited manually, to contain the right VirtualHost directives, as shown previously.
>
> To synchronize the manual configuration changes done on the middle-tier, run *ORACLE_HOME*/dcm/bin/dcmctl as follows:
>
> `dcmctl updateConfig -ct ohs`
>
> Finally, restart Oracle HTTP Server, by running the following command from *ORACLE_HOME*/opmn/bin:
>
> `opmnctl restartproc type=ohs`

> **Note:** If your site name is not registered with the DNS, you need to update the hosts file on your client machine as follows:
>
> On Windows, this file is typically located in the directory `C:\WINNT\system32\drivers\etc`. Here is an example of the hosts file on Windows.
>
> ```
> # Copyright (c) 1993-1995 Microsoft Corp.
> #
> # This is a sample HOSTS file used by Microsoft TCP/IP
> # for Windows NT/2000.
> #
> 127.0.0.1 localhost
> 196.12.67.8 www.abc.com
> 196.12.67.8 www.xyz.com
> ```
>
> On UNIX, the file is typically located in the directory `/etc/hosts`. You do not have to restart the system after making these changes.

5.4.1.4 Verify That the Virtual Hosts are Configured Correctly

Verify that both the servername, and the virtual host are working, by accessing these URLs:

- `http://www.xyz.com:7779/pls/portal`

- `http://www.abc.com:7779/pls/portal`

5.4.2 Configure OracleAS Web Cache

The site `www.abc.com` is already defined in OracleAS Web Cache. Additionally, you must create a *site alias* in OracleAS Web Cache, to make the multiple virtual hosts transparent to the OracleAS Metadata Repository. Note that `www.abc.com` must be set up as a site, while `www.xyz.com` must be defined as a site alias. This way, invalidation messages sent to OracleAS Web Cache invalidate content that is cached for both sites.

> **See Also:** *Oracle Application Server Web Cache Administrator's Guide* for information about setting up a site alias.

5.4.3 Register OracleAS Portal with OracleAS Single Sign-On

For Single Sign-On in the Oracle Application Server Single Sign-On to work properly, it must always be referenced by any partner application with the same hostname in the URL. This is because cookies are sent back only to the host that generated them. So, in the preceding example, the OracleAS Single Sign-On must always be referenced as `http://www.login.com`. Therefore, you must register `www.abc.com`, and `www.xyz.com` as partner applications. To do this:

1. Add a partner application entry for `www.abc.com`, by running OracleAS Portal Configuration Assistant (OPCA) with -mode MIDTIER and -type SSO, as follows:

```
ptlasst.csh -mode MIDTIER -type SSO -host www.abc.com -port 7779 -sdad portal
```

2. Add a partner application entry for `www.xyz.com`, by running OPCA in the SSO type of the MIDTIER mode:

```
ptlasst.csh -mode MIDTIER -type SSO -host www.xyz.com -port 7779 -sdad portal
```

3. Register the secured request with OracleAS Single Sign-On by configuring it as a partner application for `www.abc.com`. The script `ossoreg` performs this registration. `ossoreg` is located on the middle-tier in *MID_TIER_ORACLE_HOME*/sso/lib.

   a. Ensure that you have your environment configured properly to run `ossoreg`:

   ```
   ORACLE_HOME=MID_TIER_ORACLE_HOME
   LD_LIBRARY_PATH=ORACLE_HOME/lib
   ```

   b. Run `ossoreg`. The following example illustrates the usage of `ossoreg`.

   ```
   MID_TIER_ORACLE_HOME/jdk/bin/java -jar
     MID_TIER_ORACLE_HOME/sso/lib/ossoreg.jar -site_name www.abc.com
     -mod_osso_url http://www.abc.com:7779 -config_mod_osso TRUE
     -oracle_home_path MID_TIER_ORACLE_HOME -u install_user -config_file
     MID_TIER_ORACLE_HOME/Apache/Apache/conf/osso/osso.conf
     -admin_info cn=orcladmin
   ```

4. Register the secured request with OracleAS Single Sign-On by configuring it as a partner application for `www.xyz.com`, using the virtual host mode of `ossoreg`, as shown in the following example.

   ```
   MID_TIER_ORACLE_HOME/jdk/bin/java -jar
   MID_TIER_ORACLE_HOME/sso/lib/ossoreg.jar -site_name www.xyz.com
   ```

```
-mod_osso_url http://www.xyz.com:7779 -config_mod_osso TRUE    -oracle_home_
path
MID_TIER_ORACLE_HOME -u install_user -config_file
MID_TIER_ORACLE_HOME/Apache/Apache/conf/osso/osso_xyz.conf    -admin_info
cn=orcladmin -virtualhost
```

Note that the `-config_file` parameter refers to the file osso_xyz.conf.

5. You must edit the Virtual Host container for www.xyz.com as follows (changes shown in bold).

```
<VirtualHost *:7778>
   ServerName www.xyz.com
   Port 7779
   ServerAdmin you@your.address
   RewriteEngine On
   RewriteOptions inherit
   OssoConfigFile MID_TIER_ORACLE_HOME/Apache/Apache/conf/osso/osso_xyz.conf
   OssoIpCheck off
</VirtualHost>
```

Refer to the section titled "Registering mod_osso" in Chapter 4 of the *Oracle Application Server Single Sign-On Administrator's Guide* for more information.

5.4.4 Verify the Configuration

To verify that the virtual hosts are set up correctly, connect to OracleAS Portal using either of the following URLs:

- `http://www.abc.com:7779/pls/portal`

- `http://www.xyz.com:7779/pls/portal`

You should get a login screen at `http://www.login.com` on the first login and must be able to log in successfully. A subsequent login from the other virtual host should result in a successful single sign-on without a prompt for login credentials.

5.5 Configuring OracleAS Portal to Use a Proxy Server

You can configure OracleAS Portal to use proxy servers to connect to providers and Web sites outside of your firewall.

Notes:

- Oracle Text uses these proxy server settings when indexing URL content. For more information, see Section 8.3.6.4, "URL Index Proxy Settings".

- To configure OracleAS Portal to use a proxy server, you must be a portal administrator.

To specify a proxy server:

1. In the **Services** portlet, click **Proxy Settings**.

   Note: The **Services** portlet is on the **Administer** tab of the **Builder** page.

2. In the **HTTP Proxy Host** field, enter the address for the HTTP proxy server that you want to use to access applications outside your firewall, for example, `myproxy.mycompany.com`. Do not prefix `http://` to the proxy server name.

3. In the **Port** field, enter the port number for the proxy server. The port number defaults to **80** if no value is specified.

   Note: Contact your server administrator for the names of servers running proxy software and their port numbers.

4. Click **Add**.

   You can now use this proxy server for connections between the portal and Web providers. You can also use this proxy for other connections, for example, to connect to the URLs specified for URL items.

5. In the **Select Proxy** section, choose the proxy server you want to use for such connections. Choose **None** if you do not want to use a proxy server for non-provider connections.

6. In the **No Proxy Servers for Domains beginning with** field, enter the domains for which the proxy server should not be used.

> **Note:** The domains must begin with a period (.), for example,
> mycompany.com. Separate multiple domains with a comma (,).

7. Click **OK**.

**More On
Portal Center**

You'll find additional information about how to set up proxy servers in the paper "*A
Primer on Proxy Servers,*" on Portal Center, http://portalcenter.oracle.com. Click the
Search icon in the upper right corner of any Portal Center page.

5.6 Configuring Reverse Proxy Servers

A reverse proxy server is a host process that is used as part of a firewall architecture
to isolate the internal hosts from the externally accessible host(s). It does this by
providing a *proxy* through which external requests must pass to access internal
services. Typically, a proxy server takes the form of a dual-homed host. This means
that it is a host with two network interface cards. One interface connects to the
external network, and the other interface connects to the internal network, or
demilitarized zone (DMZ) of the firewall.

Figure 5–6 shows an architecture in which the browser accesses the server through
the hostname that is published by the proxy server. The proxy server then forwards
the request to the actual host within the firewall.

For this example, we will assume that you have properly configured the OracleAS
Single Sign-On server to work with the reverse proxy server.

> **See Also:** The chapter "Deploying Oracle Application Server
> Single Sign-On with a Proxy Server" in the *Oracle Application Server
> Single Sign-On Administrator's Guide*.

Figure 5–6   Reverse Proxy Server Configuration
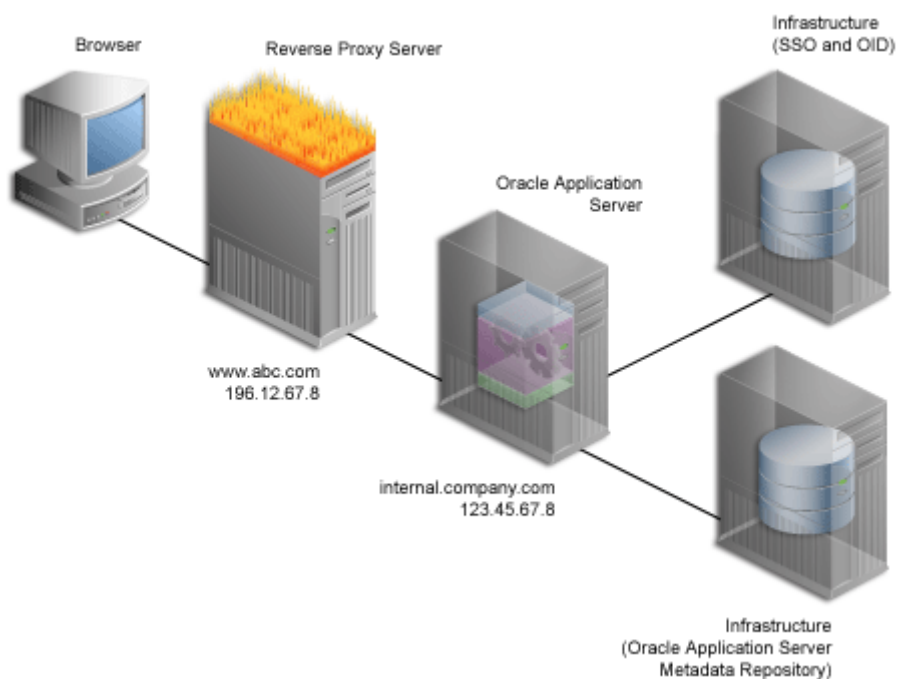
For this example, let's assume the following:

Property	Value
External server name	`www.abc.com`
External server port	80
OracleAS Web Cache server name (internal server)	`internal.company.com`
OracleAS Web Cache listening port (internal server)	`7777`
OracleAS Web Cache administration port (internal server)	`4000`
OracleAS Web Cache invalidation port (internal server)	`4001`

> **Notes:** The name and port values used in this section are for
> illustration purposes only and you will need to substitute these
> with your own.

Complete these steps to configure OracleAS Portal for the architecture shown in
Figure 5–6 in the order specified:

- Ensure That Self-Referential URLs Work

- Configure Loopback to the Internal Server

- Specify the OracleAS Portal Published Address and Protocol

- Configure Seeded Providers and Locally Hosted Web Providers

- Register the Domain Name

- Verify Your Configuration

5.6.1 Ensure That Self-Referential URLs Work

On the middle-tier, set the **ServerName** directive to point to the server name of the
reverse proxy, so that self-referential URLs rendered on OracleAS Portal pages are
valid for the browser. To do this, perform the following steps:

1. Define a virtual host, using the **Create Virtual Host** wizard, as explained in
   Section 5.4.1.1, "Create the Virtual Host for www.xyz.com", with the following
   exceptions:

   - On the **Addresses** page (step 9), specify www.abc.com in the **Server Name**
     field.

   - In step 23, specify 80 for the Port directive in the VirtualHost container.

2. Define a virtual host, using the **Create Virtual Host** wizard, as explained in
   Section 5.4.1.1, "Create the Virtual Host for www.xyz.com", with the following
   exceptions:

   - On the **Addresses** page (step 9), specify internal.company.com in the
     Server Name field.

   - In step 23, specify 7777 for the Port directive in the VirtualHost container.

   - When prompted to restart the Oracle HTTP Server (step 26), click **Yes**.

5.6.2 Configure Loopback to the Internal Server

To improve performance, and to ensure that seeded providers work correctly, the local HOSTS file must be used to resolve domain names that are not normally visible to the internal network. For more information about this loopback connection, refer to Section 1.2.2.2, "How Does Communication Flow in OracleAS Portal?".

For example, the Oracle Application Server host for `internal.company.com` makes requests to itself, but the URLs that it is requesting are referring to `www.abc.com`. You must create host entries in the local HOSTS file on that machine, allowing it to resolve this name within the firewall. The hosts entry for this example should include the following lines:

```
# This is a sample HOSTS file used by Microsoft TCP/IP
# for Windows NT/2000.
127.0.0.1    localhost
123.45.67.8  www.abc.com
```

If you do not provide these entries in the local HOSTS file, then you need to set the Oracle Application Server host to recognize a proxy server that would take the request out to the Internet and back in through the reverse proxy (`www.abc.com`), or configure the reverse proxy server's internal interface to respond to `www.abc.com`.

> **Note:** On some platforms, such as HP, there is a file that indicates the search order that should be applied to the sources for IP name mapping. For the preceding example to work, if such a file exists on your platform, make sure that it specifies the local hosts file to be checked for IP mapping, before any DNS servers.

5.6.3 Specify the OracleAS Portal Published Address and Protocol

Typically, the hostname and port number, by which OracleAS Portal is addressed, uses the OracleAS Web Cache hostname and port number. This is because, in a simple configuration, browser requests go directly to OracleAS Web Cache. However, in a configuration that has a reverse proxy server front-ending OracleAS Web Cache, the hostname and port number defined should reflect that of the reverse proxy server.

In this configuration, you want OracleAS Web Cache invalidation messages to be sent directly to the OracleAS Web Cache host, as opposed to the reverse proxy server. In the scenario where your published hostname is different from the

hostname used for OracleAS Web Cache invalidation, you cannot use the **Portal Web Cache Settings** page in the Oracle Enterprise Manager Application Server Control, or the Portal Dependency Settings file, to establish these settings.

To configure this appropriately, take the following steps:

1. Run OPCA (`ptlasst`) with -mode MIDTIER and -type OHS, as follows:

```
ptlasst.csh -i typical -mode MIDTIER -type OHS -sdad portal -host
www.abc.com -chost internal.company.com -port 80 -cport_i 4001 -cport_a 4000
```

2. Register the secured request with OracleAS Single Sign-On by configuring it as a partner application. The script `ossoreg` performs this registration. `ossoreg` is located on the middle-tier in *MID_TIER_ORACLE_HOME*/sso/lib.

   a. Ensure that you have your environment configured properly to run `ossoreg`:

   ```
   ORACLE_HOME=MID_TIER_ORACLE_HOME
   LD_LIBRARY_PATH=ORACLE_HOME/lib
   ```

   b. The following example illustrates the usage of `ossoreg`:

   ```
   MID_TIER_ORACLE_HOME/jdk/bin/java -jar
     MID_TIER_ORACLE_HOME/sso/lib/ossoreg.jar -site_name www.abc.com
     -mod_osso_url http://www.abc.com -config_mod_osso TRUE
     -oracle_home_path MID_TIER_ORACLE_HOME -u install_user -config_file
     MID_TIER_ORACLE_HOME/Apache/Apache/conf/osso/osso.conf
     -admin_info cn=orcladmin
   ```

   Refer to the section titled "Registering mod_osso" in Chapter 4 of the *Oracle Application Server Single Sign-On Administrator's Guide* for more information.

3. To prevent access to Oracle Enterprise Manager from the outside, the Oracle Enterprise Manager link provided by OracleAS Portal needs to be changed back to point to the internal server. To do this, run `ptlconfig` (typically located in the directory *MID_TIER_ORACLE_HOME*/portal/conf) as shown in the following example:

```
ptlconfig -dad portal -em
```

See Also:

- Appendix B, "Using the OracleAS Portal Configuration Assistant Command Line Utility" for more information on how to use OPCA.

- Section 7.2.3, "Portal Web Cache Settings Page" for more information about the Portal Web Cache Settings page.

- Appendix A, "Using the Portal Dependency Settings File" for more information about the Portal Dependency Settings File and Tool.

5.6.4 Configure Seeded Providers and Locally Hosted Web Providers

To configure the seeded providers (WebClipping and OmniPortlet), and locally hosted Web providers, you must take the following steps:

1. Login to OracleAS Portal as the administrator (for example, PORTAL).

2. Click the **Administer** tab.

3. Click the **Portlets** sub-tab.

4. In the **Remote Providers** portlet, type WEBCLIPPING in the **Name** field.

5. Click **Edit**.

6. Click the **Connection** tab.

7. In the **URL** field, change the URL from:

   ```
   http://www.abc.com:80/portalTools/webClipping/providers/webClipping
   ```

   to:

   ```
   http://www.abc.com:7777/portalTools/webClipping/providers/webClipping
   ```

8. Click **OK** to commit the change.

9. Repeat steps 4 through 8, with the following exceptions:

   - Type OMNIPORTLET instead of WEBCLIPPING in step 4.

   - In step 7, change the URL from:

     ```
     http://www.abc.com:80/portalTools/omniPortlet/providers/omniPortlet
     ```

     to:

```
http://www.abc.com:7777/portalTools/omniPortlet/providers/omniPortlet
```

When you register locally hosted Web Providers (such as the JPDK Sample provider), you need to register them using HTTP as the protocol, `www.abc.com` as the hostname, and 7777 as the port number. This restriction only applies to locally hosted Web Providers (that is, Web Providers running on the same middle-tier as OracleAS Portal).

For example, to register the JPDK Sample provider, the URL is:

```
http://www.abc.com:7777/jpdk/providers/sample
```

Note: If your infrastructure is located on a separate machine than your OracleAS Portal middle- tier, you need to add the following to your `/etc/host` file:

```
123.45.67.8  www.abc.com
```

where `w1.w1.w1.w1` is the IP Address of your OracleAS Web Cache machine (`internal.company.com`).

5.6.5 Register the Domain Name

Register the `www.abc.com` domain name on a domain name server on the Internet, with IP address 196.12.67.8.

Note: In order for OracleAS Portal to work with a reverse proxy server, the reverse proxy server must preserve the incoming **Host HTTP** request header from the client. If this is not the default setting in your proxy server, refer to the configuration manual of the reverse proxy to set this. For example, if you are using mod_ proxy in Apache 2.0, you will need to set the *ProxyPreserveHost* directive to **On**.

5.6.6 Verify Your Configuration

You can now access OracleAS Portal at `http://www.abc.com/pls/portal`.

> **Note:** The **Web Cache Administration** link in the **Services** portlet will not work in the new configuration. Instead, you can access OracleAS Web Cache Manager, using the Application Server Control on the middle-tier where OracleAS Web Cache is installed.

5.7 Configuring OracleAS Web Cache Caching in OracleAS Portal

Oracle Application Server Web Cache offers caching, page assembly, and compression features. OracleAS Web Cache accelerates the delivery of both static and dynamic Web content, and provides load balancing and failover features for Oracle Application Server.

This section discusses how to configure OracleAS Portal to work with OracleAS Web Cache.

This section contains the following topics:

- Accessing OracleAS Web Cache Manager
- Configuring OracleAS Web Cache Settings Using Application Server Control
- Configuring OracleAS Web Cache Settings Using OracleAS Portal
- Clearing the Cache Invalidation Queue Through SQL*PLUS
- Evaluating the OracleAS Web Cache Logs
- OracleAS Web Cache Configuration Scripts
- Troubleshooting OracleAS Web Cache Configuration

5.7.1 Accessing OracleAS Web Cache Manager

OracleAS Web Cache Manager is a graphical user interface tool that combines configuration and monitoring options to provide an integrated environment for configuring and managing OracleAS Web Cache and the Web sites for which it caches content.

You can access OracleAS Web Cache Manager in different ways:

1. From the Oracle Enterprise Manager Application Server Control on the middle-tier where OracleAS Web Cache is installed:

   a. On the Application Server home page, select **Web Cache** from the list of system components.

    b. Click **Web Cache Administration** on the Web Cache home page.

2. From OracleAS Portal:

    a. Navigate to the **Services** portlet on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

    b. In the **Services** portlet, click the **Web Cache Administration** link.

After entering the OracleAS Web Cache administrator username and password (typically the ias_admin password), you will be able to use OracleAS Web Cache Manager, as shown in the Figure 5–7.

Figure 5–7   OracleAS Web Cache Manager
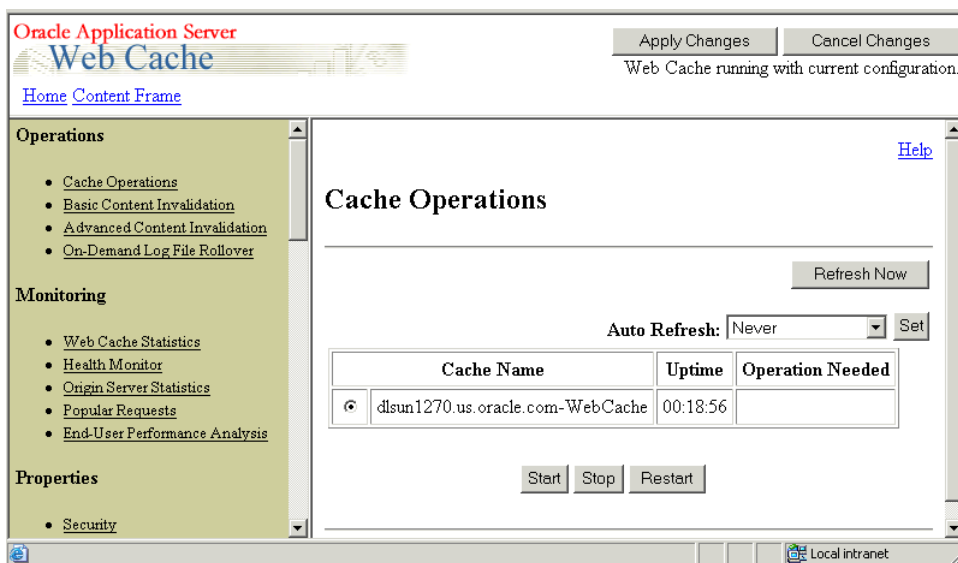
For more information on the usage of the OracleAS Web Cache Administration Manager, refer to the *Oracle Application Server Web Cache Administrator's Guide*.

5.7.2 Configuring OracleAS Web Cache Settings Using Application Server Control

You must use the Oracle Enterprise Manager Application Server Control to configure OracleAS Web Cache settings that OracleAS Portal uses, such as the hostname, and the invalidation port number. These settings can be configured on the **Portal Web Cache Settings** page.

Using Application Server Control, you can specify the OracleAS Web Cache settings that OracleAS Portal should use. Setting the OracleAS Web Cache properties on this page will automatically result in the Portal Dependency Settings file located on this middle-tier being updated, and the `ptlconfig` script being executed.

For a detailed description of how to access, and use the Application Server Control, and specifically the Portal Web Cache Settings page, refer to Section 7.2.3, "Portal Web Cache Settings Page".

5.7.3 Configuring OracleAS Web Cache Settings Using OracleAS Portal

From the OracleAS Portal user interface, you can perform various other OracleAS Web Cache related configuration tasks. The entire OracleAS Web Cache can be cleared, or it can be cleared for each user.

> **Caution:** Clearing the cache results in cache misses on subsequent requests and may degrade the portal's performance until the cache is repopulated.

You may want to clear the cache if a user's group membership has changed, in order to remove the cache entries for that user, so that he or she has new privileges. Similarly, if you change a user or group's privileges on an object, you can clear the cache entries for that object.

To clear the entire cache, or to clear the cache for a particular user, you must be the portal administrator. To clear the cache for a particular portal object, you must have at least **Manage** privileges on the object.

It is also possible to configure OracleAS Portal's OracleAS Web Cache settings, using the **Cache** tab on the **Portal Global Settings** page. However, this is not the recommended approach. You must use Oracle Enterprise Manager Application Server Control to change these settings, as this will update the `iasconfig.xml` file.

If you have used the **Portal Global Settings** page to change any of the OracleAS Web Cache settings, you must take the following manual steps to update the `iasconfig.xml` file:

1. Edit *ORACLE_HOME*/portal/conf/iasconfig.xml.

2. Locate the `WebCacheComponent` element for the portal instance you want to update and modify the properties of the `WebCacheComponent` as required.

3. Run the following script to update the Oracle Application Server Metadata Repository with the new settings:

```
ORACLE_HOME/portal/conf/ptlconfig -dad <dad> -wc
```

The following sections describe the actions that can be performed using OracleAS Portal in more detail:

- Clearing the Entire Web Cache
- Clearing the Cache for a Particular User
- Setting the Expiry Time for Invalidation-based Caching
- Clearing the Cache for a Particular Portal Object

5.7.3.1 Clearing the Entire Web Cache

You can clear the entire Web Cache by performing the following steps:

1. In the **Services** portlet, click **Global Settings**.

> **Note:** By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

---

2. Click the **Cache** tab.

3. Select **Clear The Entire Web Cache**.

4. Click **Apply** or **OK** to clear the cache.

> **Note:** This clears all the page URLs and style sheets but not the Portal images.

5.7.3.2 Clearing the Cache for a Particular User

You can clear the cache for a particular user by performing the following steps:

1. In the **Services** portlet, click **Global Settings**.

> **Note:** By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

---

2. Click the **Cache** tab.

3. In the **Clear Cache For User** field, enter the name of the user for whom you want to clear the cache.

> **Note:** If you are not sure of the user name, click the **Browse Users** icon and select from the list provided.

4. Click **Apply** or **OK** to clear the cache for the specified user.

> **Note:** Alternatively, you can clear the cache for a particular user by editing the user's portal profile.

5.7.3.3 Setting the Expiry Time for Invalidation-based Caching

With invalidation-based caching, a cache entry is purged when the portal or a provider sends a message informing OracleAS Web Cache that the object has changed (for example, when an item is edited). However you can also set an expiry time for cache entries. A cache entry that reaches this time limit is purged, even if OracleAS Web Cache has not received an invalidation message for it.

> **Note:** To set the expiry time for invalidation-based cache entries, you must be the portal administrator.

To set the expiry time for invalidation-based cache entries:

1. In the **Services** portlet, click **Global Settings**.

> **Note:** By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Click the **Cache** tab.

3. In the **Maximum Expiry Time** field, enter the maximum amount of time (in minutes) a cache entry should remain in the cache before being purged.

4. Click **OK**.

5.7.3.4 Clearing the Cache for a Particular Portal Object

You can clear cache entries for page groups, pages, page templates, portlets in the Portlet Repository, Portal database providers, and Portal database provider components, by performing the following steps:

1. In the Navigator, drill down to the object with which you want to work.

   - For page groups, pages, and page templates, click **Properties**, then click the **Access** tab.

   - For Portal database providers, and Portal database provider components, click **Grant Access**.

   - For portlets, click **Edit Root Page** next to the **Portlet Repository** page group, drill down to the page that contains the portlet, edit the portlet, and click the **Access** tab.

2. Click **Clear Cache**.

3. Click **OK**.

5.7.4 Clearing the Cache Invalidation Queue Through SQL*PLUS

Sometimes, the cache invalidation queue can grow excessively large as a result of user actions. For example, repeated granting of security privileges on a page to a group with a large number of members will place one soft invalidation in the queue for every user for every grant.

Some soft invalidations may not be necessary, but OracleAS Portal may not be able to determine this. For example, if a group's privileges on a page are upgraded from **View** to **Fully Customize**, and no member of the group has viewed the page yet, then no invalidation is necessary. However, Portal does not have a record of who has viewed the page. Therefore, it proceeds with the soft invalidation configured to use the security change.

The Portal administrator can check the number of soft invalidations in the queue by executing the following query in SQL*PLUS as the Portal schema owner:

```
select count(1) from wwutl_cache_inval_msg$ where process_type=2;
```

The Portal administrator can check the total number of invalidations, hard or soft, in the queue by executing the following query in SQL*PLUS as the Portal schema owner:

```
select count(1) from wwutl_cache_inval_msg$;
```

The number of rows in the table **wwutl_cache_inval_msg$** that can be considered excessive depends, to some extent, on the speed of the infrastructure running the database. Typically, 50000 messages will slow down the soft invalidation job, and sending 50000 invalidation messages to OracleAS Web Cache will introduce a network load, as OracleAS Portal communicates with the OracleAS Web Cache invalidation port.

If the soft invalidations are found to be unnecessary, the Portal administrator can perform the following query in SQL*PLUS as the Portal schema owner:

```
delete from wwutl_cache_inval_msg$ where process_type=2;
```

This removes soft invalidations from the queue.

If the soft invalidations are necessary but there is an excessively large number, the Portal administrator can clear the cache invalidation queue using the following command:

```
truncate table  wwutl_cache_inval_msg$;
```

The Portal administrator should then clear the entire cache through the Portal UI as described in Section 5.7.3.1, "Clearing the Entire Web Cache".

5.7.5 Evaluating the OracleAS Web Cache Logs

Log files for OracleAS Web Cache are typically stored in *ORACLE_HOME*/webcache/logs on UNIX and *ORACLE_HOME*\webcache\logs on Windows.

There are two log files:

- The access_log file.

- The event_log file.

> **See Also:** *Oracle Application Server Web Cache Administrator's Guide*

5.7.6 OracleAS Web Cache Configuration Scripts

You can configure OracleAS Portal to work with OracleAS Web Cache in a variety of ways, and some scripts are supplied to facilitate this. These scripts are covered in more detail in Section C.1, "OracleAS Web Cache Configuration Scripts".

The scripts covered in Appendix C, "Using OracleAS Portal Installation and Configuration Scripts" are:

- `cachset.sql`, which enables you to turn the use of OracleAS Web Cache on or off. Information on how to disable OracleAS Web Cache completely is covered as well.

- `cachjsub.sql`, which enables you to manage the invalidation message processing job.

5.7.7 Troubleshooting OracleAS Web Cache Configuration

> **See:** *Oracle Application Server Portal Error Messages Guide*

5.8 Changing the Infrastructure Services Used By a Middle-Tier

Oracle Application Server 10*g* enables you to change the Infrastructure services (either Oracle Identity Management or OracleAS Metadata Repository) that a middle-tier uses. You can use this feature, for example, to move middle-tiers (and their applications) from stage to production. If you are changing the OracleAS Metadata Repository that your OracleAS Portal uses, then you will also need to move application-specific data stored in the stage OracleAS Metadata Repository to an OracleAS Metadata Repository in the production environment. Changing the Infrastructure services is convenient, if you need additional computers for the production environment. In a single step, you add a computer that already has a middle-tier and deployed applications. For instructions on how to change the Infrastructure Services used by a middle-tier instance, refer to the *Oracle Application Server 10g Administrator's Guide*.

> **Note:** By default, an OracleAS Portal middle-tier is made up of one portal instance. Both the DAD name and the OracleAS Metadata Repository schema name for this instance are **portal**. You can only change the Infrastructure services of this default OracleAS Portal instance in the previously described way.

5.9 Configuring OracleAS Wireless

If Oracle Application Server Wireless is configured with OracleAS Portal during the middle-tier installation, the middle-tier installation registers the Portal on the OracleAS Wireless service. In case of multiple middle-tier installs, the last configured OracleAS Wireless service URL is stored in the OracleAS Portal instance. You can change this to your choice of OracleAS Wireless service by running the

`portalRegistrar.sh` script in the Oracle Application Server middle-tier selected for the OracleAS Wireless service:

On UNIX:

ORACLE_HOME/wireless/bin/internal/portalRegistrar.sh

On Windows:

ORACLE_HOME/wireless/bin/internal/portalRegistrar.bat

Specify the following arguments when running the `portalRegistrar` script:

- `admin_user` (typically `orcladmin`).
- `url` (for example, `http://lbr.abc.com`).

5.10 Changing the OracleAS Portal Schema Password

Refer to the chapter "Managing Administrative Passwords" in the *Oracle Application Server 10g Administrator's Guide* for information about changing the OracleAS Portal schema password.

> **Note:**   By default, an OracleAS Portal middle-tier is made up of one portal instance. Both the DAD name and the OracleAS Metadata Repository schema name for this instance are **portal**. You can only change the schema password of this default OracleAS Portal instance in the previously described way.

6

Securing OracleAS Portal

One of the most important aspects of any portal solution is security. The ability to control user access to Web content and to protect your site against people breaking into your system is critical. This chapter describes the architecture of OracleAS Portal security in the following topics:

- About OracleAS Portal Security

- Configuring OracleAS Security Framework for OracleAS Portal

- Configuring OracleAS Portal Security

> **See Also:**
>
> - *Oracle Application Server 10g Security Guide*
>
> - *Oracle Identity Management Concepts and Deployment Planning Guide*

6.1 About OracleAS Portal Security

The sections that follow provide an overview of OracleAS Portal security and how it works with the OracleAS Security Framework.

- OracleAS Portal Security Model

- Classes of Users and Their Privileges

- Resources Protected

- Authorization and Access Enforcement

- Leveraging Oracle Application Server Security Services

- Leveraging Oracle Identity Management Infrastructure

- Security for Portlets

- Securing the OmniPortlet and Simple Parameter Form

- Securing the Web Clipping Provider

- Securing the Federated Portal Adapter

- Securing OraDAV

6.1.1 OracleAS Portal Security Model

When you make content available on the Web, it is very likely that you need to restrict access to at least some parts of it. For example, it is unlikely that you want every user to be able to see every document on your site. It is even less likely that you want every user to be able to change every document on your site. OracleAS Portal provides a comprehensive security model that enables you to completely control what users can see and change on your Web site.

Before a user logs on to OracleAS Portal, they can only view the content that the content contributors designate as public. Public content can be viewed by any user who knows the URL of a portal object (for example, a page) and can connect to the machine where it is stored. The user sees only those aspects of the object that are designated as public, such as the public portlets. If the object has no public contents, then the user is denied access to it.

Once the user logs in to the portal, they may or may not be able to see and change content depending upon their access privileges. Typically, an authenticated user can see and do more in the portal than a public user. For example, an authenticated user might see items or portlets on the page that the public user cannot view. An authenticated user might also be able to add and edit content, and change properties, privileges that would typically be denied to a public user. In the portal, you can control access to objects (pages, items, or portlets) by user and group. That is, you might grant access privileges for a page to specific named users, user groups, or a combination of both.

To support this flexible approach to controlling access to Web content, OracleAS Portal leverages the other components of Oracle Application Server and Oracle9*i* Database Server to provide strong protection for your portal. OracleAS Portal interacts with all of the following components to implement its security model:

- Oracle Application Server Single Sign-On authenticates users, who are attempting to gain access to non-public areas of your portal.

- mod_osso is an Oracle HTTP Server module that redirects authentication requests to OracleAS Single Sign-On.

- OracleAS Web Cache is the cache used to serve up pages generated by OracleAS Portal (or proxied to the Oracle HTTP Server if not able to service the request). Based on invalidation caching, OracleAS Portal invalidates the cache when the underlying page/metadata changes.

- Oracle Internet Directory, Oracle's native LDAP version 3 service, acts as the repository for user credentials and group memberships.

- The Oracle Internet Directory's Oracle Delegated Administration Services (DAS) adds or updates the information stored inside the directory (users and groups).

- Oracle Directory Integration Platform notifies OracleAS Portal upon the occurrence of any directory events (for example, user deletions) to which OracleAS Portal subscribes. In essence, the directory integration server informs OracleAS Portal when a change occurs in the directory that requires a change in OracleAS Portal.

OracleAS Portal Architecture

Figure 6–1 illustrates the components and relationships of the OracleAS Portal security architecture.

Figure 6–1 OracleAS Portal Security Architecture
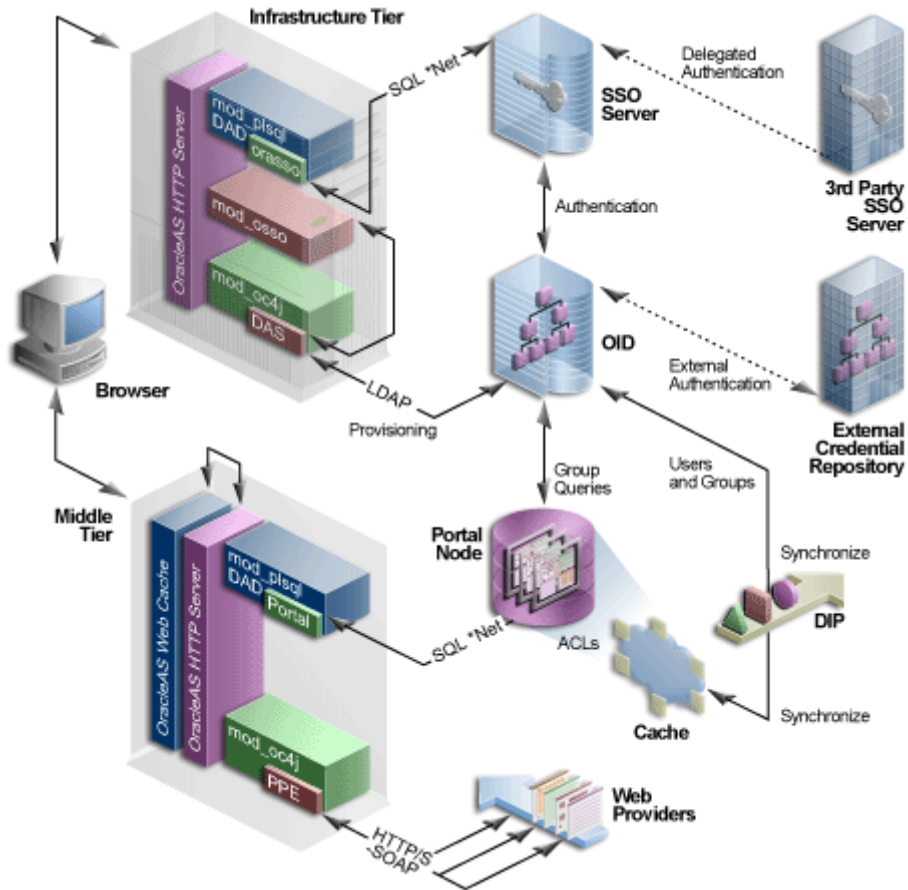
The OracleAS Portal architecture consists of three basic tiers, including the client browser, the middle-tier server, and the infrastructure servers and repositories. By default, Oracle Internet Directory and OracleAS Single Sign-On are installed on the same host as part of the infrastructure installation. This tier is subsequently used for the OracleAS Portal installation.

While the default installation has all three servers and repositories installed on the same host, we recommend that you install these functions on separate servers.

If you used releases prior to 9.0.2, notice that the middle and infrastructure tier components have changed. The DAD and mod_plsql combination still exists on the infrastructure tier but is now joined by DAS running in Oracle Application Server

Containers for J2EE (OC4J). Similarly, the Parallel Page Engine appears on the middle-tier also running in OC4J.

Furthermore, the OracleAS Single Sign-On model has expanded to include the addition of mod_osso, which allows any URL to participate in OracleAS Single Sign-On. Even if the application was not written as a partner application, mod_osso is now the recommended method to introduce third party applications into OracleAS Single Sign-On. Any application capable of reading HTTP headers may avail itself of the OracleAS Single Sign-On capability.

OracleAS Web Cache front ends these middle-tier components and thus optimizes the throughput of OracleAS Portal. When a page request comes from the browser, OracleAS Web Cache takes it. If possible, the page is delivered from the cache. If not, the request goes through to the origin Oracle HTTP Server.

As with Release 1.x, if the requested page is not a public page, the user is challenged for a username and password. This function is still carried out through a redirection to OracleAS Single Sign-On for authentication. (Note that the single sign-on DAD has been renamed to orasso for this release.)

Unlike Release 1.x, OracleAS Single Sign-On does not compare the user credentials against to its own schema tables. It verifies them on the Oracle Internet Directory through LDAP. The credentials are compared to those found within the Directory (LDAP compare) and the result returned to OracleAS Single Sign-On. Upon successful authentication, OracleAS Single Sign-On creates a single sign-on cookie.

Once the user is authenticated and an appropriate OracleAS Portal session created, she may access pages and other objects. It is necessary to determine which pages and objects the user has the necessary privileges to access. As in Release 1.x, the Access Control Lists for all portal objects are held in the OracleAS Portal Repository.

The difference in Release 2 is that all user and group membership information is stored in the Oracle Internet Directory. When a user first logs in to OracleAS Portal, the group memberships of the user are copied to the portal node and cached on that tier. This process allows for fast lookup of object privileges. Once the object and page privileges of the user are known, the Parallel Page Engine can go on to generate the page from the appropriate pieces.

In Release 2, all user provisioning is performed against the Oracle Internet Directory rather than the OracleAS Single Sign-On schema. The interface between the middle-tier and the LDAP server is the DAS servlet. The calls to the DAS servlet are protected by the mod_osso plug-in, which verifies that the user has been properly authenticated before providing access to the Oracle Internet Directory.

One important feature of the security architecture is the ability to keep the local cached group membership list synchronized with the Oracle Internet Directory. The Oracle Directory Integration Platform automatically keeps the locally cached information up to date with changes in the Oracle Internet Directory.

If you need to authenticate against an external repository, the Oracle Internet Directory performs this step rather than OracleAS Single Sign-On server as in Release 1.x. Just as the Oracle Directory Integration Platform keeps the local cache synchronized with the Oracle Internet Directory, it also keeps the Oracle Internet Directory synchronized with any external repository.

6.1.2 Classes of Users and Their Privileges

OracleAS Portal provides a number of user accounts and groups by default.

- OracleAS Portal Default, Seeded User Accounts
- OracleAS Portal Default, Seeded Groups

6.1.2.1 OracleAS Portal Default, Seeded User Accounts

Table 6–1 describes the user accounts created by default when OracleAS Portal is installed.

Table 6–1   Default OracleAS Portal Users

User	Description
PUBLIC	Is the user account that identifies unauthenticated access to the OracleAS Portal. Once a user logs in, the username changes from PUBLIC to the username by which the user authenticated herself/himself. When granting Portal privileges on individual objects that do not have an explicit checkbox for granting the object to Public, this user can be identified as the grantee of the privilege to grant access to it for unauthenticated users.
PORTAL	Is the super-user for the portal. In a standard installation, the user name is PORTAL. This user account has the highest privileges because it is granted all the global privileges available in the portal.
ORCLADMIN	Similar to portal, this account is granted the highest privileges in OracleAS Portal. This account is created for the Oracle Application Server administrators, and uses the password that is supplied during the Oracle Application Server installation.

Table 6–1   (Cont.) Default OracleAS Portal Users

User	Description
PORTAL_ADMIN	Is a privileged OracleAS Portal user account with administrative privileges excluding those that would give the user the ability to obtain higher privileges or perform any database operations. This user cannot edit any group or manage privileges on any schema or shared object. This account is typically intended for an administrator who manages pages and provisions user accounts.

6.1.2.2 OracleAS Portal Default, Seeded Groups

Table 6–2 describes the groups created by default when OracleAS Portal is installed.

Table 6–2   Default OracleAS Portal Groups

Group[1]	Description
AUTHENTICATED_USERS	Is the group that includes any authenticated, or logged in, user. The purpose of this group is to provide a means to assign the default privileges you want every logged in user to have in the portal. By default, this group is given the following privileges: ■ Create Group This group is a member of OracleDASCreateGroup.

Table 6–2 (Cont.) Default OracleAS Portal Groups

Group[1]	Description
DBA	Is a highly privileged group established for Oracle Application Server administrators. Components that are part of Oracle Application Server grant full component-specific privileges to members of this group.
	The DBA group is a member of the PORTAL_ADMINISTRATORS group.
	This group is also a member of the following Oracle Application Server privilege groups:
	■ OracleDASCreateUser
	■ OracleDASEditUser
	■ OracleDASDeleteUser
	■ OracleDASUserPriv
	■ OracleDASCreateGroup
	■ OracleDASEditGroup
	■ OracleDASDeleteGroup
	■ OracleDASGroupPriv
	■ OracleDASConfiguration

Table 6–2   (Cont.) Default OracleAS Portal Groups

Group[1]	Description
PORTAL_ADMINISTRATORS	Is a highly privileged group established for OracleAS Portal.
	By default, this group is given the following OracleAS Portal global privileges:
	■　Manage All Page Groups
	■　Manage All Pages
	■　Manage All Styles
	■　Manage All Providers
	■　Manage All Portlets
	■　Manage All Portal DB Providers
	■　Manage All Portal User Profiles
	■　Edit All Group Profiles
	■　Manage All Logs
	■　Execute All Transport Sets
	This group is a member of the following Oracle Application Server privilege groups:
	■　OracleDASCreateUser
	■　OracleDASEditUser
	■　OracleDASDeleteUser
	■　OracleDASCreateGroup
	■　OracleDASConfiguration
	Members of PORTAL_ADMINISTRATORS do not have the necessary privileges to administer OracleAS Single Sign-On. If you want members of this group to administer OracleAS Single Sign-On, then you must grant them those privileges as described in the *Oracle Application Server Single Sign-On Administrator's Guide*.
PORTLET_PUBLISHERS	Is a privileged group established for users who need to publish portlets to other users of the portal.
	By default, this group is given the following OracleAS Portal global privileges:
	■　Publish All Portlets

Table 6–2   (Cont.) Default OracleAS Portal Groups

Group[1]	Description
PORTAL_DEVELOPERS	Is a privileged group established for users who are building portlets.
	By default, this group is given the following OracleAS Portal global privileges:
	■ Create All Portal DB Providers
	■ Manage All Shared Components
	If you want PORTAL_DEVELOPERS to create database providers and portlets, you need to give this group privileges that enable them to modify schema, for example, Modify Data on all schemas. For more information, refer to Table 6–5.
RW_ADMINISTRATOR	Is the group of users who administer OracleAS Reports Services reports, printers, calendars, and servers.
	You must assign this group any desired object privileges (For example, Manage).
RW_DEVELOPER	Is the group of users who develop OracleAS Reports Services reports.
	You must assign this group any desired object privileges (For example, Execute or Manage).
RW_POWER_USER	Is the group of users who can modify OracleAS Reports Services reports.
	You must assign this group any desired object privileges (For example, Execute or Manage).
RW_BASIC_USER	Is the group of users who use OracleAS Reports Services reports.
	You must assign this group any desired object privileges (For example, Execute).

---

[1] All groups shown in this table are located in cn=<portal_group_container>,cn=Groups,dc=MyCompany,dc=com. Note that identity management realm name is determined by the domain name of the server on which the system is installed. For example, if the domain name of the server was oracle.com, the default identity management realm name would be dc=oracle,dc=com. If the domain name of the server could not be determined, Oracle Internet Directory defaults to the domain specified during installation by the administrator. The OracleDASxxxxx groups are Oracle Internet Directory privilege groups that reside under cn=groups,cn=OracleContext,dc=MyCompany,dc=com. These groups provide the privileges to perform operations in Oracle Internet Directory, such as creating or editing of users and groups, and their privileges.

6.1.3 Resources Protected

Within OracleAS Portal, you decide at what level of granularity you want to control access. You can assign privileges to any object on a per-user or per-group basis. For example, you can assign access privileges on a per-user basis for each and every item in your portal, but this approach creates considerable overhead for your content contributors.

If you want to lessen the burden on contributors, then you can assign privileges on a per-group basis at the page level and simply ensure that all of the items that you place on any given page have similar security requirements. With this approach, the security that items receive through the page that contains them is usually sufficient and content contributors only need to assign privileges for items that require higher security than the page.

> **See Also:** For information about how you might model privileges, see Section 6.1.6.9, "DAS Public Roles".

For information about how you might model privileges, refer to the white paper, *Strategies for Administering Privileges in OracleAS Portal Release 2*, on the Oracle Technology Network, http://otn.oracle.com.

6.1.3.1 Global Privileges

Use global privileges to give a user or group a certain level of privileges on all objects of a particular type.

> **Note:** Global privileges confer a great deal of power on the user to whom they are granted. As a result, they should be granted very cautiously and only to users or groups who truly require them. You should only have a small number of users with global privileges.

There are three types of privilege groups:

- Table 6–3, " Page Group Privileges"
- Table 6–4, " Portal DB Provider Privileges"
- Table 6–5, " Administration Privileges"

Table 6–3   Page Group Privileges

Object Type	Privileges
All Page Groups	**None:** No global page group privileges are granted.
	Manage All: Perform any task on any page group. This privilege supersedes any other privilege in the other global page group privileges. For example, this also allows managing of any page.
	Manage Classifications: Create, edit, and delete any category, perspective, custom attribute, custom page type, or custom item type in any page group.
	Manage Templates: Create, edit, and delete any page template in any page group. Grant access to any page template.
	Manage Styles: Create, edit, and delete any style in any page group.
	View: View any page in any page group.
	Create: Create page groups, and create any page group object in those page groups. Users or groups with these privileges can also edit and delete the page groups and page group objects they create. Note: These users cannot create any objects in the existing page groups.

Table 6–3   (Cont.) Page Group Privileges

Object Type	Privileges
All Pages	**None:** No global page privileges are granted.
	Manage: Create, edit, customize, or delete any page in any page group. Grant access to any page in any page group.
	Manage Content: Add, edit, hide, show, share, or delete any item, portlet, or tab on any page in any page group.
	Manage Items With Approval: Create new items on any page in any page group. These items are not published until approved through a specified approval process. Users or groups with these privileges can also edit the items they create. Users with these privileges cannot add portlets to a page.
	Manage Styles: Apply an available or new style to any page in any page group. Create, edit, and delete new styles. Note: Only allows editing of styles created by user (cannot modify or delete other user's styles).
	Customize Portlets (Full): Customize any page in any page group to add, show, hide, delete, move, or rearrange portlets. Customize any page to show, hide, delete, or rearrange tabs, or add tabs to existing tabbed regions. Customize any page in any page group to use a different style.
	Customize Portlets (Add-only): Customize any page in any page group to add portlets or add tabs to existing tabbed regions. Users or groups with these privileges can also delete the portlets they add. Customize any page in any page group to use a different style.
	Customize Portlets (Hide-Show): Customize any page in any page group to show or hide portlets or tabs. Customize any page in any page group to use a different style. Arrange portlets in any page in any page group.
	Customize (Style): Customize any page in any page group to use a different style.
	View: View any page in any page group.
	Create: Create sub-pages in any page group. Users or groups with these privileges can also edit and delete the sub-pages they create. Note: You must have Manage privileges on the root page in a page group in which you want to create the pages.

Table 6–3 (Cont.) Page Group Privileges

Object Type	Privileges
All Styles	**None:** No global style privileges are granted.
	Manage: Create, edit, and delete any style in any page group.
	View: View any style in any page group.
	Publish: Make any style in any page group public for other users to use.
	Create: Create styles in any page group. Users or groups with these privileges can also edit and delete the styles they create.
All Providers	**None:** No global provider privileges are granted.
	Manage: Register, edit, and deregister any provider, as well as display and refresh the Portlet Repository. Also allowed to grant edit abilities on any provider.
	Edit: Edit any registered provider.
	Publish: Register and deregister any provider.
	Execute: View the contents of any provider.
	Create: Register portlet providers. On the provider the user (or group) creates, the user gets a Manage privilege. Thus, he can do all the operations (including edit and deregister) on the particular provider that he has created.
All Portlets	**None:** No global portlet privileges are granted.
	Manage: Create, edit, or delete any portlet in any provider.
	Edit: Edit any portlet in any provider.
	Execute: Execute any portlet in any provider. Users or groups with these privileges can see all portlets even if the portlet security is enforced. The Show link appears in the Navigator for all portlets.
	Access: View any portlet in any provider.
	Publish: Publish any page, navigation page, or Portal DB provider portlet to the portal, making it available for adding to pages.

Table 6–4   Portal DB Provider Privileges

Object Type	Privileges
All Portal DB Providers	**None:** No global application privileges are granted.
	Manage: Edit, delete, or export any Portal DB provider. Create, edit, delete, or export any portlet in any Portal DB provider. Grant access to any Portal DB provider and any portlet in any Portal DB provider.
	Edit Contents: Edit or export any portlet in any Portal DB provider.
	View Source: View the package specification and body and run any portlet in any Portal DB provider. Intended primarily for users or groups who may want to look at a portlet's source so they know how to call it.
	Customize: Run and customize any portlet in any Portal DB provider.
	Run: Run any portlet in any Portal DB provider.
	Create: Create Portal DB providers. Users or groups with these privileges can edit, delete, and export the providers they create and create, edit, delete, and export any portlet in them.
All Shared Components	**None:** No global shared component privileges are granted.
	Manage: Create, view, copy, edit, delete, and export any shared component in any Portal DB provider. View and copy any system shared component. Grant access to any non-system shared component.
	Create: Create shared components in any Portal DB provider. View and copy any system shared component. View any shared component. Users and groups with these privileges can view, copy, edit, delete, and export the shared components they create.

Table 6–5   Administration Privileges

Object Type	Privileges
All User Profiles	**None:** No global user profile privileges are granted.
	Manage: Edit any user profile. Grant this privilege to other users and groups.
	Edit: Edit any user profile.

Table 6–5   (Cont.) Administration Privileges

Object Type	Privileges
All Group Privileges (profiles)	**None:** No global group profile privileges are granted.
	Manage: Edit any group profile. Grant this privilege to other groups. The Privileges tab of the group profile allows the user to assign those privileges to the group. The Manage privilege provides the edit privilege and the ability to grant it to others.
	Edit: Edit any portal group profile (setting the default home page and default mobile home page). Note: The ability to change any group's description, memberships, and owners is controlled by the Oracle Internet Directory access control policies, which are administered through membership in the OracleDASEditGroup group.
All Schemas	**None:** No global schema privileges are granted.
	Manage: Create, edit, and drop any schema. Grant access to any schema. Create, edit, drop, and rename any database object in any schema. Query, update, delete, and insert data in any table or view in any schema. Compile any function, procedure, package, or view in any schema. Execute any function, procedure, or package in any schema. Grant access to any database object in any schema.
	Modify Data: Create schemas. Query, update, delete, and insert data in any table or view in any schema. Compile any function, procedure, package, or view in any schema. Execute any function, procedure, or package in any schema. Users or groups with these privileges can edit, drop, and grant access to the schemas they create.
	Insert Data: Create schemas. Query and insert data in any table or view in any schema. Users or groups with these privileges can edit, drop, and grant access to the schemas they create.
	View Data: Create schemas. Query data in any table or view in any schema. Users or groups with these privileges can edit, drop, and grant access to the schemas they create.
	Create: Create schemas. Users with these privileges can also edit, drop, and grant access to the schemas they create. Note: If you want a user or group to access the **Schemas** portlet on the **Administer Database** tab of the **Builder** page, either make the user or group a member of the DBA group, or explicitly grant the user or group View privileges on the **Administer Database** tab. If you do not grant these privileges, the user or group will still be able to use the Navigator to access schemas.

Table 6–5 (Cont.) Administration Privileges

Object Type	Privileges
All Logs	**None:** No global log privileges are granted.
	Manage: Edit or purge any log. Grant this privilege to others.
	Edit: Edit or purge any log.
	View: View any log.
All Transport Sets	**None:** No global transport set privileges are granted.
	Execute: Export/Import objects that are not shared.
	Manage: Edit or purge any import or export sets. Grant this privilege to others.

6.1.3.2 Object Privileges

You can assign access privileges to users or groups for all of the following objects within OracleAS Portal through the **Access** tab of the object's Edit Page:

Table 6–6 OracleAS Portal Objects with Privilege Control

Type of Object	Available Privileges	Inherited Privileges
Calendar	■ Manage	From Database Provider
	■ View	
	■ Customize	
	■ Execute	
Chart (based on SQL query)	■ Manage	From Database Provider
	■ Edit	
	■ View	
	■ Customize	
	■ Execute	
Chart (based on wizard)	■ Manage	From Database Provider
	■ Edit	
	■ View	
	■ Customize	
	■ Execute	

Table 6–6   (Cont.) OracleAS Portal Objects with Privilege Control

Type of Object	Available Privileges	Inherited Privileges
Data Component	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
Data Component Cell	■ Edit ■ View	From Data Component
Database Provider	■ Manage ■ Edit ■ View Source ■ Customize ■ Execute	Not applicable
Document	■ Own ■ Manage ■ View Only	From page or item
Dynamic Page Component	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
Form[1]	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
Frame Driver	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider

Table 6–6   (Cont.) OracleAS Portal Objects with Privilege Control

Type of Object	Available Privileges	Inherited Privileges
Hierarchy	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
Image Chart	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
Link	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
List of Values	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
Menu	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
OracleAS Reports Services printer	■ Manage ■ Edit ■ View ■ Execute	From Database Provider

Table 6–6   (Cont.) OracleAS Portal Objects with Privilege Control

Type of Object	Available Privileges	Inherited Privileges
OracleAS Reports Services report	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
OracleAS Reports Services Server	■ Manage ■ Edit ■ View ■ Execute	From Database Provider
Page	■ Manage ■ Manage Content ■ Manage Items With Approval ■ Manage Style ■ Customize Portlets (Full) ■ Customize Portlets (Add-Only) ■ Customize Portlets (Hide-Show) ■ Customization (Style) ■ View	Not applicable
Page group	■ Manage All ■ Manage Classifications ■ Manage Templates ■ Manage Styles ■ View	Not applicable
Page Item	■ Own ■ Manage ■ View Only	From page

Table 6–6   (Cont.) OracleAS Portal Objects with Privilege Control

Type of Object	Available Privileges	Inherited Privileges
Portlet	■ Manage ■ Edit ■ Execute ■ Access ■ Publish	Not applicable
Provider	■ Manage ■ Edit ■ Publish ■ Execute	Not applicable
Query by example form	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
Report[2]	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider
Schema	■ Manage ■ Modify ■ Insert ■ View	Not applicable
URL	■ Manage ■ Edit ■ View ■ Customize ■ Execute	From Database Provider

Table 6–6 (Cont.) OracleAS Portal Objects with Privilege Control

Type of Object	Available Privileges	Inherited Privileges
XML	- Manage - Edit - View - Customize - Execute	From Database Provider

[1] You can have many different types of forms (stored procedure or table based, version 2 or version 3 based, and master-detail), but all of these types have the same available privileges and privilege inheritance.

[2] You can have two different types of reports (SQL and table based), but all of these types have the same available privileges and privilege inheritance.

6.1.3.3 Privileges to Create and Edit Web Providers and Provider Groups

To create and manage Web providers and provider groups through the user interface, as opposed to working with files directly, you need to grant appropriate privileges to the administrative users. The access control list is implemented differently than for the OracleAS Portal repository resident objects described in Section 6.1.3.1, "Global Privileges" and Section 6.1.3.2, "Object Privileges". Rather, the grants for provider privileges are maintained in an XML file.

> **Note:** The privileges described here are for users developing new Web providers and pertain to authorizations that are enforced by the provider user interface. These privileges are not required to register Web providers.

To grant privileges to create, edit, and delete Web providers or provider groups, you need to manually make changes to the following file:

```
MID_TIER_ORACLE_HOME/j2ee/OC4J_
Portal/applications/portalTools/providerBuilder/WEB-INF/deployment_
providerui/provideruiacls.xml
```

An example of this file follows:

> **Note:** In this example, the user names `any_provider_manage_` `user`, `any_provider_edit_user`, and so on, are just sample user names used here to illustrate the privilege codes that correspond to the privileges implied by the corresponding user names. An actual user grant would have the OracleAS Single Sign-On user name as the value of the `name` attribute in the `<user>` element, and the privilege would be populated with the appropriate privilege code.

```
<providerui xmlns="http://www.oracle.com/portal/providerui/1.0">
    <objectType name="ALL_OBJECTS">
        <object name="ANY_PROVIDER" owner="providerui">
            <user name="any_provider_manager_user" privilege="500"/>
            <user name="any_provider_edit_user" privilege="400"/>
            <user name="any_provider_execute_user" privilege="300"/>
            <user name="any_provider_create_user" privilege="100"/>
        </object>
        <object name="ANY_PORTLET" owner="providerui">
            <user name="any_portlet_manage_user" privilege="500"/>
            <user name="any_portlet_edit_user" privilege="400"/>
            <user name="any_portlet_execute_user" privilege="300"/>
        </object>
    </objectType>
    <objectType name="PROVIDER">
        <object name="TEST_PROVIDER" owner="providerui">
            <user name="provider_manage_user" privilege="500"/>
            <user name="provider_edit_user" privilege="400"/>
            <user name="provider_execute_user" privilege="300"/>
        </object>
    </objectType>
    <objectType name="PORTLET">
        <object name="PORTLET_UNDER_TEST_PROVIDER" owner="TESTPROVIDER">
            <user name="portlet_manage_user" privilege="500"/>
            <user name="portlet_edit_user" privilege="400"/>
            <user name="portlet_execute_user" privilege="300"/>
        </object>
    </objectType>
</providerui>
```

This file allows for granting of the following types of privileges, described in the following sections:

- Global Privileges

- Object Level Privileges

6.1.3.3.1 Global Privileges Table 6–7 describes the global object types and corresponding privilege codes that can be granted to users in the `provideruiacls.xml` file. When granting a privilege to the user, you should specify the numeric privilege code.

Table 6–7   Global Privilege Codes for provideruiacls.xml

Type of Object	Available Privileges
ANY_PROVIDER	**500 (Manage):** Can create/edit/delete/open any provider or provider group and portlets under them.
	400 (Edit): Can create/edit any provider or provider group and execute the portlets under them.
	300 (Execute): Can open any provider or provider group and execute the portlets under them.
	100 (Create): Can create any provider or provider group.
ANY_PORTLET	**500 (Manage):** Can edit/delete/execute any portlet under any provider.
	400 (Edit): Can edit/execute any portlet under any provider.
	300 (Execute): Can execute any portlet under any provider.

To add a privilege to a particular user, add an entry in the proper object type container, for example:

```
<objectType name="ALL_OBJECTS">
    <object name="ANY_PROVIDER" owner="providerui">
       <user name="jdoe" privilege="400"/>
         …
    </object>
</objectType>
```

For these global privileges, the `objectType` name is set to ALL_OBJECTS, the object owner is set to `providerui`, and the object name should be ANY_PROVIDER or ANY_PORTLET depending on the type of grant you are setting.

You then set the user name and privilege to the values corresponding to the OracleAS Single Sign-On username of the grantee and the privilege code you wish to assign. This model does not support any grants to groups. It only supports grants directly to users.

6.1.3.3.2 Object Level Privileges Table 6–8 describes the object level privileges that can be granted to users to give them privileges on specific object instances as referenced within the provideruiacl.xml XML file.

Table 6–8   Object Privilege Codes for provideruiacl.xml

Type of Object	Available Privileges
PROVIDER	**500 (Manage):** Can edit/delete/open the specified provider or provider group and the portlets under it.
	400 (Edit): Can edit the specified provider or provider group and execute the portlets under it.
	300 (Execute): Can open the specified provider or provider group and execute the portlets under it.
PORTLET	**500 (Manage):** can edit/delete/execute the specified portlet under the specified provider.
	400 (Edit): Can edit/execute the specified portlet under the specified provider.
	300 (Execute): Can execute the specified portlet under the specified provider.

To add a privilege to a particular user, add an entry into the proper object type container, for example:

```
<objectType name="PORTLET">
    <object name="PORTLET_UNDER_TEST_PROVIDER" owner="TESTPROVIDER">
       <user name="jdoe" privilege="400"/>
        …
    </object>
</objectType>
```

For the object level privileges, the objectType name is set to PROVIDER or PORTLET, depending upon to which object instances you are providing access. The object name is set to the provider name or the portlet name, respectively. The object owner is set to proviederui or the name of the associated provider, again respectively for providers and portlets.

Table 6–9 summarizes these rules:

Table 6–9   Attribute values for Providers and Portlets

Attribute	Provider Instance Grant	Portlet Instance Grant
ObjectType name	`PROVIDER`	`PORTLET`
Object name	Provider or provider group name	Portlet name
Object owner	`providerui`	Provider name
User name	OracleAS Single Sign-On user name	OracleAS Single Sign-On user name
User privilege	Privilege code	Privilege code

6.1.3.4 Privileges to Create/Edit URL/XML Portlets in the Portlet Repository

To create and edit URL and XML portlets in the Portlet Repository, privileges need to be granted to the users. The URL and XML portlets are available from the Portlet Builders page in the Portlet Repository. To grant access, you need to manually make changes to following file:

```
MID_TIER_ORACLE_HOME/j2ee/OC4J_Portal/applications/jpdk/jpdk/WEB-INF/
deployment_providerui/provideruiacls.xml
```

The privileges are identical to those described earlier in Section 6.1.3.3, "Privileges to Create and Edit Web Providers and Provider Groups".

6.1.4 Authorization and Access Enforcement

When users attempt to log in to OracleAS Portal, OracleAS Single Sign-On must first verify their credentials against the directory. Once their identity has been verified, OracleAS Portal checks their access privileges in the directory to determine which objects they may see and use within the portal.

1. From OracleAS Portal, the user requests to log in by clicking the Login link.

2. The login request is forwarded to OracleAS Single Sign-On for authentication.

3. OracleAS Single Sign-On verifies the user credentials against the information stored in the directory.

4. If authentication is successful, OracleAS Single Sign-On creates an SSO cookie for the user. If authentication is not successful, the user is denied access and returned to the login page to re-enter their user name and password.

5. Once the user's identity has been verified, control is returned to OracleAS Portal, which creates a portal session cookie. OracleAS Portal then connects to the directory and determines the user's group memberships and privileges.

6. OracleAS Portal caches the user's membership and privilege information locally for the duration of their session.

7. When the user attempts to access a page, OracleAS Portal performs the following checks:

   - Checks whether the page is public. If so, the user can view it.

   - If the page is not public, OracleAS Portal checks the local privilege table to determine whether the current user has privileges to view the page. If the user has viewing privileges, the user can view it.

   - If the current user does not have direct viewing privileges on the page, OracleAS Portal checks the cached membership information and privilege table to determine whether any of the groups to which the user belongs has privileges to view the page. If one of the groups to which the user belongs has viewing privileges on the page, the user can view it.

   > **Note:** If changes are made to Oracle Internet Directory that affect the user's privileges, a notification is raised and the cached information about the user is invalidated. Thus, OracleAS Portal starts enforcing the user's updated privileges as soon as it receives the notification.

6.1.5 Leveraging Oracle Application Server Security Services

OracleAS Portal leverages Oracle Application Server Security Services in the following ways:

- **SSL encryption.** The use of HTTPS and the Secure Sockets Layer (SSL) allows for the creation of a secured connection between a client and a server. Digital certificates on each end of the communication verify the validity of the server and encryption of the communication to ensure that it is not compromised. You can implement SSL encryption for OracleAS Portal through the Oracle Application Server Security Services.

- **JAZN**. JAZN is the internal name for a Java Authentication and Authorization Service (JAAS) provider. JAAS is a Java package that enables applications to authenticate and enforce access controls upon users. The use of JAZN in OracleAS Portal is limited to the authentication of external JSPs.

> **See Also:** For more information:
>
> - Section 6.3.2.1, "Configuring SSL for OracleAS Portal"
> - Section F.2, "Setting Up a JAZN File for External Communication"
> - *Oracle Application Server Containers for J2EE Services Guide*

6.1.6 Leveraging Oracle Identity Management Infrastructure

To provide a more comprehensive security solution, OracleAS Portal takes advantage of a variety of components in the Oracle Identity Management infrastructure:

- Relationship Between OracleAS Portal and OracleAS Single Sign-On
- Relationship Between OracleAS Portal and Oracle Internet Directory
- Relationship Between OracleAS Portal and Oracle Internet Directory
- Relationship Between OracleAS Portal and DAS

OracleAS Portal also takes advantage of Oracle Identity Management when it creates users and groups. The most common way to create users and groups, and set global privileges and preferences for your portal is through the following portlets:

- User Portlet
- Portal User Profile Portlet
- Group Portlet
- Portal Group Profile Portlet

> **See Also:**
>
> - *Oracle Identity Management Concepts and Deployment Planning Guide*
> - *Oracle Internet Directory Application Developer's Guide*

6.1.6.1 Relationship Between OracleAS Portal and OracleAS Single Sign-On

OracleAS Portal uses OracleAS Single Sign-On for user authentication, as discussed in Section 6.1.4, "Authorization and Access Enforcement".

> **Note:** OracleAS Portal Release 3.0.9.8.4 or later can be used with
> OracleAS Single Sign-On Release 9.0.[1] You cannot use versions
> older than Release 3.0.9.8.4 with OracleAS Single Sign-On 9.0.

---

[1] Refers to the release of OracleAS Single Sign-On that ships with Oracle
   Application Server Release 2.

OracleAS Single Sign-On manages the Single Sign-On sessions of users. In order for
single sign-on security to function properly with OracleAS Portal, the following
tasks must be completed:

- Add OracleAS Portal as a partner application for OracleAS Single Sign-On.

- Add OracleAS Portal entries to the partner application enabler configuration
  table.

The Oracle Universal Installer performs these two configuration steps for you upon
installation. If you need to make changes to your configuration after installation,
you can do so by:

- Using the Application Server Control, as described in Section 7.1, "Using the
  Application Server Control", or by using the Portal Dependency Settings tool,
  described in Appendix A, "Using the Portal Dependency Settings File".

- Invoking the Oracle Portal Configuration Assistant, `ptlasst.csh` (UNIX) or
  `ptlasst.bat` (MS Windows) with `-mode MIDTIER -type SSO`. This
  procedure adds OracleAS Portal as a partner application to an existing
  OracleAS Single Sign-On installation. To work correctly, you must already have
  installed OracleAS Portal and OracleAS Single Sign-On, and created their
  DADs.

  The `ptlasst` scripts and their documentation are located in *MID_TIER_
  ORACLE_HOME*/assistants.

  > **See Also:** Appendix B, "Using the OracleAS Portal Configuration
  > Assistant Command Line Utility".

6.1.6.2 Relationship Between OracleAS Portal and Oracle Internet Directory

Oracle Internet Directory is Oracle's highly scalable, native LDAP version 3 service
and hosts the Oracle common user identity. As stated in the previous section,
OracleAS Portal queries the directory to determine a user's privileges and what
they are entitled to see and do in the portal. In particular, OracleAS Portal retrieves

the group memberships of the user from the directory to determine what they may access and change.

Given this model, OracleAS Portal requires the following interactions with Oracle Internet Directory:

- OracleAS Portal specific entries stored in the directory

- Group attributes stored in the directory

- User attributes stored in the directory

- Caching of user and group information from the directory

- Populating user and group lists of values from the directory through DAS

6.1.6.2.1 Directory Entries in Oracle Internet Directory for OracleAS Portal In order for security to function properly, OracleAS Portal requires the following entries in the directory's Directory Information Tree (DIT) structure:

- **Default user accounts** (cn=PUBLIC, cn=PORTAL, cn=PORTAL_ADMIN) are created in the identity management realm's user base (cn=Users,dc=MyCompany,dc=com[1]). The PORTAL and PORTAL_ADMIN users are added to the DBA and PORTAL_ADMINISTRATORS groups, respectively. The PUBLIC user is created for unauthenticated users. Typically, the PUBLIC user entry is for granting viewing privileges on portal content that is accessible to any user, unrestricted.

- **Group container** (Release 9.0.4: cn=*schema_name.yymmdd.hhmi* or Release 9.0.2.6: cn=portal.iasdb.server.mycompany.com) is created within the identity management realm's group base (cn=Groups,dc=MyCompany,dc=com[1]). OracleAS Portal can leverage any group in the directory, but groups are more easily accessed for display in a list of values if they are located within the OracleAS Portal group container.

  For Release 10*g* (9.0.4) of OracleAS Portal, the name of the group container is derived from the following in OracleAS Portal:

  - Portal schema name

  - Date and time when OracleAS Portal began to use Infrastructure Services

---

[1] The default identity management realm name is determined by the domain name of the server on which the system is installed. For example, if the domain name server was oracle, the default identity management realm name would be dc=oracle,dc=com. If the domain name server cannot be determined, the default name assigned by the directory is dc=Default Company,dc=com

The format of the name is:

schema_name.yymmdd.hhmi

For Release 9.0.2.6 of OracleAS Portal, the name of the group container is derived from the following in OracleAS Portal:

- Portal schema name
- Database SID
- Database server hostname

For example, if the schema name is PORTAL, SID is iasdb, and the hostname is host1.abc.com then the name of the group container is cn=PORTAL.iasdb.host1.abc.com.

- **Groups** are created within the OracleAS Portal group container in the directory:
  - cn=AUTHENTICATED_USERS
  - cn=DBA
  - cn=PORTAL_ADMINISTRATORS
  - cn=PORTAL_DEVELOPERS
  - cn=PORTLET_PUBLISHERS
  - cn=RW_ADMINISTRATOR
  - cn=RW_DEVELOPER
  - cn=RW_POWER_USER
  - cn=RW_BASIC_USER

- **Application entity** (orclApplicationCommonName=*application_name*) is created in the root Oracle Context (cn=Portal,cn=Products,cn=OracleContext). OracleAS Portal uses this entity to bind to the directory when it needs to query it or perform actions against it (for example, adding a user) on behalf of the user. When OracleAS Portal binds to the directory for a user, it uses a proxy connection to connect as the user. This method ensures that the directory properly enforces the user's authorization restrictions. The OracleAS Portal application entity obtains the privileges to initiate proxy connections by its membership in the user proxy privileges group (cn=UserProxyPrivilege,cn=Groups,cn=OracleContext). In Release 9.0.4, the name of the application entity is derived from the schema and the time that OracleAS Portal began to use the Infrastructure Services. In Release 9.0.2.6, the

name of the application entity is derived from the schema, SID, and the hostname. For example, if the schema name is PORTAL, SID is iasdb, and the hostname is host1.abc.com then the name of the application name is orclApplicationCommonName=PORTAL.iasdb.host1.abc.com.

- **Directory synchronization subscription** A provisioning profile entry is created in the provisioning profile of the directory (cn=Provisioning Profiles,cn=changelog subscriber,cn=oracle internet directory). This entry indicates that the directory must notify OracleAS Portal when user or group privilege information has changed. It enables OracleAS Portal to keep its authorizations synchronized with the information stored in the directory.

Figure 6–2 shows where the OracleAS Portal information is located in the directory's DIT structure.

Figure 6–2   OracleAS Portal DIT Structure
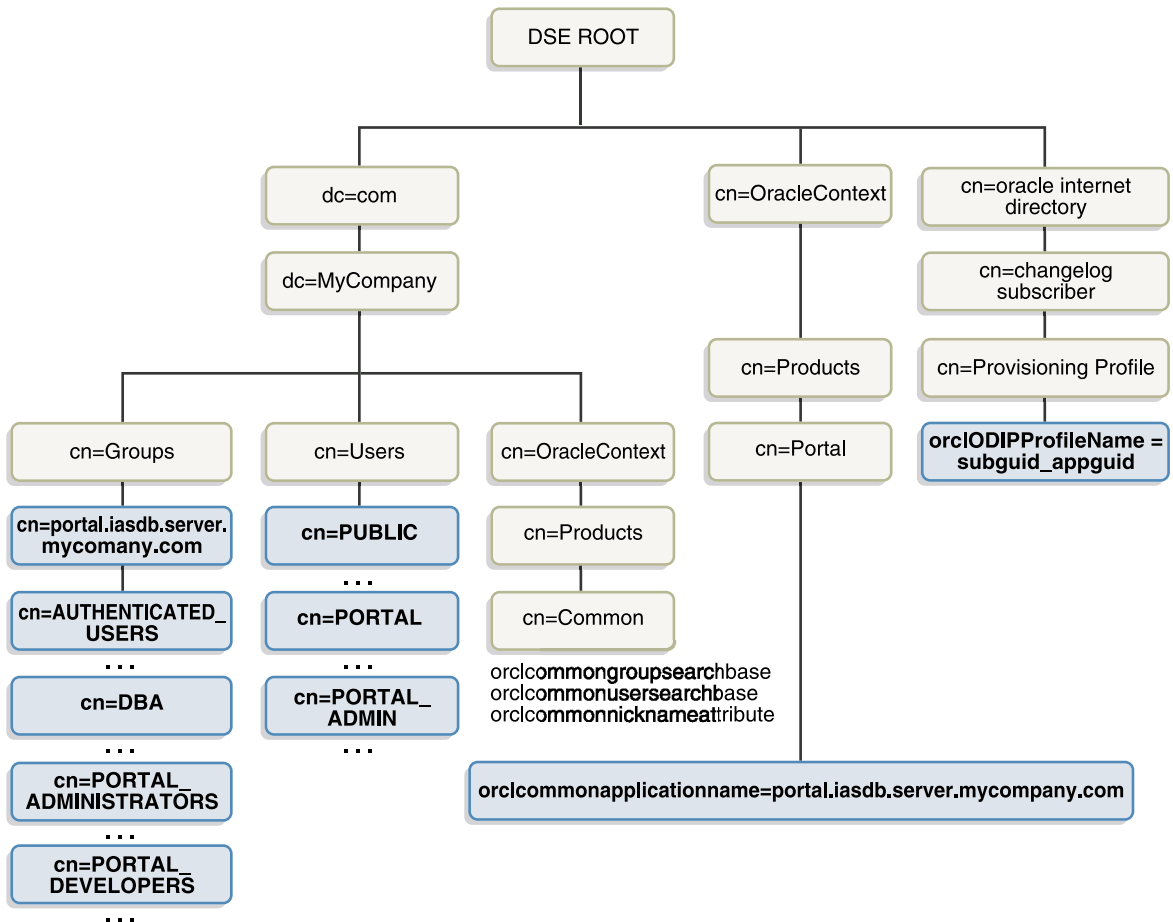
6.1.6.2.2   User Attributes Stored in Oracle Internet Directory OracleAS Portal, like all other components of Oracle Application Server, relies upon the directory to store user information. All users in the directory are defined using the following object classes:

- The inetOrgPerson object class contains the entire user attributes defined by the Internet Engineering Task Force (IETF) Request for Comments (RFC) number 2798.

- The orclUser and orclUserV2 object classes contain a set of standard, additional attributes for Oracle products.

The subsequent tables show the various user attributes stored in Oracle Internet Directory. A complete list of these attributes is available in IETF RFC 2798.

Table 6–10   inetOrgPerson Attributes

inetOrgPerson (IETF) attributes	Comment
cn	The common name of the user. This attribute is mandatory.
employeeNumber	Number used to identify employees
sn	Last name. This attribute is mandatory. If nothing is explicitly specified for this attribute, the user's nickname is used.
givenName	First name
middleName	
displayName	Preferred name
mail	e-mail address
telephoneNumber	
homePhone	
mobile	
pager	
facsimileTelephoneNumber	
street	
l	City of office
st	State of office
postalCode	Postal code of office
c	Country of office
homePostalAddress	Home address
jpegPhoto	Person's picture
o	Organization
title	

Table 6–10   (Cont.) inetOrgPerson Attributes

inetOrgPerson (IETF) attributes	Comment
manager	Employee's supervisor
uid	User ID
userPassword	
preferredLanguage	

Table 6–11   orclUserV2 Attributes

orclUserv2 attributes	Comments
orclIsVisible	A flag to indicate whether the user should be hidden from all but administrators.
orclDisplayPersonalInfo	A flag to indicate whether a user's personal information should be hidden from all but administrators.
orclMaidenName	
orclDateOfBirth	
orclHireDate	
orclDefaultProfileGroup	Default user group for the person
orclActiveStartDate	When account was activated
orclActiveEndDate	when account was (or will be) terminated
orclTimeZone	
orclIsEnabled	A flag to indicate whether the user account is active. If not active, the user will not be allowed to log in.

For users who are familiar with the user properties from previous versions of OracleAS Portal, the following table maps the old user properties to the new Oracle Internet Directory attributes.

Table 6–12   Mapping of OracleAS Portal User Properties to Oracle Internet Directory

Previous OracleAS Portal user property	inetOrgPerson or orclUserv2 attributes
ID	Not applicable because ID remains a local OracleAS Portal attribute that is linked to the corresponding directory entry by means of a globally unique identifier.
EMPNO	employeeNumber
LAST_NAME	sn
FIRST_NAME	givenName
MIDDLE_NAME	middleName
KNOWN_AS	displayName
EMAIL	mail
WORK_PHONE	telephoneNumber
HOME_PHONE	homePhone
MOBILE_PHONE	mobile
PAGER	pager
FAX	facsimileTelephoneNumber
OFFICE_ADDR(1,2,3)	street
OFFICE_CITY	l
OFFICE_STATE	st
OFFICE_ZIP	postalCode
OFFICE_COUNTRY	c
HOME_ADDR[1,2,3],CITY, STATE,ZIP,COUNTRY	homePostalAddress
IMAGE	jpegPhoto
ORGANIZATION	o
TITLE	title
MANAGER	manager
PASSWORD	userPassword

Table 6–12   (Cont.) Mapping of OracleAS Portal User Properties to Oracle Internet

Previous OracleAS Portal user property	inetOrgPerson or orclUserv2 attributes
DISPLAY	orclIsVisible
DISPLAY_PERSONAL_INFO	orclDisplayPersonalInfo
NOTIFICATION_PREFERENCE	orclWorkflowNotificationPref
USER_NAME	orclCommonNickNameAttribute, which is the nickname used in place of the user's full Dn. The full Dn attribute is quite long (cn=name,dc=domain,dc=com), hence it is simpler for users to log in with this nickname. For more information, refer to the documentation on Oracle Internet Directory.
MAIDEN_NAME	orclMaidenName
DATE_OF_BIRTH	orclDateOfBirth
HIREDATE	orclHireDate
SUBSCRIBER_ID	Not applicable because the identity management realm identifier is obtained from the user's identity management realm node.
DEFAULT_GROUP	orclDefaultProfileGroup

6.1.6.2.3   Group Attributes Stored in Oracle Internet Directory OracleAS Portal, like all other components of Oracle Application Server, relies upon the directory to store group information. All groups in the directory are defined using the following object classes:

- The `groupOfUniqueNames` object class contains all of the group attributes defined by IETF (RFC 2256).

- The orclGroup object class contains a set of standard, additional attributes for OracleAS Portal.

> **Note:**   Unlike OracleAS Portal Release 3.x, you cannot scope groups in OracleAS Portal 9.x to a specific page group.

Figure 6–3 shows where the OracleAS Portal information for groups is located in the directory's DIT structure.

Figure 6–3   DIT Structure for OracleAS Portal Groups
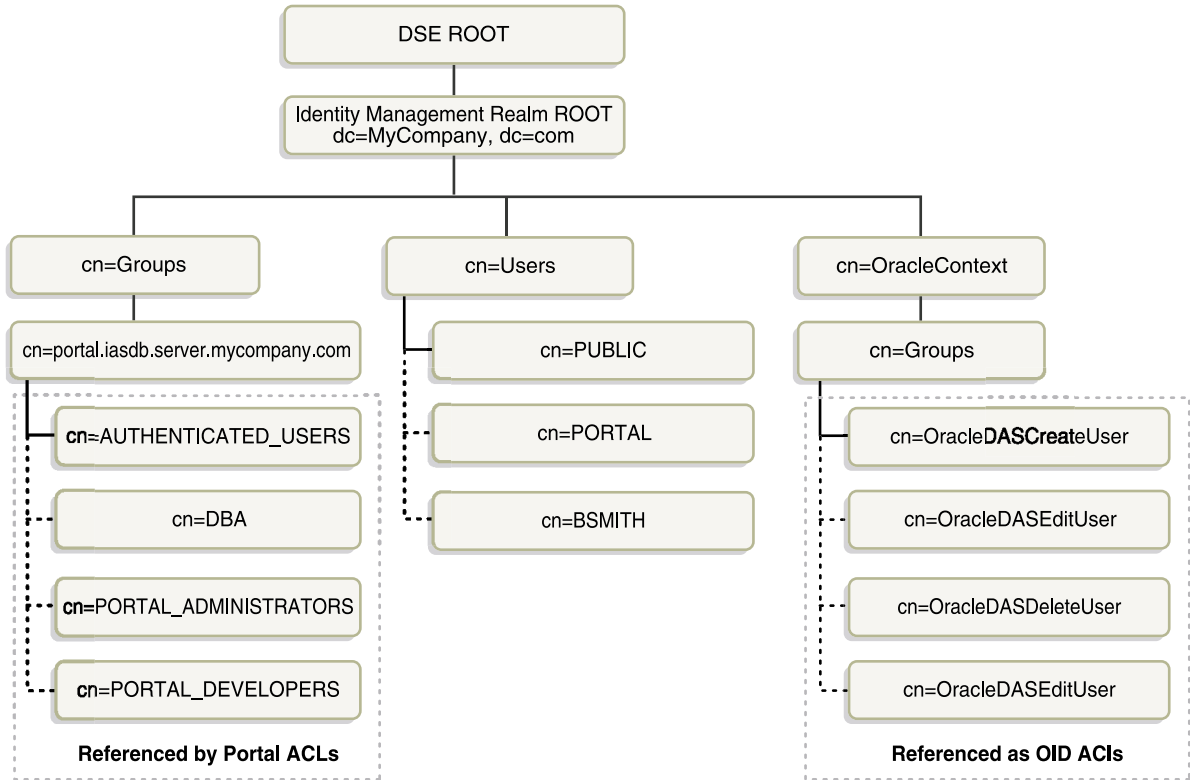
The subsequent tables show the various group attributes stored in Oracle Internet Directory:

Table 6–13   groupOfUniqueNames/groupOfNames Attributes

groupOfUniqueNames/groupOf Names (IETF) attributes	Comment
cn	The common name of the group, which can be typed into places like the Edit Group field in the Group portlet to locate the group.
description	The text description of the group, which is displayed in lists of values where the group appears.
uniqueMember	A list of the distinguished names (DNs) of all of the members of the group. The member DNs can represent a user or another group.
owner	A list of the DNs of all of the users and groups that have the privilege of administering this group.

Table 6–14   orclGroup Attributes

orclGroup attributes	Comment
orclGUID	The globally unique identifier (GUID) for this group.
orclIsVisible	A flag to indicate whether the group is public or private. Private groups only appear in lists of values for their owners. Other users cannot see them.

For users who are familiar with the group properties from previous versions of OracleAS Portal, the following table maps the old user properties to the new Oracle Internet Directory attributes:

Table 6–15   Mapping of OracleAS Portal Group Properties to Oracle Internet Directory

Previous OracleAS Portal group property	groupOfUniqueNames or orclGroup attribute
ID	local ID for the group, which can be matched to the orclGUID in the directory by a new locally stored orclGUID.

Table 6–15   (Cont.) Mapping of OracleAS Portal Group Properties to Oracle Internet

Previous OracleAS Portal group property	groupOfUniqueNames or orclGroup attribute
HIDDEN_GROUP	orclIsVisible
SUBSCRIBER_ID	Subscriber id is no longer needed because the location of the group entry under an identity management realm base indicates the identity management realm.
NAME	Cn
DESCRIPTION	Description
group membership	uniqueMember
OWNER	Owner

6.1.6.2.4   Oracle Internet Directory Cache in OracleAS Portal To improve performance, OracleAS Portal caches some directory information locally. In particular, OracleAS Portal caches the following:

- Directory connection information for OracleAS Portal

- URLs for DAS

- orclGUIDs of certain privilege groups for authorization checks on directory portlets (for example, the User and Group portlets)

- some Oracle Context information

- the locally selected group search and creation bases

- group memberships and default group for each user

The majority of the information cached by OracleAS Portal is fairly static (for example, directory connection information). For those items that are more dynamic, such as group memberships and default group, OracleAS Portal relies upon the Directory Synchronized Provisioning agent for updates. OracleAS Portal maintains a directory synchronization subscription in the directory that flags the agent to notify it of any change events that affect OracleAS Portal security (for example, adding or deleting a user from a group).

6.1.6.2.5   User and Group Lists of Values in OracleAS Portal The User, Group, Portal User Profile, and Portal Group Profile portlets include lists of values for users or groups. These lists of values must be populated with information stored in the directory. By

default, the list of values displays the groups contained under the OracleAS Portal group container in the OracleAS Portal DIT structure. You can, however, browse to any group in the tree to which you have access from the list of values.

The groups that are displayed in the list of values for groups depend on the privileges of the user viewing them. For example, if a user views the list of values from the Group portlet, the list only displays those groups that can be edited or deleted by that user.

In some cases, you may encounter problems in rendering these user and group lists of values. You can resolve these problems in one of two ways:

- Defining a Common JavaScript Domain for DAS Lists of Values
- Configuring DAS to Reside on the OracleAS Portal Middle-Tier

Defining a Common JavaScript Domain for DAS Lists of Values

If you have your directory and OracleAS Portal servers residing in different domains, you must explicitly set the JavaScript domain for OracleAS Portal such that it can resolve user and group lists of values. For example, suppose that your installation has OracleAS Portal configured to use a different Oracle HTTP Server than the DAS. In this situation, you need to have a common domain so that the values can be transferred from the list of values displayed by the DAS to the page displayed by OracleAS Portal.

To create a single domain in this case, do the following:

1. Login to SQL*Plus as PORTAL.

2. Run the following SQL script:

   ```
   secjsdom.sql <domain_name>
   ```

   where <domain_name> is something like abc.com.

Performing this procedure enables you to run directory lists of values from OracleAS Portal in either Netscape or Microsoft Internet Explorer. For more information about secjsdom.sql, refer to Section C.4, "Using the secjsdom.sql Script".

> **See Also:** Section 6.3.2.3.4, "Group Search Base Distinguished Name (DN)" for information about choosing where OracleAS Portal searches for groups.

Configuring DAS to Reside on the OracleAS Portal Middle-Tier

In the preceding section, Defining a Common JavaScript Domain for DAS Lists of Values, we described how to create a single domain for DAS lists of values using secjsdom.sql. In some cases, creating a single domain with secjsdom.sql is not sufficient to resolve the JavaScript cross-domain scripting restrictions. In these situations, listed subsequently, you may need to deploy DAS on OracleAS Portal's middle-tier:

- Your users are on Netscape 7 or higher versions, or on browsers that do not properly implement the option of setting a common JavaScript domain.

- You are doing virtual hosting with hostnames that do not have common host domain.

- You have subscribers using branded URLs, again without a common domain.

To avoid the issues of cross-domain scripting and browser restrictions with support of the common domain directives in JavaScript, you can install DAS directly on the OracleAS Portal middle-tier. In this way, DAS can be used to support the lists of values that need to write values back to the OracleAS Portal forms. Implementing this configuration involves the following high-level steps:

- Manually Deploy and Configure DAS on OracleAS Portal's Middle-Tier
- Run secdaslc.sql

Manually Deploy and Configure DAS on OracleAS Portal's Middle-Tier

1. Navigate to the *MID_TIER_ORACLE_HOME*/dcm/bin directory.

2. Create a new component using the following command:

   ```
   MID_TIER_ORACLE_HOME/dcm/bin/dcmctl createcomponent -verbose -debug -ct oc4j
   -co OC4J_SECURITY
   ```

3. Start the component by using the following command:

   ```
   MID_TIER_ORACLE_HOME/dcm/bin/dcmctl start -verbose -debug -co OC4J_SECURITY
   ```

4. Deploy the oiddas.ear file using the following command:

   ```
   MID_TIER_ORACLE_HOME/dcm/bin/dcmctl deployApplication -debug -verbose
   -a oiddas -f ORACLE_HOME/ldap/das/oiddas.ear -co OC4J_SECURITY
   ```

5. Perform the following steps to add the LD_LIBRARY_PATH and DISPLAY environment variables to the opmn.xml file:

a. Navigate to the *MID_TIER_ORACLE_HOME*/opmn/conf directory and open opmn.xml in a text editor.

b. Add the following lines in the OC4J_SECURITY section of opmn.xml:

For a UNIX environment:

```
<environment>
   <variable id="DISPLAY" value="localhost:0.0"/>
   <variable id="LD_LIBRARY_PATH" value="MID_TIER_ORACLE_HOME/lib"/>
</environment>
```

For a Windows environment:

```
<environment>
   <variable id="PATH" value="MID_TIER_ORACLE_HOME\\bin"/>
</environment>
```

Replace hostname and *MID_TIER_ORACLE_HOME* with the appropriate values. Note the following example in the OC4J_SECURITY section of opmn.xml (MS Windows environment) :

```
<process-type id="OC4J_SECURITY" module-id="OC4J">
   <environment>
      <variable id="PATH" value="D:\\oracle\\bin"/>
   </environment>
   <module-data>
      <category id="start-parameters">
         .....
         .....
      </category>
   </module-data>
   <start timeout="3500" retry="2"/>
   <stop timeout="120"/>
   <restart timeout="720" retry="2"/>
   <port id="ajp" range="3301-3400"/>
   .....
   .....
   <process-set id="default_island" numprocs="1"/>
</process-type>
```

c. Navigate to the *MID_TIER_ORACLE_HOME*/dcm/bin directory.

d. Save the changes to the repository by using the following command:

```
MID_TIER_ORACLE_HOME/dcm/bin/dcmctl updateconfig -verbose -debug
-ct opmn
```

e. Restart OPMN by using the following command:

```
MID_TIER_ORACLE_HOME/dcm/bin/dcmctl restart -verbose -ct opmn
```

f. Stop and start the OC4J_SECURITY instance by using the following commands:

```
MID_TIER_ORACLE_HOME/dcm/bin/dcmctl stop -verbose -debug -ct oc4j
  -co OC4J_SECURITY
MID_TIER_ORACLE_HOME/dcm/bin/dcmctl start -verbose -debug -ct oc4j
  -co OC4J_SECURITY
```

g. Through Oracle Directory Manager, set permissions/grant privilege to the Oracle Application Server instance where OracleAS Portal was installed by adding its DN entry (orclApplicationCommonName=*OracleAS_instance_name*,cn=IAS Instances,cn=IAS,cn=Products,cn=OracleContext) to the group entry under the DAS application that defines the associated middle-tiers.

To perform this step, do the following:

– Connect to Oracle Internet Directory using Oracle Directory Manager.

– Get your Oracle Application Server instance's DN from the following location:

```
cn=IAS Instances,cn=IAS,cn=Products,cn=OracleContext
```

Note: You may find more than one Oracle Application Server instance listed under the cn=IAS Instances container. You need to choose the one that represents the middle-tier that you are configuring.

For example:

```
orclApplicationCommonName=OracleAS_instance_name,cn=IAS
Instances,cn=IAS,cn=Products,cn=OracleContext
```

– Add the entry shown in the preceding text to the uniquemember attribute of the following entry:

```
cn=Associated Mid-tiers,orclApplicationCommonName=DASApp,cn=DAS,
cn=Products,cn=OracleContext
```

–   Apply your changes and exit the Oracle Directory Manager.

h. Verify that DAS is running by entering following URL in your browser:

```
http://midtier_hostname:port_number/oiddas
```

where *midtier_hostname* is the name of the computer on which the Oracle HTTP Server is running and *port_number* is the corresponding http port number. This URL should display the DAS home page.

Run secdaslc.sql

After you manually deploy and configure DAS on OracleAS Portal's middle-tier, you need to set the configuration setting that instructs OracleAS Portal to render the DAS list of values links as OracleAS Portal middle-tier URLs rather than using the DAS base URL value from Oracle Internet Directory. You do this step by running secdaslc.sql.

> **Note:** A prerequisite for this procedure is to have the OracleAS Portal middle-tier properly associated with an Oracle Application Server infrastructure home directory.

1. From your operating system command prompt, go to *MID_TIER_ORACLE_ HOME*/portal/admin/plsql/wwc and make it your current working directory.

2. Using SQL*Plus, connect to the OracleAS Portal instance as the PORTAL schema user and run the following command:

```
@secdaslc.sql Y
commit;
exit;
```

At this point, OracleAS Portal is configured to invoke DAS lists of values from the OracleAS Portal middle-tier. All other DAS operations will continue to be invoked on the infrastructure instance of DAS. The OracleAS Portal middle-tier based lists of values will not have any issues with cross-domain scripting problems.

6.1.6.3 Relationship Between OracleAS Portal and Oracle Internet Directory

As shown in Figure 6–4, the Oracle Directory Integration Platform provides important services to notify components of user and group change events and synchronize directories.

Figure 6–4   Oracle Directory Integration Platform Synchronization
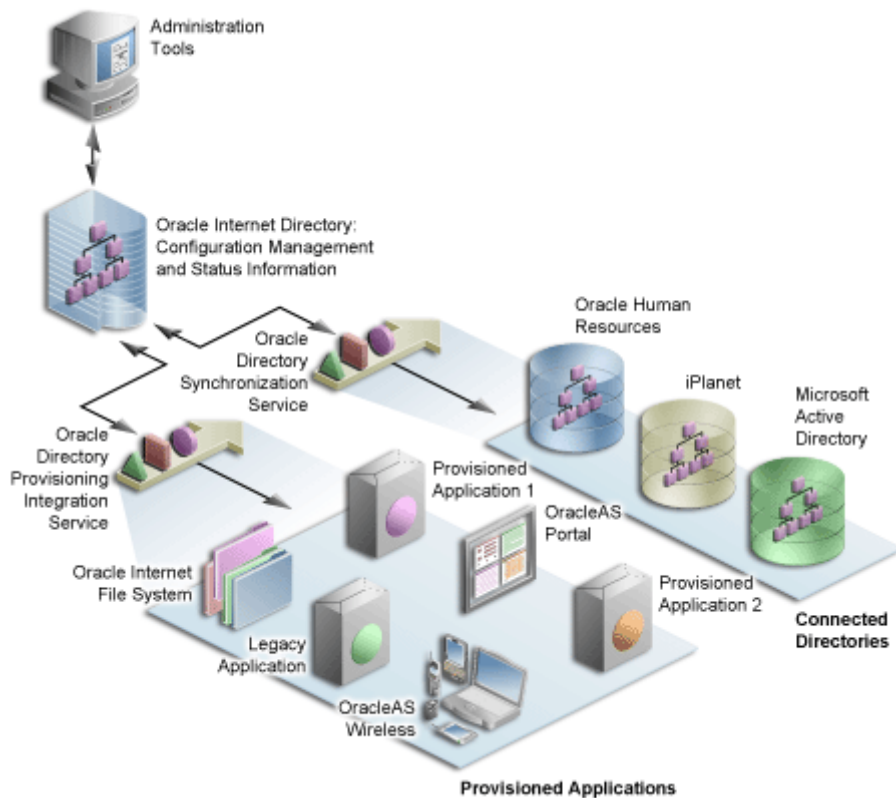
In the figure, the flow to and from the Oracle Internet Directory has two paths. The first path, labeled Oracle Directory Synchronization Service, illustrates the concept of synchronization. In this case, the Oracle Internet Directory acts as a gateway to some external directory or repository. The synchronization service ensures that changes are coordinated between the Oracle Internet Directory and its connected directories. Whenever a change occurs in one of the directories, a notification must be raised with the Oracle Internet Directory to appropriately reflect the change across all of the affected directories.

The second path, labeled Oracle Directory Provisioning Integration Service, illustrates the concept of provisioning. In provisioning, an application, such as OracleAS Portal, subscribes to changes to certain user or group information. For example, suppose that an administrator removes a user from a group through the DAS. As a result of this change, the user should no longer be allowed to access

certain pages in OracleAS Portal. The Oracle Directory Integration Platform must notify OracleAS Portal to update its local cache and immediately prevent the user from accessing the pages to which she no longer should have access.

For provisioning services, components like OracleAS Portal subscribe to provisioning events (for example, deletion of a group) in order to keep their local caches of user and group information synchronized with the central user and group repository in the Oracle Internet Directory. When a change event occurs, all of the components that are subscribed to that change event are notified by the Directory Synchronized Provisioning agent of the Oracle Directory Integration Platform. OracleAS Portal sets the Portal directory synchronization subscription flag in the directory to indicate that it should be notified whenever a subscribed change event takes place. Table 6–16 shows the events to which OracleAS Portal subscribes and the actions it takes when those events occur:

Table 6–16   Directory Synchronized Events Handled By OracleAS Portal

Subscribed event	OracleAS Portal action
USER DELETE	The local user profile entry is deleted, resulting in the deletion of the user's privileges. Pages associated with this user are invalidated in OracleAS Web Cache.
USER MODIFY (orclDefaultProfileGroup)	The default group of the user is changed in the local user profile.
GROUP DELETE	The local group profile is deleted, resulting in the deletion of the privileges assigned to this group. The WWSEC_FLAT$ table is updated accordingly.
GROUP MODIFY (uniqueMember, member)	The WWSEC_FLAT$ table is updated to reflect membership changes that affect OracleAS Portal.
	If the membership changes involve a group being added or deleted from the modified group, the pages associated with the users of the added or deleted group are invalidated in OracleAS Web Cache. The reason for this action is that the security changes might affect what is visible on the page or the access privileges of the page itself.

> **Note:** OracleAS Portal does not need to subscribe to user and group creation events. The local user profile is created automatically when a new user first logs on or is assigned some privilege that causes the user to be referenced in an access control list of OracleAS Portal. Similarly, a local group profile is created automatically when a new group is first referenced in an access control list.

In order to function properly, OracleAS Portal requires the following for its integration with Oracle Directory Integration Platform:

- The Oracle Directory Integration Platform must be running. To start the Oracle Directory Integration Platform, you use the `oidctl` command, for example:

```
oidctl instance=1 server=odisrv flags="host=iasqa-ultra1.abc.com
    port=4032" start
```

- The subscription profile must be created in the Oracle Internet Directory. The profile is normally created during the installation of OracleAS Portal by the OracleAS Portal Configuration Assistant (OPCA).

> **See Also:** *Oracle Internet Directory Administrator's Guide*

6.1.6.3.1 Update Subscription Profiles for Groups Based on groupOfNames By default, groups created in the Oracle Internet Directory by the DAS are based on the IETF object class `groupOfUniqueNames`. However, there is now support for handling groups created with the object class `groupOfNames` as well. If your portal has an existing Oracle Directory Integration Platform subscription profile in the Oracle Internet Directory (from 9.0.2), then it would be subscribing to group modifications and deletions based on groups using `groupOfUniqueNames`. If any existing groups in Oracle Internet Directory are based on the `groupOfNames` object class you must update the Oracle Directory Integration Platform subscription profile to subscribe to the events for groups based on `groupOfNames` in addition to `groupOfUniqueNames`.

If you need to change the subscription profile, we recommend you first delete the old subscription profile with this command:

```
ptlasst.csh -mode MIDTIER -type DIPUNREG
```

And then re-create a new subscription profile with this command:

```
ptlasst.csh -mode MIDTIER -type DIPREG
```

Only perform these operations during periods of downtime or no activity, so that changes pending in the change log are not lost as a result of deleting the old profile and starting the new one.

Details of the MIDTIER mode types DIPREG and DIPUNREG used to create/drop the Oracle Directory Integration Platform subscription profiles are as follows:

DIPREG

```
ptlasst.csh -i custom -mode MIDTIER -type DIPREG -s <portal_schema> -c <portal_
connect_string> -ldap_h <oid_host> -ldap_p <oid_port> -ldap_d <oid_admin_dn>
-ldap_w <oid_admin_password> -silent -verbose
```

DIPUNREG

```
ptlasst.csh -i custom -mode MIDTIER -type DIPUNREG -s <portal_schema> -c
<portal_connect_string> -ldap_h <oid_host> -ldap_p <oid_port> -ldap_d <oid_
admin_dn> -ldap_w <oid_admin_password> -silent -verbose
```

The resulting subscription profile will correctly subscribe to modifications and deletions for both types of group.

6.1.6.4 Relationship Between OracleAS Portal and DAS

In addition to querying the directory for user and group information, OracleAS Portal must provide users with the means to add and modify user and group information. To change information in the directory, use the DAS. OracleAS Portal provides links to the delegated administration server for users with the privileges to add and change users and groups.

6.1.6.4.1 Creating and updating information Stored in Oracle Internet Directory The DAS provides a comprehensive interface for making updates to the directory. Authenticated users who have the appropriate privileges can access the delegated administration server through the User and Group portlets on the **Administration** tab in OracleAS Portal. To access these portlets, a user must be a member of the OracleDASCreateUser and OracleDASCreateGroup groups, respectively. The PORTAL and PORTAL_ADMIN users are members of both of these groups by default. AUTHENTICATED_USERS may also create groups by default.

6.1.6.4.2 Relationship Between DAS, mod_osso, and the OracleAS Single Sign-On mod_osso protects URLs behind the OracleAS Single Sign-On environment by making the HTTP server effectively into a partner application. DAS functionality is single

sign-on enabled by using mod_osso to get the user's identify from the OracleAS Single Sign-On session.

Figure 6–5   Relationship between DAS, mod_osso, and OracleAS Single Sign-On
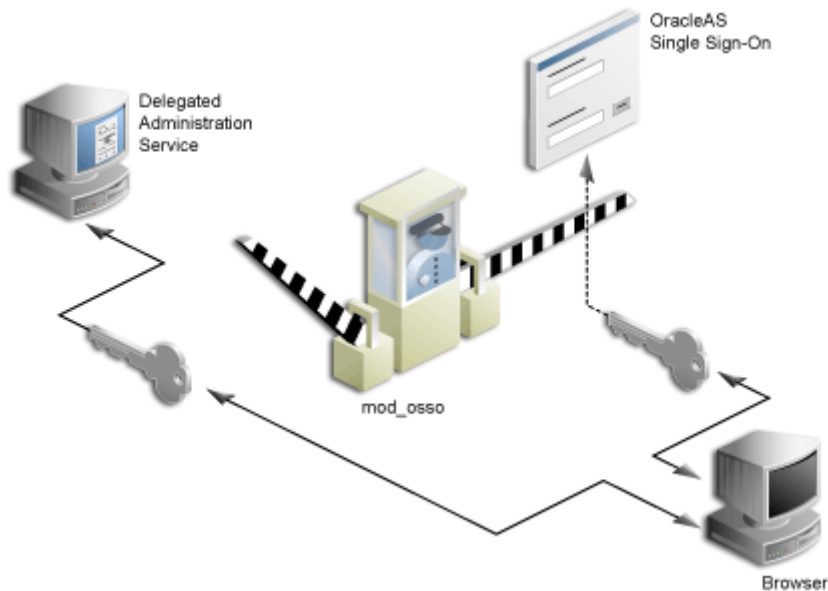
mod_osso is a module of the Oracle HTTP Server that is written as a partner application. You can use mod_osso to enable applications, including OC4J applications, for single sign-on. You achieve this by configuring mod_osso with Oracle HTTP Server directives to restrict access to the OC4J application URLs.

DAS is implemented as an OC4J application, which relies on mod_osso to authenticate users attempting access. When a user attempts to access a DAS dialog (for example, a list of users or groups, or the Create User form), mod_osso checks whether the user has been authenticated. mod_osso performs no authorization checks other than checking for authentication. If the user has not been authenticated, mod_osso, which is an OracleAS Single Sign-On partner application, redirects the user's request to OracleAS Single Sign-On. OracleAS Single Sign-On either:

- Finds a cookie that indicates the user has been properly authenticated and sends back an authenticated token to mod_osso.

- Or, if no cookie has been created yet, it brings up the login page to authenticate the user.

Once the user has been properly authenticated, they are redirected by mod_osso to the requested DAS URL. DAS then becomes accessible to the user and enforces the user's privileges, typically relying on access control items in the Oracle Internet Directory.

DAS URLs

The first request to DAS from a user session in OracleAS Portal is redirected to the OracleAS Single Sign-On so that mod_osso, which acts as a partner application on behalf of DAS, can establish the identity of the user. OracleAS Single Sign-On constructs a URLC token that includes the requested DAS URL. There is about a 2K limit on the length of the URLC token imposed by Internet Explorer. As such, the length of the DAS URL is also limited. In order to provide a seamless integration with DAS, OracleAS Portal includes the URLs of the current portal page and the portal home page within this DAS URL. A typical DAS URL appears as follows:

```
http://myportal.us.abc.com:7777/oiddas/ui/oracle/ldap/das/group/AppCreateGroupIn
foAdmin?doneURL=https%3A%2F%2Fwebsvr.us.oracle.com%3A5001%2Fportal%2Fpage%3F_
pageid%3D6%2C1%2C6_12%3A6_18%26_dad%3Dportal_9_0_2_6_7%26_schema%3DPORTAL_9_0_2_
6_7&homeURL=https%3A%2F%2Fwebserver.us.abc.com%3A5001%2Fportal%2Fpage%3F_
pageid%3D6%2C1%2C6_12%3A6_18%26_dad%3Dportal_9_0_2_6_7%26_schema%3DPORTAL_9_0_2_
6_7&amp;parentDN=cn%3Dportal_9_0_2_6_
7.s901dev0.portalserver.us.abc.com%2Ccn%3Dgroups%2Cdc%3Dus%2Cdc%3Doracle%2Cdc%3D
com&amp;enablePA=true
```

When this URL is included in the URLC token, which is then encrypted for security reasons, the length of the resulting token easily approaches the 2K threshold. If it exceeds this limit, the browser may show an error.

There is no fixed size for the URL. However, if you see browser errors when performing DAS operations, you should consider reducing the size of various parts that comprise the portal URL as this will help ensure that the URL doesn't exceed the 2k limit. For example, limit hostnames to 8 characters or less and DAD names to 6 characters or less.

In the event that you encounter this problem, the work around is to login to DAS first through a shorter URL, such as the **Directory Administration** link in the **Services** portlet. Any subsequent access to DAS will then not require SSO redirection, and will succeed.

6.1.6.5 User Portlet

> **Note:** Only a user who is a member of the OracleDASCreateUser, OracleDASEditUser, or OracleDASDeleteUser privilege groups can see the User portlet. The link to create new users is displayed only to users who are members of the OracleDASCreateUser group.

The **User** portlet on the **Portal** tab under **Administration** enables you to create and update users through DAS. To create a new user, click the **Create New Users** link in the User portlet. To update information for an existing user, type their user name in the Name field or choose it from the list of values and click **Edit**. To delete a user, type their user name in the Name field or choose it from the list of values and click **Delete**.

> **See Also:** *Oracle Internet Directory Administrator's Guide*

Figure 6–6   User Portlet
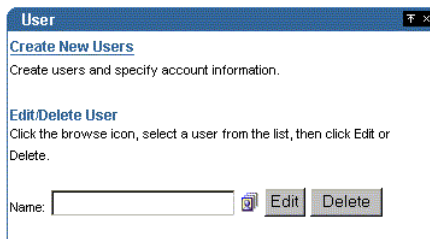
6.1.6.6 Portal User Profile Portlet

> **Note:** The Portal User Profile portlet is only visible to users with Manage or Edit privileges for All User Profiles.

To set global user privileges and preferences that pertain specifically to the portal, use the Portal User Profile portlet. To update a user's portal preferences and privileges, type their user name in the Name field or choose it from the list of values. You can set all of the following for the user's profile:

- Preferences

- – whether the user can access the portal

- – database schema name for the user

- – whether the user has a personal page

- – default user group for the user

- – default home page for the user

- – default style for the user

- – whether to clear the OracleAS Web Cache for the user

- Global Privileges

  - – page group privileges

  - – Portal DB Provider privileges

  - – administration privileges

Figure 6–7   Portal User Profile Portlet
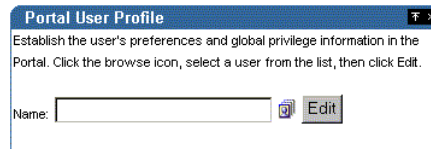
6.1.6.7 Group Portlet

> **Note:**   Every user can see the Group portlet, but the link to create new groups is displayed only to users who are members of the OracleDASCreateGroup privilege group. Users can only edit or delete a group if they are the group's owner or a member of a group with appropriate access control information (ACI) to edit or delete the group. The following privilege groups are seeded in the Oracle Internet Directory:
>
> - OracleDASCreateGroup
>
> - OracleDASEditGroup
>
> - OracleDASDeleteGroup

The **Group** portlet on the **Portal** tab under **Administration** enables you to create and update user groups through DAS. To create a new group, click the **Create New Group**s link in the Group portlet. To update information for an existing group, type its name in the Name field or choose it from the list of values and click **Edit**. To delete a group, type the group name in the Name field or choose it from the list of values and click **Delete**.

> **See Also:** *Oracle Internet Directory Administrator's Guide*

Figure 6–8   Group Portlet
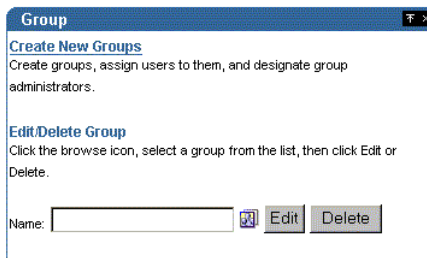
6.1.6.8 Portal Group Profile Portlet

> **Note:** The Portal Group Profile portlet is displayed to all users, but only users with the Manage or Edit privilege for All Group Profiles, or the owner of a group can edit its profile.

To set global group preferences and privileges that pertain specifically to the portal, you need to use the Portal Group Profile portlet. To update a group's portal preferences and privileges, type the group name in the Name field or choose it from the list of values. You can set all of the following for the group's profile:

- Preferences
    - default home page for the group
    - default style for the group
- Global Privileges
    - page group privileges
    - Portal DB privileges

– administration privileges

Figure 6–9   Portal Group Profile Portlet

6.1.6.9 DAS Public Roles

In many cases, it is more efficient to use DAS roles to assign privileges rather than the more granular, per-user approach. When creating users, you might notice a section called Roles Assignment on the Create User page, shown in Figure 6–10.

> **Note:** In releases prior to 9.0.4, roles were called public groups.

Figure 6–10   OracleAS Portal Create User Page
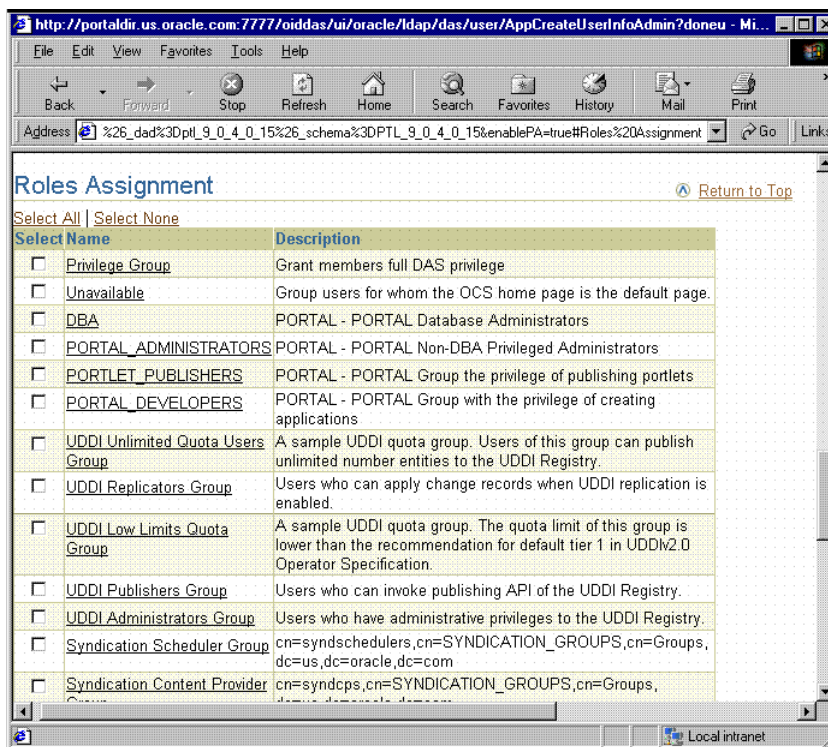
Roles provide a very convenient mechanism by which users can be created and granted a set of privileges simultaneously. When a check box for a role is checked for a given user, they are granted the designated role upon creation. As an administrator, you can create your own roles and pre-assign any combination of Oracle Internet Directory and OracleAS Portal privileges to them.

6.1.6.9.1   Example: Defining a User Administrator Role   Suppose that you want to create a role with the appropriate privileges for a user administrator. You could create such a role by following these steps:

Step 1: Create a group.

You begin by creating a group in the usual manner:

1. From Portal Builder (the design time pages), click **Administer**, if you are not already on the **Administer** tab.

2. Click **Create New Group** in the Group portlet and the Create Group page appears, as shown in the following image.
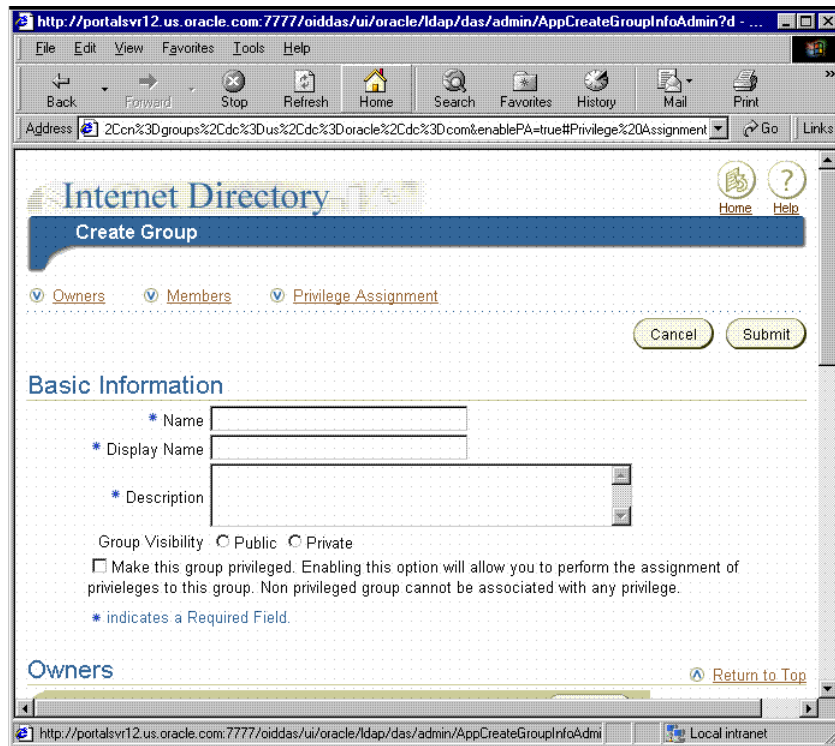
3. Enter the required fields (indicated by asterisks).

4. On the Create Group page, click **Privilege Assignment** to go to that section and choose the following privileges, as shown in the following image:

- Allow user creation

- Allow user editing

- Allow user deletion
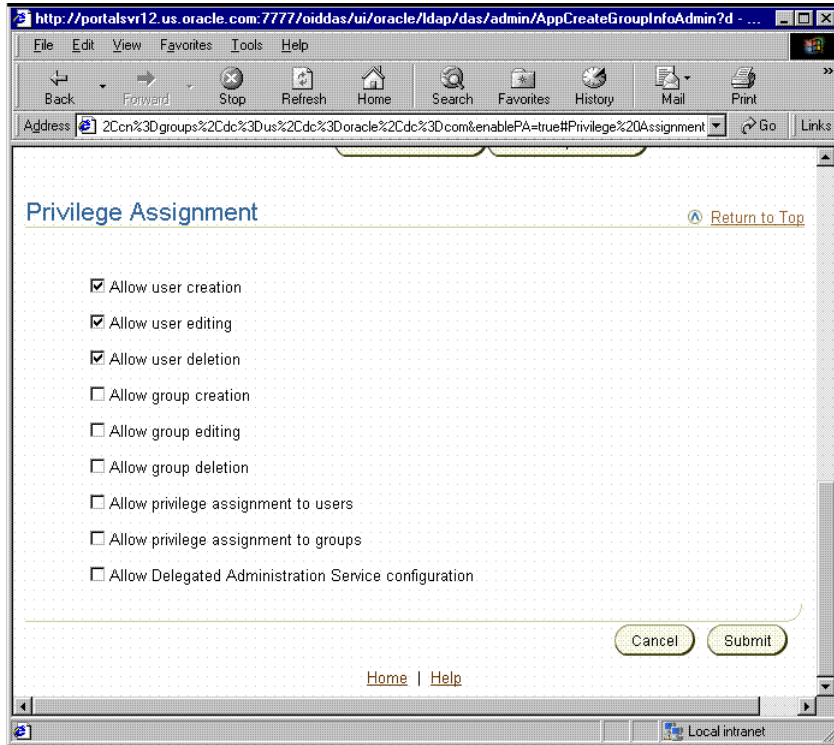
5. Click **Submit**.

Step 2: Assign the Manage privilege for all user profiles.

After you create the user administrator group, you need to assign it the Manage privilege for all user profiles. This privilege is the only global privilege that you need to assign to this group for user administration.

1. From Portal Builder (the design time pages), click **Administer**, if you are not already on the **Administer** tab.

2. From the **Portal Group Profile** portlet, enter the name of your newly created group and click **Edit**.

3. Click **Privileges** to go to that tab.

4. Scroll down to the **Administration Privileges** section, shown in the following image. From the list next to **All User Profiles**, choose **Manage**.
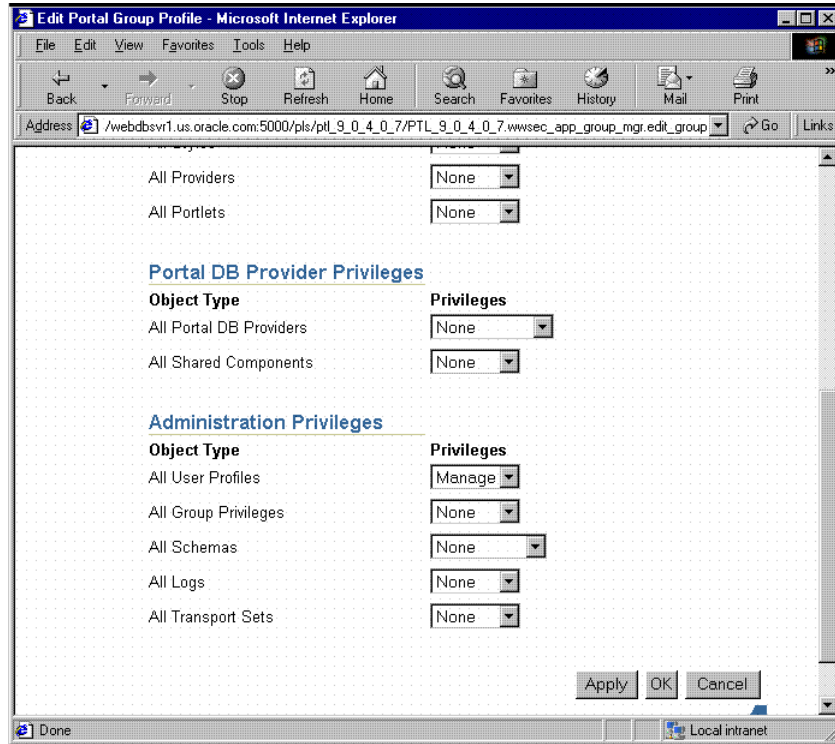
5. Click **OK**.

Step 3: Make the group a role.

Now that you have created a group representing the user administrator role, you need to enable it as a role so it appears in the list of roles on the Create User page.

1. From Portal Builder (the design time pages), click **Administer**, if you are not already on the **Administer** tab.

2. In the **Services** portlet, click **Directory Administration**.

3. Click **Configuration** to display that tab.

4. Click **User Entry**.

5. Click **Next** until you reach Step 5 of 5, **Configure Roles**, of the wizard, as shown in the following image.

6. Click **Add Role** to choose the new group and add it to the list of roles.
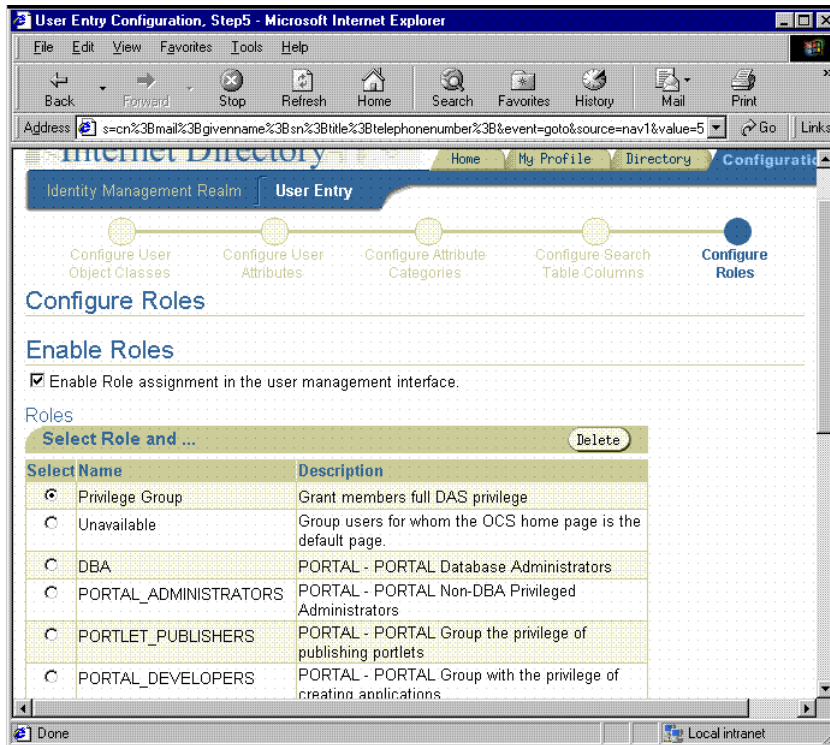
7. Click **Finish**. Your group will now appear in the list of public groups on the **Create User** page.

Step 4: Hide detailed privilege assignment section.

To encourage the usage of roles rather than direct privilege assignment, you can turn off the detailed privilege assignment section of the Create Users page. In order to implement this change, you need to update a configuration entry in the OracleAS Portal repository. This setting stops DAS from displaying the Privilege Assignment section in the Create/Edit User page when it is called from an OracleAS Portal administration page.

1. Login to the PORTAL schema through SQL*Plus.

> **Note:** The PORTAL schema password is stored in the Oracle
> Internet Directory and the entry may be viewed by an
> administrator using the `oidadmin` utility with the following path
> under Entry Management:
>
> `OrclResourceName=PORTAL,orclReferenceName=iasdb.my`
> `host.au.oracle.com,cn=IAS Infrastructure`
> `Databases,cn=IAS,cn=Products,cn=OracleContext`

2. Invoke the following commands to set the `das_enable_pa` variable in
   OracleAS Portal's Oracle Internet Directory configuration preference store.

```
$ sqlplus
...
Enter user-name: portal
Enter password:
...
SQL> set serverout on
SQL> exec wwsec_oid.set_preference_value('das_enable_pa', 'N');

PL/SQL procedure successfully completed.

SQL> commit;

Commit complete.

SQL> exit
...
```

3. Because the User Portlet is cached in OracleAS Web Cache as well as the
   OracleAS Portal middle-tier file system cache, you must invalidate the cached
   version of the portlet before you are done. Updating the configuration
   parameter changes the behavior of the portlet, but updating the parameter does
   not invalidate the cache. You can invalidate the cached version of the User
   Portlet by running `secupoid.sql` and specifying options to update the cache.

   > **See Also:** *Oracle Application Server Web Cache Administrator's Guide*

Step 5: Validate your changes.

Once you have performed steps 1 through 4, go to the Create User page to verify
that your user administrator group appears there. Note how the other OracleAS

Portal administrative roles, or groups, are already pre-seeded into the Roles Assignment list on this page.

6.1.7 Security for Portlets

Portlets act as windows on your application, displaying summary information and providing a way to access the full functionality of the application. Portlets expose application functionality directly in your portal or provide deep links that take you to the application itself to perform a task. Since portlets format information for display on a Web page, the underlying application need not be Web enabled to be integrated with OracleAS Portal.

In OracleAS Portal, portlets are managed by providers. A provider is an application that you register with OracleAS Portal. OracleAS Portal supports two types of providers:

- Web providers
- Database providers

Portlet security consists of three major areas of functionality:

- **Authentication:** When a user first accesses a secure URL, they must be challenged for information that verifies their identity, such as a username and password, or a digital certificate.

- **Authorization:** Authorization is the process that allows certain users to access parts of an application. Some parts of an application may have public access while others may be accessible only to a limited number of authenticated users.

- **Communication security:** Communication security is the means by which OracleAS Portal establishes the authenticity of communications (for example, messages) to and from providers. In a heavily networked environment, it is critical to verify that communications are authenticate.

In order to make your Web providers truly secure, you need to make sure that they are secured in each of these areas. If you only implement security features for one or two out of three areas, then your providers cannot be considered secure. The effort you expend to secure a Web provider should be proportional to the confidentiality of the data the provider exposes.

6.1.7.1 Authentication

When a user first logs in to OracleAS Portal, they must enter their password to verify their identity before being permitted access. OracleAS Single Sign-On

manages the login process. Refer to Section 6.1.7.4, "Single Sign-On" for more information.

6.1.7.2 Authorization

Authorization determines whether a particular user should view or interact with a portlet. There are two types of authorization checks:

- **Portal Access Control Lists:** When you log in to OracleAS Portal, OracleAS Single Sign-On authenticates you. Once authenticated, OracleAS Portal uses Access Control Lists (ACLs) to grant you privileges on portal objects such as pages and portlets. The actions may range from simply viewing an object to performing administrative functions. If you do not belong to a group that has been granted a specific privilege, OracleAS Portal prevents you from performing the actions associated with that privilege.

- **Programmatic Portlet Security:** The Portal Developer's Kit-Java includes APIs that are called to determine if a particular user is authorized to view a portlet. You can use these APIs to implement authorization logic that augments the Portal ACL security.

6.1.7.3 Communication Security

Authentication and authorization are important components of securing your Web providers. They do not, however, check the authenticity of messages being received by a provider and are therefore not suitable on their own for securing access to a provider. If the communication is unsecured, someone could imitate OracleAS Portal and fool the Web provider into returning sensitive information.

Communication security focuses on securing communications between OracleAS Portal and a Web provider. These methods do not apply to database providers, which execute within the portal database. There are three types of communication security:

- Portal Server Authentication ensures that incoming messages emanated from a trusted host.

- Message Authentication ensures that the incoming messages were sent from a host with a shared key.

- Message Encryption protects the contents of a message by encrypting them.

6.1.7.3.1 Portal Server Authentication Portal Server Authentication restricts access to a provider to a small number of recognized machines. This method compares the IP address or the hostname of an incoming HTTP message with a list of trusted hosts.

If the IP address or hostname is in the list, the message is passed to the provider. If not, it is rejected before reaching the provider.

6.1.7.3.2 Message Authentication Message authentication works by appending a checksum based on a shared key to provider messages. When a message is received by the provider, the authenticity of the message is confirmed by calculating the expected value of the checksum and comparing it with the actual value received. If the values are the same, the message is accepted. If they are different the message is rejected without further processing. The checksum includes a time stamp to reduce the chance of a message being illegally recorded in transit and resent later.

6.1.7.3.3 Message encryption Message encryption relies on the use of the HTTPS protocol for communication between the provider and OracleAS Portal. Messages are strongly encrypted to protect the data therein. While encryption provides a high level of security, it also of necessity impacts performance.

6.1.7.4 Single Sign-On

Portlets act as windows into an application by displaying a summary of content and a method for accessing the full functionality of the application. Portlets can expose application functionality directly in the portal or provide deep links into the application itself to perform a task.

If the application is not confidential (that is, public), then users need not be authenticated to see and use it or its associated portlets. For more restricted applications, you need to authenticate the user who is accessing the application:

- Partner applications share the same authenticated user as the OracleAS Portal user. The application user and the OracleAS Portal user are the same in this case.

- External applications use a different authentication mechanism than OracleAS Portal and usually a different repository for user credentials. The application username can be the same as in OracleAS Portal, but the external application verifies the user through its own mechanism.

6.1.7.4.1 Partner Application A partner application shares the same OracleAS Single Sign-On as OracleAS Portal for authentication. Sharing OracleAS Single Sign-On instances means that when a user is already logged into OracleAS Portal, their identity can be asserted to the partner application without logging in again.

Partner applications tightly integrate with OracleAS Single Sign-On. When a user attempts to access a partner application, the partner application delegates the authentication of the user to OracleAS Single Sign-On. Once authenticated with a

valid username and password, a user need not provide a username or password when accessing other partner applications that share the same OracleAS Single Sign-On instance. OracleAS Single Sign-On determines that the user was successfully authenticated and indicates successful authentication to the other partner applications.

Partner application providers expose portlets that integrate a partner application's content with OracleAS Portal. The partner application provider trusts OracleAS Portal to authenticate the user on the provider's behalf. This relationship is possible because OracleAS Portal is, itself, a partner application.

Partner application providers must trust OracleAS Portal to authenticate the user in this way because the provider cannot perform the authentication itself. Authenticating the user directly requires the provider to redirect the browser to OracleAS Single Sign-On and provide success and failure URLs. This method is not possible due to the provider architecture. The primary reason for it is that the authentication occurs in response to an API call from OracleAS Portal to the provider. OracleAS Single Sign-On cannot imitate that call upon successful authentication to the initSession()/dologin() method to complete its normal processing.

Authentication of users in partner applications differs from conventional applications. Partner applications delegate user authentication to OracleAS Single Sign-On. If the user has not been authenticated, OracleAS Single Sign-On displays a login page prompting the user to enter a username and password. The login page submits the username and password back to tOracleAS Single Sign-On.

If successfully authenticated, OracleAS Single Sign-On creates a special cookie containing information about the user. For security, OracleAS Single Sign-On encrypts the contents of the cookie. The cookie is sent back to the user's browser but is scoped such that only OracleAS Single Sign-On can access it. It is not passed to any other listeners. After creating the cookie, OracleAS Single Sign-On redirects the Web browser to the success URL specified by the partner application. At this point, the partner application creates an application session cookie which contains information the application needs to reestablish the session later. The contents may be encrypted using the Single Sign-on SDK. Upon making subsequent requests to the partner application, it detects the presence of the partner application session cookie and from it knows that the user is already authenticated.

If the user later accesses another partner application, that application looks for its application specific session cookie. If the cookie is not found, the application redirects the request to OracleAS Single Sign-On as described previously. This time OracleAS Single Sign-On detects the presence of the user's OracleAS Single Sign-On cookie. This cookie indicates that the user is already authenticated and OracleAS

Single Sign-On redirects the browser to the success URL of the second partner application without prompting the user for credentials again. At this point, the partner application creates its own application specific session cookie. To secure the application session cookies, the content may be encrypted using the Single Sign-On SDK.

Advantages

- Provides the tightest integration with OracleAS Portal and OracleAS Single Sign-On.

- Provides the best OracleAS Single Sign-On experience to users.

- Provides the most secure form of integration because user names and passwords are not transmitted between the portal and the provider.

- The application and the portal share the same user repository, which reduces user maintenance.

Disadvantages

- The application must share the same user repository as OracleAS Portal even though the application's user community may be a subset of the portal's user community. This minor issue can be addressed because you can restrict access to the portal pages that expose the application to the application's user community.

- The application can only be tightly integrated with one or more OracleAS Single Sign-On if they share the same user repository.

Implementation Techniques

You make an application a partner application through either of the following mechanisms:

- mod_osso is a general purpose Oracle HTTP Server module and a partner application of OracleAS Single Sign-On. Once configured, it restricts access to URLs and handles such things as the redirection to OracleAS Single Sign-On and the creation of cookies. If an application's URLs are protected by mod_osso, it is effectively a partner application.

- The Single Sign-On SDK is a group of packages that can be used by an application to become a partner application. There are also some Java wrapper classes that call these packages so they can be called from an application written in Java.

mod_osso

mod_osso is a general purpose Oracle HTTP Server module and a partner application of OracleAS Single Sign-On. It uses OracleAS Single Sign-On to do the authentication. The module does all the communication and handling of cookies between the Oracle HTTP Server and OracleAS Single Sign-On. If mod_osso is configured to protect the URLs of a Web application, then the application effectively becomes a partner application.

OracleAS Portal is also a partner application and uses OracleAS Single Sign-On to authenticate users. Provided OracleAS Portal and mod_osso use the same OracleAS Single Sign-On instance, the user can access either the Web application or OracleAS Portal by logging into either one, that is, they need only login once to be able to access both the Web application and OracleAS Portal.

Advantages

- mod_osso is simple to set up.

- You need no additional code in the application.

- mod_osso generates a partner application cookie and does all the cookie handling.

- mod_osso secures the partner application and deep links from the partner application provider.

Disadvantages

- mod_osso can only be used with Web applications

Single Sign-On SDK

The Single Sign-On SDK enables programmers to integrate their application with OracleAS Single Sign-On. This SDK consists of a number of database packages that communicate with OracleAS Single Sign-On when an application wants to authenticate a user. These packages make an application a partner application. It also includes methods to encrypt/decrypt information, which are used to secure information stored in the application cookie. The Single Sign-On SDK also has Java class wrappers to the PL/SQL packages, which enable Java applications to become either partner or external applications.

OracleAS Portal is a partner application and uses OracleAS Single Sign-On to authenticate users. Provided OracleAS Portal and the Single Sign-On SDK are configured to use the same OracleAS Single Sign-On instance, the user can access either OracleAS Portal or the application by logging into either one, that is, they

need only login once to be able to access both the Web application and OracleAS Portal.

Advantages

- The Single Sign-On SDK can create either a partner application or an external application.

- It provides utilities to encrypt/decrypt cookies.

Disadvantages

- The Single Sign-ON SDK method requires changes to the application code.

- The application must be written using a technology that can be easily integrated with Java or PL/SQL.

- All entry points to the partner application that need to be secure need to call the Single Sign-On SDK if a partner application cookie is not found and the user must be authenticated.

6.1.7.4.2 External Application An External Application is an application that uses a different authentication mechanism than OracleAS Portal. The application may use a different instance of OracleAS Single Sign-On than the used by OracleAS Portal or some other authentication method. In either case, OracleAS Single Sign-On stores user name mappings, passwords, and any other required credentials to authenticate the user in each external application. When a user is already logged into OracleAS Portal, they will be logged into the external application without having to type in their username or password.

Applications that manage their own authentication of users can be loosely integrated with OracleAS Single Sign-On by registering as external applications. An external application can be exposed as a provider using the Oracle Application Server Portal Developer Kit so that it may be accessed from a portlet on a page. External application providers are only available to Web providers.

> **See Also:** For more information about the External Applications portlet, see *Oracle Application Server Portal User's Guide.*

When a previously authenticated user accesses an external application for the first time, OracleAS Single Sign-On attempts to authenticate the user with the external application. The authentication is performed by submitting an HTTP request that combines the registration information and the user's username and password for the application. If the user has not yet registered their username and password for

the external application, OracleAS Single Sign-On prompts the user for the required information before making the authentication request. When a user supplies a username and password for an external application, OracleAS Single Sign-On maps the new username and password to the user's OracleAS Portal username and stores them. The next time the user needs to access the external application the stored credentials are used.

Advantages

- Allows integration with many portals. If there is a preferred portal, the application could be integrated as a partner application of that portal and an external application of other portals.

- Provides a single sign-on experience for users. However, users still need to maintain different user names and passwords. In addition, the external application username mapping must be maintained.

- Allows integration with multiple portals independent of their user repositories and single sign-on mechanisms.

Disadvantages

- External applications do not share the same user repository as the portal, which requires additional maintenance of user information.

- The username and password is transmitted to the provider in plain text. This approach is not as secure as a partner application.

- The application must be written using a technology that can be easily integrated with Java or PL/SQL.

6.1.7.4.3 No Application Authentication In this case, the provider trusts the portal sending the requests. The provider can determine if the user is logged in and the portal user name, but the application has not authenticated the user.

Advantages

- You can implement this form of integration most easily and very fast.

Disadvantages

- Provides the weakest integration with OracleAS Portal.

6.1.7.5 Access Control Lists

When you login to OracleAS Portal, OracleAS Single Sign-On authenticates you. OracleAS Portal then uses Access Control Lists (ACLs) to determine if you are authorized to view each piece of content, including providers and portlets. If you do not belong to a group that has been granted a specific privilege, OracleAS Portal prevents you from performing the actions associated with that privilege.

ACLs are managed by the following:

- Privileges define the actions that can be performed on the object to which they are granted. Several privileges can be granted, such as Manage, Execute, Access, and Publish. If you set any of these privileges, then the user can access the portlet.

- Users and their privileges are managed from the Portal tab under the **Administer** tab of the builder.

- Group membership in a group and the privileges granted to the group are managed from the **Portal** tab under the **Administer** tab of the builder. A privilege granted to a user group is inherited by all members of that group.

- Privileges can be granted to a provider. By default, those privileges apply to the provider and all the portlets in the provider. Provider ACLs are managed on the Provider tab of the navigator.

- Privileges for portlets can override the privilege set for the portlet's provider. Portlet ACLs are managed on the **Provider** tab of the navigator. Using Open for the Provider takes you to a page to manage the portlets of the provider.

6.1.7.5.1 Advantages

- ACLs offer a simple, yet very powerful, mechanism to secure Portal objects.

- Since the management of users and groups is centralized, you do not have to change the ACLs as the membership of groups changes.

6.1.7.5.2 Disadvantages

ACLs are applied at the provider or portlet level. You cannot vary the security rules for a portlet depending on the portal page on which the portlet is placed.

6.1.7.6 Programmatic Portlet Security

You can implement portlet security methods within a provider to verify that given users may access portlet instances. These security methods work at the portlet level, that is, each portlet may have its own user access control. By implementing access

control methods in the provider, content may only be retrieved from a portlet if the user's credentials pass the authorization logic. If you do not implement portlet security methods in the provider, any username may be passed in, even a fictitious, unauthenticated one.

A provider can implement two portlet security methods:

- Get a list of portlets
- Determine portlet accessibility

These methods have access to the following information about authorization level:

- Strong indicates that OracleAS Single Sign-On has authenticated a user in the current OracleAS Portal session, that is, the user logged in to OracleAS Portal with a valid username and password, and called the portlet in that session.

- Weak indicates a user who previously had strong authentication and returns to view a page without an active OracleAS Portal session. A persistent cookie from the user's browser indicates that in some previous session the user logged in with a valid username and password.

- Public indicates a user has not logged in within the context of the current OracleAS Portal session and does not have a persistent cookie to indicate that such a state previously existed.

Portlets can also access the OracleAS Portal user privileges and group memberships:

- User's default group
- User or group privileges
- User's highest available privilege across all groups
- Objects a user can access

6.1.7.6.1 Advantages

With portlet security methods, you can have a portlet produce different output depending on the user's level of authorization.

6.1.7.6.2 Disadvantages

Most security manager implementations use the authorization level or some other user specific element in an incoming message. A check of this type could be bypassed by an entity imitating OracleAS Portal.

6.1.7.7 OracleAS Portal Server Authentication

One way you can prevent unauthorized access to providers is to restrict access to the provider to known client machines at the server level. This method goes some way towards defending against denial of service attacks.

In the Oracle HTTP Server, you can permit or deny directives in the `httpd.conf` file based on hostnames or IP addresses. If hostnames are used as discriminators, the server needs to look them up on its Domain Name Server, which incurs overhead to the processing of each request. Using the IP address prevents this added overhead, but the IP address may change without warning.

6.1.7.7.1 Advantages

- This approach only allows trusted hosts to access the provider

- You can set the restrictions up easily.

6.1.7.7.2 Disadvantages

- OracleAS Web Cache does not have IP address checking capability. If you have OracleAS Web Cache in front of a provider, a client on any host can send show requests to OracleAS Web Cache.

- You can circumvent this approach by sending messages to a provider containing fake IP addresses and hostnames. This method is tricky to carry out effectively because return messages will go to the machine with the copied IP address, but it can still cause problems.

6.1.7.8 Message Authentication

The Oracle Application Server Portal Developer Kit supports message authentication to limit access to a specified provider instance or group of provider instances. A provider is registered with a secret shared key known only to the Portal and provider administrators.

An OracleAS Portal instance sends a digital signature, calculated using a Hashed Message Authentication Code (HMAC) algorithm, with each message to a provider. A provider may authenticate the message by checking the signature with its own copy of the shared key. This technique may be used in SSL communication with a provider instead of client certificates.

An OracleAS Portal instance calculates a signature based on user information, a shared key, and a time stamp. The signature and time stamp are sent as part of the SOAP message. The time stamp is based on UTC (coordinated universal time, the

scientific name for Greenwich Mean Time) so that time stamps can be used in messages between computers in different time zones.

When the provider receives this message it will generate its own copy of the signature. If the signatures agree, it will then compare the message time stamp with the current time. If the difference between the two is within an acceptable value the message is considered authentic and processed accordingly.

A single provider instance cannot support more than one shared key. Multiple keys could cause security and administration problems if several clients sharing a provider use the same key. For instance, if one copy of the shared key is compromised in some way, the provider administrator has to create a new key, distribute it to all of the clients, and the clients must then update their provider definition. The way around this issue is to deploy different provider services, specifying a unique shared key for each service. Each provider service has its own deployment properties file so that each service is configured independently of the others. The overhead of deploying multiple provider services within the same provider adapter is relatively small.

If a provider does not have an OracleAS Web Cache in front of it, the use of the same signature cookie over the lifetime of a provider session means you must trade off between performance and the security provided by authenticating the requests. The signature cookie value is calculated only once after the initial SOAP request establishes the session with the provider. The shorter the provider session timeout, the more often a signature will be calculated to provide greater security against an illegal show request. However, the SOAP request required to establish a session takes more time.

In a provider that uses OracleAS Web Cache to cache show request responses, you make a similar trade-off. Cached content is secured in the sense that incoming requests must include the signature cookie to retrieve the cached content, but caching content for an extended period of time leaves the provider open to illegal show requests.

The signature element provides protection against interception and the resending of messages, but it does nothing to prevent the interception and reading of message contents. Messages are still transmitted in plain text. If you are concerned about the content of messages being read by unauthorized people, you should use message authentication in conjunction with SSL.

6.1.7.8.1 Advantages

Message authentication ensures that the message received by a provider comes from a legitimate OracleAS Portal instance.

6.1.7.8.2 Disadvantages

- Message authentication can cause administration problems if a provider serves more than one OracleAS Portal instance.

- Message authentication has a performance implication if made very secure by having a short session timeout.

6.1.7.9 HTTPS Communication

Normal communication between OracleAS Portal and a provider uses HTTP, a network protocol that transmits data as plain text using TCP as the transport layer. An unauthorized agent can read an intercepted message. HTTPS uses an extra security layer (SSL) on top of TCP to secure communication between a client and a server.

Each entity (for example, an OracleAS Web Cache instance) that receives a communication using SSL has a freely available public key and a private key known only to the entity itself. Any messages sent to an entity are encrypted with its public key. A message encrypted by the public key may only be decrypted by the private key so that even if a message is intercepted it cannot be decrypted.

Certificates are used to sign communications, thereby ensuring that the public key does in fact belong to the correct entity. These certificates are issued by trusted third parties known as Certification Authorities (CA), for example, Verisign. They contain an entity's name, public key, and other security credentials. They are installed on the server end of an SSL communication to verify the identity of the server. Client certificates may also be installed on the client to verify the identity of a client, but this feature is not yet supported OracleAS Portal. Message authentication may be used instead.

Oracle Wallet Manager is the application used to manage public key security credentials. It is used to generate public/private key pairs, create a certificate request to a CA, and then install the certificate on a server.

6.1.7.10 Configuration of SSL

When a provider is registered from an OracleAS Portal instance, only one URL is entered. HTTP or HTTPS may be used, but not a combination of both.

Each port on each server that may be used to receive SSL messages must have a server side certificate installed, that is, the OracleAS Web Cache instance (if any) in front of the Web provider and the server which hosts the provider. The certificate installed on a server port ensures that communication between two points is encrypted, but it does not authenticate the source of a message. Message

authentication should be used as well to fully secure communication between a trusted OracleAS Portal instance and a provider.

> **See Also:**
>
> - Section 6.3.2.1, "Configuring SSL for OracleAS Portal"
> - *Oracle Internet Directory Administrator's Guide*
> - *Oracle Application Server Web Cache Administrator's Guide*

6.1.7.10.1 Advantages

- SSL encrypts the contents of a portlet.

6.1.7.10.2 Disadvantages

- Encryption has a performance impact on OracleAS Portal.
- If used, encryption requires all portlets from a provider to use HTTPS even if some of the content is public.

You'll find additional information on security for your Web providers in the papers:

- *Overview of Provider Security*
- *Overview of Password Authenticated Applications*

on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

6.1.8 Securing the OmniPortlet and Simple Parameter Form

The OmniPortlet and simple parameter form are located under Portlet Builders in the Portlet Repository. By default, any user who has the privilege to create pages can add these portlets to a page and define them. Furthermore, a user who only has view privileges on the page can define these portlets by clicking the **Define OmniPortlet** or **Define Simple Parameter Form**.

To control this kind of access, you can activate a privilege check. Once you perform the procedure that follows, the display of these portlets depends upon the privileges granted to the user or user group from the **Access** tab. To perform any operations on the portlet, the user or user group needs at least the Execute privilege.

1. Login to OracleAS Portal.

2. Click the **Navigator** link.

3. Click the **Portlet Repository** page group.

4. Click **Pages**.

5. Next to the **Portlet Builders** page, click **Edit**.

6. Click Page: **Access** in the upper left of the page.

7. Select **Enable Item Level Security**.

8. Click **OK**.

9. Click the **Edit Item** icon next to **OmniPortlet**.

10. Click the **Access** tab.

11. Check **Define Portlet Access Privileges**.

12. Click **Apply** and note the appearance of the Grant Access and Change Access sections of the page.

13. Use the **Grant Access** section to assign privileges to users and groups as desired.

14. Click **OK**.

15. Repeat steps 9 through 14 for the **Simple Parameter Form**.

6.1.9 Securing the Web Clipping Provider

Appendix I, "Administering Web Clipping" describes the administrative tasks that must be performed before you are able to use the Web Clipping Provider. The following sections describe some security configuration options that you should implement to enable the Web Clipping Provider to access trusted sites and encrypt the channel between itself and the database:

- Adding Certificates for Trusted Sites
- Configuring Oracle Advanced Security for the Web Clipping Provider

6.1.9.1 Adding Certificates for Trusted Sites

When a user navigates to a secure site, the Web site typically returns a certificate, identifying itself to the user when asking for secure information. If the user accepts the certificate, the certificate is placed into the list of trusted certificates of the browser so that a secure channel can be opened between the browser and that server. Like a Web browser, the Web Clipping Provider behaves as an HTTP client

to external Web sites. In order for the Web Clipping Provider to keep track of trusted sites, it makes use of a file that stores the certificates of those sites, namely the `ca-bundle.crt` file.

The shipped `ca-bundle.txt` is an exported version of the trusted server certificate file from Oracle Wallet Manager. The default trusted server certificate in Oracle Wallet Manager does not cover all possible server certificates that exist on the Web. For this reason, when a user navigates to a secure server using HTTPS, the user may get an SSL Hand-shake failed exception in the Web Clipping Studio. To solve this problem, the `ca-bundle.crt` file needs to be augmented with new trusted sites that are visited. As a Portal Administrator, you must do the following to extend the shipped `ca-bundle.crt` file:

1. Use a browser (preferably Internet Explorer) to download the root server certificate from each external HTTPS Web site in BASE64 format that is visited, and is missing from the trusted certificate file.

2. Use Oracle Wallet Manager to import each certificate.

3. Export the trusted server certificates into a file and replace the `ca-bundle.crt` file with that file.

More On OTN

For more information about Oracle Wallet Manager, see Chapter 17 "Using Oracle Wallet Manager" in *Oracle Advanced Security Administrator's Guide* in the Oracle9*i* Release 2 (9.2) documentation section on the Oracle Technology Network, http://otn.oracle.com.

6.1.9.2 Configuring Oracle Advanced Security for the Web Clipping Provider

The Web Clipping Provider can use Oracle Advanced Security Option (ASO) to secure and encrypt the channel between itself (at the middle-tier) and the database that hosts the Web Clipping Repository. As ASO is a feature available only on Oracle Application Server Enterprise Edition, or as an add-on option to the Standard Edition, this feature is disabled by default. To enable it, you must go to the Web Clipping Provider test page at

```
http://<host>:<port>/portalTools/webClipping/providers/webClipping
```

Under the **Provider Configuration** section, in the **Setting** column, there is a **Web Clipping Repository** field. Click its corresponding **Edit** link in the **Actions** column. In the **Repository Settings** section of the **Edit Provider: webClipping** page, select the **enable (secure database connections)** option in the **Advanced Security Option**

field, then select **OK** to save the settings and return to Web Clipping Provider test page.

In addition, you must set the following ASO configuration parameters in the `sqlnet.ora` file to ensure that the database connections created between the Web Clipping Provider and the database hosting the Web Clipping Repository are using ASO. See the *Oracle Advanced Security Administrator's Guide* for the list of values to use as all possible combinations of parameters are described in detail.

- `SQLNET.AUTHENTICATION_SERVICES` -- This parameter is used to select a supported authentication method in making database connections with ASO. See *Oracle Advanced Security Administrator's Guide* for more information about setting this parameter.

- `SQLNET.CRYPTO_SEED` -- This parameter denotes the cryptographic seed value (FIPS 140-1 setting), used in making database connections with ASO.

  See the *Oracle Advanced Security Administrator's Guide* for more information about setting this parameter.

> **Note:** When setting these parameters after the initial configuration (where the database parameters are already set up), the database connections are assumed to be open already. Because enabling ASO affects all connections made to the database, it is advisable to restart the OC4J instance containing the Web Clipping Provider to reset all the current connections to now use ASO. You would also need to do this when disabling ASO.

6.1.10 Securing the Federated Portal Adapter

The Federated Portal Adapter is a component of OracleAS Portal that allows portal instances to share their portlets through the Web portlet interface. For example, suppose that a user displays a page in one portal instance that contains a portlet whose source resides on another portal instance. When the Federated Portal Adapter on the remote portal receives the request for the portlet, it starts a session for the user on the remote portal. The portlet can then be run from the remote portal instance by the user. This scenario has a couple of security implications:

- Because the Federated Portal Adapter must create a session for the user on the remote portal, it would be best for the two portal instances to share the same single sign-on server. Otherwise, name collisions could occur when the Federated Portal Adapter attempts to log the user onto the remote portal instance.

■ Since the Federated Portal Adapter creates private portal sessions based on SOAP messages it receives, it is a potential security risk. A message authentication code must be used to ensure that any messages received by the Federated Portal Adapter emanate from a trusted source.

> **See Also:** Chapter 12, "Using the Federated Portal Adapter"

More On
Portal Center

You'll find additional information in the article "*How to Add Remote Portlets Using the Federated Portal Adapter*," on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

6.1.11 Securing OraDAV

WebDAV (World Wide Web Distributed Authoring and Versioning) is the IETF's standard for collaborative authoring on the Web. It defines a set of extensions to HTTP that facilitates collaborative editing and file management between users located remotely from each other on the Internet.

OraDAV, Oracle's implementation of WebDAV, is the mechanism used by the Oracle HTTP Server to communicate with WebDAV clients. OraDAV enables your users to connect to OracleAS Portal using their WebDAV clients. In terms of security, accessing OracleAS Portal through WebDAV presents two security issues for you to consider:

■ Expiration of OracleAS Portal session cookies for OraDAV

■ SSL and OraDAV

6.1.11.1 Session Cookie Expiration

The OraDAV configuration parameter, ORACookieMaxAge, specifies, in seconds, the length of time for which the DAV client should retain the cookie. The default value is 28800 (that is, 8 hours).

ORACookieMaxAge can be changed in Oracle Enterprise Manager or by directly editing it in the oradav.conf file located in *MID_TIER_ORACLE_HOME*/Apache/oradav/conf. If you choose to manually change the file, you must synchronize the changes with Dynamic Configuration Management. Once the change has been made in the configuration file, you need to restart the Oracle HTTP Server to have the changes take effect in the runtime system:

```
cd MID_TIER_ORACLE_HOME/dcm/bin
./dcmctl shell
```

```
- dcmctl> updateConfig -ct ohs
```

After you exit the dcmctl shell, execute the following command from *MID_TIER_ ORACLE_HOME*\opmn\bin to restart the Oracle HTTP Server:

```
opmnctl restartproc type=ohs
```

> **Note:** Not all WebDAV clients use cookies. Some perform their authentication on each request using HTTP basic authentication. A client may choose to record the user name and password for the duration of that WebDAV client session and thus only need to prompt the user once for their credentials. In either case, though, this behavior results in a slower response time from OracleAS Portal because every request from such clients must be authenticated, requiring added communication with the Oracle Internet Directory.

> **See Also:** *Oracle HTTP Server Administrator's Guide*

6.1.11.2 SSL and OraDAV

Access to OracleAS Portal through OraDAV using SSL was not certified for Oracle Application Server Release 2 (9.0.2). In Release 2 (9.0.4), SSL access is certified.

6.2 Configuring OracleAS Security Framework for OracleAS Portal

This section describes considerations for:

- Configuring OracleAS Security Framework Options for OracleAS Portal
- Configuring Oracle Identity Management Options for OracleAS Portal

6.2.1 Configuring OracleAS Security Framework Options for OracleAS Portal

For OracleAS Portal, the main consideration when configuring the OracleAS Security Framework is how to properly configure SSL. For a full description of SSL configuration for OracleAS Portal, refer to Section 6.3.2.1, "Configuring SSL for OracleAS Portal".

6.2.2 Configuring Oracle Identity Management Options for OracleAS Portal

As you configure OracleAS Portal for security, you should consider the following topics related Oracle Identity Management:

- Setting the Appropriate Naming and Nickname Attributes
- Configuring the Portal Administrator for Single Sign-On Administration

6.2.2.1 Setting the Appropriate Naming and Nickname Attributes

When deciding on the Directory Information Tree structure and the setting of the Oracle Context parameters for your Oracle Identity Management Infrastructure, you should consider making the naming attribute different from the nickname attribute. The naming attribute is used for the first attribute in the entry's Distinguished Name. By contrast, the nickname attribute holds the OracleAS Single Sign-On user name.

For OracleAS Portal to properly support renaming users by changing the value of the nickname attribute in the Oracle Internet Directory, the nickname attribute must be different than the naming attribute. By keeping the two separate, the Distinguished Name of the user's entry in the Oracle Internet Directory remains unchanged even when the value of the nickname attribute changes.

> **See Also:** *Oracle Identity Management Concepts and Deployment Planning Guide*

6.2.2.2 Configuring the Portal Administrator for Single Sign-On Administration

In previous releases of OracleAS Portal, the super user, PORTAL, was able to perform OracleAS Single Sign-On administration. With OracleAS Portal Release 9.0.4, the ability to perform OracleAS Single Sign-On administration out of the box is removed. The rationale for this change is that in enterprise settings it is not necessarily appropriate for an OracleAS Portal administrator to have permissions to perform Oracle Internet Directory and OracleAS Single Sign-On administration. Much like the discussion in the previous section, regarding the roles of the centralized Oracle Identity Management Infrastructure administrator and the departmental OracleAS Portal administrator, it may be inappropriate for the OracleAS Portal administrator to have the permissions to perform OracleAS Single Sign-On administration.

If you need to allow the OracleAS Portal account to perform OracleAS Single Sign-On administration, you need to explicitly give the user the privilege. This procedure can be done for each Identity Management Realm, or at the root Oracle Context level.

- *Oracle Identity Management Concepts and Deployment Planning Guide*

- *Oracle Application Server Single Sign-On Administrator's Guide*

6.3 Configuring OracleAS Portal Security

This section describes configuration considerations for OracleAS Portal.

- Configuring OracleAS Portal Security Options

- Configuring Options for OracleAS Security Framework

- Configuring OracleAS Portal Options for Database Security

6.3.1 Configuring OracleAS Portal Security Options

When you install OracleAS Portal, the installation process installs some default schemas of which you need to be aware.

OracleAS Portal Default Schemas

Table 6–17 describes the schemas created by default when OracleAS Portal is installed.

Table 6–17   Default OracleAS Portal Schemas

Schema	Description
PORTAL	Contains the OracleAS Portal database objects and code. This schema also represents the proxy user account that mod_plsql uses to connect to the database through the credentials provided in the corresponding DAD.
	To execute Web requested procedures, mod_plsql uses N-Tier authentication to connect to the schema to which the lightweight user accounts are assigned (by default, PORTAL_PUBLIC). As shown in Figure 6–11, access to the database of the portal user is proxied through the single schema user. By default, this entry is named something like `portal.iasdb.hostdomain.com`.
	The default name for this schema in a standard OracleAS Portal installation is PORTAL. If you want to give it another name, you must perform a custom installation.

Table 6–17 (Cont.) Default OracleAS Portal Schemas

Schema	Description
PORTAL_PUBLIC	Is the schema that all lightweight users are mapped to by default. All procedures publicly accessible through the Web are granted execute to PUBLIC, which makes them accessible through this schema.
	In a standard OracleAS Portal installation, this schema is named PORTAL_PUBLIC. If you want to give it another name, you must perform a custom installation.
PORTAL_DEMO	Is created to hold some demonstration code. The installation of this schema is optional.
PORTAL_APP	Is used for external JSP application authentication.

Figure 6–11   N-Tier Authentication By User Proxy
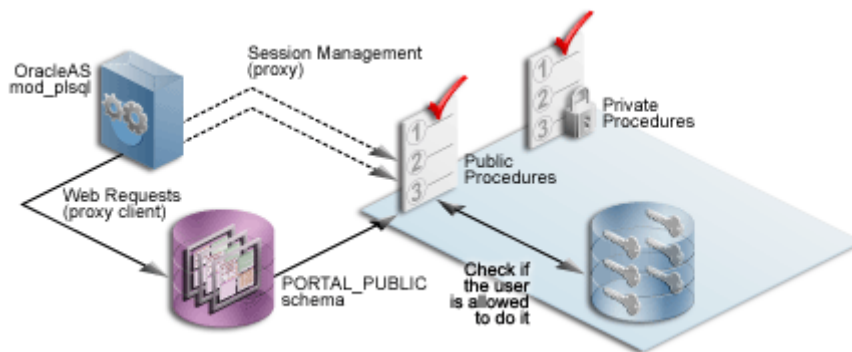
> **See Also:**   *Oracle Application Server 10g mod_plsql User's Guide*

6.3.2 Configuring Options for OracleAS Security Framework

When configuring OracleAS Portal, you should consider the following options that leverage the OracleAS Security Framework.

- Configuring SSL for OracleAS Portal

- Securing the Connection to Oracle Internet Directory (Optional)

- Changing Settings on the Global Settings Page

- Post-Installation Security Checklist

6.3.2.1 Configuring SSL for OracleAS Portal

The sections that follow provide an overview of the most common SSL configurations for OracleAS Portal and describe the procedures necessary to implement them:

- Overview of SSL Configurations

- SSL to OracleAS Single Sign-On

- SSL to OracleAS Web Cache

- SSL Throughout OracleAS Portal

- External SSL with Non-SSL Within Oracle Application Server

6.3.2.1.1 Overview of SSL Configurations OracleAS Portal uses a number of different components (such as the Parallel Page Engine, Oracle HTTP Server, and OracleAS Web Cache) each of which may act as a client or server in an HTTP communication. As a result, each component in OracleAS Portal's middle-tier must be configured individually for the protocols of HTTPS rather than HTTP.

Interacting with OracleAS Portal involves a number of distinct, "network hops." These include:

- Between the client browser and OracleAS Web Cache

- Between OracleAS Web Cache and Oracle HTTP Server

- Between the client browser and the Oracle HTTP Server of the OracleAS Single Sign-On/Oracle Internet Directory (or infrastructure) tier

- A loop back between the Parallel Page Engine (PPE) on the middle-tier and OracleAS Web Cache or front-end reverse proxy

- Between OracleAS Portal infrastructure and the Oracle Internet Directory

SSL Usage Restriction

Internal and external JSPs do not work with the Parallel Page Engine in a partial SSL configuration mode. They will work when SSL is used throughout OracleAS Portal or when SSL is implemented externally with a load balancing router. As a result, you should only use JSPs with the SSL configurations described in Section 6.3.2.1.2, "SSL to OracleAS Single Sign-On", Section 6.3.2.1.4, "SSL Throughout OracleAS Portal", and Section 6.3.2.1.5, "External SSL with Non-SSL Within Oracle Application Server".

6.3.2.1.2 SSL to OracleAS Single Sign-On

Figure 6–12 Secured Connection to OracleAS Single Sign-On
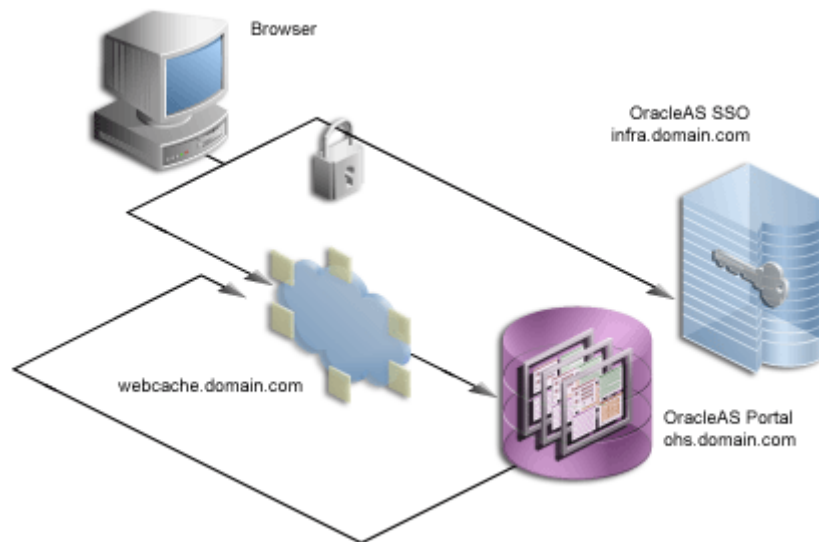
If any connection should be secured with SSL, it is the connection between the browser and OracleAS Single Sign-On. The password should be protected by SSL in transit between the browser and OracleAS Single Sign-On. For at least a minimal level of security, you should configure your installation with this option. All of the subsequent SSL configurations assume that you have configured SSL for OracleAS Single Sign-On.

To configure this option, refer to Enabling SSL in Chapter 9, Advanced Configurations, of the *Oracle Application Server Single Sign-On Administrator's Guide*. If you are going to configure OracleAS Single Sign-On behind a reverse proxy server, you should refer to Deploying OracleAS Single Sign-On with a Proxy Server in Chapter 9, Advanced Configurations, of the *Oracle Application Server Single Sign-On Administrator's Guide*.

> **Note:** In the previously described configuration of SSL, you must re-register the OracleAS Single Sign-On middle-tier partner application. Since the OracleAS Single Sign-On middle-tier partner application is still non-SSL, you must re-register it as non-SSL. Therefore, the re-registration of mod_osso needs to specify the non-SSL URL of the OracleAS Single Sign-On middle-tier for the `mod_osso_url` parameter to `ossoreg.jar`.
>
> Refer to the section titled "Registering mod_osso" in Chapter 4 of the *Oracle Application Server Single Sign-On Administrator's Guide* for more information.

After enabling SSL on OracleAS Single Sign-On following the steps listed in the *Oracle Application Server Single Sign-On Administrator's Guide*, you must complete the following configuration tasks on OracleAS Portal:

Re-registering the OracleAS Portal partner application

After OracleAS Single Sign-On is SSL-enabled, all OracleAS Single Sign-On partner applications need to be re-registered so that the updated SSL login URL is obtained by each partner application for subsequent authentication requests.

To re-register the OracleAS Portal partner applications, invoke OPCA with `ptlasst.csh` (on the OracleAS Portal middle-tier). On MS Windows, you use `ptlasst.bat`.

For example:

```
MID_TIER_ORACLE_HOME/assistants/opca/ptlasst.csh -i typical -mode MIDTIER -type
SSO -host portal_site_name -port portal_site_port
```

where:

portal_site_name is the `hostname.domain` of the portal middle-tier.

portal_site_port is the OracleAS Web Cache listen port or the reverse proxy listen port that the browser addresses.

> **Note:** If you have multiple virtual hosts configured in OracleAS Portal, you will need to re-register each of the virtual hostnames individually using the preceding command, replacing the *portal_site_name* accordingly for each virtual hostname.

Setting the OracleAS Single Sign-On Query Path URL

OracleAS Portal maintains the URL prefix of OracleAS Single Sign-On, which accesses certain information through HTTP calls from the database using the UTL_ HTTP package. These calls must be done through HTTP rather than HTTPS. As a result, even if OracleAS Portal and OracleAS Single Sign-On are configured to use HTTPS, you must still have access to an HTTP port on OracleAS Single Sign-On to support these interfaces. The calls made across this interface are required for the following reasons:

- Obtain the list of external applications to allow customization of the External Applications portlet.

- Perform the mapping of OracleAS Single Sign-On user names to external application user names.

To set this URL prefix, the OracleAS Single Sign-On Query Path URL, perform the following steps:

1. Log on to OracleAS Portal as the portal administrator.

2. Click the **Administer** tab.

3. Click the **Portal** tab.

4. Click **Global Settings** in the **Services** portlet.

5. Click the **SSO/OID** tab.

6. Edit the **Query Path URL Prefix** under the **SSO Server Settings**. Enter a URL for OracleAS Single Sign-On, for example:

   ```
   http://infra.domain.com:7777/pls/orasso
   ```

Conditionally Updating the DAS URL Base Entry in Oracle Internet Directory

After enabling the infrastructure tier's Oracle HTTP Server for SSL, you were asked to re-register all partner applications, which includes mod_osso on the infrastructure tier. You have the option of accessing DAS over non-SSL or SSL. The base URL for DAS is stored in Oracle Internet Directory, and this determines the URL that other applications render when providing links to DAS functionality.

If you want DAS accessed over SSL, then the re-registration of mod_osso needs to specify an SSL URL for the `mod_osso_url` parameter to `ossoreg.jar`. Refer to the section titled "Registering mod_osso" in Chapter 4 of the *Oracle Application Server Single Sign-On Administrator's Guide* for more information.

If you decide that you want to access DAS over SSL, then the `orcldasurlbase` attribute in the cn=OracleContext,cn=Products,cn=DAS,cn=OperationURLs entry needs to be updated in Oracle Internet Directory to reflect this fact. This attribute value is then used by OracleAS Portal for generating subsequent DAS URLs. This procedure assumes that the Oracle HTTP Server on the infrastructure tier is also listening on an HTTPS port.

1. For this step, you need Oracle Directory Manager (Integrated Management Tools : Oracle Directory Manager on Windows, or *INFRA_ORACLE_ HOME*/bin/oidadmin on UNIX). Run the Oracle Directory Manager and log in as cn=orcladmin.

2. Navigate to Entry Management, **cn=OracleContext > cn=Products > cn=DAS > cn=OperationURLs**.

3. Update the `orcldasurlbase` entry to reflect the HTTPS port being used on the infrastructure tier, that is, `https://infrahost:port`.

Once the entry is updated in Oracle Internet Directory, you must force a refresh of the OracleAS Portal cache, which holds the relevant Oracle Internet Directory information.

1. Logon to OracleAS Portal as a user with administrator privileges.

2. Go to the **Builder**.

3. Click the **Administration** tab.

4. From the **Portal** tab, open **Global Settings** and navigate to the **SSO/OID** tab.

5. Scroll to the bottom of the page.

6. Check **Refresh Cache** for the Oracle Internet Directory parameters.

7. Click **Apply**.

8. The page should refresh with the appropriate value in the **DAS Host Name** field.

Re-registering the Oracle HTTP Server Partner Application

You now need to register the secured request with OracleAS Single Sign-On by configuring it as a partner application. The script `ossoreg` performs this registration. `ossoreg` is located on the middle-tier in *MID_TIER_ORACLE_ HOME*/sso/lib.

1. Ensure that you have your environment configured properly to run `ossoreg`:

   ```
   ORACLE_HOME=MID_TIER_ORACLE_HOME
   ```

```
LD_LIBRARY_PATH=ORACLE_HOME/lib
```

2. Run `ossoreg`. The following example illustrates the usage of `ossoreg`.

```
MID_TIER_ORACLE_HOME/jdk/bin/java -jar
  MID_TIER_ORACLE_HOME/sso/lib/ossoreg.jar -site_name abc.com -mod_osso_url
  http://www.abc.com:7777 -config_mod_osso TRUE -oracle_home_path
  MID_TIER_ORACLE_HOME  -u install_user -config_file
  MID_TIER_ORACLE_HOME/Apache/Apache/conf/osso/osso.conf
  -admin_info cn=orcladmin
```

Refer to the section titled "Registering mod_osso" in Chapter 4 of the *Oracle
Application Server Single Sign-On Administrator's Guide* for more information.

At this point, configuration is complete for SSL communication to OracleAS Single
Sign-On.

6.3.2.1.3   SSL to OracleAS Web Cache

Figure 6–13   Secured Connection to OracleAS Web Cache
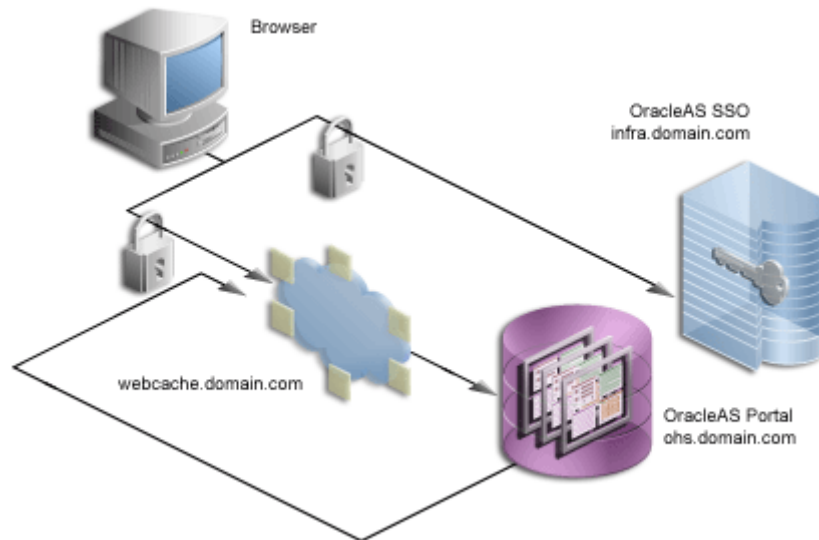
Once you secure the OracleAS Single Sign-On communication, the next option is to
secure the communication to the front door of OracleAS Portal, which is OracleAS
Web Cache. In this configuration, OracleAS Web Cache can forward the request to
the Oracle HTTP Server, which is acting as the OracleAS Portal middle-tier, using

HTTP for better performance. Similarly, the Parallel Page Engine requests for portlet content that loop back to OracleAS Web Cache can request the content using HTTP.

Creating a Wallet

The various components of OracleAS Portal use the Oracle Wallet Manager to store the certificates for the secure communication. The first step in this process is to obtain a certificate from a Certificate Authority (for example, Verisign, GTE CyberTrust).

Obtaining a Certificate

In order to obtain a digital certificate from the relevant signing authority, you submit a Certificate Request (CR) uniquely identifying your server to the signing authority.

1. Open the Oracle Wallet Manager in the middle-tier *MID_TIER_ORACLE_HOME*. On UNIX, enter `owm` from the command prompt. On Windows, invoke Oracle Wallet Manager from the Start menu.

2. Choose **Wallet > New**.

> **Note:** This action may result in a prompt if the default wallet directory does not exist. You may choose to take the default, **Yes**, and create the directory or not. If you choose to create the directory, you must confirm that the operating system account owning the Oracle Wallet Manager has the appropriate write permissions to the parent directory.

On UNIX the wallet is stored in the following location by default:

```
/etc/ORACLE/WALLETS/<Account Name creating the Wallet>
```

On MS Windows the wallet is stored in the following location by default:

```
\Documents And Settings\<Account Name creating the Wallet>\ORACLE\WALLETS
```

3. Create a password for the wallet.

4. Click **Yes** to accept the option to create a CR.

5. Fill out the **Certificate Request** dialog, shown in the following image, with details that uniquely identify your server. Provide the server name as the value of Common Name, for example, `www.abc.com`.

> **Note:** In an installation where the middle-tier components reside
> on one machine, you can share one certificate for all of the
> components. If your middle-tier is distributed across multiple
> machines, you will need to request one certificate for each server.
> The reason for this requirement is that the Common Name typically
> refers to a specific machine (server and domain), which means you
> need a separate certificate for each machine.

6. Click **OK**. A dialog will inform you that the certificate request was created
   successfully. The Certificate node in the Wallet Navigator will change to
   `Requested`.

7. Save the wallet in a convenient directory, for example:

   `MID_TIER_ORACLE_HOME\webcache\wallets\portalssl`

8. Send the CR to the chosen Certificate Authority (CA).

Cutting and Pasting   Depending on the CA, you may need to cut and paste the
certificate request in their Web page or export the CR to a file for subsequent
uploading to the site.

1. Select the **Certificate** node in the Wallet Navigator.

2. Highlight the Certificate text in the **Certificate Request** field. Make sure to
   include the `BEGIN/END NEW CERTIFICATE REQUEST` lines.

3. Copy and paste into the **Certificate Request** field on the CA's Web site.

Exporting Certificate Request   To export the certificate request:

1. Choose **Operations > Export Certificate Request**.

2. Choose the Name and Location of the CR file. A Status Line Message will confirm the successful export of the CR.

3. Once exported, the CR can be uploaded to the CA's Web site.

Managing Trusted Certificates

Once the CA has processed the request, the User Certificate is forwarded to you either as text within an e-mail or as a simple file that is downloaded from a given Web page. Note, if you are using a trial Root Certificate or have chosen a CA which is not currently installed in the Oracle Wallet Manager, you must first import the CA's Trusted Certificate before importing your server specific User Certificate.

Importing Trusted Certificate (if necessary)   To import the trusted certificate:

1. Choose **Operations > Import Trusted Certificate**.

2. Based on the CA, choose **Paste the Certificate** or **Select a file that contains the certificate**.

3. Select the appropriate certificate file or paste in the text from the e-mail. Oracle Wallet Manager expects base-64 encoded root certificates. If you do not have a base-64 encoded root certificate, then you must perform the steps described in Changing Trusted Certificate Format (if necessary).

4. Click **OK**.

A status line message should appear indicating that the certificate was successfully imported. When you import the server specific User Certificate, the certificate node in the tree structure should also display as **Ready**.

Changing Trusted Certificate Format (if necessary)   If the certificate import fails, then it is possible that the Certificate is in a format that the Oracle Wallet Manager does not support. In this case, you need to convert it to a supported format before importing. The easiest way to do this is through the certificate Import/Export Wizards within a browser. The following steps are for the Microsoft Internet Explorer Browser.

1. In MS Internet Explorer, choose **Tools > Internet Options...**.

2. Click the **Content** tab.

3. Click **Certificates...**.

4. Click the **Trusted Root Certification Authorities** tab.

5. Select **Import...** and follow the wizard to import the certificate.

6. Highlight the newly imported certificate from the list.

7. Click **Export...** and follow the wizard to the Export File Format page.

8. Choose **Base-64 encoded X.509**.

9. Click **Next** and give the certificate a file name.

10. Click **Next**.

11. Click **Finish**.

12. In Oracle Wallet Manager, choose **Operations > Import Trusted Certificate**.

Once the Trusted Root Certificate has been successfully imported into the Oracle Wallet Manager you may then import the server specific User Certificate.

Importing Server's User Certificate   To import the server's user certificate:

1. Choose **Operations > Import User Certificate**.

2. Based on the CA, choose **Paste the Certificate** or **Select a file that contains the certificate**.

3. Select the appropriate certificate file or paste in the text from the e-mail.

4. Click **OK**.

A status line message should appear indicating that the User Certificate has been successfully imported.

Having imported the certificate, it is important to save the wallet with the Autologin functionality enabled. This step is required because OracleAS Web Cache accesses the wallet as the process starts and the wallet password is not held by OracleAS Web Cache. If this property is not set, OracleAS Web Cache immediately shuts down if running in SSL mode.

1. Choose the Trusted Certificate you just imported from the list in the Oracle Wallet Manager.

2. Check **Wallet > AutoLogin**, if it is not already checked.

3. Choose **Wallet > Save**.

Securing OracleAS Web Cache

This sections that follow describe how to configure OracleAS Web Cache to accept SSL connections.

> **Note:** Changing OracleAS Web Cache settings (for example, Listening Port) can change the OracleAS Portal URL. If you do this, mobile settings need to be updated manually. For more information, see Section C.8, "Using the cfgiasw Script to Configure Mobile Settings".

Configuring OracleAS Web Cache SSL Port

1. From the OracleAS Web Cache administration page, click the **Listening Ports** link in the **Ports** section.

2. To add the SSL port, click **Add...** and enter the following information:

   - **IP Address:** ANY

   - **Port Number:** SSL port that Web Cache will listen on

   - **Protocol:** HTTPS

   - **Require Client-Side Certificate:** No (unchecked)

   - **Wallet:** Path to the Oracle Wallet directory containing the SSL server certificate

3. Click **Submit**.

For more information on the procedure in the preceding text, refer to Task 3: Configure HTTPS Operations Ports for the Cache in Chapter 8, Specialized Configurations, of the *Oracle Application Server Web Cache Administrator's Guide*.

Defining a Site for the Published SSL Hostname and Port

Since there is no out-of-box default SSL configuration, you need to add a Site definition for the SSL-based Site that OracleAS Web Cache will be caching.

1. From the OracleAS Web Cache administration page, click **Site Definitions** under **Origin Servers, Sites, and Load Balancing**.

2. Define a Site where **Host Name** is the hostname seen by the browser.

   This is the load balancer or reverse proxy server name for configurations that use a load balancer device or other reverse proxy, or it is the OracleAS Web

Cache hostname in a configuration where OracleAS Web Cache receives browser requests.

3. Set **Port** to the HTTPS port addressed by the browser requests.

4. Enter Site information as follows:

   HTTPS Only Prefix: Leave blank

   Client-Side Certificate: Not required

   Default Site: Yes

   Create Alias from Site Name with/without www: No

5. Click **Submit.**

6. Remove any sites that are no longer applicable to your modified configuration, including the default, out-of-the-box non-SSL site.

For more information on the procedure in the preceding text, refer to:

- Task 4: Create a Site for HTTPS Requests for the Cache in Chapter 8, Specialized Configurations, of the *Oracle Application Server Web Cache Administrator's Guide*.

- Create Site Definitions in Chapter 7, Basic Setup and Configuration, of the *Oracle Application Server Web Cache Administrator's Guide*.

Adding Site Aliases for Each Potential Port in the Configuration

Site aliases are necessary if content is cached across a number of different hostnames, or ports, or both, but which actually refer to the same logical content. For example, when the PPE makes a request for some portlet, and this portlet is requested on a non-SSL port, but the main Site is accessed over SSL, then an alias entry is needed. This equates the content accessed through the Site with SSL, to the content accessed over non-SSL. This way, invalidation requests that are sent to invalidate the content, will invalidate the content that is cached over either form of URL.

You need to create an alias for the non-SSL OracleAS Web Cache listening port for the OracleAS Web Cache SSL site. To create a site alias:

1. From the OracleAS Web Cache administration page, return to the **Site Definitions** page, select the newly added site, and click **Add Alias**.

2. Enter the same hostname as used by the site entry, and provide the non-SSL port that the PPE will use to request portlets from OracleAS Web Cache. Click **Apply Changes**.

3. Restart the server after making the necessary additions.

For more information on the procedure in the preceding text, refer to Create Site Definitions in Chapter 7, Basic Setup and Configuration, of the *Oracle Application Server Web Cache Administrator's Guide*.

Adding Site to Server Mappings of the New Site to the Origin Server

After adding a new Site definition and associated aliases, you must add the Site to Server mapping for the newly defined Site to the origin server. To do this:

1. Select the **Site** you are mapping from the drop-down list.

2. Check the appropriate **Origin Server** to which requests should be routed for content.

For more information on the procedure in the preceding text, refer to Map Sites to Origin Servers in Chapter 7, Basic Setup and Configuration, of the *Oracle Application Server Web Cache Administrator's Guide*.

Securing the Parallel Page Engine

In this configuration, you need to configure the PPE to make portlet requests using HTTP requests. The sections that follow describe how to implement a partial SSL PPE configuration for this purpose.

1. Open the `web.xml` file associated with the OC4J_PORTAL instance on the middle-tier.

```
MID_TIER_ORACLE_HOME\j2ee\OC4J_Portal\applications\portal\
portal\WEB-INF\web.xml
```

2. Add `useScheme` and `usePort` in additional `<init-param>` blocks in `web.xml`. The `useScheme http` indicates that the http protocol should be used for PPE loop backs and `usePort` indicates which port these non-SSL loop backs should use. The HTTP port you specify for `usePort` should be the OracleAS Web Cache non-SSL http port. For example:

```
<servlet>
<servlet-name>page</servlet-name>
    <servlet-class>oracle.webdb.page.ParallelServlet</servlet-class>
    <init-param>
      <param-name>useScheme</param-name>
      <param-value>http</param-value>
    </init-param>
    <init-param>
      <param-name>usePort</param-name>
```

```
      <param-value>7777</param-value>
   </init-param>
 </servlet>
```

3. (Optional) If you want the PPE to trust only specific certificates, add `x509certfile` in an additional `<init-param>` block in `web.xml`. For example:

```
<servlet>
<servlet-name>page</servlet-name>
   <servlet-class>oracle.webdb.page.ParallelServlet</servlet-class>
     <init-param>
        <param-name>x509certfile</param-name>
        <param-value>C:\mySSLconfig\trustedCerts.txt</param-value>
     </init-param>
 </servlet>
```

> **Note:** If you choose not to implement `x509certfile`, the PPE trusts any SSL certificate.

Re-registering the Oracle HTTP Server Partner Application

You now need to register the secured request with OracleAS Single Sign-On by configuring it as a partner application. The script `ossoreg` performs this registration. `ossoreg` is located on the middle-tier in *MID_TIER_ORACLE_HOME*/sso/lib.

1. Ensure that you have your environment configured properly to run `ossoreg`:

```
ORACLE_HOME=MID_TIER_ORACLE_HOME
LD_LIBRARY_PATH=ORACLE_HOME/lib
```

2. Run `ossoreg`. The following example illustrates the usage of `ossoreg`.

```
MID_TIER_ORACLE_HOME/jdk/bin/java -jar
  MID_TIER_ORACLE_HOME/sso/lib/ossoreg.jar -site_name abc.com -mod_osso_url
  https://www.abc.com:4443 -config_mod_osso TRUE -oracle_home_path
  MID_TIER_ORACLE_HOME  -u install_user -config_file
  MID_TIER_ORACLE_HOME/Apache/Apache/conf/osso/osso.conf
  -admin_info cn=orcladmin
```

Refer to the section titled "Registering mod_osso" in Chapter 4 of the *Oracle Application Server Single Sign-On Administrator's Guide* for more information.

At this point, configuration is complete for SSL communication to OracleAS Web Cache.

Specifying the OracleAS Portal Published Address and Protocol

To specify the published address for OracleAS Portal with the modified port for SSL, you need to use Oracle Enterprise Manager as follows:

1. Navigate to the Oracle Enterprise Manager Application Server Control.

2. Click the Standalone Instance with the Oracle Application Server that is running the OracleAS Portal middle-tier.

3. Click the OracleAS Portal system component.

4. Under **Administration**, click **Portal Web Cache Settings**.

5. For **Listen Port**, enter the OracleAS Web Cache SSL port number.

6. For **Listening Port SSL Enabled**, choose **Yes**.

7. Click **OK**. The OracleAS Portal Repository is updated with the setting and the Oracle Enterprise Manager target instance is updated to use HTTPS for its URL tests.

   If at a later time you choose to switch to HTTP, you would perform this same procedure and return Listening Port SSL Enabled to No.

   > **Note:** This procedure updates the settings in the `iasconfig.xml` file.

   > **See Also:** For more information about `iasconfig.xml`, see Appendix A, "Using the Portal Dependency Settings File".

8. Edit the *MID_TIER_ORACLE_HOME*/Apache/Apache/conf/httpd.conf file as follows:

   a. Specify 4443 for the **Port** directive.

   b. At the bottom of the file, add a line:

   ```
   SimulateHttps On
   ```

   To make the SimulateHttps directive take effect, you must load mod_certheaders in the Oracle HTTP Server by adding one of the following directives before the SimulateHttps directive:

On UNIX:

```
LoadModule certheaders_module libexec/mod_certheaders.so
```

On MS Windows:

```
LoadModule certheaders_module modules/ApacheModuleCertHeaders.dll
```

For more information, refer to the "mod_certheaders" section of Chapter 8, Oracle HTTP Server Modules, in the *Oracle HTTP Server Administrator's Guide*.

c. Save the file.

9. Run the following command to synchronize the manual configuration changes:

```
MID_TIER_ORACLE_HOME/dcm/bin/dcmctl updateconfig
```

10. Restart your Oracle Application Server instance:

```
MID_TIER_ORACLE_HOME/opmn/bin/opmnctl stopall
MID_TIER_ORACLE_HOME/opmn/bin/opmnctl startall
```

6.3.2.1.4 SSL Throughout OracleAS Portal

Figure 6–14   Secured Connections Throughout the System
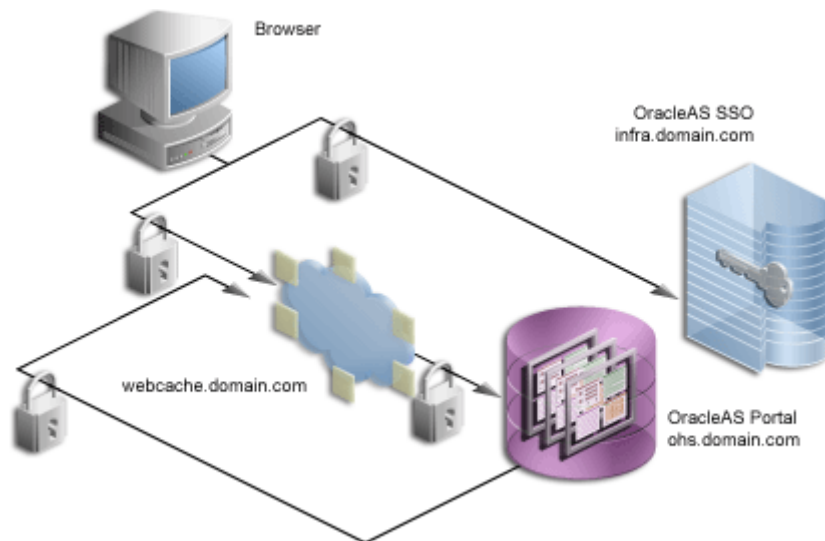
For installations that require the utmost security, it is possible to configure SSL throughout the system. In this configuration, the loop back from the PPE to OracleAS Web Cache uses a wallet and the hop between the PPE and the Web providers uses a certificate directly rather than through a wallet.

> **Note:** If you have already followed the steps described in Section 6.3.2.1.3, "SSL to OracleAS Web Cache", you must revert all the changes you have made prior to configuring SSL throughout OracleAS Portal.

Creating a Wallet

It is possible to share a single wallet across both the listener and origin server (and all other available ports in OracleAS Web Cache) if OracleAS Web Cache and the Oracle HTTP Server are on the same machine. Conversely, specific wallets may be created for each port. In this case the two servers/ports will be sharing the same wallet specified earlier.

In some cases, such as when the Oracle HTTP Server is on a different machine than OracleAS Web Cache, you need to create a separate wallet for the Oracle HTTP Server. For this situation, refer to the steps in "Creating a Wallet" on page 6-90 to create the wallet for the Oracle HTTP Server.

In the default case, where the Oracle HTTP Server is on the same machine as OracleAS Web Cache, you can share the wallet between the two.

Securing the Oracle HTTP Server

You need to configure the Oracle HTTP Server as the OracleAS Web Cache's origin server to accept HTTPS-based communication. The Oracle HTTP Server implements SSL by the use of mod_ssl. As such, the configuration to use HTTPS is fairly straightforward.

> **Note:** The default installation of the Oracle HTTP Server provides a basic implementation of SSL with a demo certificate. For production use, you should obtain a server certificate from a certificate authority following the instructions in "Creating a Wallet" on page 6-90.

The SSL configuration of the Oracle HTTP Server is defined within `ssl.conf`. This file may be edited directly or by using the Advanced Server Properties page under

the Oracle HTTP Server node of the appropriate Oracle Application Server instance within the Oracle Enterprise Manager Administration pages. If you edit the files manually, it is recommended that you run the `dcmctl` utility with the following options to make sure that the file is synchronized with the DCM repository.

MID_TIER_ORACLE_HOME/dcm/bin/dcmctl updateConfig -ct ohs

1. Open `ssl.conf` in *MID_TIER_ORACLE_HOME*/Apache/Apache/conf.

2. Search for the following directives and change the values accordingly:

Table 6–18   Wallet Entries in ssl.conf

Default Entry	Updated Entry
SSLWallet file:	SSLWallet file:
MID_TIER_ORACLE_ HOME/Apache/Apache/conf/ ssl.wlt/default	/Directory where the wallet has been saved
SSLWalletPassword	SSLWalletPassword
.... (hashed out)	password used when creating the wallet

3. In `ssl.conf` in *MID_TIER_ORACLE_HOME*/Apache/Apache/conf, verify that the default virtual host for SSL communication specifies the correct, preallocated port number for SSL.

   > **Note:**   When setting up OracleAS Portal to communicate through HTTPS, it is common to configure both the middle and infrastructure tiers to operate in this mode. You must leave an HTTP port open on the infrastructure tier for OracleAS Portal to communicate with OracleAS Single Sign-On for External Application information. This call is made directly from the repository using the UTL_HTTP package.

4. Ensure that the start mode of the Oracle HTTP Server is set to `ssl-enabled` in *MID_TIER_ORACLE_HOME*/opmn/conf/opmn.xml. For example:

```
<ias-component id="HTTP_Server">
  <process-type id="HTTP_Server" module-id="OHS">
    <module-data>
```

```
            <category id="start-parameters">
                <data id="start-mode" value="ssl-enabled"/>
            </category>
        </module-data>
        <process-set id="HTTP_Server" numprocs="1"/>
    </process-type>
</ias-component>
```

Securing OracleAS Web Cache

The sections that follow describe how to configure OracleAS Web Cache to accept SSL connections and forward SSL requests to the SSL-enabled origin server.

> **Note:** Changing OracleAS Web Cache settings (for example, Listening Port) can change the OracleAS Portal URL. If you do this, mobile settings need to be updated manually. For more information, see Section C.8, "Using the cfgiasw Script to Configure Mobile Settings".

Configuring the OracleAS Web Cache SSL Port

1. From the OracleAS Web Cache Administration page, click the **Listening Ports** link in the **Ports** section.

2. To add the SSL port, click **Add...** and enter the following information:

   IP Address: ANY

   Port Number: SSL port that Web Cache is listening on

   Protocol: HTTPS

   Require Client-Side Certificate: No (unchecked)

   Wallet: Path to the SSL server certificate

3. Click **Submit**.

For more information on the procedure in the preceding text, refer to Task 3: Configure HTTPS Operations Ports for the Cache in Chapter 8, Specialized Configurations, of the *Oracle Application Server Web Cache Administrator's Guide*.

Adding the SSL Origin Server

To add the SSL origin server:

1. From the OracleAS Web Cache administration page, click **Origin Servers** under **Origin Servers, Sites, and Load Balancing**.

2. Click **Add...** to add the SSL Origin Server.

3. Enter the information as follows:

   Host Name: Physical hostname of the machine in which Oracle HTTP Server is running

   Port: Oracle HTTP Server's SSL port

   Routing: Enable

   Capacity: 100

   Failover Threshold: 5

   Ping URL: /

   Ping Interval: 10

   Protocol: HTTPS

4. Click **Submit**.

For more information on the procedure in the preceding text, refer to Task 9: Configure Origin Server, Load Balancing, and Failover Settings in Chapter 7, Basic Configuration and Setup, of the *Oracle Application Server Web Cache Administrator's Guide*.

Defining a Site for the Published SSL Host Name and Port

Since there is no out-of-box default SSL configuration, you need to add a site definition for the SSL-based Site that OracleAS Web Cache will be caching.

1. From the OracleAS Web Cache Administration page, click **Site Definitions** under **Origin Servers, Sites, and Load Balancing**.

2. Define a site where **Host Name** is the hostname seen by the browser, which would be the OracleAS Web Cache hostname.

3. Set **Port** to the HTTPS port addressed by the browser requests, which would be the OracleAS Web Cache SSL listen port.

4. Enter site information as follows:

   HTTPS Only Prefix: Leave blank

   Client-Side Certificate: Not required

Default Site: Yes

Create Alias from Site Name with/without www: No

5. Click **Submit.**

6. Remove any sites that are no longer applicable to your modified configuration.

For more information on the procedure in the preceding text, refer to:

■ Task 4: Create a Site for HTTPS Requests for the Cache in Chapter 8, Specialized Configurations, of the *Oracle Application Server Web Cache Administrator's Guide*.

■ Create Site Definitions in Chapter 7, Basic Setup and Configuration, of the *Oracle Application Server Web Cache Administrator's Guide*.

Adding Site to Server Mappings of the New Site to the Origin Server

After adding a new site definition, you must add the site to server mapping for the newly defined site to the origin server. To do this:

1. Select the **Site** you are mapping from the drop-down list, which should be the OracleAS Web Cache site with the SSL port.

2. Check the **Origin Server** representing the OracleAS Portal SSL port, to which requests should be routed for content.

For more information on the procedure in the preceding text, refer to Map Sites to Origin Servers in Chapter 7, Basic Setup and Configuration, of the *Oracle Application Server Web Cache Administrator's Guide*.

Securing the Parallel Page Engine

In this configuration, SSL is used throughout as the request comes to OracleAS Web Cache through HTTPS and the PPE loops back over HTTPS as well. The server specifies the chain that goes with its certificate. As long as the chain is valid and leads to a self-signed root certificate, it is validated without determining whether it is trusted, assuming that you have not loaded any trust points into it.

To implement this configuration, perform the following steps on the OracleAS Portal middle-tier:

1. Open the `ssl.conf` file.

   MID_TIER_ORACLE_HOME/Apache/Apache/conf/ssl.conf

2. Add `SSLWallet` and `SSLWalletPassword`. For example:

```
SSLWallet file:/usr/local/adeviews/webdb/webdb_3000_ias902PW/Apache/Apache/conf/ssl.wlt/default
```

```
SSLWalletPassword serverWalletPassword
```

> **Note:** The previous example of `SSLWalletPassword` shows the password as plaintext. In many cases, you may want to obfuscate the password using the `iasobf` utility.

> **See Also:** For more information about `SSLWallet` and `SSLWalletPassword`, and `httpd.conf`, see the *Oracle HTTP Server Administrator's Guide*.

3. Open the `web.xml` file associated with the OC4J_PORTAL instance on the middle-tier.

   ```
   MID_TIER_ORACLE_HOME\j2ee\OC4J_Portal\applications\portal\portal\
   WEB-INF\web.xml
   ```

4. Add `httpsports` in an additional `<init-param>` block in `web.xml`. This should point to the OracleAS Web Cache HTTPS listening port. For example:

   ```
   <servlet>
   <servlet-name>page</servlet-name>
      <servlet-class>oracle.webdb.page.ParallelServlet</servlet-class>
      <init-param>
        <param-name>httpsports</param-name>
        <param-value>4443</param-value>
      </init-param>
    </servlet>
   ```

   > **Note:** If your current `web.xml` file contains the useScheme and usePort directives, you need to remove them. The configuration with SSL throughout should only use the `httpsports` directive.

5. (Optional) If you want the PPE to trust only specific certificates, add `x509certfile` in an additional `<init-param>` block in `web.xml`. For example:

   ```
   <servlet>
   <servlet-name>page</servlet-name>
      <servlet-class>oracle.webdb.page.ParallelServlet</servlet-class>
        <init-param>
          <param-name>x509certfile</param-name>
   ```

```
        <param-value>C:\mySSLconfig\trustedCerts.txt</param-value>
      </init-param>
  </servlet>
```

> **Note:** If you choose not to implement `x509certfile`, the PPE trusts any SSL certificate.

Securing the Event Servlet

The Smart Page functionality of OracleAS Portal allows for the publishing of Events and Page Parameters to allow portlets to communicate information to each other. The Event servlet, which runs in the same container as the Parallel Page Engine itself, implements this functionality. Since the Event servlet also must create the appropriate ACTION URLs resulting from the user interaction, it also requires knowledge of the protocol used to generate the page. The required parameters to indicate the use of HTTPS are the same ones used for the Page servlet.

1. Open the `web.xml` file associated with the OC4J_PORTAL instance on the middle-tier.

MID_TIER_ORACLE_HOME\j2ee\OC4J_
Portal\applications\portal\portal\WEB-INF\web.xml

2. Add an additional `<init-param>` block to the file to indicate the ports that are using HTTPS. This block should point to the OracleAS Web Cache HTTPS listening port.

```
<servlet>
<servlet-name>event</servlet-name>
   <servlet-class>oracle.webdb.event.EventServlet</servlet-class>
     <init-param>
       <param-name>httpsports</param-name>
       <param-value>4443</param-value>
     </init-param>
 </servlet>
```

Specifying OracleAS Portal Published Address and Protocol

To specify the published address for OracleAS Portal with the modified port for SSL, you need to use Oracle Enterprise Manager as follows:

1. Navigate to the Oracle Enterprise Manager Application Server Control.

2. Click the Standalone Instance with the Oracle Application Server that is running the OracleAS Portal middle-tier.

3. Click the OracleAS Portal system component.

4. Under **Administration**, click **Portal Web Cache Settings**.

5. For **Listening Port SSL Enabled**, choose **Yes**.

6. Click **OK**. The OracleAS Portal Repository is updated with the setting and the Oracle Enterprise Manager target instance is updated to use HTTPS for its URL tests.

   If at a later time you choose to switch to HTTP, you would perform this same procedure and return Listening Port SSL Enabled to No.

   > **Note:** This procedure updates the settings in the `iasconfig.xml` file.

   > **See Also:** For more information about `iasconfig.xml`, see Appendix A, "Using the Portal Dependency Settings File".

Re-registering the Oracle HTTP Server Partner Application

You now need to register the secured request with OracleAS Single Sign-On by configuring it as a partner application. The script `ossoreg` performs this registration. `ossoreg` is located on the middle-tier in *MID_TIER_ORACLE_HOME*/sso/lib.

1. Ensure that you have your environment configured properly to run `ossoreg`:

```
ORACLE_HOME=MID_TIER_ORACLE_HOME
LD_LIBRARY_PATH=ORACLE_HOME/lib
```

2. Run `ossoreg`. The following example illustrates the usage of `ossoreg`.

```
MID_TIER_ORACLE_HOME/jdk/bin/java -jar
  MID_TIER_ORACLE_HOME/sso/lib/ossoreg.jar -site_name abc.com -mod_osso_url
  https://www.abc.com:4443 -config_mod_osso TRUE -oracle_home_path
  MID_TIER_ORACLE_HOME  -u install_user -config_file
  MID_TIER_ORACLE_HOME/Apache/Apache/conf/osso/osso.conf
  -admin_info cn=orcladmin
```

Refer to the section titled "Registering mod_osso" in Chapter 4 of the *Oracle Application Server Single Sign-On Administrator's Guide* for more information.

Associating the Federated Portal Adapter with SSL

The Federated Portal Adapter uses the Oracle HTTP Server rewrite rules to simplify URLs for registering providers. By default, these rewrite rules are only specified for HTTP communication.

1. Copy these rewrite rules from `portal.conf` to the Oracle HTTP Server configuration files. The rewrite rules in `portal.conf` are as follows:

```
# Portal PL/SQL Adapter URL Simplification
RewriteEngine on
# This is to match '/adapter/<dad_name>/<schema_name>' and an optional trailing '/'
RewriteRule ^/adapter/(.+)/([^/]+)/?$ /pls/$1/!$2.wwpro_app_adapter.process_request [PT]
# This is to match '/adapter/<dad_name>' and an optional trailing '/'
RewriteRule ^/adapter/([^/]+)/?$ /pls/$1/!$1.wwpro_app_adapter.process_request [PT]
```

You need to add these same rules to the virtual hosts section of your Oracle HTTP Server file as follows:

```
## SSL Virtual Host Context
##
#
# NOTE: this value should match the SSL Listen directive set previously in this
# file otherwise your virtual host will not respond to SSL requests.
#
<VirtualHost _default_:3011>
  #  General setup for the virtual host
  DocumentRoot /usr/local/adeviews/webdb/webdb_3000_ias902PW/Apache/Apache/htdocs
  ServerName host1.abc.com
  ServerAdmin you@your.address
  ErrorLog /usr/local/adeviews/webdb/webdb_3000_ias902PW/Apache/Apache/logs/error_log
  TransferLog "/usr/local/adeviews/webdb/webdb_3000_ias902PW/Apache/Apache/logs/access_log"
  Port 3001
  SSLEngine on
  SSLCipherSuite
SSL_RSA_WITH_RC4_128_MD5:SSL_RSA_WITH_RC4_128_SHA:SSL_RSA_WITH_3DES_EDE_CBC_SHA:SSL_RSA_WITH_DES_CBC_SHA:SSL_
RSA_EXPORT_WITH_RC4_40_MD5:S

SL_RSA_EXPORT_WITH_DES40_CBC_SHA
  SSLWallet file:/usr/local/adeviews/webdb/webdb_3000_ias902PW/Apache/Apache/conf/ssl.wlt/default

  <Files ~ "\.(cgi|shtml)$">
   SSLOptions +StdEnvVars
  </Files>
  <Directory /usr/local/adeviews/webdb/webdb_3000_ias902PW/Apache/Apache/cgi-bin>
   SSLOptions +StdEnvVars
  </Directory>

        SetEnvIf User-Agent ".*MSIE.*" nokeepalive ssl-unclean-shutdown
        CustomLog /usr/local/adeviews/webdb/webdb_3000_ias902PW/Apache/Apache/logs/ssl_request_log "%t %h
```

```
%{SSL_PROTOCOL}x
%{SSL_CIPHER}x \"%r\" %b"

        RewriteEngine on
        # This is to match '/adapter/<dad_name>/<schema_name>' and an optional trailing '/'
        RewriteRule ^/adapter/(.+)/([^/]+)/?$ /pls/$1/!$2.wwpro_app_adapter.process_request [PT]
        # This is to match '/adapter/<dad_name>' and an optional trailing '/'
        RewriteRule ^/adapter/([^/]+)/?$ /pls/$1/!$1.wwpro_app_adapter.process_request [PT]

    </VirtualHost>
```

2. Run the following command to synchronize the manual configuration changes:

*MID_TIER_ORACLE_HOME*/dcm/bin/dcmctl updateconfig

3. Restart your Oracle Application Server instance:

MID_TIER_ORACLE_HOME/opmn/bin/opmnctl stopall
MID_TIER_ORACLE_HOME/opmn/bin/opmnctl startall

At this point, configuration is complete for SSL communication throughout OracleAS Portal.

6.3.2.1.5 External SSL with Non-SSL Within Oracle Application Server

Figure 6–15 External SSL Only
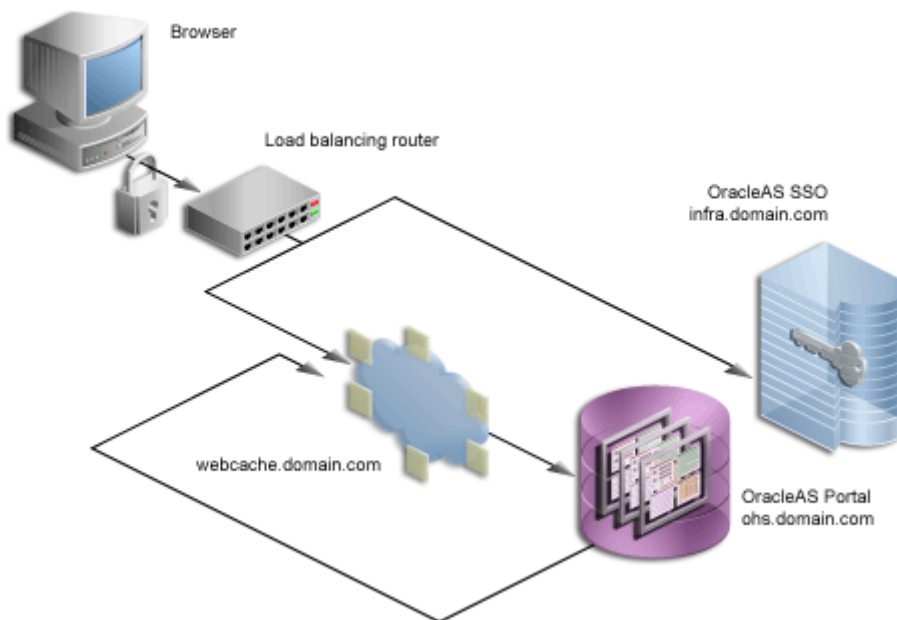
The previous configurations discuss how to configure OracleAS Portal in such a way that communications within Oracle Application Server are secured through SSL. In some cases, you may want to have OracleAS Portal set up such that the site is externally accessible through SSL URLs but the Oracle Application Server is internally running in non-SSL mode. Note that in this latter scenario, you need to have the SSL to non-SSL translation done either by a load balancer or an SSL accelerator. The section that follows outlines the steps you would use with an accelerator on a load balancer or a reverse proxy server performing the SSL translation.

With this option, the SSL features of the OracleAS Security Framework are not used, but, instead, an external component is used for providing the SSL connection point. These external accelerators may be coupled with load balancers or reverse proxy servers. Oracle Application Server enables you to configure these external devices to provide SSL, thus allowing Oracle Application Server to use HTTP internally for the best performance.

For the purposes of this procedure, assume the following:

■ Your load balancer is running on `lbr.abc.com` and the load balancer port used for accessing the site is 443.

- OracleAS Web Cache is on machine `w1.abc.com` and the listening port is 7777, the administration port is 4000, and invalidations are sent to port 4001.

- The Oracle HTTP Server is running on machine `m1.abc.com` and the port is 7778.

> **Note:** In a typical configuration, `w1.abc.com` and `m1.abc.com` would reside on the same machine, but, for illustration purposes, they are separated here.

> **See Also:** For more information, refer to:
>
> - Section 5.6, "Configuring Reverse Proxy Servers"
> - Section 5.3, "Configuring Multiple Middle-Tiers with a Load Balancing Router"

1. Configure the OracleAS Portal middle-tier to allow the underlying components to construct URLs based on the load balancer's hostname, `lbr.abc.com`, and port (443). To do this, perform the following steps:

   a. Define a virtual host, using the **Create Virtual Host** wizard, as explained in Section 5.4.1.1, "Create the Virtual Host for www.xyz.com", with the following exceptions:

   – On the **Addresses** page (step 9), specify the hostname of the LBR (`lbr.abc.com`) in the **Server Name** field for your virtual host.

   – In step 23, specify `443` for the Port directive in the VirtualHost container. In that same VirtualHost container, add a line:

   ```
   SimulateHttps On
   ```

   To make the `SimulateHttps` directive take effect, you must load `mod_certheaders` in the Oracle HTTP Server by adding one of the following directives before the `SimulateHttps` directive:

   On UNIX:

   ```
   LoadModule certheaders_module libexec/mod_certheaders.so
   ```

   On MS Windows:

   ```
   LoadModule certheaders_module modules/ApacheModuleCertHeaders.dll
   ```

For more information, refer to the "mod_certheaders" section of Chapter 8, Oracle HTTP Server Modules, in the *Oracle HTTP Server Administrator's Guide*.

b. Define a second virtual host, using the **Create Virtual Host** wizard, as explained in Section 5.4.1.1, "Create the Virtual Host for www.xyz.com", with the following exceptions:

– On the **Addresses** page (step 9), specify the hostname (`m1.abc.com`) in the **Server Name** field for your virtual host.

– In step 23, specify `7777` for the Port directive in the VirtualHost container.

– When prompted to restart the Oracle HTTP Server (step 26), click **Yes**.

2. Configure the Parallel Page Engine to attempt loop backs using a different protocol and port than what is used by the site.

a. Make the following changes to the Page servlet section in *MID_TIER_ORACLE_HOME*/j2ee/OC4J_Portal/applications/portal/portal/WEB-INF/web.xml:

```
<servlet>
<servlet-name>page</servlet-name>
      <servlet-class>oracle.webdb.page.ParallelServlet</servlet-class>
         <init-param>
            <param-name>useScheme</param-name>
            <param-value>http</param-value>
         </init-param>
         <init-param>
            <param-name>usePort</param-name>
            <param-value>7777</param-value>
         </init-param>
</servlet>
```

b. Run the following command to sync up the manual configuration changes:

```
MID_TIER_ORACLE_HOME/dcm/bin/dcmctl updateconfig
```

c. Restart your Oracle Application Server instance:

```
MID_TIER_ORACLE_HOME/opmn/bin/opmnctl stopall
MID_TIER_ORACLE_HOME/opmn/bin/opmnctl startall
```

3. Configure the machine `m1.abc.com` such that it resolves the load balancer's hostname to be the IP address of the machine running OracleAS Web Cache.

You can either rely on DNS resolution for this step or make entries in the /etc/hosts file as follows:

```
w1.w1.w1.w1  lbr.abc.com
```

> **Note:** If OracleAS Web Cache is local you can use `127.0.0.1` instead of `w1.w1.w1.w1`.

This change coupled with the previous steps causes the Parallel Page Engine to loop back locally to OracleAS Web Cache.

4. Register new URLs with OracleAS Portal using the LBR's hostname and port instead of the OracleAS Web Cache hostname and port by running the OracleAS Portal Configuration Assistant (OPCA) in MIDTIER -type OHS mode. On UNIX, you invoke OPCA with `ptlasst.csh`. On MS Windows, you use `ptlasst.bat`.

   For example:

   ```
   ptlasst.csh -i typical -mode MIDTIER -type OHS -sdad portal
   -host lbr.abc.com -chost w1.abc.com -port 443 -cport_i 4001 -cport_a 4000
   -wc_i_pwd welcome1 -ssl
   ```

5. To prevent access to Oracle Enterprise Manager from the outside, the Oracle Enterprise Manager link provided by OracleAS Portal needs to be changed back to point to the internal server. To do this, run `ptlconfig` (typically located in the directory `MID_TIER_ORACLE_HOME/portal/conf`) as shown in the following example:

   ```
   ptlconfig -dad portal -em
   ```

6. From the OracleAS Web Cache administration page, click **Site Definitions** under **Origin Servers, Sites, and Load Balancing**.

7. Click **Add Site**.

8. Enter site information as follows:

   - **Host Name:** The published hostname and fully qualified domain of the external SSL accelerator device or reverse proxy server.

   - **Port Number:** The SSL port number, for example, 443 for the default SSL port.

- **HTTPS Only Prefix:** Leave blank.

- **Client-Side Certificate:** Not required.

- **Default Site:** Yes.

- **Create Alias from Site Name with/without www:** No.

Please refer to the *Oracle Application Server Web Cache Administrator's Guide* for detailed instructions on the configuration mentioned earlier.

9. Set up an alias for OracleAS Web Cache. In the configuration where the Parallel Page Engine loops back to OracleAS Web Cache and OracleAS Web Cache is listening on a different port than the load balancer, loop back content gets cached with a URL key of `lbr.abc.com:7777`, whereas OracleAS Portal sends invalidation requests to invalidate URLs with the format `lbr.abc.com:443`. To get around this inconsistency, you need to set up an alias in OracleAS Web Cache to let it know that `lbr.abc.com:7777` and `lbr.abc.com:443` are the same, invalidation requests for one should invalidate requests for the other as well, and the cached content should also be leveraged based on this alias.

   a. Go to the Oracle Application Server Web Cache administration page and log in as the administrator.

   b. Click **Site Definitions**.

   c. Select the radio button in the **Select** column that corresponds to the site for which the alias will be added, in this case `lbr.abc.com`.

   d. Click **Add Alias**.

   e. On the window that comes up, enter `lbr.abc.com` as the Host Name and 7777 as the port where 7777 is the value for `usePort` in the `web.xml` configuration file for the Parallel Page Engine.

   f. Click **Submit**.

10. After adding a new site definition, you must add the site to server mapping for the newly defined site to the origin server. To do this:

    a. In the navigation frame, select **Site-to-Server Mapping** under **Origin Servers, Sites, and Load Balancing**.

    b. In the **Site-to-Server Mapping** page, select the first mapping in the table and click **Insert Above**.

    c. In the **Edit/Add Site-to-Server Mapping** page, select the **Select from Site definitions** option and then select a site definition created in the previous step (`lbr.abc.com`).

    d. In the **Select Application Web Servers** section, select the application server (`m1.abc.com`) specified in the **Origin Servers** page.

    e. Click **Submit**.

    f. Click **Apply Changes** on the top of the page.

    g. In the **Cache Operations** page, click **Restart** to restart Web Cache.

    To verify that the site has been mapped correctly, navigate to the **Site-to-Server Mapping** page, and ensure that `m1.abc.com` is mapped to the site `lbr.abc.com`.

For more information on the procedure in the preceding text, refer to Map Sites to Origin Servers in Chapter 7, Basic Setup and Configuration, of the *Oracle Application Server Web Cache Administrator's Guide*.

11. Configure the load balancer, `lbr.abc.com`, to accept requests on port 443 and forward them to the OracleAS Web Cache port (7777) running on machine `w1.abc.com`, while converting HTTPS requests to HTTP.

> **Note:** For information on how this configuration might affect your Web Providers, refer to Section 5.3.6, "Step 6: Configure Portal Tools and Web Providers (Optional)".

Configuring Seeded Providers (WebClipping and OmniPortlet) and Locally Hosted Web Providers

1. Login to OracleAS Portal as the administrator (for example, `PORTAL`).

2. Click the **Administer** tab.

3. Click the **Portlets** sub-tab.

4. In the **Remote Providers** portlet, type `WEBCLIPPING` in the **Name** field.

5. Click **Edit**.

6. Click the **Connection** tab.

7. In the **URL** field, change the URL from:

```
https://lbr.abc.com:443/portalTools/webClipping/providers/webClipping
```

to:

```
http://lbr.abc.com:7777/portalTools/webClipping/providers/webClipping
```

8. Click **OK** to commit the change.

9. Repeat steps 4 through 8 but with the following exceptions:

   ■ In step 4, type OMNIPORTLET instead of WEBCLIPPING.

   ■ In Step 7, change the URL from:

   ```
   https://lbr.abc.com:443/portalTools/omniPortlet/providers/omniPortlet
   ```

   to:

   ```
   http://lbr.abc.com:7777/portalTools/omniPortlet/providers/omniPortlet
   ```

When you register locally hosted Web Providers (such as the JPDK Sample provider), you need to register them using HTTP as the protocol, lbr.abc.com as the hostname, and 7777 as the port number. This restriction only applies to locally hosted Web Providers (that is, Web Providers running on the same middle-tier as OracleAS Portal).

For example, to register the JPDK Sample provider, the URL is:

```
http://lbr.abc.com:7777/jpdk/providers/sample
```

> **Note:** If your infrastructure is located on a separate machine than your OracleAS Portal middle-tier, you need to add the following to your /etc/host file:
>
> w1.w1.w1.w1 lbr.abc.com
>
> where w1.w1.w1.w1 is the IP Address of your OracleAS Web Cache.

Re-registering the Oracle HTTP Server Partner Application

You now need to register the secured request with OracleAS Single Sign-On by configuring it as a partner application. The script ossoreg performs this registration. ossoreg is located on the middle-tier in *MID_TIER_ORACLE_ HOME*/sso/lib.

1. Ensure that you have your environment configured properly to run ossoreg:

```
ORACLE_HOME=MID_TIER_ORACLE_HOME
LD_LIBRARY_PATH=ORACLE_HOME/lib
```

2. Run ossoreg. The following example illustrates the usage of ossoreg.

```
MID_TIER_ORACLE_HOME/jdk/bin/java -jar
  MID_TIER_ORACLE_HOME/sso/lib/ossoreg.jar -site_name lbr.abc.com
  -mod_osso_url https://lbr.abc.com -config_mod_osso TRUE
  -oracle_home_path MID_TIER_ORACLE_HOME -u install_user -config_file
  MID_TIER_ORACLE_HOME/Apache/Apache/conf/osso/osso.conf
  -admin_info cn=orcladmin
```

Refer to the section titled "Registering mod_osso" in Chapter 4 of the *Oracle Application Server Single Sign-On Administrator's Guide* for more information.

6.3.2.2 Securing the Connection to Oracle Internet Directory (Optional)

In Section 6.3.2.1, "Configuring SSL for OracleAS Portal", we were mainly concerned with the HTTP-based network hops. However, you can also secure the network connection to the Oracle Internet Directory itself, which is LDAP-based communication. In this case the Oracle Internet Directory should be configured to use LDAP over SSL (LDAPS). You can find further information about configuring the Oracle Internet Directory for LDAPS in the *Oracle Internet Directory Administrator's Guide*.

Once Oracle Internet Directory is configured to use SSL, you must update the OracleAS Portal repository to use the new port on the LDAP server. To perform this step, you run the SQL script, secupoid.sql, located in *MID_TIER_ORACLE_HOME*/portal/admin/plsql/wwc. This script allows for the setting of the following Oracle Internet Directory related parameters:

- Directory Host Name
- Directory Port
- Application Directory Password
- SSL Settings

When you run the script, it displays the current settings and gives you the ability to change them accordingly. In this case, you want to set the following:

```
use_ssl_to_connect_to_ldap=Y
```

The script will then give you the option of automatically refreshing OracleAS Portal's Oracle Internet Directory cache.

> **Note:** In Release 10*g* (9.0.4), you can optionally install OracleAS Portal using LDAPS rather than having to implement it after installation.

6.3.2.3 Changing Settings on the Global Settings Page

Once you have installed OracleAS Portal and performed the appropriate tasks from Section 6.3.2.4, "Post-Installation Security Checklist", you can change all of the following settings that pertain to security from the **Global Settings** page of OracleAS Portal:

- Cache for Oracle Internet Directory Parameters

- Oracle Directory Integration Platform Synchronization

- Group Search Base Distinguished Name (DN)

- Group Creation Base Distinguished Name (DN)

6.3.2.3.1 Cache for Oracle Internet Directory Parameters As pointed out in Section 6.1.6, "Leveraging Oracle Identity Management Infrastructure", OracleAS Portal maintains a cache of information from the directory. From the **Global Settings** page, you can refresh this cache with the updated information from the directory. Refresh Cache for the Oracle Internet Directory parameters immediately updates the cache with the latest parameters values from the directory. The cached information is relatively static information, hence you do not need to refresh the cache frequently.

6.3.2.3.2 Oracle Directory Integration Platform Synchronization Because OracleAS Portal caches group membership information, it requires a mechanism for updating the cache when the information is changed in the directory. The directory integration server notifies OracleAS Portal whenever a change is made in the directory that must be reflected in OracleAS Portal. In **Global Settings**, you can set:

- **Enable directory synchronization** defines whether the directory integration server notifies OracleAS Portal when a relevant change is made in the directory. If this setting is not checked, then OracleAS Portal will not be notified of any directory integration server subscribed events.

- **Send event notifications every n seconds** defines the interval of time between event notifications sent by the directory integration server to notify OracleAS Portal of relevant changes. This setting is available only when Enable directory synchronization is checked.

When the Oracle Directory Integration and Provisioning server is running and configured to work with OracleAS Portal, group membership changes in Oracle Internet Directory will result in soft cache invalidations in OracleAS Portal.

> **See Also:**
>
> - Section 1.3.3, "Understanding Cache Invalidation in OracleAS Portal"
>
> - Section 5.7, "Configuring OracleAS Web Cache Caching in OracleAS Portal".
>
> - Section 6.1.6.3, "Relationship Between OracleAS Portal and Oracle Internet Directory"

Some examples of group membership cache invalidations are:

- If you add a user to a group, the Oracle Directory Integration and Provisioning server notifies OracleAS Portal of the change. OracleAS Portal will then issue a single soft invalidation message that will be processed by the soft invalidation job. This is because of the possibility that the user may have new privileges that can affect what data can be viewed.

- If you add group _A to group _B, the Oracle Directory Integration and Provisioning server notifies OracleAS Portal of the change. OracleAS Portal will then issue a soft invalidation message for each user in group _A. This is because of the possibility that the users in group _A may have new privileges that affect what data they can view.

6.3.2.3.3 Group Creation Base Distinguished Name (DN) OracleAS Portal maintains its user group information in the directory. When groups are created through the Group portlet, they are created under a node of the LDAP Directory Information Tree (DIT). A node is identified by its distinguished name (DN). Therefore, in OracleAS Portal, you need to specify in which node you wish to create groups:

Group Creation Base DN is the DN of the node in which you want OracleAS Portal to maintain its user groups. For example:

```
cn=ptl_schema_name.031009.0430,cn=Groups,dc=MyCompany,dc=com
```

This setting is particularly useful if you adapt OracleAS Portal to interact with an existing DIT.

6.3.2.3.4 Group Search Base Distinguished Name (DN) Just as you need to define the node in which you want to create groups, you must also define the node in which

you want OracleAS Portal to search for existing groups. For example, you need to specify where OracleAS Portal searches when it displays the group's list of values in the **Group** portlet.

Local Group Search Base DN is the DN of the node in which you want OracleAS Portal to maintain its user groups. For example:

```
cn=ptl_schema_name.031009.0430,cn=Groups,dc=MyCompany,dc=com
```

This setting is particularly useful if you adapt OracleAS Portal to interact with an existing DIT.

6.3.2.4 Post-Installation Security Checklist

After OracleAS Portal is installed, you should consider performing the following steps to complete the security configuration:

- Configure mod_plsql Settings

- Safeguard Passwords for Lightweight OracleAS Portal Users

- Remove Unnecessary Objects

- Review Default Seeded Privileges

- Revoke Public Access to Provider Components

- Control Access to Administration Pages

- Protect PL/SQL Packages

- Consider SSL and the Login Portlet

- Consider LDAP over SSL for Oracle Internet Directory Connections

- Change the Application Entity Password

6.3.2.4.1 Configure mod_plsql Settings The mod_plsql settings are configured in Oracle Enterprise Manager, which can be accessed from OracleAS Portal as follows:

1. From the OracleAS Portal Design-Time Pages page, click the **Administer** tab.

2. Click the **Portal** tab if it is not already selected.

3. In the **Services** portlet, click **Portal Services Monitoring**.

4. Click **mod_plsql Services** in the list of **Components**.

5. In the DADs section, edit the PORTAL DAD and change the password of the corresponding database user.

6. Delete all of the DADs that you do not need. For example, SAMPLE_DAD is unnecessary.

7. Add a new DAD for the portal you are building and set the default name or location.

> **See Also:** *Oracle Application Server 10g mod_plsql User's Guide*

6.3.2.4.2 Safeguard Passwords for Lightweight OracleAS Portal Users Unscrupulous users might try to learn the passwords of your default users, which could result in an account lock. This lock can be released from the server, but it is far better that you not depend on the default user accounts for administrative purposes. To safeguard the passwords for these accounts do the following:

1. Immediately change the default passwords for all of the following default users:

   ■ PORTAL

   ■ PORTAL_ADMIN

   ■ PUBLIC

2. Create new lightweight administrator accounts with the same access rights as the default users, and set the account termination date in OracleAS Single Sign-On for the default users. Alternatively, you can also uncheck the Allow User To Log In setting in the Edit User page for the default users.

3. Once you have disabled login or changed the passwords for the default users, try logging in to the portal as the default users with the default passwords to ensure that your changes have been successful.

6.3.2.4.3 Remove Unnecessary Objects In order to prevent users from entering your portal through obsolete or unnecessary pages, you should remove any unused objects from your OracleAS Portal and database environment. For example:

■ Delete page groups that are no longer in use.

■ Delete OracleAS Portal providers that are no longer in use.

6.3.2.4.4 Review Default Seeded Privileges When OracleAS Portal is installed, the seeded groups listed in Table 6–2 are provisioned with a set of privileges that are typically required by users in those roles. You should review these initial set of privileges to ensure that they are consistent with your security policy.

Users or groups can obtain privileges from one of the following sources:

■ OracleAS Portal access control entries

- Oracle Internet Directory privilege groups

To edit the privileges granted through OracleAS Portal access control entries, you edit the user or group profile from the **Administer** tab with the User Profile Portlet or Group Profile Portlet. Click the User or Group Profile dialog's **Privilege** tab. You can revoke or assign privileges from this list.

To edit the privileges granted through Oracle Internet Directory privilege groups, use the User Portlet or Group Portlet to edit the User or Group in Oracle Internet Directory. Select or deselect the checkmarks by the Privilege Assignment list to grant or revoke the appropriate privileges in Oracle Internet Directory.

Privileges granted to the AUTHENTICATED_USERS group are given to any user that logs on to OracleAS Portal through OracleAS Single Sign-On upon successful authentication. This is the group that you will want to establish with the default privileges for all your logged in users.

For example, if you don't want authenticated users to be able to create groups, then edit the AUTHENTICATED_USERS group through the Group Portlet and remove the check mark beside **Allow group creation** under Privilege Assignment.

6.3.2.4.5 Revoke Public Access to Provider Components In some cases, OracleAS Portal provider components may give users the option to view or modify records in application tables. To tighten security, you should revoke public access from such components if it is unnecessary. You can also use a menu component with specific access rights on the menu options to more tightly control application access.

6.3.2.4.6 Control Access to Administration Pages In order to prevent users who should not have access to administration interfaces from entering administration pages, you should ensure that you control access rights for the following page groups and the pages they contain:

- Corporate Pages is the page group that contains the OracleAS Portal Home Page.

- Portal Design-Time Pages is the page group that contains the Builder and Navigator pages.

- Portlet Repository

To control access to the page groups mentioned earlier, perform the following steps:

1. In the Navigator, click **Page Groups**.

2. Click **Edit Properties** next to the page group for which you want to change the access settings.

3. Click the **Access** tab.

4. Grant MANAGE ALL to specific users or to certain groups. For example, DBA, PORTAL_ADMINISTRATORS, PORTAL_DEVELOPERS, and your own groups.

5. When you are done, click **OK**.

To control access to individual administration pages in these page groups, perform the following steps:

1. In the Navigator, click **Page Groups**.

2. Click **Contents** next to the page group that contains the pages on which you want to change the access settings.

3. Click **Pages**.

4. Click **Properties** next to the page for which you to change the access settings.

5. Click the **Access** tab.

6. Grant MANAGE ALL to specific users or to certain groups. For example, DBA, PORTAL_ADMINISTRATORS, PORTAL_DEVELOPERS, and your own groups.

7. When you are done, click **OK**.

> **Note:** The **Builder** page is the root page of the Portal Design-Time Pages page group. In order to alter its access settings, you must click **Edit Root Page** next to the Portal Design-Time page group.

6.3.2.4.7 Protect PL/SQL Packages You must protect the execution of PL/SQL procedures granted to PUBLIC in the database. These procedures pose a security hole when they are executed through a Web browser. For example, some procedures in the dbms_% packages allow access to sensitive information. You can specify which packages to protect with the PlsqlExclusionList directive in the mod_plsql configuration file called dads.conf.

> **Note:** In earlier releases, you also needed to protect monitoring packages (wwmon_%). For Release 9.0.4 and later releases, these packages have been removed and therefore no longer need protection.

To ensure the best security, specify the following system default settings with the PlsqlExclusionList directive for each DAD:

```
PlsqlExclusionList sys.*
PlsqlExclusionList dbms_*
PlsqlExclusionList utl_*
PlsqlExclusionList owa_util.*
```

6.3.2.4.8 Consider SSL and the Login Portlet To secure passwords going across the Internet you can use Secure Sockets Layer (SSL) communications by configuring OracleAS Portal to run in HTTPS. However, to enable or disable SSL, you must have portal administrator privileges.

Login Portlet Versus Login Page

Portal has the option to place a Login portlet on a page (typically the home page). This portlet shows user name and password fields and a login button when the user is not logged in. If the user is logged in, it shows a logout link. This portlet provides an easy way to log in without having to navigate to a dedicated login page. It also displays in the OracleAS Portal page layout style.

However, if you use this portlet, you must ensure that the pages on which it appears are SSL-encrypted. Bear in mind, that SSL encryption for your complete site could adversely affect performance because it requires more resources. Furthermore, the Login portlet presents a security risk because you cannot prevent showing the login screen since it is shown when the user is not logged in. Hence, in situations where you want SSL encryption on passwords, you should not use the Login portlet when you want SSL encryption. To enforce this restriction, you must remove access rights for the Login portlet in the Portlet Repository.

6.3.2.4.9 Consider LDAP over SSL for Oracle Internet Directory Connections By default, OracleAS Portal connects to the directory using LDAP without SSL. If the directory server is configured for an SSL port, though, OracleAS Portal can be configured to use LDAP over SSL, also known as LDAPS.

> **See Also:** *Oracle Internet Directory Administrator's Guide*

To configure OracleAS Portal to use SSL to connect to the directory, you must run the `secupoid.sql` script, located in *ORACLE_HOME*`/portal/admin/plsql/wwc`. This script enables you to change the following OracleAS Portal configuration parameters related to the directory:

- Directory hostname
- Directory port

- Application directory password
- SSL setting

When you install OracleAS Portal, it is automatically configured with a directory server. However, you may want to change some settings, such as whether to use SSL, after installation. To change to an SSL connection for the directory, simply run the `secupoid.sql` script in the PORTAL schema to specify the LDAPS port instead of the LDAP port, and indicate that you want to use SSL.

Running the secupoid.sql script

The section that follows illustrates a sample execution of `secupoid.sql` from SQL*Plus.

In the example, the directory was initially configured to run LDAP on port 389. Later, an LDAPS port was activated on 636. Since the server name does not change, we retain the old value, update the port, and indicate that we want to use SSL by setting the Use SSL? value to Y. When you run the script, it displays the current configuration and lets you replace any of the configurable settings. The script also enables you to update OracleAS Portal's directory cache after running it. Since activating SSL does not change any of the directory information cached by OracleAS Portal, it is not usually necessary to refresh the cache in this case.

```
SQL> @secupoid
Current Configuration
--------------------
OID Host: oid.domain.com
OID Port: 389
Application DN:
orclApplicationCommonName=PORTAL,cn=Portal,cn=Products,cn=OracleContext
Application Password: 3E8C2D1B87CB61011757239C5AA9B390
Use SSL? N

PL/SQL procedure successfully completed.

Updating OID Configuration Entries
Press [Enter] to retain the current value for each parameter
For SSL Connection to LDAP, specify "Y"es or "N"o
-----------------------------------------------
Enter value for oid_host:
Enter value for oid_port: 636
Enter value for app_password:
Enter value for use_ssl_to_connect_to_ldap: Y
Enter value for refresh_with_new_settings: N
```

```
PL/SQL procedure successfully completed.

No errors.
```

After executing the script, OracleAS Portal is configured for LDAPS access of the directory.

6.3.2.4.10   Change the Application Entity Password OracleAS Portal never passes a user's password to the directory. Only OracleAS Single Sign-On performs that task. However, OracleAS Portal authenticates itself to the directory through its application entity and password.

If you want to change the application entity's password, you need to first change its entry in the directory, using command line utilities or the Oracle Directory Manager. To locate the application entry in the directory, you need its DN, which is reported by the secupoid.sql script. By default, OracleAS Portal's application entry is:

```
orclApplicationCommonName=PORTAL,cn=Portal,cn=Products,cn=OracleContext
```

To change the password, you set the userPassword attribute for the application entry to the new password.

After you have changed the password in the directory, you run secupoid.sql script in the PORTAL schema and specify the new password there, too. Running the script enables OracleAS Portal to encrypt the password and store it for retrieval when it needs to connect to the directory.

For more information about the secupoid.sql script, refer to Section C.3, "Using the secupoid.sql Script".

> **See Also:**   Section 6.1.6.2.1, "Directory Entries in Oracle Internet Directory for OracleAS Portal", for more information about the application entity.

6.3.3 Configuring OracleAS Portal Options for Database Security

Fine-grained access controls and secure application contexts add a new dimension to your ability to secure your data in the database.

Fine-grained access control is sometimes referred to as virtual private database or row level security. Fine-grained access control in the Oracle9*i* Database Server is the ability to dynamically attach, at runtime, a predicate (WHERE clause) to any and all queries issued against a database table or view. This feature gives you the ability to procedurally modify the query at runtime. You may evaluate who is running the

query, where they are running the query from, when they are running the query and develop a predicate given those circumstances. With the use of application contexts, you may securely add additional information to the environment (such as an application role the user may have) and access it in your procedure or predicate as well.

As an example of fine-grained access control, you might have a security policy that determines what rows different groups of people may see. Your security policy will develop a predicate based on who is logged in and what group they are in.

More On Portal Center

You'll find additional information about fine-grained access and application contexts in the technical note "*How to Implement Fine Grained Access Controls and Secure Application Contexts*," on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

7

Monitoring and Administering OracleAS Portal

This chapter provides information about the monitoring and administration tools that are available, and how to use them to successfully monitor and administer OracleAS Portal.

You can monitor and administer OracleAS Portal through the Oracle Enterprise Manager Application Server Control. Additionally, you can view OracleAS Portal Analytics to monitor OracleAS Portal performance and analyze OracleAS Portal access characteristics.

> **See Also:** For additional OracleAS Portal monitoring and administration information, see the *Portal Administrator Zone* on Portal Center, at http://portalcenter.oracle.com.

This chapter contains the following sections:

- Using the Application Server Control
- Using Application Server Control to Monitor and Administer OracleAS Portal
- Viewing OracleAS Portal Analytics
- Viewing Oracle Application Server Port Information

7.1 Using the Application Server Control

The Application Server Control is included when you install Oracle Application Server. From OracleAS Portal's perspective, consider this to be the administration console for the Oracle Application Server. The Application Server Control enables you to perform the following administration and configuration operations:

- Enable and disable components

- Administer clusters

- Start and stop services

- View logs and ports

- Perform real-time monitoring

- Modify the Infrastructure services used by an Oracle Application Server middle-tier.

This section contains information about:

- Accessing the Application Server Control

- Using Application Server Control to Configure OracleAS Portal

7.1.1 Accessing the Application Server Control

You can access the Application Server Control by entering the following URL in your Web browser:

```
http://<hostname>:<port>
```

For example, `http://mgmthost.company.com:1810`. The port is typically on 1810, however the possible port range for the Application Server Control varies upwards in increments of 1.

If there is more than one standalone application server instance, your start page for the Application Server Control is the Oracle Application Server **Farm** home page. Clicking an instance, takes you to the Oracle Application Server instance home page. This page contains a table of System Components. From this table you can display the home page for each component of the application server for monitoring and administrative purposes.

If OracleAS Portal is configured, **Portal:<portal schema name>** appears in this table. The default portal schema name is **portal**.

7.1.2 Using Application Server Control to Configure OracleAS Portal

If **Portal:portal** is not listed in the **System Components** table, it means it is not yet configured. The **Configure Component** button appears above the System Components table if you have installed, but not configured, some Oracle Application Server components.

> **Note:** Only components that have the checkbox selected can be started or stopped.

To configure OracleAS Portal perform these steps:

1. On the Oracle Application Server home page, click the **Configure Component** button.

2. Select **Portal** from the **Select Component** dropdown list.

3. Enter the administration password for the Oracle Application Server instance in the **Password** field.

4. Click **Finish**.

> **Note:** By default, an OracleAS Portal middle-tier is made up of one portal instance. Both the DAD name and the OracleAS Metadata Repository schema name for this instance are **portal**. You cannot use the **Configure Component** button to configure additional OracleAS Portal instances for a given middle-tier if **Portal:portal** is already listed in the **System Components** table.

7.2 Using Application Server Control to Monitor and Administer OracleAS Portal

To monitor and administer OracleAS Portal, click **Portal:<portal schema name>** in the list of system components on the Oracle Application Server instance home page. The default portal schema name is **portal**. Note that **OC4J:Portal** is the container for portal servlets, and not an actual portal servlet to monitor.

The main page for monitoring OracleAS Portal that is displayed, is shown in Figure 7–1.

You can also access this page directly from OracleAS Portal. Click the **Administer** tab on the Portal Builder page and then click **Portal Service Monitoring** (located on the **Portal** sub-tab).

> **Note:** If any Oracle Enterprise Manager Application Server Control details change, for example, the port or protocol, you must update the link provided by OracleAS Portal otherwise it will not work. For instructions, see Section 7.2.8, "Updating OracleAS Portal Link to Oracle Enterprise Manager".

Figure 7–1 Application Server Control - Main OracleAS Portal Monitoring Page

The main OracleAS Portal monitoring page, shown in Figure 7–1, contains various sections and links:

- General

- OracleAS Metadata Repository

- Portal Web Cache Settings Page

- Component Status

- Severity Status

- Related Link
- Logs Link

7.2.1 General

Use this section to establish the general status of an OracleAS Portal instance, that is, whether it is currently 'up' or 'down'.

You can also see the average number of page requests for each hour, as well as the current home page download speed. Furthermore, you can check if the monitoring services are up and running.

7.2.2 OracleAS Metadata Repository

Use this section to view metrics relating to the OracleAS Metadata Repository. This is the repository containing the OracleAS Portal schema.

You can see if the database that contains the OracleAS Metadata Repository is up and running, the version number, name of the database, and the version number of the OracleAS Metadata Repository.

7.2.3 Portal Web Cache Settings Page

From the Application Server Control, you can specify the OracleAS Web Cache settings that OracleAS Portal should use. Setting the OracleAS Web Cache properties on this page will automatically result in the Portal Dependency Settings file located on this middle-tier being updated, and the ptlconfig script being executed. See Appendix A, "Using the Portal Dependency Settings File" for more details.

When you click the **Portal Web Cache Settings** link, under **Administration**, the **Portal Web Cache Settings** page shown in Figure 7–2 is displayed.

Note: Changing OracleAS Web Cache settings (for example, Listening Port) can change the OracleAS Portal URL. If you do this, mobile settings need to be updated. For more information, see Appendix C.8, "Using the cfgiasw Script to Configure Mobile Settings".

Figure 7–2 Application Server Control - Oracle Application Server Web Cache Settings
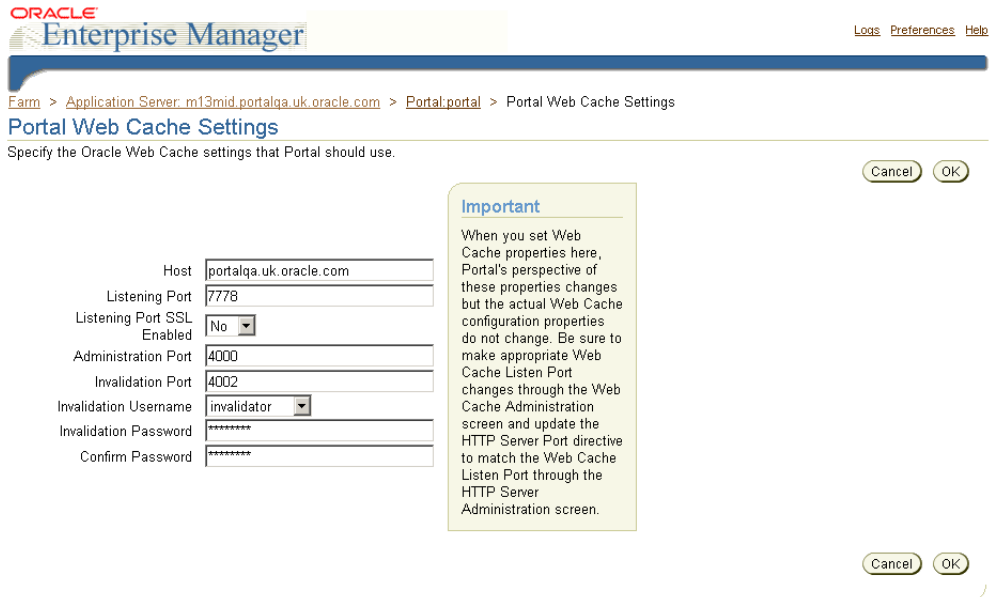
In the **Portal Web Cache Settings** page, you can modify the settings detailed in Table 7–1:

Table 7–1 Portal Web Cache Settings

Setting	Description
Host	The hostname that OracleAS Web Cache should use. For example, `abc.company.com`.
Listening Port	The port on which OracleAS Web Cache listens. For example, `7778`.
Listening Port SSL Enabled	Indicates whether OracleAS Web Cache is SSL enabled. Valid values are 'Yes' and 'No'.
Administration Port	The OracleAS Web Cache administration port. For example, `4000`.
Invalidation Port	The OracleAS Web Cache invalidation port, to which invalidation messages are sent. For example, `4001`.
Invalidation Username	The username used for sending the invalidation messages. Either `invalidator` or `administrator`.

Table 7–1 (Cont.) Portal Web Cache Settings

Setting	Description
Invalidation Password	The invalidation password. The default is `invalidator`.
Confirm Password	Repeat the password specified earlier.

> **Note:** When you set OracleAS Web Cache properties, Portal's perspective of these properties changes but the actual OracleAS Web Cache configuration properties do not change. You must make corresponding changes to the appropriate OracleAS Web Cache configuration pages. Refer to the *Oracle Application Server Web Cache Administrator's Guide* for more information about OracleAS Web Cache.
>
> Typically, the hostname and port number, by which OracleAS Portal is addressed, uses the OracleAS Web Cache hostname and port number. This is because, in a simple configuration, browser requests go directly to OracleAS Web Cache. However, in a configuration that has a load balancing router (LBR), or reverse proxy server front-ending OracleAS Web Cache, the hostname and port number defined on this page may need to reflect that of the LBR, or reverse proxy server.
>
> In this configuration, you want OracleAS Web Cache invalidation messages to be sent directly to the OracleAS Web Cache host, as opposed to the LBR, or reverse proxy server. In the scenario where your published hostname is different from the hostname used for OracleAS Web Cache invalidation, you cannot use the **Portal Web Cache Settings** page in the Oracle Enterprise Manager Application Server Control, to establish these settings. Instead, you must use the OracleAS Portal Configuration Assistant (OPCA) in the MIDTIER mode with -type OHS, using the `host` parameter to specify the hostname of the LBR, or reverse proxy server, and the `-chost` parameter to define the OracleAS Web Cache hostname.

In the following example, the **Portal Web Cache Settings** page is used to configure OracleAS Portal to use OracleAS Web Cache on a different host:

Example 7–1   Example: Configuring OracleAS Portal to use OracleAS Web Cache on a Different Host

To configure OracleAS Portal to use OracleAS Web Cache on a different host from the one the OracleAS Portal middle-tier is installed on, you must follow these steps:

1. Access the Application Server Control on the middle-tier where OracleAS Portal is installed.

2. Select the portal instance you want to configure. Typically this is **Portal:portal**.

3. Select **Portal Web Cache Settings**.

4. Update the **Hostname** property with the new host name, along with any other property changes you want to add.

5. Click **OK**.

7.2.4 Component Status

Lists the Oracle Application Server components used by OracleAS Portal and indicates their status. You can also drill down to find more information about individual Oracle Application Server components. These components are:

- HTTP Server

- mod_plsql Services

- Web Cache

- Parallel Page Engine Services

- Providers

- Syndication Services

- Ultra Search

For performance reasons, less critical metric data, that is, non-response metric, are collected by the Application Server Control metric cache and may be slightly out of date. To display the most up to date metric data, click the **Refresh** link in the top left corner of a page.

7.2.4.1 HTTP Server

Clicking the **HTTP Server** link takes you to the Oracle HTTP Server home page. This is the starting point for managing a single instance of Oracle HTTP Server. For example, you can restart the Oracle HTTP Server from here.

7.2.4.2 mod_plsql Services

Clicking the **mod_plsql Services** link takes you to the **mod_plsql Services** home page shown in Figure 7–3. From here you can configure, as well as monitor, mod_plsql related settings and metrics.

Figure 7–3   Application Server Control - mod_plsql Services Monitoring Page

The following sub-sections in this page help you monitor and configure mod_plsql services:

- **General**

  Indicates whether the mod_plsql module loaded into memory successfully. It also shows the number of requests for every hour, received by mod_plsql since the server was started.

- **HTTP Response Codes**

  Displays the HTTP response and error codes. For each response code, the number of requests and the percentage this number represents overall is shown.

- **Cache**

Displays the Cache Settings, as shown in Figure 7–4. Here you can see the number of requests for cached content, including the percentage of cache hits and misses. In the case of a cache miss you can see if the miss was due to stale content, or new content being added. If you see a high percentage of misses, you can tune the cache.

Update the cache configuration settings on this page

> **See Also:** *Oracle Application Server 10g mod_plsql User's Guide*

Figure 7–4 Application Server Control - Cache Settings

- **DADs**

  Displays the status of existing DADs. A Database Access Descriptor (DAD) is a set of values that specify how an application connects to an Oracle database to fulfill an HTTP request. You can also create, edit, and delete DADs here. See Section 4.5.3, "Configuring a Portal DAD".

- **Errors and Response Codes**

  Provides links to the HTTP error and SQL error pages. You can get the following details:

  - Breakdown of HTTP response codes that mod_plsql has returned.

  - Last ten SQL errors mod_plsql has encountered.

  - SQL errors that mod_plsql has encountered, grouped by type of error.

7.2.4.3 Web Cache

Clicking the **Web Cache** link takes you to the OracleAS Web Cache home page. Use this page to gather overall performance statistics for OracleAS Web Cache, including status, resource utilization, and cache efficiency. The following sections in this page enable you to monitor and configure Oracle Application Server Web Cache settings:

- **General** - Displays the current status of OracleAS Web Cache.

- **Activity** - Displays cache resource and performance information.

- **Performance** - Enables you to monitor overall cache performance, monitor the status and performance of the origin servers, and view the most popular requests since the cache was started.

- **Administration** - Provides access to the OracleAS Web Cache Manager URL (`http://web_cache_hostname:admin_port/webcacheadmin`).

7.2.4.4 Parallel Page Engine Services

Clicking the **Parallel Page Engine Services** link takes you to the Parallel Page Engine (PPE) monitoring page shown in Figure 7–5. From here you can get detailed PPE statistics, such as:

- Portlet response codes

- Page level caching

- Status of the OC4J Portal container

- PPE request queue statistics

> **See Also:** For more information about the PPE, refer to Appendix D, "Configuring the Parallel Page Engine".

Figure 7–5   Application Server Control - Parallel Page Engine Services Monitoring Page

7.2.4.5 Providers

Clicking the **Providers** link takes you to the Providers monitoring page shown in Figure 7–6. From here you can get an overview of the performance, status, and HTTP response codes (portlets only) of providers and portlets that are requested by the Parallel Page Engine (PPE) in the Application Server Control.

> **Note:** For performance reasons, the Provider's *Up/Down* value in the Component Status table (see Figure 7–1) is always set to 'Unknown'.

Figure 7–6   Application Server Control - Providers Monitoring Page

On the Providers page, Web Providers are distinguished from Database Providers. You can click a provider to get details about individual portlets, owned by a provider.

Metrics you can monitor include:

- Avg Time (seconds) - The average response time to request a portlet.

- Max Time (seconds) - The maximum response time to request a portlet.

- Requests - The number of requests serviced by this provider.

- Cache Hits - The number of times the cache has been accessed.

- Online - Indicates if a provider is currently online.

- Performance - Indicates whether the providers are performing as expected.

- Status - Indicates whether a specific provider is up or down.

7.2.4.6 Syndication Services

Clicking the **Syndication Services** link takes you to the Oracle Application Server Syndication Services monitoring page. From this page you can manage Oracle Application Server Syndication Services, which automates the establishment of syndication relationships (offers and subscriptions), content transfers based on delivery rules (contracts), and results analysis (access logs).

The following administration options are available from this page:

- **Offer Management** - View, select, create, and manage offers and associated offer contracts from content providers.

- **Content Providers** - Register and manage content providers and view content provider connectors that provide the interface to external content repositories belonging to content providers.

- **Subscriptions** - View and manage accepted offer agreements, subscription state, and purge expired and terminated subscriptions.

- **System Properties** - Enable access logging, edit domain information, edit scheduler properties, and edit HTTP/S and SMTP transport properties.

- **Access Logs** - View and manage user access records for all access to Oracle Application Server Syndication Services.

For more information, see Chapter 11, "Syndicating Content Into OracleAS Portal".

7.2.4.7 Ultra Search

Clicking the **Ultra Search** link takes you to Oracle Ultra Search administration pages. From here you can configure Oracle Ultra Search. For more information, see Chapter 8, "Configuring the Search Features in OracleAS Portal".

7.2.5 Severity Status

Lists the Oracle Application Server components used by OracleAS Portal that indicate severity status. Table 7–2 describes the severity status levels that are reported.

> **Note:** Severity level threshold are set in `targets.xml`.

Table 7–2   Severity Level Status Descriptions

Item	Description
OK	The component is running normally.
Warning	There is some problem with the component.
Critical	The component is having critical problems.
Unknown	There is not enough information to establish status as the component is down.

7.2.6 Related Link

Contains the link **Portal End User Default Homepage**, which takes you to the home page of the OracleAS Portal being monitored.

7.2.7 Logs Link

To perform detailed diagnostics, using log files, click the **Log** link. In the Application Server Control, this link is located at the top and bottom of every Oracle Application Server component home page.

> **See Also:**   *Oracle Application Server 10g Administrator's Guide*

7.2.8 Updating OracleAS Portal Link to Oracle Enterprise Manager

OracleAS Portal provides a link to the Oracle Enterprise Manager Application Server Control that is monitoring and managing the portal. To access the **Portal Service Monitoring** link, click the **Administer** tab in OracleAS Portal and locate the **Services** portlet.

If any Oracle Enterprise Manager Application Server Control details change, for example, the port or protocol, you must update the link provided by OracleAS Portal otherwise it will not work.

To do this, follow these steps:

1. Edit the file `iasconfig.xml` on the Portal middle-tier.

This is usually located in *ORACLE_HOME*/portal/conf. For details, see Appendix A, "Using the Portal Dependency Settings File".

2. Update the **EMComponent** element for your Portal instance, as required.

3. Run the following script to update the Oracle Application Server Metadata Repository with the new settings:

```
ORACLE_HOME/portal/conf/ptlconfig -dad <dad> -em
```

4. Clear the OracleAS Web Cache cache to view the updated link in OracleAS Portal, that is, the **Services** portlet.

   In the **Services** portlet, click **Global Settings**, the **Cache** tab and then select **Clear The Entire Web Cache**.

7.2.9 Enabling Monitoring For Oracle9*i*AS Portal Repository (9.0.2)

An Oracle Application Server 10*g* (9.0.4) middle-tier can use an Oracle9*i*AS Portal repository version 9.0.2. You must complete some additional steps after installation to enable Oracle Enterprise Manager Application Server Control to monitor this version of Oracle9*i*AS Portal.

If you do not do this, Oracle9*i*AS Portal version information and Oracle9*i*AS Portal Metadata Repository information (database version and start time) is not accessible in Oracle Enterprise Manager Application Server Control. Also, missing package errors (WWC_MONITORING) are displayed in the Oracle HTTP Server logs.

1. In SQL*Plus, connect as SYS.

2. Run the script `cfgvr902.sql`, located in the middle-tier Oracle home, under `<upgrade_directory>/wwc/`.

   Use the `<portal schema name>` as an argument.

   Note that this script only exists in the Oracle Application Server (9.0.4) middle-tier with which the Oracle9*i*AS Portal repository (9.0.2) is connected. The script will not exist in the Oracle home running the infrastructure.

   For example, if the Oracle9*i*AS Portal repository (9.0.2) schema name is `portal` and the middle-tier is running from `/homes/portalMid904/`, enter:

   ```
   /homes/portalMid904/portal/admin/plsql/wwc/cfgvr902.sql portal
   ```

3. If there are no errors, run the following grant when connected to the `portal` schema:

   a. In SQL*Plus, connect as PORTAL.

   b. Make the new WWC_MONITORING package accessible to the monitoring component, enter:

   ```
   grant execute on WWC_MONITORING to PUBLIC;
   ```

7.3 Viewing OracleAS Portal Analytics

OracleAS Portal analytics includes:

- Performance reports
- Activity reports

7.3.1 OracleAS Portal Activity Reports from mod_plsql Logs

A set of OracleAS Portal Activity reports are available that execute against data collected by the performance logging service of mod_plsql. For a full description of how to implement this logging service, refer to Section 9.5, "Generating Performance Reports". These reports return information such as:

- Peak login time each day
- Number of logins the portal receives each day
- Portlet execution time
- Slowest portlet
- Total hits received by the portal each day
- Most/least popular portlets
- Frequency of pages or portlets viewed by users
- Number of unique users login each day
- Portlets that were accessed
- Number of hits received by every page each day
- Number of hits received by every portlet each day
- Breakdown of information by IP address or host name

More On
Portal Center

You can find additional information in the technical note *Object Access Reporting from the Performance Logs in OracleAS Portal*, on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

7.3.2 OracleAS Portal Activity Reports from the Portal Activity Log Tables

You can log objects and actions in OracleAS Portal and generate reports for analyzing the data. For example, you can add an entry into the Activity Log tables every time OracleAS Portal users create, edit or delete a particular page.

Any authorized user can view the OracleAS Portal Log Registry records. However, only the portal administrator can set up what information is to be logged. See Section 7.3.2.2, "Choosing Which Events are Logged" for more information.

> **Note:** With the introduction of OracleAS Web Cache into the OracleAS Portal architecture, some of the actions logged in OracleAS Portal Activity Log tables have become inaccurate. These actions include View, Execute (for Reports, Charts, and Hierarchies), and Show. The Activity Log tables and views still remain in the OracleAS Metadata Repository, as all other actions logged are still accurate

7.3.2.1 Logged Events

Table 7–3 lists the events that can be logged for different portal objects.

Table 7–3   Logged Events for OracleAS Portal Objects

Portal Object	Event
Pages	Create, Edit, Delete, Customize
Items	Create, Edit, Delete, Move, Check Out, Check In
Application Components	Create, Edit, Delete, Execute (except for Reports, Charts, and Hierarchies), Copy, Export, Rename, Generate, Access Control, Manage, Insert, Update, Save
Portlets	Add to Page, Delete from Page
Portlet Instances	Hide, Customize
Searches	Search

> **Note:** User and Group actions such as Create, Edit, and Delete are
> logged by Oracle Internet Directory and may be viewed from
> Oracle Directory Manager, if logging is enabled. For more
> information, refer to the *Oracle Internet Directory Administrator's
> Guide*.

7.3.2.2 Choosing Which Events are Logged

You can choose which events are logged in the Log Registry records.

1. In the **Services** portlet, click **Log Registry Administration**.

> **Note:** By default, the **Services** portlet is on the **Portal** sub-tab of
> the **Administer** tab on the **Portal Builder** page.

The Administer Log Registry page is displayed as shown in the following
image.

The preceding image shows two logging requests. The first creates an entry in
the Activity Log every time a portlet is customized. The second creates an entry

every time a page is created. If you want to log all possible requests, choose %
for each field.

2. Do one of the following:

Click **Add New Log Registry Record** to create a new Log Registry record and
specify logging criteria.

Or,

Edit logging criteria for an existing Log Registry record. To do this, perform the
following steps:

a. Click the **Edit** icon to edit logging criteria for an existing Log Registry
record (under **Edit/Delete Log Registry Record**).

The Edit Log Registry Record page is displayed as shown in the following
image.

b. Choose the objects that you wish to log, from the **Sub Domain** list. Valid
objects are listed in Table 7–3.

c. Choose which actions (or events) you want to log, from the **Action** list.
Valid actions are listed in Table 7–3.

d. Specify other logging criteria as required.

e. Click **OK**.

7.3.2.3 Activity Log Views

Several Activity Log views are available (named wwlog_*). These views exist in the
schema in which OracleAS Portal is installed. These views are granted to public;

however, the logs are secure according to the object's security. For example, information about pages is available only on pages for which the user has access privileges.

Table 7–4 lists all the Activity Log views and their descriptions. You can create simple OracleAS Portal DB Provider reports and charts based on these views if required.

Table 7–4   Activity Log Views

Log View	Description
wwlog_portal_admin_logs	All logs (only has records if the user is the portal administrator).
wwlog_user_logs	All logs created by current user.
wwlog_all_portlet_logs	Portlet instances on pages that the current user can view.
wwlog_all_document_logs	Documents that the current user can view.
wwlog_all_search_logs	Searches that the current user can view.
wwlog_all_item_logs	Items that the current user can view.
wwlog_all_component_logs	Components that the current user can view.
wwlog_all_object_logs	Summary view, which encompasses all the preceding views.

7.3.2.4 Accessing Activity Log Views Externally

You can also access information in the Activity Log views from outside of the OracleAS Portal browser-based interface, that is, using SQL*Plus, OracleAS Reports Services, and so on. To do this, you must first set the portal security context for your database session using the **wwctx_api.set_context** API:

```
wwctx_api.set_context (
   p_user_name => 'portal_username',
   p_password  => 'portal_pw'
);
```

7.4 Viewing Oracle Application Server Port Information

In Application Server Control, the Application Server **Ports** page shows a list of all the ports currently in use by the components of a particular Oracle Application

Server instance. This page is important when you are troubleshooting port conflicts among the various application server components.

Whenever possible, Application Server Control provides a link to the appropriate Oracle Enterprise Manager configuration page where you can modify the port settings for the component.

To access the Application Server **Ports** page:

1. Access the Application Server Control. See Section 7.1.1, "Accessing the Application Server Control" for details.

   If there is more than one standalone application server instance, your start page for the Application Server Control is the Oracle Application Server **Farm** home page.

2. Click an instance to take you to the Oracle Application Server instance home page.

3. Click the **Ports** link below the application server name to view the Application Server **Ports** page, as shown in Figure 7–7.

Figure 7–7   Oracle Application Server Ports Page

For information on managing ports, see the chapter "Managing Ports" in the *Oracle Application Server 10g Administrator's Guide*.

8

Configuring the Search Features in OracleAS Portal

This chapter provides information on setting up the search capabilities in OracleAS Portal. This includes how to set up Oracle Text.

This chapter contains the following sections:

- Search Options in OracleAS Portal
- Configuring OracleAS Portal Search Options
- Oracle Text
- Oracle Ultra Search

8.1 Search Options in OracleAS Portal

OracleAS Portal offers powerful search capabilities that you can customize according to your needs. A robust set of built-in search portlets enables you to perform searches on the portlet repository, portal pages and external sites.

Furthermore, you can perform searches against more than 100 document types including HTML, XML, PDF, word processing formats, spreadsheets formats, presentation formats, and other common business formats.

This section introduces the search options that are available in OracleAS Portal and gives some guidance on how you can choose which option is best for you:

- OracleAS Portal Search
- Oracle Ultra Search
- Default Search Functionality

8.1.1 OracleAS Portal Search

OracleAS Portal includes a set of built-in features tuned for searching content stored and managed within the OracleAS Portal Repository. These features are incorporated within these four search portlets that can be configured in a variety of ways:

- **Basic Search** -- this portlet allows simple keyword searches.

- **Advanced Search** -- this portlet enables you to enter more detailed search criteria, including operators on multiple attributes values.

- **Custom Search** -- this portlet is fully customizable and enables you to design a search portlet to suit your needs, including pre-defined searches that display results in place.

- **Saved Searches** -- this portlet enables you to repeat saved searches.

This form of search indexes metadata associated with content in the OracleAS Portal Repository, for example, display name, keyword, description, and similar attributes.

> **Note:** The following metadata is indexed: item attributes (Display Name, Description, Keywords, Author), page attributes (Display Name, Description, Keywords) and category/perspective attributes (Display Name, Description).

When full text indexing of the content within the OracleAS Portal Repository is required, these search features can be extended by enabling Oracle Text.

Oracle Text (optional)

You can extend the searching capabilities of OracleAS Portal using Oracle Text. When Oracle Text is enabled, all text-type attributes are indexed and in addition the following content is indexed:

- **Files** -- files in binary format can be indexed providing the file format is filterable by Oracle Text.

- **Web pages that URLs (in URL attributes) point to** -- the content must be plain text or HTML.

8.1.2 Oracle Ultra Search

Oracle Ultra Search is an application built on Oracle Text that provides an enterprise search capability over a variety of content repositories and data sources, including the OracleAS Portal Repository. Oracle Ultra Search is installed and pre configured for use within OracleAS Portal and includes a search portlet that can be embedded in OracleAS Portal pages.

From this portlet, a user can enter a search term and launch a search that returns a single result set that includes content from all configured data sources. When OracleAS Portal is configured as one of the data sources, the search can return only *public* OracleAS Portal content.

8.1.3 Default Search Functionality

After a standard OracleAS Portal installation you can start using the search features in OracleAS Portal right away. Without any additional configuration, you can place one of the built-in, OracleAS Portal search portlets on a page and use it to search portal content.

During installation, Oracle Text indexes are created and synchronized and Oracle Text searching is enabled in OracleAS Portal. However, it is important to note that new or modified content (items, pages, categories, perspectives) is not returned in search results until the Oracle Text indexes are synchronized again. To synchronize Oracle Text indexes, or to set up a regular synchronization schedule, see Section 8.3.5.1, "Synchronizing Oracle Text Indexes" and Section 8.3.5.2, "Scheduling Index Synchronization".

> **Note:** If you do not want to make use of the additional features provided by Oracle Text, you can disable this feature. See Section 8.2.2.1, "Enabling and Disabling Oracle Text in OracleAS Portal".

Table 8–1 shows some other default search settings. For information how to change these values, see Section 8.2.1, "Configuring OracleAS Portal Search Portlets".

Table 8–1   Default Search Settings

Search Setting Option	Default
Basic Search Portlets and Basic Search Box Items	Basic Search Results Page
Advanced, Custom and Saved Search Portlets	Search Results Page
Advanced Search Link	Advanced Search Page
Internet Search Engine Link	None
Hits per Page	20

The following images show default search portlets and pages:

Figure 8–1   OracleAS Portal Basic Search Portlet
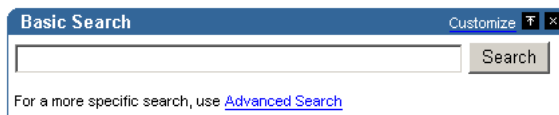
Figure 8–2   OracleAS Portal Basic Search Results Page

Figure 8–3   OracleAS Portal Advanced Search Portlet

Figure 8–4   OracleAS Portal Custom Search Portlet

Figure 8–5   OracleAS Portal Search Results Page
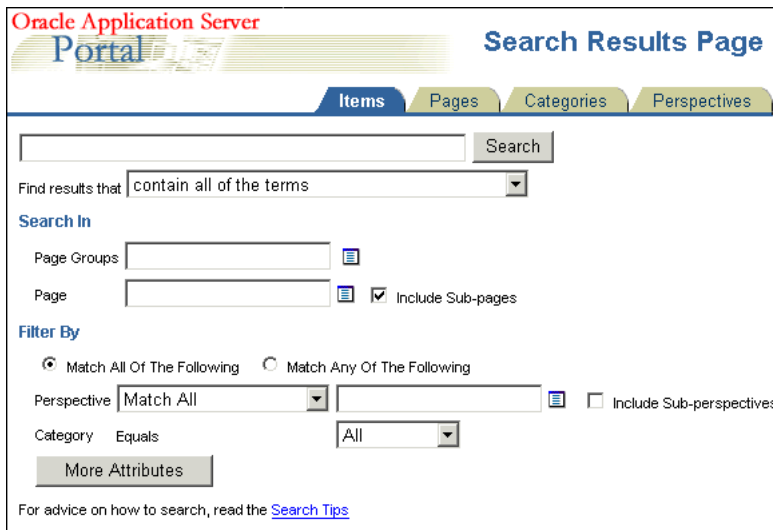
Figure 8–6   OracleAS Portal Saved Searches Portlet

Figure 8–7   Oracle Ultra Search Portlet

8.1.4 Deciding Which Search Options to Use

Choosing how to configure searching within OracleAS Portal begins with a careful examination of your goals for the search experience and understanding of your portal content. Some key questions include:

- **Searching 'breadth'** - do you wish to limit the results returned from your portal search to content managed within the OracleAS Portal Repository, or do you want to return results from other repositories?

- **Searching 'depth'** - is full text indexing of document content a key requirement, or is a metadata only index sufficient?

- **Content security policies and portal user profiles** - is your search experience targeted at primarily public, unauthenticated users searching public content or is it more targeted at individual users who have various levels of access privileges to the content?

- **Advanced searching features** - is the ability to order results by relevancy, view document themes and gists and other features of Oracle Text important capabilities to offer your users?

- **Administration** - how much time are you willing to invest in administering and maintaining indexes, data sources, and so on?

Use Table 8–2 to help match your search requirements to the most appropriate search configuration:

Table 8–2   OracleAS Portal Search Options

	OracleAS Portal (Oracle Text disabled)	**OracleAS Portal (Oracle Text enabled)**	**Oracle Ultra Search**
Searching 'Breadth'	OracleAS Portal Repository only	OracleAS Portal Repository only	OracleAS Portal Repository and other repositories
Searching 'Depth'	OracleAS Portal metadata only	Full text index	Full text index. For OracleAS Portal, public content only.
Content security and user profiles	Returns secure and public content in search results	Returns secure and public content in search results	Returns public content only
Advanced searching features	No	Yes	Yes
Administration	Minimal	Maintain full text indexes	Maintain full text indexes and configure data sources

8.1.5 Differences Between Oracle Ultra Search and OracleAS Portal Searches

This section highlights the main differences between Oracle Ultra Search and OracleAS Portal Search.

- Oracle Ultra Search only crawls public content

  OracleAS Portal is exposed to Oracle Ultra Search as a file system, and in order to see content in a folder, the folder must be public. If it is not public, none of the content from the folder or the sub-folder hierarchy is crawled. If you create a piece of content and make it public, it is only indexed if all the containing folders are also public.

- Oracle Ultra Search returns a single list of pages and items

  To Oracle Ultra Search, both OracleAS Portal pages and items are resources with metadata and content, or a visual representation that can be crawled, indexed, and returned in search results. This means that, Oracle Ultra Search can return a search result list that contains both pages and items. OracleAS Portal Search searches for distinct types of data (pages, items, categories and perspectives) and only one type of data can be searched at a time. Whilst Oracle Ultra Search does not treat categories and perspectives as separate searchable entities, it can (like OracleAS Portal Search), search for items and pages that have a particular perspective or category.

- Oracle Ultra Search searches content of displayed pages in addition to metadata

  OracleAS Portal Search searches page and item metadata. The Oracle Ultra Search crawler sees the rendered content plus the metadata. This means that Oracle Ultra Search can return results when OracleAS Portal search does not.

- OracleAS Portal Search excludes some item types

  OracleAS Portal Search can only return items of the following base item types:

  <None> that is, no base item type

  Base File

  Base URL

  Base Text

  Base PL/SQL

  Base Page Link

  Base Image

  Base Image Map

Simple Portlet Instance

Oracle Ultra Search indexes the visualization of any item type that appears on a page, irrespective of the base item type since it is the page rendition that is indexed. This means that all the content on the page, static and dynamic, is indexed by Oracle Ultra Search including banners and template items, login/logout links and so on.

- Oracle Text and scoring systems

  Both Oracle Ultra Search and OracleAS Portal Search use Oracle Text to index their content, however their implementations are different. Furthermore, Oracle Ultra Search uses a different scoring system to OracleAS Portal Search. In particular, a search term hits in the title section scores more highly than hits in the document content. For more information and details of how this can be customized, see *Oracle Ultra Search User's Guide*. OracleAS Portal Search treats all metadata and content with equal weighting.

8.1.6 Where to Find Configuration Information

OracleAS Portal Search Portlets
- To configure OracleAS Portal search portlets for use in OracleAS Portal, see Section 8.2.1, "Configuring OracleAS Portal Search Portlets".

You'll find additional information on using these search portlets to add search functionality to OracleAS Portal pages, in the *Oracle Application Server Portal User's Guide*.

Oracle Text
- To enable, disable and configure Oracle Text for use in OracleAS Portal, see Section 8.2.2, "Configuring Oracle Text Options in OracleAS Portal".

- For more information about Oracle Text, how to maintain Oracle Text indexes and troubleshooting information, see Section 8.3, "Oracle Text".

- To check that Oracle Text is installed and working correctly, see Appendix H, "Using TEXTTEST to Check Oracle Text Installation".

Oracle Ultra Search
- To set up Oracle Ultra Search and make the Ultra Search portlet available for use in OracleAS Portal, see Section 8.2.3, "Configuring Oracle Ultra Search Options in OracleAS Portal".

- For more information about Oracle Ultra Search, see Section 8.4, "Oracle Ultra Search".

8.2 Configuring OracleAS Portal Search Options

The OracleAS Portal search feature is installed with defaults so you can start using the search features right away. These initial defaults are described in Section 8.1.3, "Default Search Functionality".

This section describes how you, the portal administrator, can configure aspects of the search feature that affect *all* search portlets:

- Configuring OracleAS Portal Search Portlets
- Configuring Oracle Text Options in OracleAS Portal
- Configuring Oracle Ultra Search Options in OracleAS Portal

8.2.1 Configuring OracleAS Portal Search Portlets

This section describes how to configure aspects of the search feature that affect *all* OracleAS Portal search portlets:

- Choosing Search Result Pages
- Limiting the Number of Search Results on a Page
- Choosing an Advanced Search Link (Basic/Custom Search Portlets)
- Choosing an Internet Search Engine (Advanced/Custom Search Portlets)

8.2.1.1 Choosing Search Result Pages

You can determine the pages used to display search results from *all*:

- Basic Search portlets and Basic Search Box items
- Advanced, Custom and Saved Searches portlets

If you choose a new search result page, it is applied to both new and existing search portlets.

> **Note:** If page caching is enabled, the change may not immediately be seen in existing search portlets.

You can override this setting for a particular Custom Search portlet, if required. A Custom Search portlet only uses the result page specified here, if the **Where should the search results be displayed?** option is set to the Default Search Results Page. For more information on how to set Custom Search portlet options, refer to the *Oracle Application Server Portal User's Guide*.

To specify a search result page for your search portlets:

1. In the **Services** portlet, click **Search Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. In the Search Results Pages section, for **Basic Search Portlets and Basic Search Box Items**, choose a suitable search results page.

   You can choose any portal page that contains a search portlet. If you select a page without a search portlet, no results are displayed. The default is the Basic Search Results Page.

3. For **Advanced, Custom and Saved Search Portlets**, choose a suitable search results page.

   You can choose any portal page that contains a search portlet. If you select a page without a search portlet, no results are displayed. The default is the Search Results Page.

4. Select **OK**.

If a page you select is subsequently deleted, the associated **Page** field is empty. Choose another page and then click **OK**. If you click **Cancel**, you will see **Page Not Found** errors after search operations.

8.2.1.2 Limiting the Number of Search Results on a Page

You can limit the number of search results that are displayed on all search result pages. The limit is applied to results from Basic, Advanced and Custom Search portlets.

If the number of results returned by a search exceeds this number, the search results pages include Next and Previous icons that enable users to view all the results. See Figure 8–8.

Figure 8–8   Hits per Page Setting on Search Portlets

Results: Items 21 - 40 of 1053
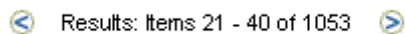
For example, if you specify Hits Per Page to be 10, the first 10 results are displayed on the first search results page, the next 10 on the second page, and so on.

> **Note:** If you change the limit, the new value does not effect existing search portlets, only new ones.

To specify the number of search results for every page:

1. In the **Services** portlet, click **Search Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. In the **Search Properties** section, for **Hits Per Page**, enter the number of search results to display on a page.

3. Click **OK**.

You cannot change this value for individual Basic or Advanced Search Portlets.

You can override this setting for a Custom Search portlet, if required. You can also hide the Next and Previous icons. For more information on how to set Custom Search portlet options, refer to the *Oracle Application Server Portal User's Guide*.

8.2.1.3 Choosing an Advanced Search Link (Basic/Custom Search Portlets)

An advanced search link is displayed on Basic Search portlets. Typically, the advanced search allows the user to specify additional search criteria. See Figure 8–9.

Figure 8–9 Advanced Search Link on Basic/Custom Search Portlets

For a more specific search, use Advanced Search

The advanced search link can be to an external site, another portal page, or a package call within OracleAS Portal.

Optionally, this link can be displayed on Custom Search portlets. For more information on how to set Custom Search portlet options, refer to the *Oracle Application Server Portal User's Guide*.

You can determine the destination of the Advanced Search Link, for all Basic/Custom Search portlet instances. When you specify a new Advanced Search Link, it is applied to both new and existing search portlets that display an Advanced Search link.

To enter advanced search link details:

1. In the **Services** portlet, click **Search Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. In the Advanced Search Link section, do one of the following:

   ■ Specify a destination **Page** for the Advanced Search link.

   The default is the **Advanced Search** Page, which contains the built-in **OracleAS Portal Advanced Search** portlet. However, you can select any portal page displaying advanced search options, the page does not have to contain one of the OracleAS Portal search portlets. For example, you can use a JSP page containing advanced search options if one existed in your portal.

   If the page you select is subsequently deleted, this field is empty. Choose another page and then **OK**. If you click **Cancel**, the advanced search links will all still point to the deleted page.

   ■ Specify a **URL** for the Advanced Search link.

   Enter the URL you want to use. If you have created a customized search engine that you want to use for advanced searches throughout the portal, you can specify its link here.

   You can specify an absolute URL, or a relative URL. For example, `http://www.myfavoritesearchengine.com` creates a link directly to this Internet search site.

   If you enter a relative URL (that is, a portal package), the value specified here is appended to the Portal schema URL and this results in a call to the portal package. Note how the value is appended, depending on whether the value specified begins with '/':

   /value results in this URL: `http://<webserver>:<port>/<value>`

   value results in this URL:
   `http://<webserver>:<port>/pls/<dad>/<value>`

3. Select **OK**.

8.2.1.4 Choosing an Internet Search Engine (Advanced/Custom Search Portlets)

An Internet search engine link is displayed on Advanced Search portlets. So, if users do not find the information they need when they search OracleAS Portal, they can extend their search using an Internet Search Engine. See Figure 8–10.

Figure 8–10   Internet Search Engine Link on Advanced/Custom Search Portlets

For searching the Internet, use YAHOO

Optionally, this link can be displayed on Custom Search portlets. For more information on how to set Custom Search portlet options, refer to the *Oracle Application Server Portal User's Guide*.

When you set the URL of an Internet search engine and the link text that users click to access the specified Internet search engine, it applies to all new and existing Advanced/Custom Search portlet instances that display an Internet search link.

1. In the **Services** portlet, click **Search Settings**.

    By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. In the **Internet Search Engine** section, for **URL**, enter the URL of an Internet search engine. For example, `http://www.yahoo.com`.

    The URL must be fully formed, and include any associated parameters.

3. For **Link Text,** enter the text that users click to access the specified Internet search engine. For example: YAHOO

    If you enter YAHOO, this text is displayed as a link in Advanced Search portlets and optionally in Custom Search portlets. See Figure 8–10.

4. Select **OK**.

If the Internet Search Engine properties (URL and Link Text) are not specified, no Advanced or Custom Search portlets will display a link to an Internet search engine.

8.2.2 Configuring Oracle Text Options in OracleAS Portal

This section describes how to configure Oracle Text features in OracleAS Portal:

- Enabling and Disabling Oracle Text in OracleAS Portal

- Setting Oracle Text Search Result Options

- Setting a Base URL for Oracle Text

- Configuring Proxy Settings for Oracle Text

8.2.2.1 Enabling and Disabling Oracle Text in OracleAS Portal

You can enable and disable the use of Oracle Text when searching in OracleAS Portal. For more information, see Section 8.3, "Oracle Text".

1. In the **Services** portlet, click **Search Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Select **Enable Oracle Text Searching** to make use of Oracle Text when searching OracleAS Portal.

   Deselect this option at any time to disable the use of Oracle Text.

   > **Note:** If you see the message Oracle Text is not installed, Oracle Text is not installed in the database and is not available in OracleAS Portal. Arrange with your database administrator to have Oracle Text installed. Once installed, you must run the following command in SQL*Plus to create the Oracle Text role:
   >
   > inctxgrn.sql
   >
   > This file is located in the directory *ORACLE_HOME*/portal/admin/plsql/wws.
   >
   > Log on using the user name and password for the PORTAL schema. You must also create Oracle Text indexes, see Section 8.3.4, "Creating and Dropping Oracle Text Indexes".

3. Click **OK**.

8.2.2.2 Setting Oracle Text Search Result Options

When Oracle Text is enabled, you can display additional information for items (documents/files) when they are returned as search results. For each item returned you can:

- View major **themes** in a chart. A theme shows the nouns and verbs that occur most frequently.

- View a short summary about the content (**gist**). Gists are derived from how frequently those nouns and verbs appear.

- View an HTML version

- View an HTML version of the file with search terms highlighted in a specific color and font

Themes and gists are optional and HTML highlighting can be customized as follows:

1. In the **Services** portlet, click **Search Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Select **Enable Themes And Gists** to create a theme and gist for each item returned by the search.

   > **Note:** Themes and gists are not available for all languages.

3. For **Highlight Text Color**, choose the color to highlight search terms found in the HTML version of items returned by the search.

4. For **Highlight Text Style**, choose the style to apply to search terms found in the HTML version of the items returned by the search.

5. Click **OK**.

8.2.2.3 Setting a Base URL for Oracle Text

Oracle Text needs a *base URL* to resolve relative URLs into fully qualified absolute URLs. For more information, see Section 8.3.6.1, "Relative URLs".

To specify the Base URL for Oracle Text:

1. In the **Services** portlet, click **Search Settings**.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Enter the **Oracle Text Base URL** in the format:
   `http://<host>:<port>/pls/<dad>`

   For example: `http://myportal.com:4000/pls/design`

If no value is specified, no relative URLs are indexed and therefore, any URL content that relative URLs points to, cannot be searched.

3. Click **OK**.

8.2.2.4 Configuring Proxy Settings for Oracle Text

Oracle Text uses OracleAS Portal proxy server settings to access URL content. This is necessary when OracleAS Portal lies behind a firewall and URL items point to content beyond this firewall. For more information, see Section 8.3.6.4, "URL Index Proxy Settings".

To configure the global proxy settings for OracleAS Portal, see Section 5.5, "Configuring OracleAS Portal to Use a Proxy Server".

8.2.3 Configuring Oracle Ultra Search Options in OracleAS Portal

This section describes how to set up Oracle Ultra Search for use in OracleAS Portal. You must complete the tasks in this section, before you can add the Ultra Search portlet to a portal page and use this feature:

- Accessing the Oracle Ultra Search Administration Tool

- Registering OracleAS Portal as a Content Source

- Registering the Ultra Search provider with OracleAS Portal

Note: Before using Oracle Ultra Search features in OracleAS Portal, also ensure that all necessary database and middle-tier configuration is complete. For detailed information, see Section 8.4, "Oracle Ultra Search".

8.2.3.1 Accessing the Oracle Ultra Search Administration Tool

1. Click **Ultra Search Administration** in the **Services** portlet.

   By default, the **Services** portlet is on the **Portal** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Log in.

   See Also: *Oracle Ultra Search User's Guide*

8.2.3.2 Registering OracleAS Portal as a Content Source

1. Access the Oracle Ultra Search administration tool as described in Section 8.2.3.1, "Accessing the Oracle Ultra Search Administration Tool".

2. On the **Instances** tab, click **Apply** to set the instance.

   If you have more than one instance make sure to select the instance you want to manage first.

3. On the **Crawler** tab, enter the **Cache Directory Location** and the **Crawler Log File Directory**.

   These directory locations are on the machine where Oracle Application Server middle-tier is installed. For example, `/tmp` for the Cache Directory Location and `/tmp` for the Crawler Log File Directory.

4. On the **Sources** tab, click the **Oracle Source** sub-tab, choose **Oracle Portal (Crawlable)** from the **Create Source** drop-down list and click **Go**.

5. Enter OracleAS Portal registration details:

   a. Enter the **Portal Name**.

   b. For **URL base**, enter the base URL for the portal.

      Use the format: `http://<hostname>:<port>/pls/<portal_DAD>/<portal_schema>`

      For example, `http://myserver.abc.com:7778/pls/portal/portal`

   c. Click **Register Portal**.

6. Select the page groups that you would like to create data sources for and then click **Create portal data sources**.

   You can optionally edit each of the portal data sources to add content types for processing. For example, you can add the MS Word Doc, MS Excel Doc, PDF Doc types.

> **Note:** A page group is available as a crawlable data source, when either:
>
> - The option **Display Page to Public Users** is set on its root page (**Edit Page:Access** tab).
>
> - The *View* privilege is granted to PUBLIC (**Edit Page Group: Access** tab).
>
> See *Oracle Application Server Portal User's Guide* for more information.

7. Finally, on the **Schedules** tab, schedule the indexing of the portal data sources:

   a. Click **Create New Schedule** and enter a **Name** for the schedule.

   b. Click **Proceed to Step 2** and specify synchronization schedule details.

   c. Click **Proceed to Step 3**, select Portal from the drop down list and then click **Get Sources**.

   d. Move the sources over to the **Assigned Sources** box and click **Finish**.

   Clicking the Status link for the source enables you to optionally run the synchronization immediately.

8.2.3.3 Registering the Ultra Search provider with OracleAS Portal

OracleAS Portal comes with a pre-built sample portlet for Oracle Ultra Search. To access the portlet the provider must first be registered with OracleAS Portal.

1. In the **Remote Providers** portlet, click **Register a Provider**.

   By default, the **Remote Providers** portlet is on the **Portlet** sub-tab of the **Administer** tab on the **Portal Builder** page.

2. Fill in all the fields on the first step of the wizard.

   - Your **Timeout** setting effects how long pages take to render if the portlet is not responding, so do not set it too high.

   - Leave **Implementation Style** set to Web.

   - Click **Next** to continue.

3. Enter the **URL** for the Ultra Search provider.

   By default this is:

```
http://machine.domain:7778/provider/ultrasearch/servlet/soaprouter
```

4. Set the **Service ID** to be 'ultrasearch'.

5. Change the **Login Frequency** to **Once per User Session** and then click **Next**.

6. Click the **Browse Groups icon**, select AUTHENTICATED_USERS and grant Execute privileges.

7. Finally, click **Finish**.

Now, the **Ultra Search** portlet can be added to a portal page.

8.3 Oracle Text

Oracle Text adds powerful text search and intelligent text management to the Oracle database. OracleAS Portal uses the Oracle Text functionality to extend its search capabilities.

Use of Oracle Text with OracleAS Portal is an optional feature that can be enabled and disabled by the portal administrator. See Section 8.2.2.1, "Enabling and Disabling Oracle Text in OracleAS Portal".

The use of Oracle Text with OracleAS Portal is described in the following sections:

- Understanding OracleAS Portal Searches with Oracle Text Enabled

- Oracle Text Prerequisites

- Oracle Text Indexes

- Creating and Dropping Oracle Text Indexes

- Maintaining Oracle Text Indexes

- Indexing and Searching URL Content

- Viewing the Status of Oracle Text Indexes

- Monitoring Oracle Text Indexing Operations

- Viewing Indexing Errors

- Translating Indexing Errors to Objects in OracleAS Portal

- Handling Indexing Hangs or Crashes

- Troubleshooting Oracle Text Installation Problems

- Updating Oracle Text Indexes When Upgrading to Oracle Database 10g

You'll find additional information in the Oracle Text documentation on the Oracle Technology Network, http://otn.oracle.com/documentation.

8.3.1 Understanding OracleAS Portal Searches with Oracle Text Enabled

If Oracle Text is disabled and you perform a basic search, that is, enter a search term only, the item attributes Display Name, Description, Keywords and Author and the page attributes Display Name, Description and Keywords are searched. General searches such as these do not match against custom attributes.

Searches that specify criteria against selected attributes, that is, an advanced search, matches against the selected attributes. If the attribute is a file attribute, the file name is searched. If the attribute is a URL attribute, the URL HREF is searched, that is, the literal string `http://www.google.com`.

If Oracle Text is enabled when you perform a basic search, all text-type attributes, including custom text attributes are searched. Furthermore, the content of files are searched. Files in binary format can be searched providing the file format is filterable by Oracle Text.

Likewise, when Oracle Text is enabled, the content of pages that URLs point to are also searched. This content must be plain text or HTML to be searchable.

8.3.2 Oracle Text Prerequisites

Oracle Text is a standard component of the Oracle9*i* Database Server. If you want to use the Oracle Text functionality in OracleAS Portal, it is essential that the Oracle Text component is correctly installed and functioning properly.

Ensure that:

- **Oracle Text is installed in the OracleAS Portal Repository database**. Since OracleAS Portal 9.0.2.2 and from the 3.0.9.8.4 patchset onwards, the Oracle Text component is required to be in the OracleAS Portal Repository database before the OracleAS Portal Repository can be installed. This is because some OracleAS Portal packages make reference to the `ctx_ddl` packages in the `CTXSYS` schema in which the Oracle Text component resides.

- **Oracle Text upgrade steps are complete.** In particular, during database upgrades, it is essential that any manual steps that pertain to Oracle Text are completed correctly.

- **Library path for Oracle Text INSO filters is set correctly.** For the Oracle Text INSO filters to function correctly, the ctxhx executable (called during indexing) needs to be able to load the appropriate shared libraries.

  - For UNIX platforms, ensure that the library path used by ld includes *ORACLE_HOME*/ctx/lib for both the TNS listener and the environment where the database is started. The library path environment variable for the different UNIX platforms are as follows:

    Solaris, Tru64 UNIX, Linux -> $LD_LIBRARY_PATH

    HP/UX   -> $SHLIB_PATH and $LD_LIBRARY_PATH

    IBM AIX   -> $LIBPATH

    For more, detailed information, see *About Inso Filtering Technology* in the *Oracle Text Reference*.

    Whenever you change the library path you must restart both the database and the listener for Oracle Text indexing operations to work. If one or both environment variables are not set, documents are not indexed as expected and the table ctx_user_index_errors may be full of DRG-11207, status 137 errors. See Also Section 8.3.11.1, "Common Document Indexing Errors".

  - On Windows platforms, ensure that the appropriate DLLs are located in *ORACLE_HOME*\bin and that this path is included in the PATH environment variable, that is, in the environment from where the Oracle server is started.

You can use the TEXTTEST utility to check that Oracle Text functionality is installed and working correctly. The TEXTTEST utility is located at *ORACLE_HOME*/portal/admin/texttest/textest. For more information, see Appendix H, "Using TEXTTEST to Check Oracle Text Installation".

8.3.3 Oracle Text Indexes

If you want to use the Oracle Text functionality in OracleAS Portal, several Oracle Text indexes are required in the OracleAS Portal schema. Details of these indexes are described in the following sections:

- Oracle Text Index Overview

- Oracle Text Index Preferences

- Datastore Procedures

- Granting CTXAPP Role to the OracleAS Portal Schema

- [Multilingual Functionality (Multilexer)](#)
- [STEM Searching](#)

8.3.3.1 Oracle Text Index Overview

All required Oracle Text indexes are built automatically during OracleAS Portal installation by procedures in the package `wwv_context`.

> **See Also:** [Appendix G, "Using the wwv_context APIs"](#)

Procedures in this package can also be used after portal installation to manage the indexes, including removing or creating them. For more information, see [Section 8.3.4.3, "Dropping All Oracle Text Indexes Using ctxdrind.sql"](#) and [Section 8.3.4.1, "Creating All Oracle Text Indexes Using ctxcrind.sql"](#).

> **Note:** Oracle Text can be disabled, even when Oracle Text indexes are present. See [Section 8.2.2.1, "Enabling and Disabling Oracle Text in OracleAS Portal"](#).

[Table 8–3](#) describes the Oracle Text indexes that are required.

Table 8–3   Oracle Text Indexes In the OracleAS Portal Schema

Index	Table.column	Purpose	Datastore type	Filter Type
WWSBR_ CORNER_CTX_ INDX	wwpob_ page$.ctxtxt	Index page metadata	user datastore	Null
WWSBR_DOC_ CTX_INDX	wwdoc_ document$.blob _content	Index document content	direct datastore	INSO
WWSBR_ PERSP_CTX_ INDX	wwv_ perspectives.ctx txt	Index perspective metadata	user datastore	Null
WWSBR_ THING_CTX_ INDX	wwv_ things.ctxtxt	Index item metadata	user datastore	Null
WWSBR_ TOPIC_CTX_ INDX	wwv_ topics.ctxtxt	Index category metadata	user datastore	Null

Table 8–3   (Cont.) Oracle Text Indexes In the OracleAS Portal Schema

Index	Table.column	Purpose	Datastore type	Filter Type
WWSBR_URL_ CTX_INDX	wwsbr_ url$.absolute_ url	Index URL content	URL datastore	Null

Most of the Oracle Text indexes use a user datastore, that is, for each row that needs to be indexed, a PL/SQL procedure is called which produces a document that gets indexed for that row.

The exceptions are the indexes WWSBR_DOC_CTX_INDX (Document index) and WWSBR_URL_CTX_INDX (URL index):

- **Document index:** Uses a direct datastore, that is, it indexes the document content held directly in the BLOB type blob_content column of the wwdoc_ document$ table.

- **URL index:** Fetches the content to be indexed for each row in the wwsbr_url$ table from the location pointed to by the absolute_url$ column.

Only the Document index uses filters. This index uses the INSO filter to convert documents into a plain text format. No document is excluded from filtering, that is, the INSO filter processes all documents, including those which are in plain text or HTML.

You'll find additional information in the Oracle Text documentation on the Oracle Technology Network, http://otn.oracle.com/documentation.

8.3.3.2 Oracle Text Index Preferences

Preferences are used to configure the Oracle Text indexes used by OracleAS Portal. The preferences are created and owned by the OracleAS Portal schema, that is, they are created using the ctx_ddl package, which resides in the CTXSYS schema, and the data representing the preferences is actually stored in relational tables in the CTXSYS schema.

The Oracle Text index preferences must exist before the indexes are created. Subsequent changes to these preferences do not take affect until the Oracle Text indexes are dropped and re-created.

The Oracle Text index preferences that are used during OracleAS Portal installation to create Oracle Text indexes can be re-created using the package wwv_context. Some Oracle Text index preferences can also be configured by you, the portal administrator. For example, when you set the global OracleAS Portal proxy settings

they are used by Oracle Text to populate the proxy preferences used in Oracle Text indexes.

See Also: Appendix G, "Using the wwv_context APIs".

In addition, the Oracle Text indexes use a number of Lexer preferences to control the linguistic aspects of the indexing. The Lexer preferences are created by the script `sbrimtlx.sql`. You can run this script at any time to re-create the Lexer preferences. The script is located in the directory *ORACLE_ HOME*/portal/admin/plsql/wws.

You'll find additional information in the Oracle Text documentation on the Oracle Technology Network, http://otn.oracle.com/documentation.

8.3.3.3 Datastore Procedures

For each of the Oracle Text indexes that use *user datastores*, a procedure is created in the `CTXSYS` schema where Oracle Text is installed. The procedures are called for each row that is to be indexed for the given index. These procedures in turn call procedures in the OracleAS Portal schema.

The datastore procedures are named:

- WWSBR_THING_CTX_<user_id>

- WWSBR_CORNER_CTX_<user_id>

- WWSBR_PERSP_CTX_<user_id>

- WWSBR_TOPIC_CTX_<user_id>

Where <user_id> is the user_id (as found in the ALL_USERS view) of the OracleAS Portal Repository schema. This postfix is required so that the procedure names do not clash, if multiple OracleAS Portal repositories exist in the same database.

If for any reason these procedures do not exist, Oracle Text functionality will not work. This might happen, for example, if the `CTXSYS` schema is dropped and re-installed. In this situation, the procedures can be re-installed by running the script `inctxgrn.sql` as the OracleAS Portal schema owner:

```
SQL> @inctxgrn.sql
```

This script also grants the `CTXAPP` role to the OracleAS Portal schema. See Section 8.3.3.4, "Granting CTXAPP Role to the OracleAS Portal Schema". The script is located in the directory *ORACLE_HOME*/portal/admin/plsql/wws.

8.3.3.4 Granting CTXAPP Role to the OracleAS Portal Schema

To use Oracle Text functionality, the role CTXAPP must be granted to the OracleAS Portal schema. This is done automatically during OracleAS Portal Repository installation and normally no further action is required.

If for any reason this grant is revoked, Oracle Text functionality will not work. For example, this may occur if the CTXAPP role is dropped when the CTXSYS schema is re-installed.

To restore the necessary grants, run the script inctxgrn.sql as the OracleAS Portal schema owner:

```
SQL> @inctxgrn.sql
```

This script also creates the OracleAS Portal user datastore procedures, which are required in the CTXSYS schema. See Section 8.3.3.3, "Datastore Procedures". The script is located in the directory *ORACLE_HOME*/portal/admin/plsql/wws.

8.3.3.5 Multilingual Functionality (Multilexer)

OracleAS Portal uses the Oracle Text Multilexer to enable language-specific searching in OracleAS Portal. The Multilexer:

- Controls the way that the linguistic aspects of searching are carried out.

- Allows content, items, pages, categories and perspectives and their translations, to be treated in a way that is appropriate to their language.

Lexer preferences are used to configure the Multilexer used for all the Oracle Text indexes. The lexer preferences are created by the script file sbrimtlx.sql. You can modify these preferences if required, but if you do, you must drop and re-create the Oracle Text indexes for the changes to take a effect.

For more information on the Multilexer, refer to Oracle Text documentation on the Oracle Technology Network, http://otn.oracle.com/documentation.

8.3.3.6 STEM Searching

By default, STEM searching is used when Oracle Text is enabled in OracleAS Portal. STEM searching enables you to search for words that have the same root as the specified term. For example, a stem of $sing expands into a query on the words sang, sung, sing.

However, STEM searching is used only when logged in to OracleAS Portal in one of the languages where STEM searching is supported in Oracle Text, that is, the following languages:

```
AMERICAN ENGLISH
CANADIAN FRENCH
DUTCH
UK ENGLISH
FRENCH
GERMAN DIN
GERMAN
ITALIAN
LATIN AMERICAN SPANISH
MEXICAN SPANISH
SPANISH
```

In all other languages, the STEM operator is not used.

8.3.4 Creating and Dropping Oracle Text Indexes

All the required Oracle Text indexes are created automatically during OracleAS Portal Repository installation. However, if the indexes are subsequently dropped, it may be necessary to re-create them.

Creating and dropping indexes is a very time-consuming and resource-intensive operation, so plan this task during non-business hours.

> **Note:** Dropping and re-creating Oracle Text indexes changes search results. It also changes the operators that are shown in the submission form, and the result attributes that are shown (The attributes *score*, *view as HTML*, *view as HTML with highlight themes*, and *gist* are only shown if you use Oracle Text).
>
> Dropping or creating the Oracle Text indexes does not invalidate OracleAS Web Cache, so autoquery portlet results, and search submission forms will still be returned until they expire from the cache, or until you go into the Edit Defaults screen of the portlet.

These sections describe how to create and drop Oracle Text indexes:

- Creating All Oracle Text Indexes Using ctxcrind.sql
- Creating a Single Oracle Text Index
- Dropping All Oracle Text Indexes Using ctxdrind.sql
- Dropping a Single Oracle Text Index

8.3.4.1 Creating All Oracle Text Indexes Using ctxcrind.sql

You can re-create all the Oracle Text indexes using scripts and packages provided with OracleAS Portal. The primary script for creating the Oracle Text indexes is `ctxcrind.sql` and it is located in the directory *ORACLE_HOME*/portal/admin/plsql/wws.

When you run the script `ctxcrind.sql` as the OracleAS Portal Repository schema owner:

- All the required Oracle Text indexes and preferences are created. For more information, see Section 8.3.3, "Oracle Text Indexes".

- If there are existing Oracle Text indexes, all existing preferences and valid indexes are dropped and re-created. Indexes are judged to be valid if:

    - The row in view `user_indexes` for the relevant index has index_status, domidx_status, and domidx_opstatus all set as 'VALID'.

    - The index has an entry in `ctx_user_indexes` with the idx_status set to 'INDEXED'.

- Any indexes that are not present are also created.

This process can take several hours.

To create Oracle Text indexes using the script ctxcrind.sql:

1. Navigate to the directory *ORACLE_HOME*/portal/admin/plsql/wws.

2. In SQL*Plus, log on using the user name and password for the PORTAL schema.

3. In SQL*Plus, type this command:

   ```
   ctxcrind.sql
   ```

If the operation is successful, all the Oracle Text indexes and preferences are created in the OracleAS Portal Repository schema. If it fails, check that your system has met all the requirements in Section 8.3.2, "Oracle Text Prerequisites".

> **Note:** The time it takes to create the Oracle Text indexes, depends on how many items and page groups exist in your portal.

The script `ctxcrind.sql` makes a call to the procedure:

```
wwv_context.createindex( p_message => l_message );
```

Where p_message is an `out` parameter that passes a completion message. The call `wwv_context.createindex()` is in turn equivalent to:

```
wwv_context.drop_prefs;  /* Drop all Oracle Text preferences for the indexes,
except Lexer preferences */
wwv_context.drop_invalid_indexes; /* Drop all valid indexes */
wwv_context.create_prefs; /* Create all Oracle Text preferences,except Lexer
preferences */
wwv_context.create_missing_indexes(l_indexes);  /* Create missing indexes and
record them in l_indexes */
wwv_context.touch_index(l_indexes); /* Mark all rows for created indexes as
requiring synchronization */
wwv_context.sync;     /* Synchronize indexes */
wwv_context.optimize; /* Optimize indexes */
```

> **See Also:**   Appendix G, "Using the wwv_context APIs".

8.3.4.2 Creating a Single Oracle Text Index

If you want to create a specific index, use the procedure `wwv_context.create_index(p_index)`.

Use `p_index` to specify which index you want to create, that is, one of the following:

```
wwv_context.PAGE_TEXT_INDEX
wwv_context.DOC_TEXT_INDEX
wwv_context.PERSPECTIVE_TEXT_INDEX
wwv_context.ITEM_TEXT_INDEX
wwv_context.CATEGPRY_TEXT_INDEX
wwv_context.URL_TEXT_INDEX
```

This procedure creates an empty index, that is, it contains no content and therefore no search results can be returned from it. For information on how to mark an index for update and to synchronize an index, see Section 8.3.5.4, "Synchronizing All the Index Content".

8.3.4.3 Dropping All Oracle Text Indexes Using ctxdrind.sql

You can drop all of the Oracle Text indexes and preferences (except for the Lexer preferences), using the script `ctxdrind.sql`. This script is located in the directory *ORACLE_HOME*/portal/admin/plsql/wws.

To drop all the Oracle Text indexes using the script ctxdrind.sql:

1. Navigate to the directory *ORACLE_HOME*/portal/admin/plsql/wws.

2. In SQL*Plus, log on using the user name and password for the PORTAL schema.

3. In SQL*Plus, type this command:

   ```
   ctxdrind.sql
   ```

   This script makes a call to:

   ```
   wwv_context.dropindex(p_message  =>l_message);
   ```

   Where p_message is an `out` parameter that passes a completion message.

   > **Note:**   When the Oracle Text indexes are dropped, any views and packages that reference tables on which the indexes were created will become invalid.
   >
   > These views and packages are automatically validated when they are next accessed. Alternatively, it is possible to validate the views and packages manually.

8.3.4.4 Dropping a Single Oracle Text Index

If may want to drop a specific Oracle Text index. For example, you may want to drop the URL index so that it can be re-created with a different proxy setting, without having to drop and re-create all the other indexes.

To do this, drop the index directly using the command:

```
SQL> drop index <index_name> force;
```

For example, to drop the URL index, enter:

```
SQL> drop index WWSBR_URL_CTX_INDX force;
```

8.3.5 Maintaining Oracle Text Indexes

Oracle Text indexes must be maintained to ensure that search results are returned accurately and efficiently. There are two aspects to consider when maintaining Oracle Text indexes, synchronization and optimization:

- **Synchronization** Updates an Oracle Text index based on a queue.

■ **Optimization** Compacts fragmented rows and removes old data in an Oracle Text index. As an index is synchronized, it grows in such a way as to consume more disk space than necessary and this reduces the efficiency of queries.

Oracle Text gives you full control over how often each index is synchronized and optimized. For example, you can choose to synchronize every five seconds, if it is important to reflect text changes quickly in the index. Alternatively, you can choose to synchronize once a day, for more efficient use of computing resources and a more optimal index.

For more information about synchronization, see:

■ Synchronizing Oracle Text Indexes

■ Scheduling Index Synchronization

■ Deciding How Often to Synchronize Oracle Text Indexes

■ Synchronizing All the Index Content

For more information about optimization, see:

■ Optimizing Oracle Text Indexes

■ Scheduling Index Optimization

■ Choosing the Optimization Interval

8.3.5.1 Synchronizing Oracle Text Indexes

When new content is added to an Oracle Text index it must be indexed before it can be searched. Furthermore, when any row in a table on which the indexes are created are updated, that row is marked as needing synchronization. These are referred to as *pending rows* and they are not returned in search results until the index is synchronized.

In OracleAS Portal this means that any content (items, pages, categories, perspectives) that is added or modified is not searchable until the indexes are synchronized, that is, the new content is not returned in search results.

You can see which rows are marked pending, using the view ctx_user_pending. You can also use the script textstat.sql to see the number of rows that need to be synchronized for each index. For more information, see Section 8.3.7, "Viewing the Status of Oracle Text Indexes".

To keep your indexes up to date so you can search on new content, use the procedure wwv_context.sync(). This procedure synchronizes all the Oracle Text indexes, indexing all pending rows.

To synchronize Oracle Text indexes:

Execute this procedure as the Portal schema owner from SQL*Plus, using the command:

```
exec wwv_context.sync();
```

This procedure operates across all virtual private portal subscribers.

8.3.5.2 Scheduling Index Synchronization

In most installations, it is desirable to schedule index synchronization to run automatically at regular intervals so that newly added or updated content gets indexed periodically. You can schedule a job using the script `textjsub.sql`. This uses `dbms_job` to call `wwv_context.sync` at regular intervals.

The script takes three parameters and it can also be used to alter or remove a synchronization job:

```
start_time       - a valid date or 'START' or 'STOP'
start_time_fmt   - start time format mask.
                     Ignored if start_time is 'START' or 'STOP'
interval_minutes - minutes between each run. Ignored if 'STOP'
```

If you set start_time to 'START', the second argument is ignored and the next job is scheduled to run immediately. Subsequent jobs are run after the interval specified.

If you set start_time to 'STOP', the job is removed and other arguments are ignored.

To schedule Oracle Text index synchronization:

Run the script `textjsub.sql`. For example, to schedule index synchronization every 60 minutes, enter:

```
SQL> @textjsub.sql START NOW 60
```

8.3.5.3 Deciding How Often to Synchronize Oracle Text Indexes

The appropriate interval between index synchronization jobs depends on:

- How often new content is added to your portal site.

- Whether it matters that newly added or altered content is not searchable immediately.

- How long is it reasonable to have to wait before added or updated content is searchable.

Depending on your requirements, the synchronization interval could be anything from a few minutes to several days.

When OracleAS Portal is initially installed, a job is set up that synchronizes the Oracle Text indexes every hour, starting immediately at the time of installation.

It is more efficient to synchronize a larger number of rows on a single occasion than to repeatedly synchronize a smaller number of rows, as the index becomes less fragmented. If an index is less fragmented, then it needs to be optimized less frequently. For more information, see Section 8.3.5.5, "Optimizing Oracle Text Indexes".

However, indexing a larger number of rows at once places a heavier load on the server. Synchronizing more frequently increases the total amount of work but spreads the load on the server. The job only synchronizes the rows that are pending, however, there is always some overhead, however small, in starting up the synchronization job.

8.3.5.4 Synchronizing All the Index Content

You can synchronize *all* the content for a particular Oracle Text index by marking *all* the rows for that index as *requiring synchronization*.

For example, when an index is initially created it is empty, so you would need to update the entire index content. This involves performing an update for the column that the index is created on. For every row in the indexed table use the procedure wvv_context.touch_index(p_index) to update the column.

After running this procedure, there is an entry in the table ctx_user_index_pending for every row in the table upon which the index was created.

Note also that this procedure works across all virtual private portal subscribers.

To synchronize all the content of an index:

Use the procedure wvv_context.touch_index(p_index). Where p_index enables you to specify one of these index names:

```
wwv_context.PAGE_TEXT_INDEX
wwv_context.DOC_TEXT_INDEX
wwv_context.PERSPECTIVE_TEXT_INDEX
wwv_context.ITEM_TEXT_INDEX
wwv_context.CATEGPRY_TEXT_INDEX
wwv_context.URL_TEXT_INDEX
```

To synchronize all the content of multiple indexes:

Use the procedure `wvv_context.touch_index(p_indexes)`. Where `p_indexes` enables you to specify a varray of index names to be synchronized (`wwsbr_array`).

8.3.5.5 Optimizing Oracle Text Indexes

Synchronizing Oracle Text indexes causes them to become fragmented. Each Oracle Text index is an inverted index where search terms are listed in a form that is efficient to look up. Each search term references the location of the term.

When new terms are added during synchronization, duplicate terms are not removed, so the index may contain the same term several times. This inflates the size of the index and causes the performance of search queries to deteriorate.

The solution is to optimize the Oracle Text indexes. This process compacts the indexes and (optionally) removes old data.

To optimize all of the Oracle Text indexes:

To optimize all of the Oracle Text indexes, use the procedure `wwv_context.optimize()`. This procedure takes the following parameters:

```
wwv_context.optimize
(
  p_optlevel in varchar2 default CTX_DDL.OPTLEVEL_FULL, -- FULL, FAST, TOKEN
  p_maxtime in number default null,  -- Maximum time for full optimization, in
minutes
  p_token in varchar2 default null -- Token to optimize (when TOKEN)
);
```

Internally this procedure calls the Oracle Text procedure `ctx_ddl.optimize_index` for each Oracle Text index and passes these parameters. It performs full index optimization as opposed to *fast* or *token* optimization.

You'll find additional information in the Oracle Text documentation on the Oracle Technology Network, http://otn.oracle.com/documentation.

> **Note:** If no Oracle Text indexes exist, the procedure `wwv_context.optimize` has no affect.

`wwv_optimize` only optimizes an Oracle Text index if it is sufficiently fragmented to require optimization. The measure of the fragmentation used is the average number of times a token that appears more than once, is found in the index. If this

average is greater than 10, the index is judged to require optimization. The fragmentation query used is as follows:

```
SELECT AVG(COUNT(*)) FROM DR$<index_name>$I
GROUP BY TOKEN_TEXT HAVING COUNT(*) > 1
```

Where <index_name> is the name of the index to be measured.

8.3.5.6 Scheduling Index Optimization

In most installations it is desirable to schedule the index optimization process to run automatically at regular intervals. You can schedule a job using the script optjsub.sql. This uses dbms_job to call wwv_context.optimize at regular intervals.

This script optjsub.sql takes three parameters and it can also be used to alter or remove an optimization job:

```
start_time        - A valid date or 'START' or 'STOP'
start_time_fmt    - Start time format mask.
                    Ignored if start_time is 'START' or 'STOP'
interval_minutes  - Minutes between each run. Ignored if 'STOP'
```

If you set start_time to 'START', the second argument is ignored and the next job is scheduled to run immediately. Subsequent jobs are run after the interval specified.

If you set start_time to 'STOP', the job is removed and other arguments are ignored.

To schedule Oracle Text index optimization:

Run the script optjsub.sql. For example, to schedule index optimization very 60 minutes, enter:

```
SQL> @optjsub.sql START NOW 60
```

This script is located in the directory *ORACLE_HOME*/portal/admin/plsql/wws. If there are no Oracle Text indexes present when you run this optimization job, the procedure has no affect.

8.3.5.7 Choosing the Optimization Interval

It is difficult to predict how often Oracle Text indexes need to be optimized as the frequency depends on the amount of content that is being loaded, the type of content being loaded, the synchronization schedule and many other factors.

However, if you measure the index fragmentation at regular intervals, you can determine how rapidly it is becoming fragmented. Using this information, you can set an appropriate optimization interval.

The procedure `wwv_context.optimize` only optimizes the index if it is judged to be fragmented. So, other than the minimal overhead of calling the job, it is quite safe to run this job more often than perhaps is required.

During OracleAS Portal installation, a job is set up to optimize all of the Oracle Text indexes, every 24 hours.

8.3.6 Indexing and Searching URL Content

If Oracle Text is enabled in OracleAS Portal, the content of URL attributes attached to items or pages are indexed. Once this URL content is indexed, it is searchable. When you enter search criteria for URL attributes, it is this URL content that is searched.

8.3.6.1 Relative URLs

In OracleAS Portal you can enter a relative URL for an URL attribute. When these URLs are rendered as links on a portal page they are relative to the base HREF that is set in the HTML <head> section for a portal page. The format of the base HREF is:

```
<protocol>://<server>:<port>/pls/<dad>/
```

For example, in the HTML <head> section you might see:

```
<base href="http://myserver.abc.com/pls/portal/">
```

In this example:

- The relative URL `/help/index.html` is resolved by the browser to:

  `http://myserver.abc.com/help/index.html`

- The relative URL `!PORTAL.mypackage.proc` (with no leading /) is resolved by the browser to:

  `http://myserver.abc.com/pls/portal/!PORTAL.mypackage.proc`

The base HREF on a page is dependent on the URL used to request the page. As it is possible to use more than one URL to access the page, the base HREF reflects the URL used to access the page.

Oracle Text Base URL Setting

When indexing URL content, Oracle Text needs to know how to resolve relative URLs into fully qualified absolute URLs. As Oracle Text does not have the context of an initial request from which to determine the correct base HREF, you must specify the base HREF that is used. You set this option, by specifying the **Oracle Text Base URL** property on the **Search Settings** page. See Section 8.2.2.3, "Setting a Base URL for Oracle Text".

During OracleAS Portal installation, this option is set automatically.

The format of the Oracle Text Base URL is:

```
<protocol>://<server>:<port>/pls/<dad>/
```

For example: `http://myserver.abc.com/pls/portal/`

> **Note:** Do not specify an Oracle Text Base URL beginning with https, as https URLs are not indexed by Oracle Text.

If you change the Oracle Text Base URL, it does not take affect immediately. When a URL is edited, it is marked as requiring synchronization and Oracle Text will use the new preference the next time the index is synchronized. If you want to force all URLs to immediately use a new Oracle Text Base URL value, you can mark the entire content of the URL Index as requiring synchronization, using the procedure:

```
SQL> wwv_context.touch_index(wwv_context.URL_TEXT_INDEX);
```

This procedure acts across all subscribers. In a single virtual private portal subscriber, this is equivalent to:

```
SQL> update wwsbr_url$ set absolute_url = null;
...
SQL> commit;
```

8.3.6.2 Unsupported URLs

Oracle Text cannot index URLs that use these protocols:

- https
- javascript

If a URL item specifies one of these protocols it is not indexed. You will not see a corresponding error in the Oracle Text error logs.

Also, since Oracle Text cannot index https URLs, you should not enter an https URL for the Oracle Text Base URL option. If you do this, no relative URLs are indexed.

8.3.6.3 Supported URLs

Oracle Text can index URLs that use these protocols:

- **http**
- **file** - File URLs must be accessible from the database server.
- **ftp** - FTP URLs must point to locations that do not require authentication as Oracle Text is not able to authenticate — even as an anonymous user.

8.3.6.4 URL Index Proxy Settings

When indexing URL content, Oracle Text can use proxy servers to access URLs. This may be necessary when OracleAS Portal lies behind a firewall and URLs items point to content beyond this firewall. As indexing takes place from the OracleAS Portal Repository server, it is the proxy settings required on this machine that are important.

The URL index uses the same proxy settings that are used globally for OracleAS Portal. These are set on the Proxy Settings page, available from the **Services** portlet. See Section 8.2.2.4, "Configuring Proxy Settings for Oracle Text".

The proxy settings are used when Oracle Text indexes are created. So, if you change the proxy settings the indexes must be re-created. If you need to drop all your indexes and re-create them, use the scripts ctxdrind.sql (drop indexes) and ctxcrind.sql (create indexes), For more information, see Section 8.3.4, "Creating and Dropping Oracle Text Indexes":

```
SQL> @ctxdrind.sql
...
SQL> @ctxcrind.sql
...
```

These scripts drop and re-create *all* of the indexes and this can take a long time if your indexes are large. Alternatively, you can drop and re-create the Oracle Text preferences and URL index only:

```
begin
   -- Drop and recreate the Oracle Text preferences
   -- to pick up the new proxy settings.
   wwv_context.drop_prefs();
   wwv_context.create_prefs();
end;
```

```
/
-- Check that the proxy settings used by the index are correct
select prv_attribute attribute, prv_value value
  from ctx_user_preference_values
  where prv_attribute in ('TIMEOUT','HTTP_PROXY','NO_PROXY')
/

begin
    -- Drop and recreate the URL index
    wwv_context.drop_index(wwv_context.URL_TEXT_INDEX);
    wwv_context.create_index(wwv_context.URL_TEXT_INDEX);

    -- Mark all of the rows for the index as pending
    wwv_context.touch_index(wwv_context.URL_TEXT_INDEX);

    -- Syncronize and optimize
    wwv_context.sync();
    wwv_context.optimize();
end;
/
```

8.3.7 Viewing the Status of Oracle Text Indexes

You can determine the status of Oracle Text indexes from several tables and views accessible from the OracleAS Portal schema.

You'll find additional information in the Oracle Text reference documentation on the Oracle Technology Network, http://otn.oracle.com/documentation.

To view a status report for Oracle Text indexes, use the script `textstat.sql`:

```
SQL> @textstat.sql
```

This script is located in the directory *ORACLE_HOME*/portal/admin/plsql/wws. Here is an example of the information that is generated by this script:

```
SQL> @textstat
Oracle Text Indexes (there should be 6):

INDEX_NAME          STATUS   DOMIDX_STATUS DOMIDX_OPSTATUS IDX_STATUS
------------------- -------- ------------- --------------- ------------
WWSBR_CORNER_CTX_INDX VALID    VALID         VALID           INDEXED
WWSBR_DOC_CTX_INDX    VALID    VALID         VALID           INDEXED
WWSBR_PERSP_CTX_INDX  VALID    VALID         VALID           INDEXED
WWSBR_THING_CTX_INDX  VALID    VALID         VALID           INDEXED
WWSBR_TOPIC_CTX_INDX  VALID    VALID         VALID           INDEXED
```

```
WWSBR_URL_CTX_INDX     VALID     VALID         VALID           INDEXED

6 rows selected.

Indexes with rows waiting to be indexed:

Index                   Rows to Index
----------------------  -------------
WWSBR_CORNER_CTX_INDX      2677

PL/SQL procedure successfully completed.

Scheduled Text Jobs:

LAST_DATE LAST_SEC   NEXT_DATE NEXT_SEC   B  FAILURES INTERVAL          WHAT
--------- ---------  --------- ----------  - ------ ------------------------
25-MAR-03 04:57:32   26-MAR-03 04:57:32   N   0     SYSDATE + 24/24 wwsbr_stats.gather_
stale;
25-MAR-03 04:57:32   26-MAR-03 04:57:32   N   0     SYSDATE + 1440/(24*60) wwv_
context.optimize(CTX_DDL.OPTLEVEL_FULL,1440,null);
25-MAR-03 06:59:30   25-MAR-03 07:59:30   N   0     SYSDATE + 60/(24*60)  wwv_context.sync;

Running Text Jobs:
no rows selected

SQL>
```

From this script you can view the following status information:

- **Oracle Text Index Status** - The first section in the status report shows whether all of the Oracle Text indexes exist and their current status. All working, valid indexes display 'VALID' for the first three status columns and 'INDEXED' for the final column as shown in this example. All six indexes should be listed.

- **Number of Pending Rows Per Index** - The next section lists any indexes that are waiting to be indexed. An entry is listed for every index that has rows waiting to be indexed, or are pending. The number of pending rows is also shown.

- **Scheduled Oracle Text Job Details** - The scheduled text job section lists any jobs that are scheduled for Oracle Text index maintenance. The report shows the last date and time that the job was run and the next date when the job is due to be run. The column labeled **B** shows whether the job is broken or not, that is, if the job is marked **Y** it is broken and is not run. The **Interval** column indicates the next time that a job will run and finally the **What** column indicates the procedure that will be run for each job.

- **Active Oracle Text Job Details** - The final section details any jobs that are running when the textstat.sql report is run.

8.3.8 Monitoring Oracle Text Indexing Operations

Oracle Text logs information to a file when indexes are created and populated. This enables you to monitor the progress of indexing operations, keep track of indexes and troubleshoot any problems that may arise.

8.3.8.1 Using start_log to Monitor Index Operations

You can use the ctx_output.start_log (filename) command to log output from the indexing process. In the subsequent example, the log file is named textindex.log.

```
ctx_output.start_log('textindex.log');
ctx_output.add_event(ctx_output.event_index_print_rowid);
...
-- Create or syncronize the indexes
...
ctx_output.end_log;
```

You can determine the location of the log file using the LOG_DIRECTORY parameter in ctx_adm.set_parameter. In the subsequent example, the log output directory is set to /tmp. Once the directory is set, all subsequent Oracle Text logs are output log files to this directory.:

```
ctxsys.ctx_adm.set_parameter('LOG_DIRECTORY', '/tmp');
```

8.3.8.2 Using logcrind.sql to Monitor Index Creation

You can use the script logcrind.sql (instead of ctxcrind.sql) to create the Oracle Text indexes with logging enabled. The script takes one parameter which is the name of the log file, for example:

```
SQL> @logcrind.sql textindex.log
```

This script sets the LOG_DIRECTORY to be the same as the database udump directory, as specified by the user_dump_dest initialization parameter.

The add_event call (used in the preceding example) is also used in the script logcrind.sql and this outputs the rowid of every row indexed to the log. This logging allows indexing operations to be tracked and also indicates whether the indexing of each row is successful or not.

Here is a sample from an Oracle Text indexing log:

```
13:53:27 05/06/03 begin logging
13:53:27 05/06/03 event
13:53:42 05/06/03 log
13:53:42 05/06/03 event
13:53:48 05/06/03 Creating Oracle index "RCLEWLEY2"."DR$WWSBR_CORNER_CTX_INDX$X"
13:53:48 05/06/03 Oracle index "RCLEWLEY2"."DR$WWSBR_CORNER_CTX_INDX$X" created
13:53:49 05/06/03 Creating Oracle index "RCLEWLEY2"."DR$WWSBR_DOC_CTX_INDX$X"
13:53:49 05/06/03 Oracle index "RCLEWLEY2"."DR$WWSBR_DOC_CTX_INDX$X" created
13:53:49 05/06/03 Creating Oracle index "RCLEWLEY2"."DR$WWSBR_PERSP_CTX_INDX$X"
13:53:49 05/06/03 Oracle index "RCLEWLEY2"."DR$WWSBR_PERSP_CTX_INDX$X" created
13:53:50 05/06/03 Creating Oracle index "RCLEWLEY2"."DR$WWSBR_THING_CTX_INDX$X"
13:53:50 05/06/03 Oracle index "RCLEWLEY2"."DR$WWSBR_THING_CTX_INDX$X" created
13:53:51 05/06/03 Creating Oracle index "RCLEWLEY2"."DR$WWSBR_TOPIC_CTX_INDX$X"
13:53:51 05/06/03 Oracle index "RCLEWLEY2"."DR$WWSBR_TOPIC_CTX_INDX$X" created
13:53:51 05/06/03 Creating Oracle index "RCLEWLEY2"."DR$WWSBR_URL_CTX_INDX$X"
13:53:51 05/06/03 Oracle index "RCLEWLEY2"."DR$WWSBR_URL_CTX_INDX$X" created
13:54:16 05/06/03 sync index: RCLEWLEY2.WWSBR_CORNER_CTX_INDX
13:54:17 05/06/03 Begin document indexing
13:54:17 05/06/03 INDEXING ROWID AAAUUcAAJAAAlhMAAA
13:54:17 05/06/03 INDEXING ROWID AAAUUcAAJAAAlhMAAI
..
13:54:18 05/06/03 INDEXING ROWID AAAUUcAAJAAAlhQAAk
13:54:18 05/06/03 Errors reading documents: 0
13:54:18 05/06/03 Index data for 159 documents to be written to database
13:54:18 05/06/03    memory use: 225971
13:54:18 05/06/03 Begin sorting the inverted list.
13:54:18 05/06/03 End sorting the inverted list.
13:54:18 05/06/03 Writing index data to database.
13:54:18 05/06/03    index data written to database.
13:54:18 05/06/03 End of document indexing. 159 documents indexed.
```

8.3.9 Viewing Indexing Errors

Any errors that occur when an index is created or synchronized are logged in the view `CTX_USER_INDEX_ERRORS`. You can see details for these errors, using the command:

```
SQL> desc ctx_user_index_errors;
 Name                   Null?    Type
 ---------------------- -------- ---------------
 ERR_INDEX_NAME         NOT NULL VARCHAR2(30)
 ERR_TIMESTAMP                   DATE
 ERR_TEXTKEY                     VARCHAR2(18)
```

```
ERR_TEXT                          VARCHAR2(4000)

SQL>
```

This view gives the index name, the rowid (ERR_TEXTKEY column) corresponding to the row in the indexed table and an error message that indicates the cause of the failure. Furthermore, the error log file indicates the rowid for the row in the table that is being indexed and a success or failure message.

Typically, you do not see errors for the item (WWSB_THING_CTX_INDX), page (WWSBR_CORNER_CTX_INDX), category (WWSBR_TOPIC_CTX_INDX) or the perspective (WWSBR_PERSP_CTX_INDX) indexes as these index content that is produced by OracleAS Portal which is easy to index. It is more common to see errors when indexing document and URL content.

For the Document index, the content may have to be filtered in order to turn a binary document into plain text for indexing. There are a number of reasons this may fail. For example, the document format may not be supported by the Oracle Text filter.

For the URL index, the URL content has to be fetched and this could fail for a number of reasons. For example, the URL may indicate a location that is not accessible as the OracleAS Portal server is behind a firewall and the proxy settings are not set correctly. Or, maybe the URL is incorrect, or perhaps the site that is being access is down.

8.3.10 Translating Indexing Errors to Objects in OracleAS Portal

The indexing errors shown in the view `CTX_USER_INDEX_ERRORS` or the Text indexing logs, show the rowid of the row in the table being indexed when the error occurred. You can use this information to determine which row is causing an indexing problem and you can also determine exactly which portal item or page this row corresponds to.

8.3.10.1 Item Indexing Errors

The rowid gives the row in the items table that caused problems. You can use a direct query to find out more information about that row. For example:

```
select i.name, i.title,        -- item title
       p.name page_name,       -- page name
       p.title page_title,     -- page display name
       pg.name page_group,     -- page group name
       sl.title page_group_title -- page group display name (default language)
```

```
 from wwv_things i,
      wwpob_page$ p,
      wwpob_item$ pi,
      wwsbr_sites$ pg,
      wwsbr_site_languages$ sl
where i.masterthingid = pi.master_thing_id
  and i.siteid = pi.site_id
  and pi.page_id = p.id
  and sl.siteid = pg.id
  and sl.language = pg.defaultlanguage
  and pi.page_site_id = p.siteid
  and pg.id = i.siteid
  and i.rowid = 'AAAOwMAAJAAAWISAAF
```

8.3.10.2 Page Indexing Errors

The rowid gives the row in the pages table. You can use a direct query to find out more information about the page that was being indexed. For example:

```
select p.name page_name,
      p.title page_title,
      pg.name page_group,
      sl.title page_group_title
  from wwpob_page$ p,
      wwsbr_sites$ pg,
      wwsbr_site_languages$ sl
 where sl.siteid = pg.id
   and sl.language = pg.defaultlanguage
   and pg.id = p.siteid
   and p.rowid = 'AAAOv/AAJAAAaSSAAB'
```

8.3.10.3 Category Index Errors

You can use a direct query against the category table to determine faulty categories. You can also use a join to show the page group. This query shows the category name and display name, and the page group name and display name.

```
select c.title, c.name, pg.name, sl.title
  from wwv_topics c,
      wwsbr_sites$ pg,
      wwsbr_site_languages$ sl
 where sl.siteid = pg.id
   and sl.language = pg.defaultlanguage
   and pg.id = c.siteid
```

```
and rowid='AAAOv/AAJAAAaSSAAB'
```

8.3.10.4 Perspective Indexing Errors

These are similar to categories. If you use a direct query against the perspective table will illustrate the faulty perspectives. You can also use a join to show the page group.

```
select p.title, p.name, pg.name, sl.title
  from wwv_perspectives p,
       wwsbr_sites$ pg,
       wwsbr_site_languages$ sl
 where sl.siteid = pg.id
   and sl.language = pg.defaultlanguage
   and pg.id = p.siteid
   and p.rowid = 'AAAOv/AAJAAAaSSAAB'
```

8.3.10.5 Document Index Errors

You are more likely to see errors with the Document index. In this case the index is on the table where the documents are actually stored. Therefore, you have to join back to the item table to determine the associated item.

The following query gives the document filename and item's Name and Display Name that a document query is associated with. select d.filename, i.name, i.title

```
  from wwv_things i,
       wwdoc_document$ d,
       wwv_docinfo di
 where
    d.name = di.name(+)
    and di.thingid = i.id(+)
    and di.masterthingid = i.masterthingid(+)
    and di.siteid = i.siteid(+)
    and d.rowid = 'AAAOYyAAJAAAWAaAAF'
```

Note that not all documents are necessarily associated with items, in which case the query would need to be modified to join in a similar way to the page table.

8.3.10.6 URL Index Errors

Like the Document index, you have to join back to the item table to determine the associated item.

The following query shows the URL, and item Name and Display Name.

```
select u.url, u.absolute_url, i.name, i.title
  from wwv_things i,
       wwsbr_url$ u
 where u.object_id = i.id
   and u.object_siteid = i.siteid
   and u.object_type = 'ITEM'
   and u.rowid = 'AAAOYyAAKAAAWAaAAB'
```

Note that the URL may not be attached to an item, it may be attached to a page, in which case the query must be modified to join in a similar way to the page table.

8.3.11 Common Indexing Errors

8.3.11.1 Common Document Indexing Errors

Typically, document indexing errors are in the format:

```
DRG-11207: user filter command exited with status n
```

The actual exit status indicates the cause of the problem. For a description of common exit status values and their meanings, log on to Oracle Metalink, at `http://metalink.oracle.com` and read the article *Troubleshooting DRG-11207 errors*. This article has **DocId 210319.1**.

8.3.11.2 Common URL Indexing Errors

Here are some common URL indexing errors. The list is not exhaustive but it highlights some of the more common errors you may see:

```
DRG-11604 URL store: access to %(1)s is denied
```
Access to the document is denied to the indexing user agent. The crawler is not capable of authenticating or managing cookies returned by the site. Check that the URL can be accessed. If it is protected, it may not be possible to index the content.

```
DRG-11609 URL store: unable to open local file specified by %(1)s
DRG-11610 URL store: unable to read local file specified by %(1)s
```
These occur for file:// URLs where the file indicated cannot be opened or read. Remember that the file needs to be accessible from the machine on which the OracleAS Portal repository database is running. Check that the file exists and that it is accessible from the database machine as the database user.

```
DRG-11611 URL store: unknown protocol specified in %1)s
```

The protocol specified in the URL is not one that the Oracle Text user agent recognizes. This can happen if no protocol is specified. A common cause of this problem is that a relative URL is specified but the Oracle Text Base URL option is not set to fully qualify the URL. Also, Oracle Text can only index http, file and ftp URLs. Look at the URL that has failed and make sure that it is in a supported fully qualified format, including a valid protocol.

`DRG-11612 URL store: unknown host specified in %(1)s`
The URL specified a host in the URL that cannot be resolved from the OracleAS Portal repository database server. It may be that a firewall lies between the OracleAS Portal repository server and the location specified by the URL. In this case it might be necessary to use a proxy server to access the URL. Check that the URL is correct and that the host is accessible from the OracleAS Portal database server. Also check that the OracleAS Portal proxy settings are correct and that the index is using the proxy settings. See Section 8.2.2.4, "Configuring Proxy Settings for Oracle Text".

`DRG-11613 URL store: connection refused to host specified by %(1)s`
This means that the host specified in the URL was resolved but the http request was refused. Check that the URL is correct and that it is accessible.

`DRG-11614 URL store: communication with host specified in %(1)s timed out`
The request timed out. Check that the URL is correct and accessible.

`DRG-11616 URL store: too many redirections trying to access %(1)s`
When accessed, a URL can cause a redirect to another URL. This in turn can cause a redirect and so on. If a large number of redirects occur, this error will result. This can occur if a redirection loop is found.

`DRG-11622 URL store: unknown HTTP error getting %(1)s`
An HTTP error that is not explicitly handled by Oracle Text has occurred. The HTTP error is reported in the error message.

8.3.12 Handling Indexing Hangs or Crashes

If for any reason a document or URL cannot be indexed, an error is logged. This situation should not prevent the indexing operation completing normally. However, any content that fails to be indexed is not searchable.

Sometimes an indexing operation can fail catastrophically, that is, the index operation is terminated before the indexes are properly populated. In most cases, such problems should be reported to Oracle Support. However, in some instances you may be able to work around the problem temporarily, that is, create the indexes

but exclude any content causing failure. For more information, see Section 8.3.12.2, "Preventing Indexes From Hanging and Crashing".

Rarely, an indexing operation causes a disastrous failure, that is, the server process performing the indexing is terminated. When this happens, this message is displayed in the client running the indexing operation:

```
ORA-03113 End of file on communication channel
```

> **Note:** If you are unsure whether an indexing operation completed successfully, repeat the operation from SQL Plus where *end of file* errors are clearly reported.

If the server process is terminated, the event should also be recorded in the database logs. Use the database alert log to determine the location of any trace files that are written. The trace files may indicate errors such as ORA-0600 or ORA-7445. For example, this trace file shows errors that occurred when creating Oracle Text indexes using the script logcrind.sql:

```
ksedmp: internal or fatal error
ORA-7445: exception encountered: core dump [drsfdatam()+308] [SIGSEGV]
[Address not mapped to object] [0x0] [
] []
Current SQL statement for this session:
declare
l_dump_dest varchar2(512);
p_logfile varchar2(100) := 'sync_2012.log';
begin
dbms_output.enable(10000);
select value into l_dump_dest from v$parameter
where name = 'user_dump_dest';
ctxsys.ctx_adm.set_parameter('LOG_DIRECTORY',l_dump_dest);
ctx_output.start_log(p_logfile);
ctx_output.add_event(ctx_output.event_index_print_rowid);
dbms_output.put_line('Log file is: '||ctx_output.logfilename);
wwv_context.sync();
ctx_output.end_log;
end;
----- PL/SQL Call Stack -----
object line object
handle number name
8198f83c 244 package body CTXSYS.DRIDISP
8198f83c 377 package body CTXSYS.DRIDISP
```

```
8198f83c 334 package body CTXSYS.DRIDISP
8178acc8 403 package body CTXSYS.DRIDML
827124b0 2033 package body CTXSYS.DRIDDL
827124b0 2090 package body CTXSYS.DRIDDL
817ea0f0 1324 package body CTXSYS.CTX_DDL
8185a488 828 package body TOOLS.WWV_CONTEXT
82d83ed8 18 anonymous block
----- Call Stack Trace -----
```

8.3.12.1 Identifying Whether an Index Operation is Hanging

The easiest way to determine if an indexing operation is hanging is to run the indexing operation with Oracle Text logging enabled. For more information, see Section 8.3.8, "Monitoring Oracle Text Indexing Operations".

With logging enabled, the rowid of each row is recorded when it is indexed and you can see when an indexing operation hangs on the same row for a prolonged period. It may be normal for some rows to take a few minutes to process but if an operation takes much longer than expected, this could indicate a problem.

In general, looking in view CTX_USER_INDEX_ERRORS is not useful when trying to find out why an indexing process is hanging or crashing. This is because information is only visible in this view after it is committed and a commit will not occur whilst an indexing operation is hanging and may not occur at all if the operation crashes.

Operations such as URL indexing and document filtering can take quite a long time to process. Both of these operations are subject to timeout mechanisms to avoid lengthening this process even further:

- **URL indexing timeout -** The default timeout for fetching URL content is 30 seconds. If URL content is not retrieved within 30 seconds, the attempt is abandoned, a failure error is reported in the view CTX_USER_INDEX_ERRORS and the indexing process continues to the next row. In most cases, 30 seconds is sufficient time to fetch URL content. However, once the content is retrieved it must be indexed, so the total time can be slightly more than the URL timeout value.

- **Document filtering timeout -** The timeout for document filtering operations is not a *hard* timeout limit. The timeout setting, which by default is 120 seconds, is the time that is waited while no output is produced by the INSO filter. If the timeout is exceeded the current filtering operation is terminated, the content for the current document is not indexed and the indexing process proceeds to the

next document. If the INSO filter output file is still growing after 120 seconds, the filtering operation is allowed to continue.

These timeout mechanisms help to avoid problems with URL and document indexing, two areas where issues are likely to arise. However, you may still encounter situations where an indexing operation hangs indefinitely.

8.3.12.2 Preventing Indexes From Hanging and Crashing

If certain content is causing indexing operations to fail, you can exclude the content from the indexing process. First, you must identify the row that is causing the problem. This section describes how to do this and the additional steps required to exclude such content.

Step 1 Identify the rowid Causing Indexing Problems

You can do this using the Oracle Text logging facility, with `print rowid event` enabled. If you look at the generated log file you can determine the rowid (of the row being processed) when failure occurred. In most cases it is this rowid that is causing indexing problems.

However, in some cases the actual rowid being processed may not be written to the log file when the failure occurs. In this case you must determine the *next* rowid:

- If the entire table is being synchronized, for example, when an index is first created, the rowid is the next rowid from the table. To determine the rowid, select from the table without an order by clause.

- When only a few pending rows are being updated, look at the view `ctx_user_pending` to determine the next rowid.

When you have identified which row is causing your indexing problems, you should verify that it is the correct row. You do this by reproducing the failure while synchronizing that row only.

If the Oracle Text indexes do not exist, create the indexes (but do not populate them) using these command:

```
SQL> exec wwv_context.drop_prefs;
PL/SQL procedure successfully completed.
SQL> exec wwv_context.create_prefs;
PL/SQL procedure successfully completed.
SQL> declare
  2      l_indexes wwsbr_array;
  3  begin
  4      wwv_context.create_missing_indexes(l_indexes);
  5  end;
```

```
   6  /
PL/SQL procedure successfully completed.
SQL>
```

This creates all of the indexes, with no rows pending.

Step 2 Mark the Problem rowid As Pending

The next step is to mark the row suspected of causing indexing problems as pending. The column you need to update depends on which index you are updating. The names of these columns are indicated in the subsequent examples. You must replace the rowid given in these examples, with the rowid you wish to verify:

URL index (WWSBR_URL_CTX_INDX) The `absolute_url` column is populated by a trigger, so set it here to null:

```
update wwsbr_url$ set absolute_url=null where rowid = 'AAAOwQAAJAAAU0+AAL';
```

Document index (WWSBR_DOC_CTX_INDX) Update the `blob_content` column, but preserve the original `blob_content` value:

```
update wwdoc_document$ set blob_content = blob_content where rowid = 'AAAOYyAAJAAAWAaAAF'
```

Item index (WWSBR_THING_CTX_INDX) This index uses a user datastore created on the `ctxtxt` column. The value of this column is irrelevant and in OracleAS Portal is always 1.

```
update wwv_things set ctxtxt = '1' where rowid = 'AAAOwMAAJAAAU0eAAB'
```

Page index (WWSBR_FOLDER_CTX_INDX) Similar to the item index.

```
update wwpob_page$ set ctxtxt = 1 where rowid = 'AAAOwMAAJAAAWITAAA'
```

Category index (WWSBR_TOPIC_CTX_INDX) Similar to the item index.

```
update wwv_topics set ctxtxt = 1 where rowid = 'AAAOwMAAJAAAWITAAA'
```

Perspective index (WWSBR_PERSP_CTX_INDX) Similar to the item index.

```
update wwv_perspectives set ctxtxt = 1 where rowid = 'AAAOwMAAJAAAWITAAA'
```

If you have a site with several subscribers installed then you may need to switch subscriber before you can see the row that you are interested in. To change subscribers, use the following procedure to set the session context for a lightweight user:

```
wwctx_api.set_context
(
        p_user_name      IN varchar2,
        p_password       IN varchar2 default null,
        p_company        IN varchar2 default null
);
```

The package `wwctx_api` is a public API package.

More On Portal Center

You'll find additional information on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

After the column update, the suspect row is placed in the pending queue.

Step 3 Synchronize the Index

Now you can synchronize the index and see if the same problem occurs, using the command:

```
SQL> exec wwv_context.sync();
```

This command synchronizes the suspect row only as it is the only row in the pending queue. The row can be updated again to repeat the test.

Step 4 Exclude the Content Causing Problems

You can prevent the indexing operation from hanging or crashing in the future, by modifying, or even removing the row causing indexing problems. For example, if it is a document, you can edit the associated item in OracleAS Portal and remove the document.

> **Note:** Contact Oracle Support if your system hangs or crashes during indexing operations. If you can provide specific detail relating to the content causing the problem, it will help them to reproduce the problem more readily.

8.3.12.3 Preventing Document Filter Operations from Hanging

If the INSO filter hangs for some reason, it can cause a document filtering operation to hang. A timeout mechanism is supposed to prevent this from happening but sometimes the INSO filter hangs before any output is logged.

In this case you can prevent the filter operation from hanging, by terminating the INSO filter process. When you do this, the document being indexed at the time is

not indexed and therefore the content of this document is not searchable. However, the indexing operation can resume.

When documents are filtered a separate INSO filter executable ctxhx is called (by the Oracle server) to filter each document:

On UNIX: *ORACLE_HOME*/ctx/bin/ctxhx

On Windows: *ORACLE_HOME*\bin\ctxhx

Here, *ORACLE_HOME* relates to the database home for the database where the OracleAS Portal repository is installed.

The commands used to terminate the INSO filter process depends on your database platform. For example, on most UNIX platforms, you can use ps to find the process ID of the hung ctxhx process and then the kill command to terminate the ctxhx process.

> **Note:** This is not a supported procedure. Only take this action when investigating indexing problems.

8.3.12.4 Running Document Filter Operations Manually

You can call the INSO filter ctxhx directly from the operating system. If you are having a problem filtering documents, you can use ctxhx to:

- Verify the problem, that is, by isolating and testing the filtering stage directly.

- Determine whether the document filtering operation is exiting abnormally, or is hanging.

For this to work, ensure that ctxhx can link with any dependent shared libraries at run time:

- For UNIX platforms, ensure that the library path used by ld includes *ORACLE_ HOME*/ctx/lib for both the TNS listener and the environment where the database is started. The library path environment variable for the different UNIX platforms are as follows:

    Solaris, Tru64 UNIX, Linux -> $LD_LIBRARY_PATH

    HP/UX   -> $SHLIB_PATH and $LD_LIBRARY_PATH

    IBM AIX   -> $LIBPATH

    For more, detailed information, see *About Inso Filtering Technology* in the *Oracle Text Reference*.

- On Windows, ensure that `ORACLE_HOME`\bin is included in the PATH environment variable.

The INSO filter `ctxhx` is located:

On UNIX platforms `ORACLE_HOME`/ctx/bin/ctxhx

On Windows `ORACLE_HOME`\bin\ctxhx

When you run this command with no arguments, some help information is displayed. However, typically, run the command as follows:

```
ctxhx infile.doc outfile.out ascii8 unicode
```

The last parameter must be the characterset of the OracleAS Portal repository database, that is, unicode in this example.

8.3.13 Troubleshooting Oracle Text Installation Problems

If you are experiencing Oracle Text-related problems, use the TEXTTEST utility to check that Oracle Text functionality is installed and setup correctly. See Appendix H, "Using TEXTTEST to Check Oracle Text Installation".

8.3.14 Updating Oracle Text Indexes When Upgrading to Oracle Database 10*g*

If a database containing an OracleAS Portal Repository schema is upgraded to Oracle Database 10*g*, some modifications are required before Oracle Text functionality works correctly in OracleAS Portal. This is because, in an Oracle9*i* database the datastore procedures are created in the CTXSYS schema, whereas in Oracle Database 10*g* they must be created in the index owning schema.

New OracleAS Portal Repository installations into Oracle Database 10*g* will work correctly with no additional modification.

To make the required modifications, run the following, as the OracleAS Portal schema owner, for each pre-existing OracleAS Portal schema in the upgraded Oracle Database 10*g*:

```
begin
    wwv_context_util.drop_context_procs();
    wwv_context.drop_prefs();
    wwv_context.create_prefs();
    wwv_context.update_index_prefs();
end;
```

This code drops the datastore procedures from the CTXSYS schema and re-creates the Oracle Text preferences used by OracleAS Portal. If any Oracle Text indexes exist, the re-created preferences are used to update the settings used by these Oracle Text indexes. The Oracle Text functionality will now work correctly.

8.4 Oracle Ultra Search

This section provides information about Oracle Ultra Search and how to perform the required database and middle-tier configuration. Specific topics in this section include:

- Oracle Ultra Search Overview
- Configuring the Oracle Application Server Infrastructure
- Configuring the Database for Oracle Ultra Search
- Configuring the Oracle Ultra Search Middle-Tier Component
- Configuring Remote Crawler Hosts
- The Oracle Ultra Search Portlet Sample

8.4.1 Oracle Ultra Search Overview

This section covers the following topics:

- About Oracle Ultra Search
- About the Oracle Ultra Search Sample Query Applications
- About the Oracle Ultra Search Administration Tool

8.4.1.1 About Oracle Ultra Search

Oracle Ultra Search is built on the Oracle database server and Oracle Text technology that provides uniform search-and-locate capabilities over multiple repositories: Oracle databases, other ODBC compliant databases, IMAP mail servers, HTML documents served up by a Web server, files on disk, and more.

Oracle Ultra Search uses a *crawler* to collect documents. You can schedule the crawler to suit the Web sites that you want to search. The documents stay in their own repositories, and the crawled information is used to build an index that stays within your firewall in a designated Oracle database. Oracle Ultra Search also provides APIs for building content management solutions.

In addition, Oracle Ultra Search offers the following:

- A complete text query language for text search inside the database

- Full integration with the Oracle database server and the SQL query language

- Advanced features like concept searching and theme analysis

- Attribute mapping to facilitate attribute search across disparate repositories

- Indexing of all popular file formats (150+)

- Full globalization, including support for Chinese, Japanese and Korean (CJK), and Unicode

Oracle Ultra Search is made up of these components:

- Ultra Search Crawler

- Ultra Search Server Component

- Ultra Search Administration Tool

- Ultra Search APIs and Sample Applications

> **See Also:** *Oracle Ultra Search User's Guide*

*More On
Portal Center*

You'll find additional information in:

- The paper "How do I use Ultra Search with OracleAS Portal?" located on Portal Center, http://portalcenter.oracle.com.

- The Oracle Ultra Search papers and presentations on the Oracle Technology Network, http://otn.oracle.com/documentation.

Oracle Ultra Search is integrated with OracleAS Portal. This allows OracleAS Portal users to add a powerful multi repository search to their portal pages. It also has the capability to crawl OracleAS Portal's own repository and search *public* content.

Figure 8–11 shows an overview of the Oracle Ultra Search architecture:

Figure 8–11 Oracle Ultra Search Architecture
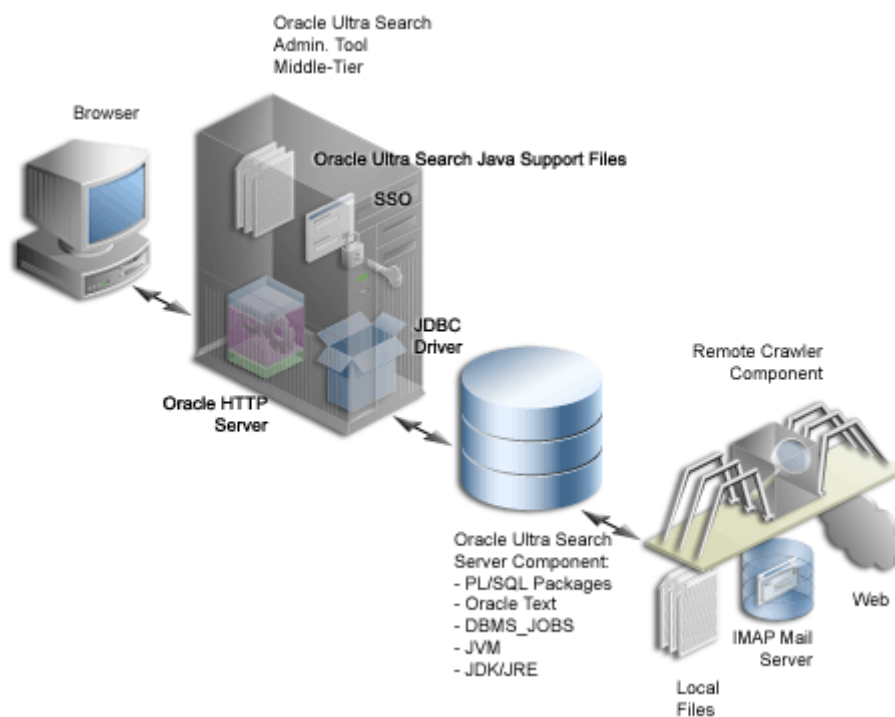
See Also: *Oracle Ultra Search User's Guide*

8.4.1.2 About the Oracle Ultra Search Sample Query Applications

Oracle Ultra Search includes fully functional sample query applications to query and display search results. The query applications are written as J2EE-compliant Web applications.

The sample query applications also includes the Ultra Search portlet, shown in Figure 8–12.

Figure 8–12   Oracle Ultra Search Portlet
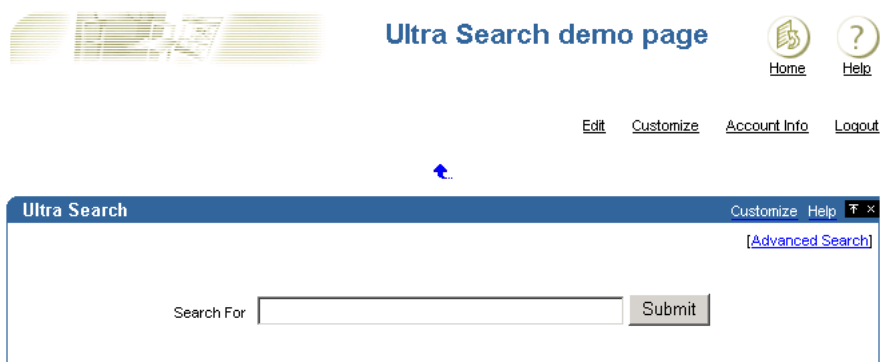
The Oracle Ultra Search portlet demonstrates how to write a search portlet for use in OracleAS Portal.

When the user issues a query in any of the query applications, a hit list containing query results is returned. The user can select a document to view from the hit list. A hit list can include HTML documents, files, database table content, archived e-mails, or other items as shown in Figure 8–13. The Oracle Ultra Search sample query applications also incorporate an E-mail browser for reading and browsing e-mails.

To use the Oracle Ultra Search portlet in OracleAS Portal, see Section 8.2.3, "Configuring Oracle Ultra Search Options in OracleAS Portal".

Figure 8–13   Example of Query Results in the Oracle Ultra Search Portlet

If you do not want to use the Oracle Ultra Search sample query applications, you can build your own query application by directly invoking the Oracle Ultra Search Java query API. Because the API is coded in Java, you can invoke the API methods from any Java-based application, such as from a Java servlet or a JavaServer page (as in the case of the provided sample query applications). For rendering e-mails that have been crawled and indexed, you can also directly invoke the Oracle Ultra Search Java Mail API methods.

> **See Also:**
>
> - *Oracle Ultra Search User's Guide*
>
> - README file located at *ORACLE_HOME*/ultrasearch/sample/sample_readme.htm

8.4.1.3 About the Oracle Ultra Search Administration Tool

The Oracle Ultra Search administration tool is a Web application that lets you manage Ultra Search instances. It allows user management operations on either database users or SSO users. Authenticated SSO users never see the Oracle Ultra Search login screen. Instead, they can immediately choose an Oracle Ultra Search instance.

From the Oracle Ultra Search administration tool you can:

- Define Oracle Ultra Search instances

- Manage administrative users

- Define data sources and assign them to data groups

- Configure and schedule the Oracle Ultra Search crawler

- Set query options

The Oracle Ultra Search administration tool and the Oracle Ultra Search sample query applications are part of the Oracle Ultra Search middle-tier components module. However, the Oracle Ultra Search administration tool is independent from the Oracle Ultra Search sample query applications. Therefore, they can be hosted on different machines to enhance security or scalability.

You can access the Oracle Ultra Search administration tool through OracleAS Portal. In the **Services** portlet, go to the **Ultra Search Administration** page. See Section 8.2.3.1, "Accessing the Oracle Ultra Search Administration Tool".

> **See Also:** *Oracle Ultra Search User's Guide*

8.4.2 Configuring the Oracle Application Server Infrastructure

The Oracle Ultra Search server tier is installed with the Oracle Application Server infrastructure. By default, the following activity occurs during this process:

- All Oracle Ultra Search server component files are copied into a directory named ultrasearch. This directory resides immediately under the *ORACLE_HOME* of the designated database installation.

- The database user WKSYS is created, with password wksys. You should change this password immediately for security purposes. All Oracle Ultra Search database objects are installed in this user's schema. After the infrastructure database is installed, all user schema passwords are randomized. To change the password, log on as user WKSYS (or WKPROXY), change the WKSYS (or WKPROXY)

schema password by following the link **Change Schema Password** from the **Oracle Enterprise Manager Infrastructure** page.

- Various PL/SQL scripts are run against the database as user `WKSYS`. These scripts install and create various database objects.

> **See Also:** Your installation guide for information on setting necessary environment variables.

8.4.3 Configuring the Database for Oracle Ultra Search

To configure the database for Oracle Ultra Search, follow the steps outlined in the "Post-Installation Information" chapter of the *Oracle Ultra Search User's Guide*.

8.4.4 Configuring the Oracle Ultra Search Middle-Tier Component

If you checked the OracleAS Portal option on the *Configuration Options* Oracle Installer screen, the OracleAS Portal Configuration Assistant automatically configures Oracle HTTP Server and Oracle Application Server Containers for J2EE with Ultra Search. If not, then you must manually perform the steps under "Configuring Oracle Ultra Search Middle Tier Component with Oracle HTTP Server and OC4J" in the *Oracle Ultra Search User's Guide* to configure your existing Web server.

In addition, you must edit `data-sources.xml` to add the `UltraSearchDS` datasource, and then unlock the `WK_TEST` schema and reset its password to `WK_TEST`. This is described in Section 8.4.4.1, "Editing the data-sources.xml File" subsequently.

You do not need to configure the `ultrasearch.properties` file, containing configuration information used by the Oracle Ultra Search middle-tier component. This is automatically configured by the Oracle installer. For more information, see Section 8.4.4.2, "Editing the ultrasearch.properties File".

8.4.4.1 Editing the data-sources.xml File

> **Caution:** Storing clear text passwords in `data-sources.xml` poses a security risk. Avoid this by using password indirection to specify the password. This lets you enter the password in `jazn-data.xml`, which automatically gets encrypted, and point to it from `data-sources.xml`. For more information, see *Oracle Application Server Containers for J2EE Services Guide*.

The Oracle Ultra Search query API uses the data source functionality of the J2EE container. Under directory *ORACLE_HOME*/j2ee/OC4J_Portal/config, edit the file data-sources.xml. Under tag <data-sources> add the following:

```
<data-source
   class="oracle.jdbc.pool.OracleConnectionCacheImpl"
   name="UltraSearchDS"
   location="jdbc/UltraSearchPooledDS"
   username="<username>"
   password="<password>"
   url="jdbc:oracle:thin:@<database_host>:<oracle_port>:<oracle_sid>"
/>
```

Where *username* and *password* are the Oracle Ultra Search instance owner's database user name and password, *database_host* is the host name of the back-end database machine, *oracle_port* is the port to the user's Oracle database, and *oracle_sid* is the SID of the user's Oracle database. In addition to user name, password, and JDBC URL, data-sources.xml also allows configuration of the connection cache size, as well as the cache scheme. The following tag specifies the minimum and maximum limits of the cache size, the inactivity time out interval, and the cache scheme.

```
<data-source
   class="oracle.jdbc.pool.OracleConnectionCacheImpl"
   name="UltraSearchDS"
   location="jdbc/UltraSearchPooledDS"
   username="wk_test"
   password="wk_test"
   url="jdbc:oracle:thin:@<database_host>:<oracle_port>:<oracle_sid>"
   min-connections="3"
   max-connections="30"
   inactivity-timeout="30">
 <property name="cacheScheme"  value="1"/>
</data-source>
```

For security purposes, WK_TEST is locked after the installation. The administrator should login to the database, and unlock the WK_TEST user account. To do this, run the following statement as the SYSTEM or SYS database user:

```
ALTER USER WK_TEST ACCOUNT UNLOCK;
```

After that, set the password to be WK_TEST. (The password expires after the installation.) If the password is changed to anything other than WK_TEST, then you must also update the cached schema password using the administration tool **Edit Instance** page after you change the password in the database.

> **Note:** The URL of the JDBC data source can be provided in the
> form of `jdbc:oracle:thin:@<hostname>:<port>:<sid>` or
> in the form of a TNS keyword-value syntax, such as
> `jdbc:oracle:thin:@(DESCRIPTION=(LOAD_`
> `BALANCE=yes)(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)`
> `(HOST=cls02a)(PORT=3999))(ADDRESS=(PROTOCOL=TCP)`
> `(HOST=cls02b)(PORT=3999)))(CONNECT_DATA= (SERVICE_`
> `NAME=acme.us.com)))`

There are three types of caching schemes:

- `DYNAMIC_SCHEME = 1`

- `FIXED_WAIT_SCHEME = 2`

- `FIXED_RETURN_NULL_SCHEME = 3`

> **See Also:** *Oracle Application Server Containers for J2EE Security
> Guide*

8.4.4.2 Editing the ultrasearch.properties File

The *ORACLE_*
HOME`/ultrasearch/webapp/config/ultrasearch.properties` file
contains configuration information used by Oracle Ultra Search middle-tier
component. You do not need to edit this file, as it is automatically configured by the
Oracle installer.

Here is an example of the `ultrasearch.properties` file:

```
connection.driver=oracle.jdbc.driver.OracleDriver
connection.url=jdbc:oracle:thin:@ldap://dlsn8888.cn.oracle.com:3060/iasdb,cn=ora
clecontext
oracle.net.encryption_client=REQUESTED
oracle.net.encryption_types_client=(RC4_56,DES56C,RC4_40,DES40C)
oracle.net.crypto_checksum_client=REQUESTED
oracle.net.crypto_checksum_types_client=(MD5)
oid.app_entity_cn=m16bi.sgtcnsn03.cn.oracle.com
domain=us.oracle.com
```

Where:

- `connection.driver` specifies the JDBC driver you are using.

- `connection.url` specifies the database to which the middle-tier connects. Ultra Search supports following formats:

    - `host:port:SID` (where host is the full host name of the Oracle base instance running Ultra Search, port is the listener port number for the Oracle Database instance, and SID is the Oracle Database instance ID)

    - `HA-aware string` (for example, TNS keyword-value syntax) Here is an example connection.url string:

    ```
    connection.url=jdbc:oracle.thin:@ultrasearch.us.oracle.com:1521:myInstance
    ```

- `oracle.net.encryption_client`, `oracle.net.encryption_types_client`, `oracle.net.crypto_checksum_client`, and `oracle.net.crypto_checksum_types_client` control the properties of the secure JDBC connection made to the database.

- `oid.app_entity_cn` specifies the Oracle Ultra Search middle-tier application entity name.

- `domain` specifies the common domain for the identity management machine and the Oracle Ultra Search middle-tier machine. This enables Oracle Delegated Administration Services (DAS) lists of values to work with Internet Explorer. For example, if the Oracle Ultra Search middle-tier is `us.company.com` and the identity management machine is `uk.company.com`, then the common domain is `company.com`. In this case, you would add the following line in `ultrasearch.properties`:

    ```
    domain=company.com
    ```

8.4.4.3 Restarting the OC4J_Portal Instance

Restart the OC4J_Portal instance using Oracle Enterprise Manager Application Server Control.

To restart the OC4J_Portal instance:

1. Access the Application Server Control. See Section 7.1.1, "Accessing the Application Server Control" for details.

    If there is more than one standalone application server instance, your start page for the Application Server Control is the Oracle Application Server **Farm** home page.

2. Click an Oracle Application Server instance to go to its home page.

3. Select **OC4J_Portal** in the **System Components** area.

4. Click **Restart**.

5. When prompted to confirm the restarting of OC4J_Portal, click **Yes**.

> **See Also:** *Oracle Application Server 10g Administrator's Guide*

8.4.4.4 Testing the Oracle Ultra Search Administration Tool

You can test your changes by attempting to log on to the Oracle Ultra Search administration tool at:

```
http://<hostname>.<domainname>:<port>/ultrasearch/admin/index.jsp
```

Where `hostname.domainname` is the full name of the host where you have just installed the Oracle Ultra Search middle-tier component, and `port` is the default Web server port.

During the installation of the Oracle Ultra Search server component, you should have created a new Oracle Ultra Search instance owner. Log on to the Oracle Ultra Search administration tool by entering the Oracle Ultra Search instance owner's database username and password.

If you log on to the Oracle Ultra Search administration tool successfully, then you have completed the Oracle Ultra Search administration tool configuration process.

You can also access the Oracle Ultra Search administrative interface through OracleAS Portal. In the **Services** portlet, go to the **Ultra Search Administration** page. See Section 8.2.3.1, "Accessing the Oracle Ultra Search Administration Tool".

8.4.4.5 Testing the Oracle Ultra Search Sample Query Applications

After you verify that the Oracle Ultra Search administration tool is working, you should be able to run the Oracle Ultra Search sample query applications. Refer to the section "Testing the Ultra Search Sample Query Applications" in the chapter "Installing and Configuring Ultra Search" of the *Oracle Ultra Search User's Guide* for more information on how to run the Oracle Ultra Search sample query applications.

8.4.5 Configuring Remote Crawler Hosts

The Oracle Ultra Search remote crawler functionality allows multiple crawlers to run in parallel on different hosts. All remote crawler hosts must share common resources, such as common directories and a common Oracle Ultra Search database.

> **See Also:** *Oracle Ultra Search User's Guide*

8.4.6 The Oracle Ultra Search Portlet Sample

Oracle Ultra Search provides a search portlet that can be embedded in OracleAS Portal pages. It is implemented as a JavaServer Page (JSP) application and called the Oracle Ultra Search Portlet Sample. The Oracle Ultra Search Portlet Sample is a Web application that complies with the OracleAS Portal portlet interface. By complying with the portlet interface, OracleAS Portal users can create pages and embed Oracle Ultra Search portlets within those pages.

More On Portal Center

You'll find additional information about Oracle Application Server Portal Developer Kit and the OracleAS Portal portlet interface, on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

The portlet sample implements a provider that contains exactly one portlet. The provider name is *Ultra Search Provider* and it belongs to the *Oracle Application Server Providers* provider group. The portlet contained within the Ultra Search provider is also called *Ultra Search*.

Note that Web providers are not registered with OracleAS Portal as part of the Oracle Application Server installation, as the provider must be up and running for registration to take place. This is not possible since the very last step performed during the installation is the starting of OC4J.

To register the Ultra Search provider, see Section 8.2.3.3, "Registering the Ultra Search provider with OracleAS Portal".

8.4.6.1 Searching Public Data

The Oracle Ultra Search portlet enables you to add Oracle Ultra Search functionality to portal pages. However, remember that Oracle Ultra Search does not support any security model for search end-users. This means that all data crawled and indexed by Oracle Ultra Search is accessible to all users of a particular Oracle Ultra Search instance. There is no way to specify that a particular portal user has access to a subset of search results returned by Oracle Ultra Search.

8.4.6.2 Connecting to an Oracle Ultra Search Instance

Oracle Ultra Search supports the creation of multiple Oracle Ultra Search instances. Each Oracle Ultra Search instance contains its own distinct index that can be queried against by the Oracle Ultra Search portlet. Each Oracle Ultra Search index

requires its own database schema and the Oracle Ultra Search portlet must be configured to query against a specific Oracle Ultra Search instance schema.

To do this, you must configure the file *ORACLE_ HOME*/j2ee/home/config/data-sources.xml as follows:

```
<data-source
              class="oracle.jdbc.pool.OracleConnectionCacheImpl"
              name="UltraSearchDS"
              location="jdbc/UltraSearchPooledDS"
              username="<ultrasearch_instance_schema>"
              password="<ultrasearch_instance_schema_password>"
              url="jdbc:oracle:thin:@<hostname>:<port>:<sid>"
/>
```

The parameters are listed and described in Table 8–4.

Table 8–4   Oracle Ultra Search Connection Parameters

Parameter	Description
ultrasearch_instance_ schema	name of the schema
ultrasearch_instance_ schema	password of the schema
hostname	Oracle Ultra Search database host name
port	Oracle Ultra Search database listener port
sid	Oracle Ultra Search database instance identifier

Note that the sample portlet shares the same data source entry as the Complete Sample Application.

8.4.6.3 Restrictions

OracleAS Portal users should only embed Oracle Ultra Search portlets that are hosted on the same OC4J instance as OracleAS Portal.

If OracleAS Portal is installed on host A, Oracle Ultra Search is installed on host A and the Oracle Ultra Search provider is also hosted as a Web application on host A.

It is possible that the Oracle Ultra Search provider running on host A is registered with a second OracleAS Portal instance running on host B. However, if the Oracle Ultra Search portlet hosted on A is embedded within pages created in Portal B, the

pop-up list-of-values will not work correctly. This is because of a security bug inherent in JavaScript.

Portal pages created within Portal A should *only* embed the Oracle Ultra Search portlet from the provider running on host A and not from host B or any other host.

8.4.6.4 Portlet Sample Files

The portlet sample files are located in the following file:

ORACLE_HOME/ultrasearch/sample.ear

When the application server first deploys `sample.ear`, the content of this file is expanded into the following directory:

ORACLE_HOME/ultrasearch/sample/query

You can view the source code using your preferred text editor.

> **See Also:** The file *ORACLE_HOME*/ultrasearch/sample/query/portlet/README.html for a complete list and description of all the files used by the Portlet Sample, as well as a full description of how the portlet sample works.

9

Tuning Performance in OracleAS Portal

This chapter discusses how you can tune the performance of your OracleAS Portal on the configuration, after you have set up the basic configuration of your Portal system.

This chapter contains the following list of options for tuning the performance of OracleAS Portal:

- Setting the Number of Server Processes

- Setting the Number of Idle Processes

- Setting the Number of PPE Fetchers

- Tuning the Oracle HTTP Server

- Generating Performance Reports

- Tuning File System Cache to Improve Caching Performance

> **See Also:**   *Oracle Application Server 10g Performance Guide*

9.1 Setting the Number of Server Processes

Oracle HTTP Server processes Web requests by distributing them to HTTP processes. Oracle HTTP Server can serve all types of requests originating in users' browsers, such as those for static files, Java servlets, or PL/SQL procedures.

MaxClients is an HTTP Server configuration directive that controls the maximum number of Web requests that the HTTP Server can handle at any given time. When the *MaxClients* value is exceeded, the HTTP Server refuses to handle any new requests until it handles the current load and the HTTP processes are freed. In fact, client browsers may be "locked out" if the number of allowable sessions has been exceeded by other browsers.

One way to think of the *MaxClients* directive is that it's a throttle that permits just the right flow of concurrent Web requests to your server. Set it too low, and your Web portal performance may suffer. Even though you may have the server and database resources to handle more traffic with quicker response intervals, Web requests can't get through because you haven't set enough processes in *MaxClients*.

Setting *MaxClients* too high unnecessarily consumes resources, because each HTTP process server consumes resources, such as CPU time, memory, and I/O. And it may result in poorer rather than better performance. Why? Keep in mind that the HTTP Server can handle all sorts of requests, including those for PL/SQL procedures. When the HTTP Server receives such a request, it hands it off to mod_plsql to communicate with the Portal database. For each server process that executes a mod_plsql request, there will be a need to cache a database connection. The value you set for *MaxClients*, therefore, sets the upper limit of database connections that mod_plsql can open.

Say you set *MaxClients* to the maximum number, 1024. At any given time, the HTTP Server is ready to handle 1024 simultaneous Web requests, including some that require database connections. Even if your server is large enough to deal with this, the database it is connected to may not be. And if the ratio of requests for PL/SQL procedures versus other types of requests suddenly becomes very high, you risk overloading your database.

> **Note:** For Windows platforms, you should look at tuning the Oracle HTTP Server parameter *ThreadsPerChild*

The key to good performance is determining the number of Web requests the servers in your configuration can process, as well as how much traffic your database can handle. So if your Portal configuration includes multiple middle-tier servers connected to a single database, the number of possible Web requests you can handle is probably going to be limited more by database capacity than the middle-tiers.

See Also:

- *Oracle HTTP Server Administrator's Guide*
- "Configuring the MaxClients Setting" in Section 9.4, "Tuning the Oracle HTTP Server".

9.2 Setting the Number of Idle Processes

MinSpareServers is a UNIX specific HTTP Server directive that sets the minimum number of idle sessions. An idle session is one that is not currently handling a Web request. If the number of idle sessions is fewer than the number specified in *MinSpareServers*, new processes are created at a maximum rate of 1 in every second.

You should consider tuning this parameter only on very busy sites. The default setting is 5. Setting this parameter to a large number is almost always a bad idea. A rule of thumb is to set *MinSpareServers* at a little over the average number of Web requests your Portal typically handles. Ideally, you can set it so user requests are filled all the time by open ports without having to open a new one, but this is possible if you have the database resources to support a lot of ports.

Unlike UNIX, Windows is a thread-based operating system where one process is started and then additional child processes are threaded as required. For Windows NT machines, the directive is called *MaxThreadsPerChild*. This is the number of concurrent requests the server will allow. Set this value according to the responsiveness of the server and the amount of system resources you want to allow the server to consume. *MaxThreadsPerChild* on Windows is equivalent to *MaxClients* on UNIX.

> **See Also:** *Oracle HTTP Server Administrator's Guide*

9.3 Setting the Number of PPE Fetchers

A request for a Portal page originates in the form of a URL sent from a user's browser to the HTTP server. If the request is for a Portal page, it is forwarded to the Parallel Page Engine (PPE). The PPE then asks each Web provider that owns a portlet on the page to execute the portlet and return content to the Portal page.

There are two options available to enable you to increase the concurrency of the PPE:

Option 1: Create a New OC4J Instance to Create Another Set of PPE Threads

Complete these steps to change the number of *OC4J_Portal* processes:

1. Access the Application Server Control.

    Typically the Application Server Control is located at
    `http://www.abc.com:1812`. Refer to Chapter 7, "Monitoring and

Administering OracleAS Portal" for more information about using the Application Server Control.

2. Click the link for the application server middle-tier where OracleAS Portal is installed.

3. Click the **OC4J_Portal** link.

4. Click the **Administration** link.

5. Click the **Server Properties** link.

6. In the Under the **Multiple VM Configuration** section, change the **Number of Processes** for the default_island as shown in Figure 9–1.

Figure 9–1 Multiple VM Configuration Section

Multiple VM Configuration

☑ **TIP** If OC4J is running, newly added islands and associated processes will be automatically started.

Islands

Island ID	Number of Processes
default_island	2

(Add Another Row)

7. Click **Apply**.

8. Navigate back to the **OC4J_Portal** home page.

9. Click **Restart**, to restart the OC4J_Portal instance.

Alternatively, you can edit the file opmn.xml manually, though the use of Application Server Control is the recommended approach.

The parameter to create multiple Oracle Application Server Containers for J2EE instances is called *numProcs* and is configured in the file *ORACLE_HOME*/opmn/conf/opmn.xml

The changed file would look something like this:

```
<oc4j instanceName="OC4J_Portal" gid="OC4J_Portal" numProcs="2">
  <config-file path="E:\Ora902\j2ee\OC4J_Portal\config\server.xml"/>
  <java-option value="-server -Xincgc -Xnoclassgc -Xmx100m "/>
  <oc4j-option value="-properties"/>
  <port ajp="3001-3100" rmi="3101-3200" jms="3201-3300"/>
  <environment>
    <prop name="PATH" value="E:/Ora902/bin"/>
```

```
      <prop name="DISPLAY" value="localhost:0"/>
    </environment>
  </oc4j>
```

For the configuration changes to take effect follow the following steps:

1. Run the following command:

```
ORACLE_HOME/dcm/bin/dcmctl updateconfig -ct opmn
```

2. Restart the Oracle Application Server middle-tier as follows:

```
ORACLE_HOME/opmn/bin/opmnctl stopall
ORACLE_HOME/opmn/bin/opmnctl startall
```

Option 2: Increase the Value of Default Number of Threads

The PPE uses a pool of *fetchers* to forward requests over the Internet to Web providers and wait for data to be returned. Once it is finished with the request, the fetcher is available to handle another new request.

The parameter to tune the number of PPE threads is called *poolSize* and is configured in the file *ORACLE_HOME*/j2ee/OC4J_ Portal/applications/portal/portal/WEB-INF/web.xml.

The default setting is 25. For most Web portals, you should never have to change pool size. But keep in mind that if pool size is too low, the user notices that pages take too long to draw at peak periods. If pool size is set too high, a possible resource drain may occur because too many concurrent URL requests can overwhelm the PPE.

The changed file would look something like this:

```
<web-app>
  <servlet>
    <servlet-name>page</servlet-name>
      <servlet-class>oracle.webdb.page.ParallelServlet</servlet-class>
    <init-param>
      <param-name>logpath</param-name>
      <param-value>./</param-value>
    </init-param>
    ...
    <init-param>
      <param-name>poolSize</param-name>
      <param-value>50</param-value>
    </init-param>
```

```
      ...
   </servlet>
...
```

For the configuration changes to take effect:

1. Run the following command:

   ORACLE_HOME/dcm/bin/dcmctl updateconfig

2. Restart the Oracle Application Server middle-tier.

> **Note:** Even for a site with high traffic, the *PoolSize* parameter should not be set to beyond the range of 50-125. If there is a need to set the value higher than this, then consider adding more OC4J instances as mentioned in "Option 1: Create a New OC4J Instance to Create Another Set of PPE Threads".

9.4 Tuning the Oracle HTTP Server

However you choose to configure the Oracle HTTP Server listener, you can optimize performance by setting an approximate number of simultaneous requests that can be handled by the Oracle HTTP Server listener.

On UNIX, in particular, since the Oracle HTTP Server is process-based, each process must open a database connection for each DAD that has requested it. As a result, the number of requests can be quite high, which may result in clients being "locked out" if the number of sessions allowable has been exceeded. However, setting too high of a value unnecessarily consumes resources.

The scenario in the preceding text is described here:

1. For every service request from an OracleAS Portal DAD, there is one network connection and two sessions (the two sessions use the same physical connection).

   The first session is for *portal* and the second is for *portal_public*.

2. If you are logging in to OracleAS Portal, then you have to open a connection for the OracleAS Single Sign-On DAD (SSO DAD). This consumes one network connection and two sessions.

   In this case, the first session is for *orasso* and the second session is for *orasso_public*.

3. The Oracle HTTP Server configuration setting that determines the maximum number of requests being handled simultaneously is named *MaxClients*. It defaults to 150.

   If each user were logging in and working in OracleAS Portal, then scenario (1) and (2) earlier would result in four sessions for every process. The total number of sessions for such a scenario is calculated as follows:

   ```
   150 * 4 = 600
   ```

   600 sessions and approximately 300 database connections (2 sessions for every connection)

Configuring the MaxClients Setting

Since login frequency is generally lower than OracleAS Portal access frequency, it makes sense to configure the OracleAS Single Sign-On on a separate Oracle HTTP Server listener. The objective is to tune down the *MaxClients* setting to a value that is reasonable, without affecting the needs of the portal system.

OracleAS Portal makes extensive use of mod_plsql, which maintains a pool of connections to the database. The *MaxClients* parameter tunes the number of processes, which directly relates to the number of database connections pooled by mod_plsql.

> **See Also:** *Oracle Application Server 10g Performance Guide*

Follow these steps to configure the `MaxClients` setting:

1. For the OracleAS Single Sign-On's listener, once you've determined the approximate value to set for the `MaxClients` parameter, edit this accordingly in the configuration file, `httpd.conf`, which is located in:

   ```
   ORACLE_HOME/Apache/Apache/conf/
   ```

   Tune down the `MaxClients` setting to control the number of requests that Oracle HTTP Server services on the Oracle HTTP Server listener. This controls the maximum number of sessions that can be established.

2. For the OracleAS Portal listener, you can separately tune the *MaxClients* parameter according to the needs of the OracleAS Single Sign-On and the needs of OracleAS Portal, without creating a conflict. This parameter directly corresponds to the number of sessions established and to the maximum workload that the Oracle HTTP Server listener can handle on the Portal listener.

The following example illustrates the `MaxClients` section in the `httpd.conf` file:

```
# Limit on total number of servers running, i.e., limit on the number
# of clients who can simultaneously connect --- if this limit is ever
# reached, clients are LOCKED OUT, so it should NOT BE SET TOO LOW.
# It is intended mainly as a brake to keep a runaway server from taking
# the system with it as it spirals down...
#
MaxClients 150
```

> **Notes:**
>
> - If you tune the OracleAS Single Sign-On and the OracleAS Portal separately, each will have a separate listener. The OracleAS Portal will control the resources (sessions) on the portal database and the OracleAS Single Sign-On will control the resources on the OracleAS Single Sign-On database.
>
> - The number of sessions and connections that the database permits is limited by the value set in the Oracle9*i* Database Server's `init.ora` file. Refer to the Oracle9*i* Database Server documentation library for more information.

9.5 Generating Performance Reports

This release includes a set of SQL scripts that can generate performance reports for OracleAS Portal. Other than using these scripts, there is no way to obtain performance-reporting information. These scripts allow a portal administrator to load OracleAS Portal log files into a database table and create reports based on that information. The scripts are located in the following directory:

ORACLE_HOME/portal/admin/plsql/perf

The file `README.html` in the `scripts` subdirectory explains how the scripts can be used to monitor OracleAS Portal performance.

The statistics collected indicate, among other things, how long overall requests take to complete, how much of that time was spent in the user's procedure, which user made the request, whether a database connection was obtained from the connection pool, and what type of caching was used. The performance scripts also enable you

to extract information similar to that, which was available in earlier releases of OracleAS Portal. Some of the performance reports that you can generate include:

- Unique logins for each day, or hour

- Page views for each day, or hour

- Top ten pages and portlets and their response time

- Response times

- Peak login time each day

- Logins for each day

- Portlets execution time

- Slowest portlet

- Total hits for each day

- Most and least popular portlets

- Unique users logged in each day

- Page hits for each day

- Portlet hits for each day

- Request breakdown by IP address and hostname

9.6 Tuning File System Cache to Improve Caching Performance

Tuning the File system cache can increase caching performance. Two ways of tuning the file system cache are:

- Configuring File System Cache to Reside on a Faster File System.

- Moving Session Cache Directory to More Performant File System.

More information on how to do this can be found in the chapter titled "Optimizing PL/SQL Performance" of the *Oracle Application Server 10g Performance Guide*.

10

Exporting and Importing Content

OracleAS Portal provides a set of export/import utilities that enable you to move content between portal installations. This chapter provides a summary of recommendations and best practices developed for export/import functionality as provided in OracleAS Portal version 10*g* (9.0.4). This chapter contains the following sections:

- How Does Export and Import Work?
- What are the Most Common Use Cases?
- What Do I Need to Check Before I Begin?
- How Does Export Work?
- How Does Import Work?
- How Do I Manage My Transport Sets?
- How Do Objects Behave After Migration?
- What Are the Recommended Best Practices?

10.1 How Does Export and Import Work?

The export and import process consists of the following steps:

- Create *transport sets* and extract the content to transport tables. Transport sets contain the portal objects that you are planning to export to your target portal environment. This information is displayed in a *manifest*. The manifest is simply the list of objects in a transport set, used to provide a granular level of control over the export.

- Move the transport sets from one system (source) to another (target) using Portal export/import command-line scripts to generate a transport set dump file.

- Transfer the script and dump file to the target system using FTP or other file transfer utilities.

- Invoke the command line script to import the dump file to the transport tables on your target system.

- Import the objects from the transport tables to the target portal repository using the Transport Set Manager portlet.

10.2 What are the Most Common Use Cases?

OracleAS Portal supports the ability to copy or update page groups and portal content between your source and target destination portal instances. This section introduces some of the most common uses.

10.2.1 Case 1: Importing/Exporting Between Development to Production Instances

This case illustrates the steps involved in copying or updating portal page groups and portlets between a development instance and a production instance of OracleAS Portal.

> **Note:** User customizations are not exported, therefore **any customizations on a page or portlet on the source are not exported or imported.**

Scenario 1. Exporting your pages and content to a target portal system. The first export to your target system should migrate the entire page group. The subsequent steps provide an overview of the process:

1. Develop page groups, applications and content on the source system.

2. Identify pages, applications and content to export, then create transport sets accordingly and export to the target system.

3. Import the transport sets on the target system, into your portal repository.

Scenario 2. Updating content on your target instance. OracleAS Portal supports the updating of item, region-level content on your target system ONLY under the following circumstance:

- Export/import of ALL changes from the source to the target instance. All page structure, content and user preferences on your target system are replaced with the content from your source system. The first export to your target system should migrate the entire page group from the source portal to the target portal instance.

Refer to Section 10.8, "What Are the Recommended Best Practices?", for a detailed overview of the recommended practices.

> **Note:** The current release does not support editing the same content on both source and target portal instances.

10.2.2 Case 2: Deploying Identical Content Across Multiple Portal Instances

This case illustrates the process of deploying the same set of OracleAS Portal objects across multiple portal instances. Oracle EXP and IMP utilities can be useful when deploying identical content across multiple OracleAS Portal instances. In this case, the OracleAS Portal objects (portlets, page groups, and so on), can be created in one instance, and propagated to multiple instances using the Oracle EXP and IMP utilities. See Section 10.8.8, "Migrating Your Portal Across Databases" for details.

10.3 What Do I Need to Check Before I Begin?

Before proceeding with the export/import process, make sure you have the following information:

- System Requirements
- Privileges for Exporting and Importing Your Content
- Portal instance information:
  - Portal schema name
  - Portal schema password
  - Portal connect string information
  - Portal user name
  - Portal user password
  - Company name (used only for hosted portal installations) - leave blank in most cases.

> **Note:** The Portal schema password is a randomized password created on install. You may want to update the password to something more meaningful.

10.3.1 System Requirements

Before exporting and importing, ensure that your system meets the minimum system requirements, as described in this section.

> **Note:** Export and import functions only within the same release of OracleAS Portal, for example, 9.0.4 to 9.0.4 You cannot export and import between two different releases, such as, 3.0.9 to 9.0.4.

- **Using Different Releases and Versions of Export**. Whenever you are moving data between different releases of the Oracle database server, the following rules apply:

  - The Oracle IMP utility and the database to which data is being imported (the target database) must be the same version.

  - The version of the Oracle EXP utility must be equal to the lowest version of the source or target database.

  > **Note:** Oracle EXP and IMP are the export and import utilities used to dump and restore data in an Oracle specific format for backup and transfer of user data.

The choice of whether to use the database Oracle home or the middle-tier Oracle home depends on the version of the database used for the source and target portal installations. The 9.0.4 middle-tier uses a 9.0.1.4 Oracle home.

Based on the recommendations given earlier, the following conditions apply when a 9.0.4 portal and 9.0.4 middle-tier is involved:

- Always use the middle-tier Oracle home for export. 9.0.1.4 is the lowest version of the database supported for a 9.0.4 portal installation.

- Always use the target database Oracle home for import. The version of the import utility and the target database must be the same.

> **Note:** If you have configured a 9.0.2 portal (9.0.2.2 or 9.0.2.3) to use a 9.0.4 middle-tier then the rules described in the Using Different Releases and Versions of Export must be followed properly.

For example, to create an export file for an import into a higher release database, use a version of the Oracle EXP utility that is equal to the source database. To create an export file for an import into a lower release database, use a version of the Oracle EXP utility that is equal to the version of the target database.

> **Note:** It is strongly recommended that you use the same database version for the source and target portal installations.

- The Oracle EXP utility always exports user data, including Unicode data, in the character sets of the export server. The character sets are specified at database creation.

  The Oracle IMP utility automatically converts the data to the character sets of the import server.

  Some 8-bit characters can be lost (that is, converted to 7-bit equivalents) when you import an 8-bit character set export file. This occurs if the client system has a native 7-bit character set or if the NLS_LANG operating system environment variable is set to a 7-bit character set. Most often, you notice that accented characters lose their accent mark.

  Both the EXP and IMP utilities provide indications of any required character set conversion before exporting or importing the data.

  > **Note:** When the character set width differs between the export client and the export server, truncation of data can occur if conversion causes expansion of data. If truncation occurs, the export displays a warning message.

- Understand your source and target portal instances.

  - **Do you have command line access to appropriate directories on the source and target machines?** You must have command line access to run

the shell or command utilities generated by the export-import process. The command line utilities in turn access the Oracle EXP and IMP utilities, as well as the Portal instance.

- **Is your database configured to allow the execution of background jobs?** Each export or import process sets up a background process. Therefore, verify that the `job_queue_processes` database parameter is set appropriately.

  To check the value of the `job_queue_processes` parameter, perform the following query from SQL*Plus:

  ```
  %select name, value from v$parameter where name='job_queue_processes'
  ```

  The value for `job_queue_processes` should be at least 2 to allow the execution of background jobs.

  An alternative way of checking the `job_queue_processes` parameter is to examine the `init.ora` file in your database's *ORACLE_HOME*.

- Plan to perform the export and import process during non-business hours and to disable access to OracleAS Portal during the process. One way to disable access to the portal temporarily for all other users, is to configure your listener for a different port number during the duration of the export and revert it back to the original port when your export is complete.

10.3.2 Privileges for Exporting and Importing Your Content

This section describes the privileges required to successfully export and import your content.

10.3.2.1 Privileges for Exporting Your Content

To allow for secured control over the export of shared objects (objects in the Shared page group), there are now two privileges defined at the infrastructure level.

- **Any Transport Set - Manage** enables you to perform export/import of portal objects, including 'shared' objects. This privilege is granted to the DBA group by default during the portal installation process.

- A user with the **Any Transport Set - Execute** privilege can export/import portal objects, excluding shared objects. This privilege is granted to the PORTAL_ADMINISTRATORS group by default during portal installation process.

Table 10–1 provides a description of export user privileges.

Table 10–1   Export User Privileges

User Privileges	Export non - shared objects?	Export shared objects?
Any Transport Set - Manage	Yes	Yes
Any Transport Set - Execute	Yes	No
Any Transport Set - None	No	No

10.3.2.2 Privileges for Importing Your Content

In addition to the **Any Transport Set - Manage** privilege, you must also have Manage privilege on objects of a given type to successfully import your content.

For example, a page group containing Web providers require you to have **Manage All** privileges on All Providers and All Page Groups in order to import that page group. Table 10–2 provides a description of each object type and the required privilege level.

> **Note:**   The ORCLADMIN and Portal users are granted **Manage All** on all page groups at the time of install or upgrade. Members of the DBA group are also granted **Manage All** on all page groups by default.

Table 10–2   Import User Privileges

Object Type	Privileges
All Page Groups	**Manage All**: This privilege is required, along with the 'All Providers Manage' privilege to import page groups and shared objects.
All Providers	**Manage**: This privilege is required for the import of Page Groups and Portal DB Provider objects.
All Portal DB Providers	**Manage**: This privilege is required for the import of Portal DB Provider objects.
All Shared Components	**Manage**: This privilege is required for the import of shared components if the Portal DB Provider objects reference the shared components.

> **Note:** If you import a page which is based on a style belonging to the shared objects group and do not have the necessary privileges to import shared objects, then the style of the page is reset to "Main Style" by default (provided the **Ignore Warnings** option was selected in the Transport Set Manager).

10.4 How Does Export Work?

This section describes the export process and the steps required to successfully move content from the source portal system, including:

- Creating Transport Sets
- Exporting Your Data

10.4.1 Creating Transport Sets

Once the system requirements are verified, your goal is to create a transport set. The subsequent diagram illustrates the process.

> **Note:** Limit any possible conflict issues by making one person responsible for maintaining a transport set.

Figure 10–1   Export Process
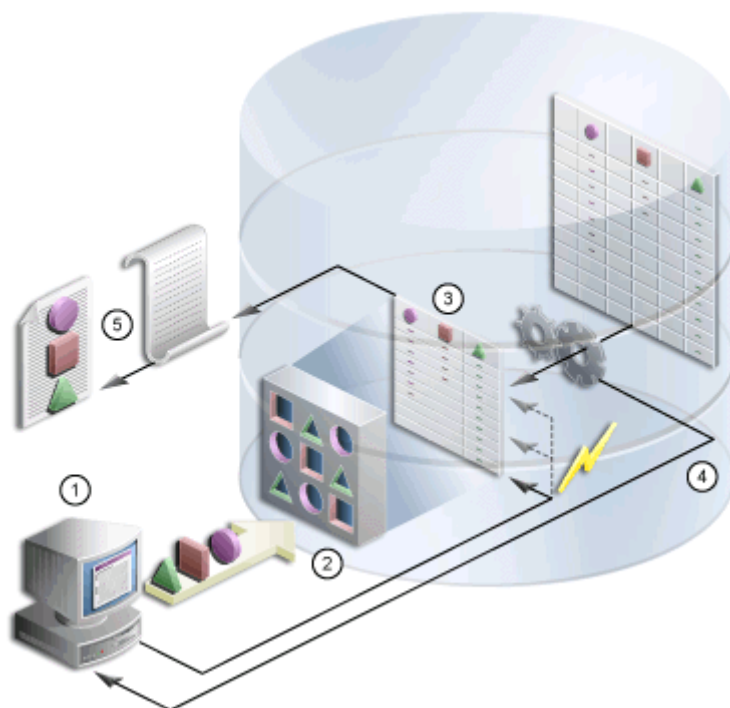
1. You select the objects to be exported from the Navigator or Bulk Actions (enables you to add multiple pages at once to the export transport set). The Transport Set Manager is automatically displayed.

2. You choose a name and select the export options for the Transport Set, click **Export Now** from the Transport Set Manager to initiate the export.

3. The procedure extracts the data and populates the transport tables.

4. You generate a migration script and log information from the Transport Set Manager.

5. You execute the script to generate a dump file.

The Export/Import Dependency Manager ensures that all the dependencies of objects in the transport set are correctly extracted. Specifically, the Dependency Manager classifies each object as explicitly selected, referenced, external or child,

based on how the object is related to the objects being explicitly exported. The information is displayed in the manifest, see Figure 10–2.

- **Explicitly Selected Objects**. Objects, that were explicitly selected, from the Navigator or Bulk Actions for export. When a page contains a portlet from an external provider, the manifest displays the external provider as a dependency.

- **Referenced Objects**. Objects that are directly or indirectly referenced by the explicitly selected objects. For example, a style used by a page is a referenced object, when it belongs to the same page group.

- **External Objects**. External objects ensure that the explicitly selected objects perform on the target portal. For example, external-providers and database schemas could be considered external objects. Generally, shared objects and components are external objects unless explicitly selected.

- **Child Objects**. Objects that are part of a hierarchy. For example, sub-pages, sub categories and sub-perspectives are child objects of a page, category and perspective.

> **Note:** When a referenced object contains child objects then the child objects are always imported in reuse mode. You should therefore explicitly select the referenced object and include it in the transport set. This will enable you to set the import mode to **Replace on Import**. Before importing the page group in reuse mode, note the page group properties and after import manually update any changes to reflect the old properties.

Working with Import Modes

The manifest provides a granular level of control over the import mode. The manifest is simply the list of objects in a transport set. There are two modes available during import:

- **Replace on Import**. If the object exists on the target, it is replaced. If it doesn't exist, then it is created. When this mode is not selected and if the object exists, the object on the target portal is retained as is. However, if the object doesn't exist on the target, then it is created.

- **Reuse on Import**. If the object does not exist on the target, it is created. If it already exists, it remains as is.

The following table describes the object classification and the default modes.

Table 10–3   Default Modes

Object Classification	Default Import Mode
Explicitly Selected Objects	Replace on Import
Referenced Objects	Reuse
Child Objects	Replace on Import
External Objects	Reuse

Figure 10–2   Transport Set Manifest
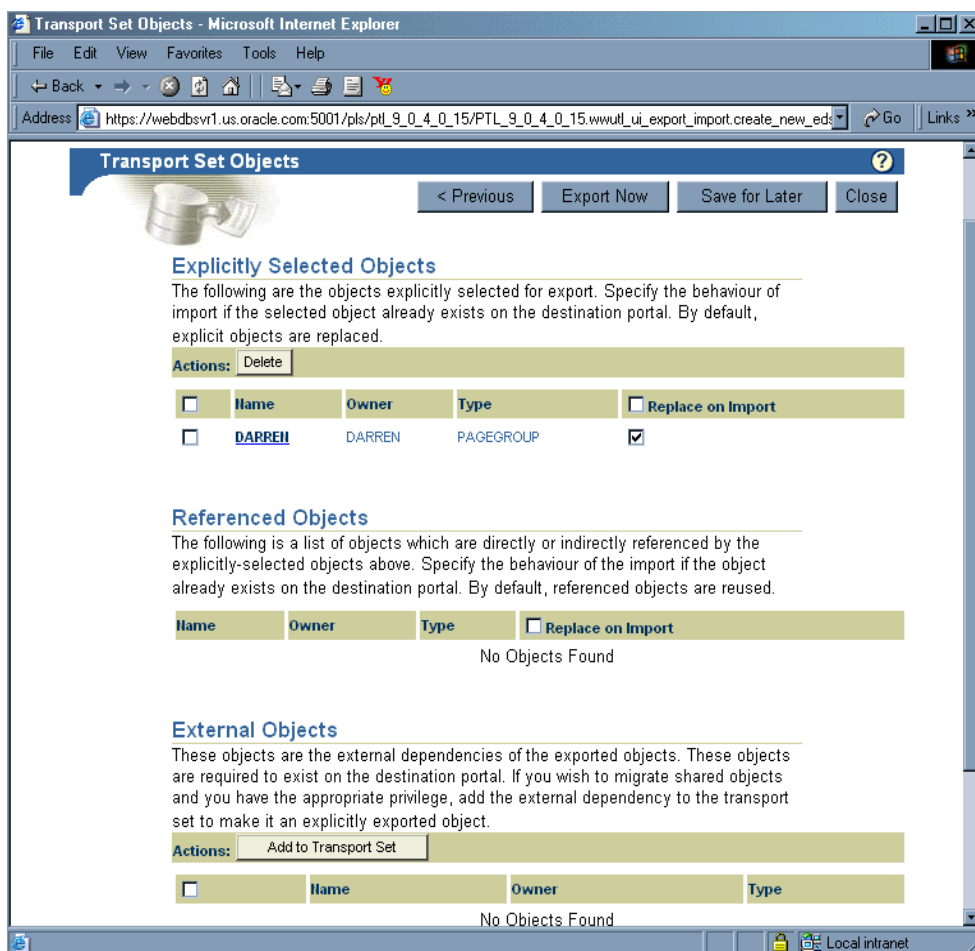
Clicking the name of an object, for example, an explicitly selected object, displays a detailed read-only screen of child, referenced and external objects, as shown in Figure 10–3.

Figure 10–3   Manifest Detailed Screen
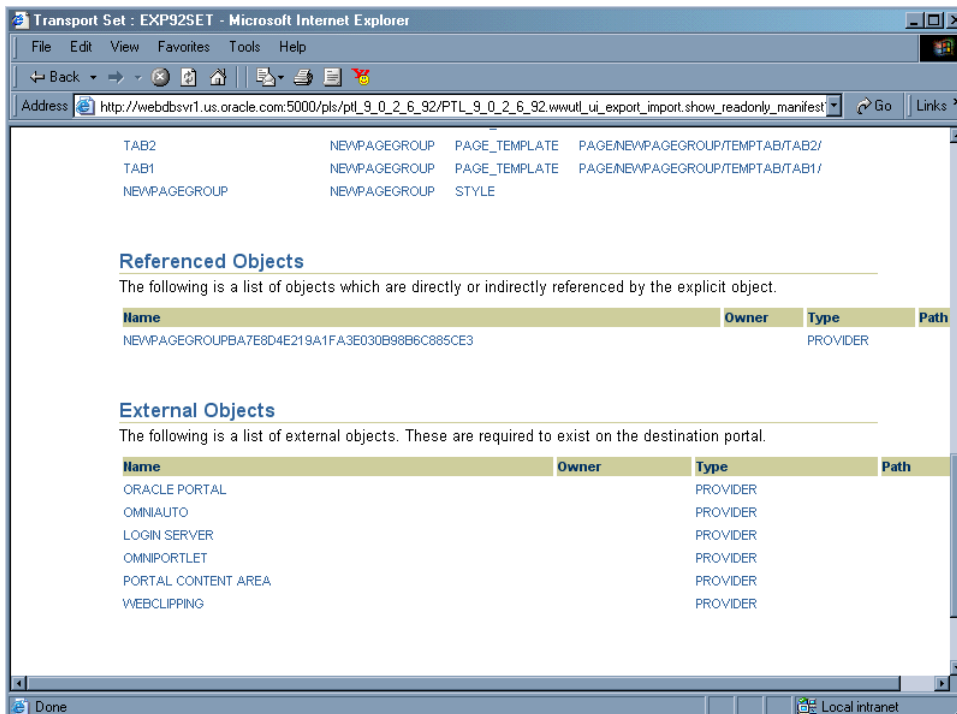
> **Note:**   To simplify the manifest, seeded types are not extracted. If you want to extract them, create custom types in the Shared Objects page group based on the existing seeded types. The Dependency Manager includes these in the manifest.

10.4.2 Exporting Your Data

Review the Section 10.7, "How Do Objects Behave After Migration?" before migrating your portal content from a source to a target instance.

> **Note:** Portlet repository information (security, organization, and so on) related to the portlet is not migrated during the export/import process.

To create a transport set for export:

1. Select the objects for export (from the Navigator, or search results > Bulk Actions for page groups). See Figure 10–4

> **Note:** Be sure to export portlets (Portal Forms, Portal Reports, Charts, Dynamic Pages), before exporting portal pages/page groups that reference them.

Figure 10–4   Portal Navigator

2. Click the **Export** link to display the Transport Set Manager, as shown in Figure 10–5. Make the transport set name as descriptive as possible and avoid using any special characters at the start of the name. For example, *My Company Transport Set 18-JAN-2003*.

Figure 10–5   Transport Set Manager
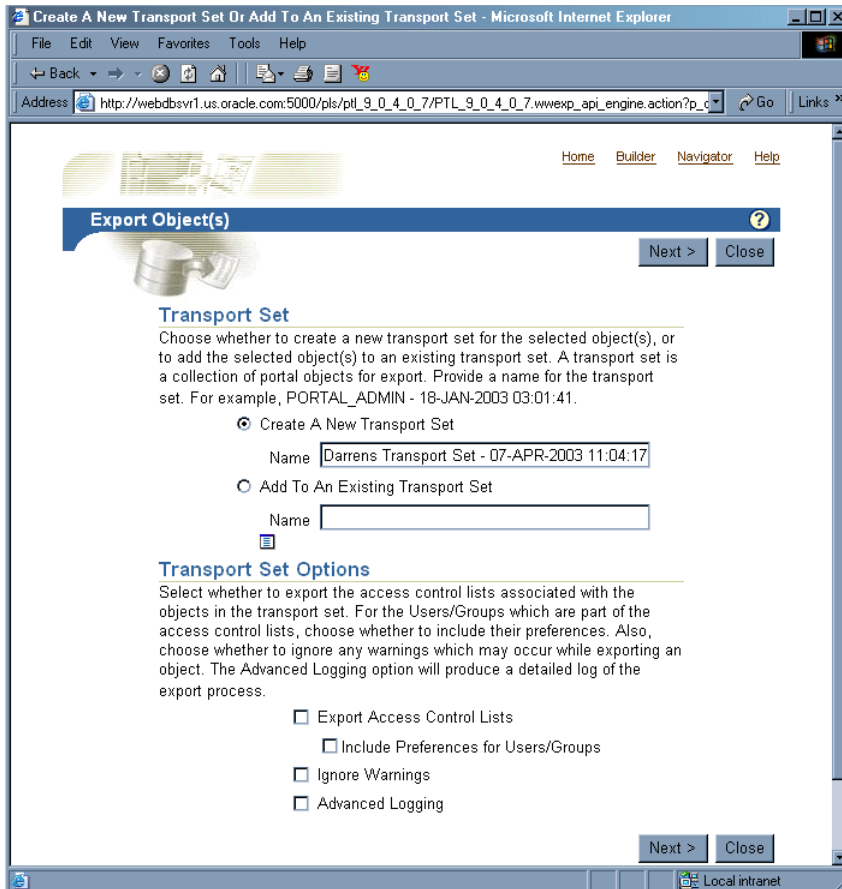
3. Check the appropriate boxes under the Transport Set Options:

    ■   **Access Control Lists**. Includes Access Control Lists associated with the
        objects in the transport set.

    ■   **Include Preferences for Users/Groups**. Includes the users and groups
        global privileges when object **Access Control Lists** are selected for export.

    ■   **Ignore Warnings.** Allows the export to proceed when a warning is
        encountered.

- **Advanced Logging.** Provides a detailed log of the export process, including debug messages.

4. Choose the import modes, delete any explicitly selected objects and promote (make explicit) any external objects. Making an external object explicit enables you to add a new object to a transport set 'in-place' instead of going back to the portal Navigator and adding it. External objects are not exported or imported by default until they are promoted as explicitly selected objects. See Figure 10–6

5. Select either, **Export Now** if you are finished, or **Save for Later** if you want to add more objects. See Section 10.6, "How Do I Manage My Transport Sets?" for details on how to edit and browse the transport sets currently on the system.

> **Note:** When you select some of the transport set options and choose **Save for Later**, the next time you add an object to the transport set all of the previously selected options are reset. Therefore, you should select the options each time until you finalize the transport set.

Figure 10–6   Transport Set Manager Objects
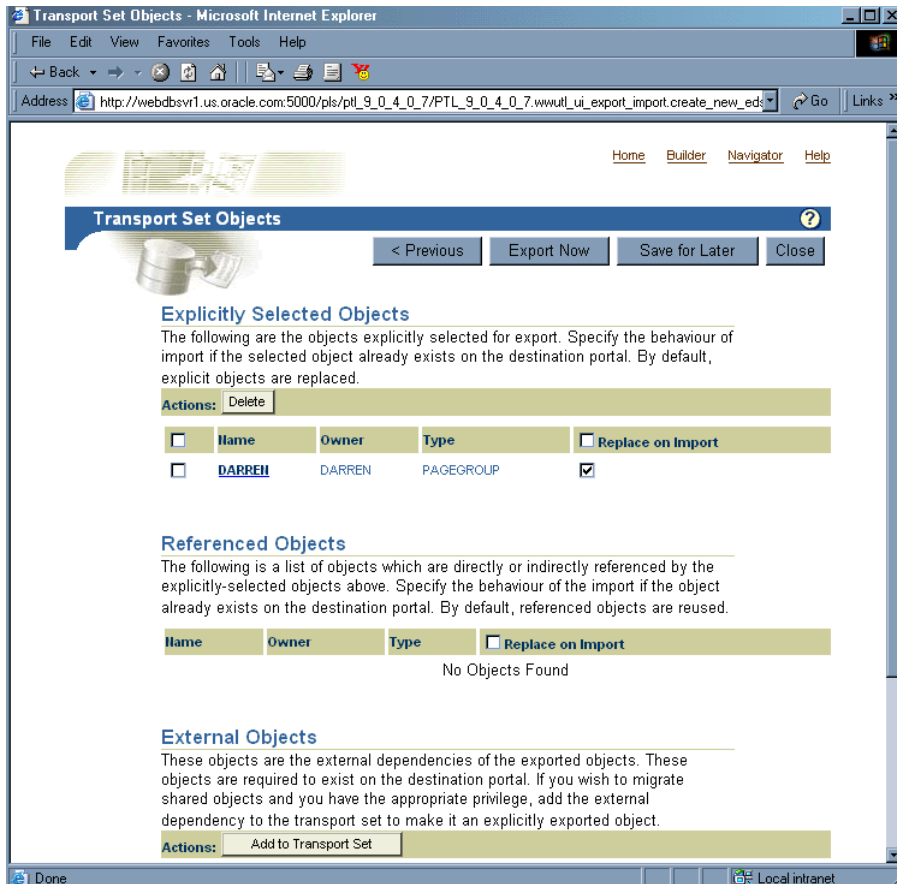
6. Click **Export Now** to finalize the transport set. The objects marked for export are copied to the transport tables for migration. These operations happen in the background.

7. Check the log in your transport set manager for any errors by clicking the **View Log Of Actions** link.

Figure 10–7   Transport Set Log Output
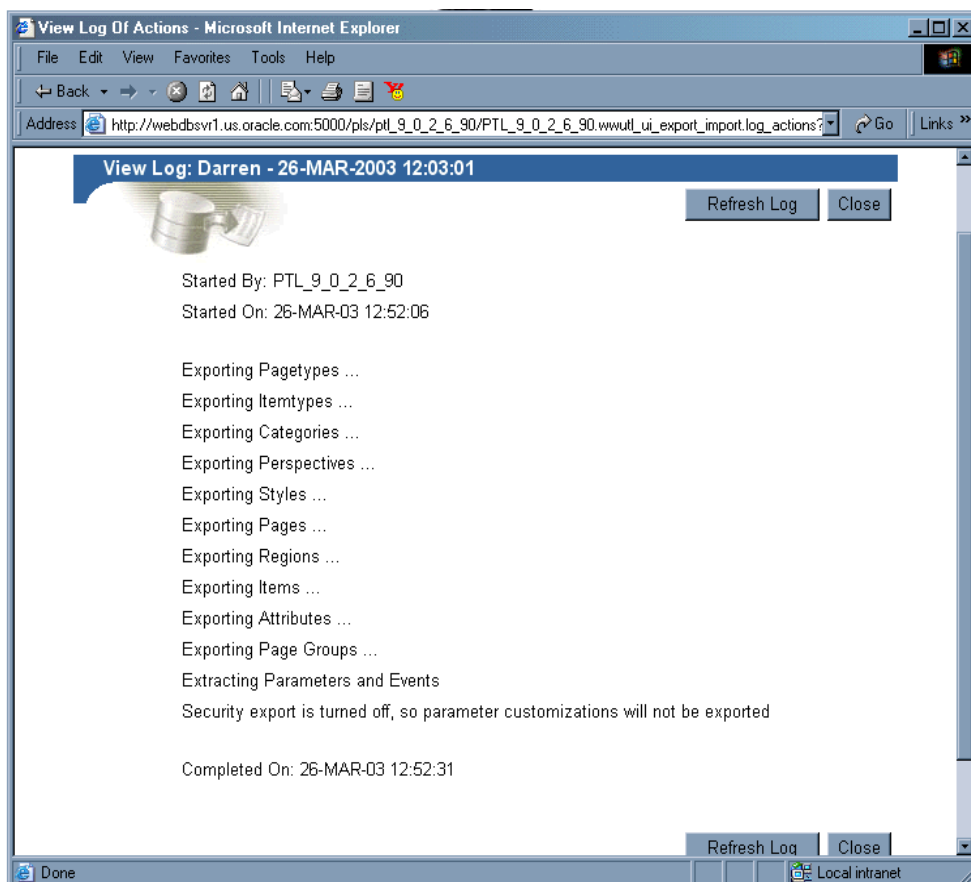
8. Choose an appropriate export script based on your operating system. See Figure 10–8.

Figure 10–8   Portal Migration Scripts

For Netscape users:

a. Click the selected script, then 'Save Target As'.

b. Change the name and remember to include the correct filename extension, **.csh** for UNIX or **.cmd** for NT, for example, `MyScript.csh`.

c. Save the file to the directory on your file system where you will want to run the export script (generally this directory should be where your export portal resides).

> **Note:** UNIX users should save the file to a local directory and move the script to the middle-tier machine where the IMP utilities reside to create the dump file.

For Internet Explorer users:

a. Right-click the selected script, then 'Save Target As'.

b. Change the name and remember to include the correct filename extension, **.csh** for UNIX or **.cmd** for NT, for example, MyScript.csh.

c. Save the file to the directory on your file system where you will want to run the export script (generally this directory should be where your export portal resides).

> **Note:** This location must have access to the database. On some systems, the downloaded UNIX script requires you to set the execute permissions correctly before running it.

Running Your Script to Create an Export Dump File

The next steps in the export process are to create a transport set dump file using the script you just created in the last section, and then transfer your export data to your target system.

To create a dump file:

■ The parameters in bold are only applicable for export and are mandatory. The subsequent example assumes that the name of the script is MyScript.csh.

```
%MyScript.csh
Usage: MyScript.csh <-mode export_or_import> <-s portal_schema>
<-p portal_password> <-pu portal_username> <-pp portal_userpassword>
<-company company_name> <-c connect_string> <-d dump_file_name(s)>
<-automatic_merge>
```

> **Note:** The value for the **company_name** parameter is the company name you see in the login page when working in a hosted portal. When working in a non-hosted portal, the value for the parameter should be 'none'. If you're running the script in interactive mode no value should be passed.

The following table provides a description of the parameters you can use in this process.

Table 10–4   Parameter Descriptions

Parameters	Description
-mode	Mode for invoking the Export Import Command Line Utility
	EXPORT mode: Exports content to dump files using Oracle EXP utility
	IMPORT mode: Imports content from dump files using Oracle IMP utility
-s portal_schema	Oracle database account for portal
-p portal_password	Oracle database password for portal
-pu portal_username	Lightweight username for logging into portal
-pp portal_userpassword	Lightweight user password for logging into portal
-company company_name	Company name (for example, ORACLE)
-c connect_string	TNS Connection Information to remote database
-d dump_file_name(s)	Name(s) of files for Oracle export or import utilities to write to or read from. If filename(s) are used, they must be separated by commas and enclosed in double-quotes. For example: "FILE1.DMP,FILE2.DMP"
-automatic_merge	Automatically import contents of dump file

To transfer your export data:

1. Run the script using -mode export as the option.

```
%MyScript.csh -mode export
```

This prompts you for information such as schema name (source), password, dump file name(s), and so on. It also creates a dump file upon completion.

2. Finally, using FTP, transfer your dump file and export/import script to the machine where your target OracleAS Portal repository resides.

10.4.3 Exporting Large Page Groups

You can use the `opeasst.csh` (Oracle Portal Export Assistant) script to export large page groups, which may time out in the browser while calculating the page

group dependencies. These timeout issues are due to the Dependency Manager and the pre-check routines that are run as foreground processes. The actual data extraction and the data merge are performed in the background.

The script can be found in the `/portal/admin/plsql/wwu` directory. An example of the script follows:

```
%opeasst.csh
Usage: opeasst.csh <-s portal_schema> <-p portal_password> <-c connect_string>
<-ts transportset_name> <-pgrps pgrp_names> <[-export_acls [-include_prefs]]>
<[-ignore_warnings]> <[-advanced_logging]>
```

The following table provides a description of the parameters used in this process.

Table 10–5   OPEASST.CSH Parameter Descriptions

Parameters	Description
`-s portal_schema`	Oracle database account for portal.
`-p portal_password`	Oracle database password for portal
`-c connect_string`	TNS Connection Information for the source database.
`-ts transportset_name`	Name of the transport set to be created
`-pgrps pgrp_names`	Comma-delimited list of Page groups for export
`-export_acls`	Export object level privileges
`-include_prefs`	Include preferences for users/groups
`-ignore_warnings`	Ignore any trivial warnings/errors generated during the data extraction process
`-advanced_logging`	Generate a very detailed log for the echo data extraction process

Perform the export from the command line, then:

1. Check the log in your transport set manager for any errors by clicking the **Status** link. See Section 10.6, "How Do I Manage My Transport Sets?" for details on how to edit and browse the transport sets currently on the system.

2. When the export is complete browse your transport sets and select the appropriate script for your operating system. See Section 10.4.2, "Exporting Your Data" for details.

3. Run the script using -mode export as the option.

```
%MyScript.csh -mode export
```

This prompts you for information such as schema name (source), password, dump file name(s), and so on. It also creates a dump file upon completion.

4. Finally, using FTP, transfer your dump file and export/import script to the machine where your target OracleAS Portal repository resides.

5. To import your objects, the contents of the transport set dump file must first be imported to the transport set tables on the target system. See Section 10.5.2, "Importing Your Data".

The following features and limitations currently exist:

- The script supports only exporting page groups.

- Multiple page groups can be exported at once using comma-delimited values.

- Exporting '**Account Control Lists**', '**Include Preferences for User/Groups**', '**Ignore Warnings**' and '**Advanced Logging**' are all supported.

- There is no import mode option available, that is, replace on import or reuse.

- Exporting database providers is not supported.

- If the Dependency Manager results in some external objects for the page group being exported, all the external objects are automatically promoted by the script without any user intervention. Those objects that are promotable are recursively promoted to become part of the transport set until there are no remaining external objects in the transport set.

- The script name cannot be changed.

> **Note:** Remember to set the infrastructure Oracle home when trying to connect to the database to run the `opeasst.csh` script.

10.5 How Does Import Work?

This section describes the import process and the steps required to successfully move content to the target portal system, including:

- Running Your Script on Your Target System

- Importing Your Data

10.5.1 Running Your Script on Your Target System

To import your objects, the contents of the transport set dump file must first be imported to the transport set tables on the target system. This is done by calling the same script (used in the export) with -mode set to import. The parameters in bold are only applicable for import and are mandatory.

```
%MyScript.csh
Usage: MyScript.csh <-mode export_or_import> <-s portal_schema>
<-p portal_password> <-pu portal_username> <-pp portal_userpassword>
<-company company_name> <-c connect_string> <-d dump_file_name(s)>
<-automatic_merge>
```

To perform the entire import from the command line, which initiates a background process, you must include the portal username and password parameters. This is required to validate your role on the target portal instance.

> **Note:** The value for the **company_name** parameter is the company name you see in the login page when working in a hosted portal. When working in a non hosted portal the value for the parameter should be 'none'. If you are running the script in interactive mode, no value should be passed.

The contents of the dump files are imported and the transport set is made available from the UI for merging on the target portal system. Figure 10–9 illustrates how the import process works.

Figure 10–9 Import Process
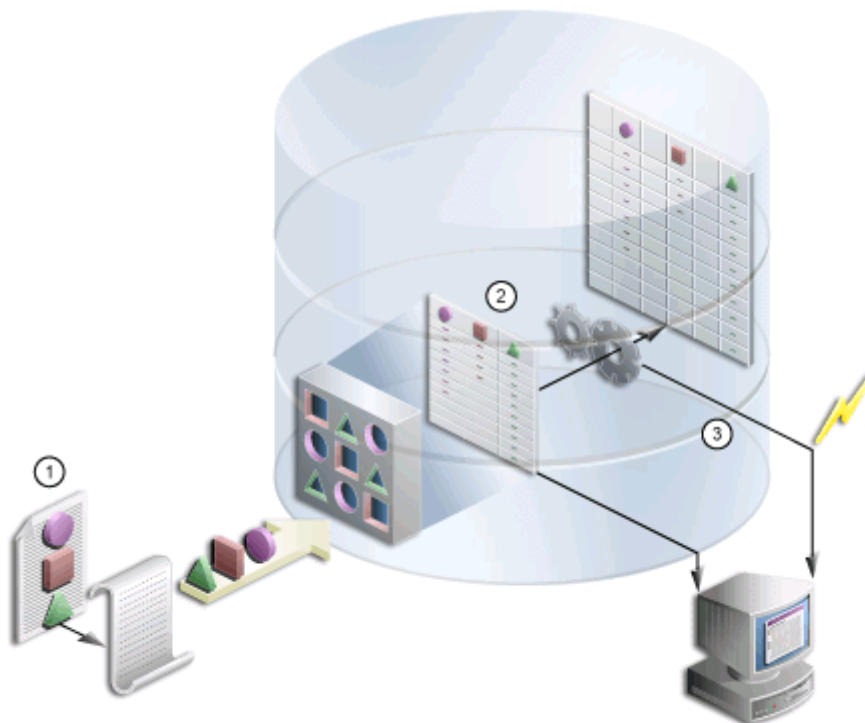
1. You import the contents of the transport set dump file to the transport set tables utilizing the same script used in the export.

2. A background job is submitted to initiate the import and log information is generated.

3. Once the import is complete, you can access the transport set from the User Interface.

> **Note:** To preserve data integrity, avoid:
>
> - Importing an object, changing its name then re-importing it.
>
> - Importing an object, promoting (moving it to shared objects) then re-importing it.
>
> - Importing an object, then moving it from one hierarchy to another.

10.5.2 Importing Your Data

To import an object, the contents of the transport set must first be imported to the target system. When you select a transport set for import, a pre-check process determines if the objects already exist on the target.

To import your content:

1. Locate the **Export/Import Transport Set** portlet, installed by default on the **Administer** tab.

> **Note:** When you import a transport set and click the **Browse Transport Sets** link, you will see the newly imported transport set with a status of 'Export Complete' and links to the export scripts.
>
> Selecting a transport set on the target for **Reuse** resets the transport set. This makes the transport unusable because it was not exported from the target instance and therefore no objects exist that match the objects in the transport set.
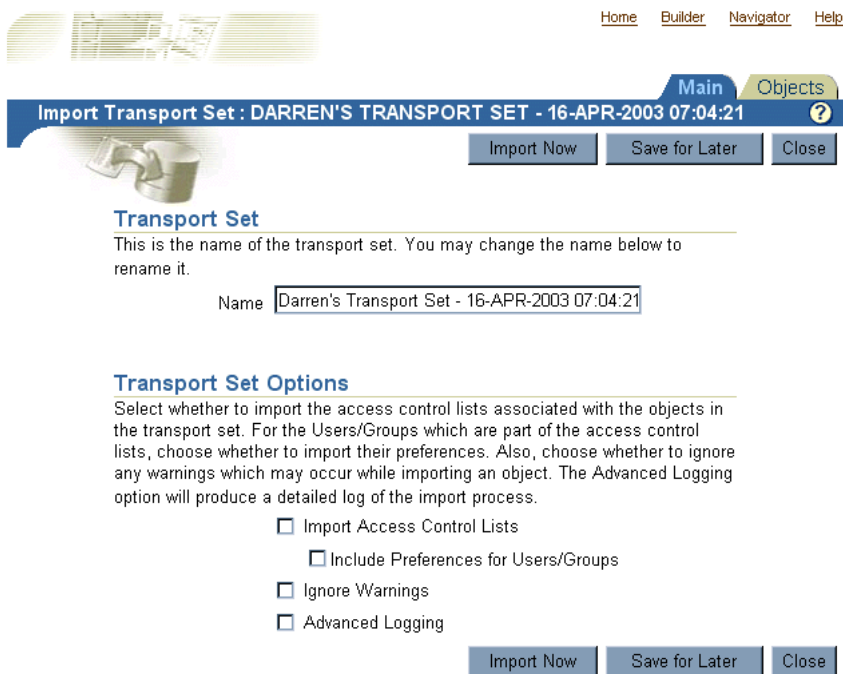
2. Select the imported transport set; click **Import.**

The Import Manager is displayed.

Check the appropriate boxes under the Transport Set Options:

> **Note:** The Access Control Lists and user preferences cannot be selected if you chose not to select them during the export process.

- **Access Control Lists**. Includes the Access Control Lists associated with the objects in the transport set.

- **Include Preferences for Users/Groups**. Includes the users and groups global privileges when object **Access Control Lists** are selected for import.

- **Ignore Warnings.** Allows the import to proceed when a warning is encountered.

- **Advanced logging.** Provides a detailed log of the import process, includes debug messages.

3. Click the **Objects** tab to view the list of objects included for import.

4. If you select **Replace on Import**, the object is replaced if it is found in the target portal.

> **Note:** **Replace on Import mode** is the default mode for explicitly selected objects; reuse is the default mode for referenced objects. The import modes are not applicable to the external objects until they are "promoted" to explicitly selected objects.

Figure 10–10   Transport Set Manager Import Objects

5. To view the log output, click the **Status** icon. The following table provides a description of each status type.

Status	Description
✔	Pass.

Status	Description
❌	Fail.
⚠	Pass with Warnings. The Pass with Warnings status only appears when the **Ignore Warnings** option is selected in the transport set. Otherwise the object status will be set to Fail.

Figure 10–11   Transport Set Manager Import Log Output
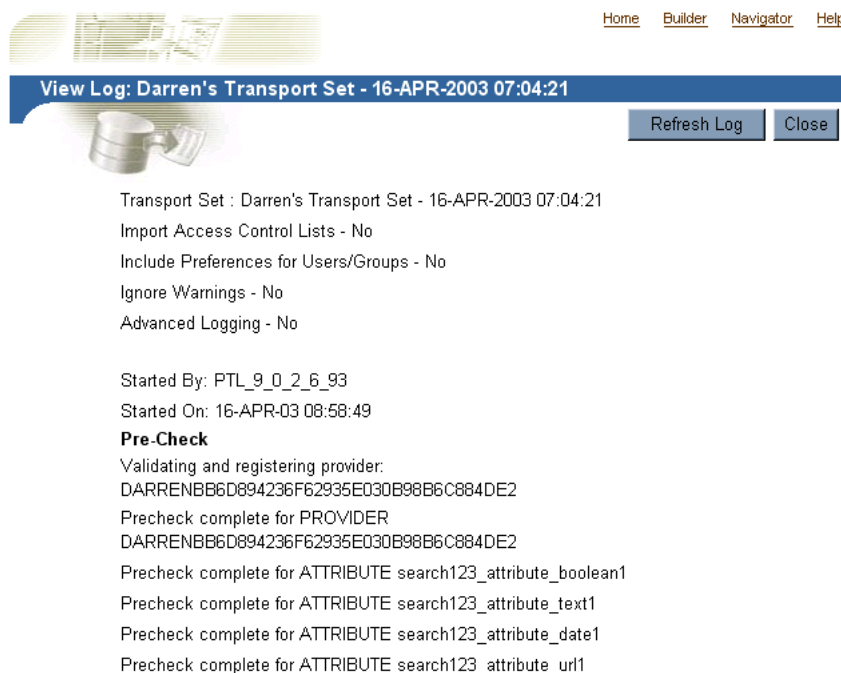
6. Click **Close** to return to the Objects page.

7. Select either, **Import Now** if you are finished or **Save for Later**.

   When you select **Import Now**, the exported objects are imported in the background. Clicking **Save for Later** saves changes to the transport set for later resolution and import.

8. Check the log for errors. If you select **Ignore Warnings**, then any warnings generated are ignored and the import proceeds. However, if the **Ignore Warnings** option is not selected, and warnings exist, the import will fail.

To ensure that everything has been imported correctly, check the following:

– In the Navigator, verify that the content in each portal page group that you have imported has been imported correctly. Specifically, for each portal page, verify that the appropriate portlets all appear in each region of your portal page. When these portlets (navigation pages, pages exposed as portlets, DB provider components or web portlets) occur as external dependencies and they do not exist on the target when **Ignore Warnings** is enabled then the portlet entry is deleted from the page.

Note: During import a two-step pre-check process is performed. Clicking the initial **View Log** shows both the first stage of the process, and the pre-check as complete. This is done prior to the actual import and prior to data populating the portal tables.

Clicking the **Refresh Log** will show both the second stage of the process and the pre-check with a different timestamp.

What happens when I select 'Ignore Warnings'?

Objects that are being imported can be classified into two types:

- Warning Types - Objects that, on failure cascade warnings to explicitly selected objects.

- Failure Types - Objects that, on failure cascade failures to explicitly selected objects.

When the **Ignore Warnings** option is selected the warning types will raise warnings and the explicitly selected objects will be imported. However, if there is a failure type object and it fails, then the explicitly selected object will also fail irrespective of the **Ignore Warnings** value.

If an explicitly selected object has two dependencies, a warning type and a failure type and if both the dependencies fail the pre-check process, then the failure type will dominate and the explicitly selected object will also fail even if **Ignore Warnings** is selected.

If **Ignore Warnings** is not selected, then the warning type objects will fail, that is, the explicitly selected object will fail.

Ignore Warnings impacts explicitly selected objects more than any other kind of object. Referenced and external objects raise failure/warnings for the explicitly selected object based on their type and whether the **Ignore Warnings** option is set. Figure 10–6 describes the expected behavior for each object when selecting the **Ignore Warnings** option.

Table 10–6   Warning/Failure Types

Object	Type	Expected Behavior
Attribute	Failure	The explicitly selected object will fail when the dependent attribute fails.
Itemtype	Failure	The explicitly selected object will fail when the dependent itemtype fails.
Pagetype	Failure	The explicitly selected object will fail when the dependent pagetype fails.
Style	Warning	The style will default to the main style of the page group that it belongs to.
Category	Warning	The category is set to 'none'.
Perspective	Warning	The perspective associated with an item/page is removed.
Page Template	Failure	The explicitly selected object will fail when the dependent template fails.
Page	Warning	There can be three possible outcomes when a page is a dependent of another object: ■ **Page exposed as a portlet**. The portlet entry is removed from the region that contained the page portlet. ■ **Page link pointing to a page**. The page link item is removed from the region, since the page to which the link is pointing to has failed. ■ **Pronto Dependency**. The link that was pointing to the page that failed, is reset to point to the same page in which the Pronto link is located.
Navigation Page	Warning	The navigation page portlet is removed from the page. You can associate the page with another navigation page after import.

Table 10–6   (Cont.) Warning/Failure Types

Object	Type	Expected Behavior
Color, Font, JavaScript, Application Template, Image	Warning	Set to default at runtime.
DB Provider Component	Warning	The portlet entry where the component is placed is deleted from the page.

When the container objects listed subsequently appear as an external dependency, because their child objects have been selected for export and they do not exist on the target, then the explicitly selected object (child object of the container object) will always fail irrespective of the **Ignore Warnings** value.

- Page group
- Portal DB Provider
- Category
- Perspective
- Page

10.6 How Do I Manage My Transport Sets?

The Export/Import Transport Set portlet, shown in Figure 10–12, is installed by default on the Administer tab and enables you to export, import, edit and browse the transport sets currently on the system. This section discusses the following:

- Editing a Transport Set
- Browsing Transport Sets

Figure 10–12   Export/Import Transport Set Portlet
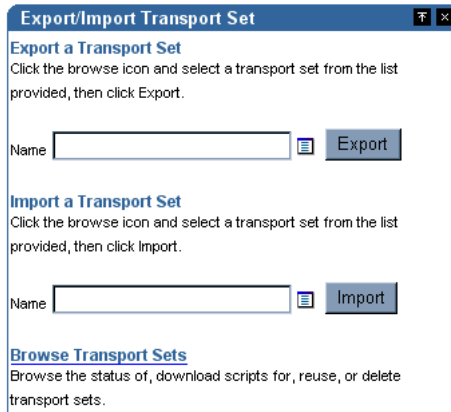
10.6.1 Editing a Transport Set

You can view and edit the list of objects selected for a transport set. Once you have created a new transport set and selected the **Save for Later** option:

- Navigate to the Export/Import Transport Set portlet.

- Select the Transport Set from the export list of values.

- Edit the preferences.

10.6.2 Browsing Transport Sets

You can view all of the transport sets that are on the system and their current status. You can also view the log of actions, referenced objects and download export/import scripts. Additionally, you can delete transport sets from the system or to reuse a transport set, select the transport set and click **Reuse**.

> **Note:**   The Reuse option is only valid for transport sets in the source portal with a status of 'Export Complete' or 'Export Failed'.

Figure 10–13 Browse Transport Sets

10.7 How Do Objects Behave After Migration?

The following considerations should be made before migrating portal content from a source to a target instance using OracleAS Portal export/import. This section discusses the behavior of portal objects after migration.

Table 10–7   Behavior of Objects

Object Type	Behavior
Page Groups	On the first export/import, if a page group doesn't exist, it is created on your target system. Any settings at page group level are replicated on the target system. On the second import, depending on the mode selected:
	Replace on Import mode. The page group properties from the source replace those on the target. All objects within the page group are created/updated depending on whether they existed or not.
	Reuse mode. When page groups already exist on the target the properties are reused and not updated. New objects within the page group are created, existing objects are reused.
	Notes:
	■ **New pages are currently not created when page groups are imported using reuse mode.**
	■ The order of visible objects (in the **Configure** tab) may differ between the source and target portal. This will result in the drop-down lists (when selecting an item, category, and so on) to look different in the target portal. You can manually re-order the visible objects in the target.
Attributes	On the first export/import, the attributes are created on the target system**.** The second import, depending on the mode selected for your target:
	Replace on Import mode. The properties of the attribute are updated.
	Reuse mode. When the attribute already exists on the target it is reused and not updated.
	Notes:
	■ Attributes that are marked as external cannot be created on the target even with **Any Transport Set - Manage** privilege.
	■ Attributes on the source and the target can only be considered the same when they have the same name, are the same type and have the same unique internal identifier. If the two attributes have the same unique internal identifier but different names then they can be only imported in replace on import mode. If the name and the type are the same but the unique internal identifier is different then the attribute import will fail and cascade to any other related objects.

Table 10–7   (Cont.) Behavior of Objects

Object Type	Behavior
Approvals	To view the approvers, Access Control Lists must be exported and imported along with the page group or page that has an approval defined on them.
	Replace on Import mode. The Approval process can be established for a page or page group. If a page group or a page is marked for either insert or update, then the approval object will be processed in replace on import mode. All the information in the target will be deleted and re-created.
	Reuse mode. No action is performed.
Items	Item information comes as a part of page export. They follow the import mode of the page.
	Replace on Import mode. When a page is imported in Replace on Import mode, items in page regions from the source are copied to the target. Any items found only on the target are removed, items that exist on both the source and target are updated, and items that exist only on the source are created.
	Reuse Mode. No items are imported from the source. The page from the source is only used as a reference, and will determine the import mode of items.
	Notes:
	■ If PL/SQL items are present in the pages being exported, the Dependency Manager does not mark the PL/SQL execute schema as an external dependency. For this reason, ensure that the target database instance already has the schema that is referenced by PL/SQL items, or manually migrate them before importing the items.
	■ The list of object items will show differently between source and target unless you migrate those referenced objects (pages, categories, and perspectives) within the same transport set as the list of objects. Note that the Dependency Manager will not mark the objects referenced in the list of objects for export. For this reason, you need to explicitly mark those referenced objects for export, or ensure that they are already in the transport set.
	■ If portlet instance items are moved from one region to another between subsequent imports of the same page, any customization made by the user(s) on those portlet instances are removed.

Table 10–7   (Cont.) Behavior of Objects

Object Type	Behavior
Pages	Exports the page and the page type, template, and style it references along with content (item and portlets).
	Replace on Import mode. The properties of the page are replaced. For region import behavior see, 'Regions'. For item behavior see, 'Items'.
	Reuse mode. The original page on the target is reused.
	Notes:
	■ The current release does not support locking and unlocking content using WebDAV. Content contributors can lock a file, which in turn will check out the item. On import no owned locks will be displayed.
	■ Edit defaults and customizations for web portlets are not currently migrated.
	■ When a page exposed as a portlet appears in the external objects list, make sure to include the page in the transport set.
Regions	Region information comes as part of page export. They follow the import mode of the page.
	Replace on Import mode. When a page is imported in replace on import mode, page regions from the source are copied to the target. Any regions found only on the target are removed, including all content in those regions.
	Reuse Mode. No regions are imported from the source. The page from the source is only used as a reference, it will determine the import mode of regions.

Table 10–7   (Cont.) Behavior of Objects

Object Type	Behavior
Templates	Exports the template and the style it references and any content on the template. The layout and content of pages that depend upon the template are synchronized with the revised template on the target.
	Replace on Import mode. The template properties are replaced on import.
	Reuse mode. Template information is reused on the target and is not updated from the settings on the source system.
	Notes:
	■ **Changes to the template are currently not replicated in pages that use the template in the target when the pages are not imported along with the template. You should always export the template and the pages (using the template) together. Avoid creating pages that use the imported template directly on the target as these would not be synchronized correctly with the updated template.**
	■ Do not export or import the following templates found in the shared objects or page group (they are present only if a category or perspective is created in that page group), Category Pages Template or Perspective Pages Template.
	■ A template can force all pages based on the template to use the template's style, or it can allow pages based on it to have their own style. When importing a template whose style has changed, the changes are only propagated to the pages based on the template if the template forces the pages to use the template's style.
Categories	Exports the category and its sub-categories.
	Reuse mode. The original category on the target is reused.
	Notes:
	■ The category page (the page that appears when a category is clicked) and the category template are not exported. They are created each time on import. The category is always reused, therefore you should make any changes once on the target and it will never be lost during subsequent imports. This applies to the category, the category page and the category template.
	■ There is no replace on import mode. The replace on import option will not apply, the category will always be reused.

Table 10–7   (Cont.) Behavior of Objects

Object Type	Behavior
Perspectives	Exports the perspective and its sub-perspectives. **Reuse mode**. The original perspective on the target is reused. **Notes**: ■ There is no replace on import mode. The replace on import option will not apply, the perspective will always be reused. ■ The perspective page (the page that appears when a perspective is clicked) and the perspective template are not exported. They are created each time on import. The perspective is always reused, therefore you should make any changes once on the target and it will never be lost during subsequent imports. This applies to the perspective, the perspective page and the perspective template.
Navigation pages	Exports the navigation page and the style it references and any links on the navigation page. **Replace on Import mode**. The properties of the navigation page are replaced. **Reuse mode**. The original navigation page on the target is reused.
Styles	Exports the style. **Replace on Import mode**. The properties of the style are replaced. **Reuse mode**. The style on the target is reused. **Note**: Styles on the source and the target can only be considered the same when they have the same name and the same unique internal identifier. If the two styles have the same unique internal identifier but different names then they can be only imported in replace on import mode.

Table 10–7   (Cont.) Behavior of Objects

Object Type	Behavior
Item Types	Exports the item type and the attributes it references.
	Seeded item types can be modified, such as file item but these modifications are not reflected on the target.
	Notes:
	■ It is recommend that if you modify a seeded item type, that you make a copy of the seeded item type and modify the attributes of the copy.
	■ Item Types on the source and the target can only be considered the same when they have the same name, are the same type and have the same unique internal identifier. If the item types on the source and the target have same unique internal identifier but different names then they can only be imported in replace on import mode.
	■ Currently when the attributes associated with the custom types (item type, page type) are modified or the functions associated with the custom type are modified between imports, the changes are not always correctly migrated. You should delete and re-create the custom type on the target. This will result in all the items/pages (based on the custom type) being deleted.
Page Types	Exports the page type and the attributes it references.
	Note: Page Types on the source and the target can only be considered the same when they have the same name, are the same type and have the same unique internal identifier. If the page types on the source and the target have same unique internal identifier but different names then they can only be imported in replace on import mode.
Portal DB Providers	Exports the entire Portal DB Provider and the shared components it references. Make sure that the schema referenced in the registration information (in your source portal) is exported and imported first.
	Replace on Import mode. The properties of the Portal DB Provider are replaced.
	Reuse mode. The Portal DB Providers on the target are reused.
	Note: Whenever a database provider component is exported, the provider associated with the component is included as an external dependency.

Table 10–7   (Cont.) Behavior of Objects

Object Type	Behavior
Web Providers	All web providers referenced by your transport set must either exist already on your target system or be able to be registered successfully during the import on your target system.
	Reuse mode. Providers are always reused.
	Note: If the provider registration generates an error, due to insufficient privileges then the provider object fails the pre-check stage. This is then cascaded to the explicitly selected objects, a provider failing always fails the explicitly selected objects irrespective of the ignore warnings value.

10.8 What Are the Recommended Best Practices?

The following is a summary of important recommendations and best practices developed for migrating portal content from a development or test environment to a production instance using OracleAS Portal export/import:

- Migrating Your Users and Groups
- Migrating Your Page Groups and Components
- Migrating Your Web Providers
- Migrating Your Portal DB Provider Schemas
- Migrating Your Portal DB Providers and Components
- Migrating Your Oracle Reports Components
- Migrating Your Search Components
- Migrating Your External Applications
- Migrating Your Portal Across Databases

10.8.1 Migrating Your Users and Groups

Oracle recommends the following procedure for export/import:

- Develop your portal objects (page groups, content, portlets, and so on) on your source/development system.
- To simplify the task of export/import, assign users, groups and privileges ONLY on your production system.

- Use export/import to migrate your portal objects to your target/production system.

- Apply users and privileges to imported portal objects as needed.

Users and groups are defined in Oracle Internet Directory. When you choose to include Access Control Lists and User and Group Preferences during a Portal Export, the user and group profiles held in the Portal schema are included in the transport set. However this does not migrate the user and group definitions that are held in Oracle Internet Directory.

For the user and group profiles to be properly imported on the target portal, the user and groups that they refer to, must exist in the target portal's associated Oracle Internet Directory.

If you are building your portal content on a test or development server, with the intention to then move that content to a production server, you have the option of assigning your security privileges on the test server and then migrating them, along with the content, to your production server.

In this scenario, assign the privileges to groups, so there is no need to ensure the consistency of the user population between the test and production infrastructures.

If you want to precisely model your user population on both the production and test servers, the best approach is to use Oracle Internet Directory's Directory Integration Platform - Directory Synchronization capabilities to synchronize the data from the production directory server to the test server. Synchronizing the data from production to test also provides you the option of adding test users and groups to the test Oracle Internet Directory server without affecting the production server.

> **Note:** See the *Oracle Internet Directory Administrator's Guide* for more information on setting up directory synchronization. Note that it is advisable to automatically synchronize the data from production to test, but not the other way around.

With the production groups also present on the test server, you can model and test all your access privileges on the test server and then safely migrate the Portal Access Control Lists with your exported objects onto the production system.

If you are introducing new groups and access privileges for those groups on the test system, then before you move the Portal content and access control lists to production, make sure you migrate the group definitions to production first. You

can actually create the groups on production first, and let the synchronization process reflect the new group on the test system before applying the test access control entries, if you need to actually create the group on the test instance first, you can create the group on production with the same means you used to generate the group on test. If this was done manually, and you want to avoid repeating the manual step on production, you can issue an LDAP query on the test instance to generate an LDIF file, which you can then load onto the production instance. For example:

```
%ldapsearch –h testoid.domain.com –p 389 –D cn=orcladmin –w password123 -b
'cn=portal.iasdb.domain.com,cn=groups,dc=domain,dc=com' –s sub –L 'cn=groupname'
> newgroup.ldif
```

> **Note:** Before loading the LDIF file containing the group information into the production Oracle Internet Directory instance, you may need to edit the file to correct the portal instance name to match the name for that portal instance on the production Oracle Internet Directory instance. This name will typically be different between the test and the production instances and the name is part of the group DN, so it will have to be modified before loading the file.

In this example, `cn=portal.iasdb.dbserver.domain.dcom, cn=groups, dc=us, dc=oracle, dc=com` is the location under which the portal groups are located. Refer to the Security chapter for more information on the organization of the entries in the Directory Information Tree in Oracle Internet Directory. This creates a file called `newgroup.ldif` containing the group definition. You can then load the file on the production Oracle Internet Directory instance by using `ldapadd`:

```
%ldapadd –h prodoid.domain.com –p 389 –D cn=orcladmin –w password123 –v -f
newgroup.ldif
```

You may only want to deploy default privileges granted to some of the seeded Portal groups, or no privileges at all. If no privileges are deployed, then the user performing the import will own the objects. The user can then further grant privileges on the target system as necessary for the specific deployment.

There is no need to synchronize seeded groups or users, assuming that, if privileges are granted to seeded groups in Portal, and those seeded groups are still present on

the target system, then the privileges will be correctly associated with those seeded groups.

When migrating group profiles from the source to the target, the import will remap the DNs of the groups to the local group base on the target system if the exported profile was one for a local group on the source. A local group is one that is under the portal group container (the group install base). For groups that were not under the group install base, the DN will remain unchanged.

> **Note:** The `ssoexp` and `ssoimp` scripts found in the `wwu` directory are obsolete for 9.0.x and not compatible with the 9.0.x login server. These should not be used.

10.8.2 Migrating Your Page Groups and Components

Page groups and their associated components may be moved from development to production using the export/import utilities described in this document. In addition to page groups as a whole, individual components within page groups such as sub-pages, categories, perspectives and page styles can be moved individually to the target system, only if the entire page group has been imported to the target system earlier.

> **Note:** The current release does not support the use of circular references. If you import a page group which contains a circular reference then this will produce an `ORA-00001: unique constraint` error, however the import should still finish successfully.

- Some considerations and best practices to keep in mind are the following:

    - The first export to your target system should migrate the entire page group from the source portal to the target portal instance. Subsequent transport sets can then export an individual page, or other page group component on the target portal installation.

> **Note:** The pre-check process will fail for an object if the page group does not exist on the target. Whenever a page group object is exported, the page group that owns the object is included as an external dependency. You can chose to promote the page group if you do not know if the page group exists on the target and therefore avoid any potential pre-check failures.
>
> The same applies to other objects included in a hierarchy. Categories, perspectives and pages when exported display the parent category, perspective or page as an external dependency in addition to the page group to which they belong. All database provider components display the provider as an external dependent when they are exported by themselves.
>
> The default settings of a page group, for example, the default template, style, navigation page, and so on, are also extracted by the Dependency Manager and classified as either reference or external (that is, local or shared).

- All new or existing content on a page is replaced when a page with the same name is being re-imported to the target.

- You can only move objects within a page group to the same page group of the same name on the target portal.

- A page is migrated along with any sub-pages.

- After an initial import operation to your target system, if you change the name of the page group on the target system, subsequent import attempts to that page group will fail.

- Categories, itemtypes, perspectives and pagetypes that are configured in the source are not automatically configured in the target. You must explicitly configure these objects unless you are doing a page group export.

- **Page URL Behavior**. Always use page link item types or direct access URLs when creating links to portal pages. Do NOT use "raw" portal page URLs.

  By default, portal page URLs generated by OracleAS Portal contain installation specific ID numbers that change when the object is exported. This causes broken links when pages are imported into a different site.

  Here is an example of a URL generated for a page. If the page is imported on another site, this PAGEID will change.

```
http://my.portal.com/servlet/page?_pageid=47,49&_dad=portalr2&_schema=portal
```

If you are using such URLs as manually entered links, we recommend instead the use of Direct Access URLs or Page Link item types.

The same page has this direct access URL:

```
http://my.portal.com/pls/portal/url/PAGE/HRPAGEGROUP/HRHOME/HRBENEFITS
```

To find the direct access URL for a page, look at the page property sheet. A link to the property sheet can be displayed by adding a Property Sheet Smart Link item to the page.

You can also use a Page Link item type to create a link to a page. The Page Link item type dynamically generates the correct link at runtime.

- **Page Portlets**: When you replace a page, the content as well as the structure is replaced on the target.

> **Note:** This release does not support the import/export of the OracleAS Portal Survey components or the Favorites portlet.

To preserve content in a page (items, portlets) on the target, but import a style, layout or rendering changes from the source then expose your content through the Federated Portal Adapter portlet. The key here is to separate your content from your page structure into two separate page groups. One for content only, exposed through the Federated Portal Adapter, and the other is your 'display' page group. Users can use this to access, view, and customize their portal. Follow these steps:

1. On the source system, create a page group that only contains pages that have one region that you will later expose to other pages. This region is to be populated with either portlets or items. Name this page group "Content Page Group".

2. Export this content page group to the target system.

3. On the target system, register the content page group through the Federated Portal Adapter. Expose these pages as portlets through the Federated Portal Adapter provider on the target system.

4. On the source system, register the same provider (using the same name as the Federated Portal Adapter provider).

5. On the source system, create another page group called "Display Pages". In this page group, construct pages with regions that expose the portlets from the Federated Portal Adapter provider. You can also include tabs, and other portlet regions in this page group if required.

6. Export the "Display Page" group to the target system.

7. From the target system, update, delete, modify, and add new items to the regions, pages in the content page group exposed through PL/SQL provider.

8. On the source system, make changes to the page structure (tabs, new regions, and so on) to the "Display Page" page group.

9. Export the latest "Display Page" page group to the target system.

10. Verify that the "Content Page Group" contains the new changes that you made in step 7 on both the source and the target environments.

11. Verify that the target system contains the latest changes to the pages in the "Display Page" page group that you recently changed.

Note: When a page containing a portlet from an adapter rendered provider (the loop-back case) is imported and the provider is automatically registered on the new portal, it will have the old URL, referencing the old portal.

When a loop-back provider is required in the new portal, you will have to create one or update the default provider.

- **Page and Portlet Customizations and Edit Defaults Migration. You can preserve the user customizations on a page or portlet on the target system while replacing or reusing the edit properties or edit defaults of that page or portlet.**

  Note: Edit Defaults and Customizations for web portlets are not currently preserved.

Base objects that no longer exist on the page in the source portal will be removed from the target page after subsequent imports. This will ensure that all customizations for base portlet regions are also removed. Base objects are

regions, portlets/items and tabs that are imported as part of the core definition of the page, defining its structure and content.

Portlets that already exist on a page behave in the following way when the page is imported in replace on import mode:

- Edit Defaults will be replaced.

- User Customizations will be preserved.

Properties of the page behave in the following way when the page is imported in replace on import mode:

- Edit Properties will be replaced.

- User Customizations will be preserved, subject to the user customizations being valid.

> **Note:** You can customize, add, hide/show, delete and move portlets and tabs. The page must have at least one portlet region and one tab (tab related customizations) in that region. The customized objects inherit the properties of the page. When a region is deleted, for example, a second import removes the region or tab from the page, then customized objects will also be deleted.

When you import the page with an increase in the number of portlets on a page, then the source takes precedence even if you have customized the page in the target and deleted a portlet. The next time you import the same page, the deleted portlet is considered to be a new portlet to be added to the structure on the target. This also applies to tabs.

The order of appearance of these portlets (customizations) and the portlets that form the content of the page are determined by the source and mode of import.

- **Replace on Import mode**. The portlets from the source are arranged in the order found in the source followed by the portlets in target (customizations).

- **Reuse Mode**. The customizations are preserved and there will be no changes to the target page.

10.8.3 Migrating Your Web Providers

Before importing on your target system, all providers referenced by your transport set must either already exist on your target system or be able to be registered successfully during the import on your target system. The pre-check process determines if a provider of the same name already exists on the target. If the provider does not exist, then the pre-check attempts to register the provider.

To ensure successful registration, check that your providers meet the following conditions on your target system:

- Ensure that you have sufficient privileges to register the web providers.

- Ensure that you have connections to your providers during the import operation. An alternative is to remove portlets of those providers that may not be contactable or available during the import process from the pages before exporting. Then add them back manually to the pages on your target system after importing.

- If you are using proxies on either your import or export portal installations, ensure that your proxies are configured correctly on your import installation before importing.

- Consider registering your providers manually in advance of performing your import on the target system to help ensure that your import operation goes smoothly.

    > **Note:** If you start the import process then decide not to proceed, some stray providers may remain on your target portal due to the pre-check process of registering the providers.

- If you register your providers manually, they need to have the same name as the corresponding providers on your source system.

    > **Note:** A different URL for the development (source) provider and the production (target) provider can be used. Pre-register the production provider on the target portal server with the same name as the development provider on the source portal server but pointing to the appropriate URL for the production provider. When pages referencing the provider are imported from the source to the target, they will point to the production provider instead of the development provider.

10.8.4 Migrating Your Portal DB Provider Schemas and Components

A Portal DB Provider Schema is the schema specified when registering the DB provider. The subsequent steps must be performed for the successful migration of pages that reference Portal DB Provider portlets.

1. Use the Oracle EXP utility to export the Portal DB Provider schemas. See Section 10.8.4.1, "Migrating Your Portal DB Provider Schemas" for details.

2. Export the Portal DB Providers and Components. See Section 10.8.4.2, "Migrating Your Portal DB Providers and Components" for details.

3. Use the Oracle IMP utility to import the Portal DB Provider schemas.

4. Import the Portal DB Providers and Components.

> **Note:** Make sure that the schema referenced in the registration information (in your source portal) is exported and imported first and has the same name on target as the source. If your Portal DB Provider components references schema objects outside this schema then ensure that the other referenced schemas are also present on the target. Grant any necessary privileges between these schemas and the Portal DB Provider schema.

The simple example that follows provides an overview of the process. *SAMPLE_APPLICATION* is a Portal DB Provider that needs to be migrated. It uses the *PORTAL_SRC_DEMO* schema for the database objects referenced by the components.

Assumptions

- PORTAL_SRC_DEMO is the Portal DB Provider building schema in the source.

- PORTAL_SRC is the source portal instance.

- PORTAL_DST is the target portal instance.

Steps

Ensure that the source portal instance is setup in one database instance and that the target portal is setup in a different database instance.

1. Use the Oracle EXP utility to export the PORTAL_SRC_DEMO schema from the source.

2. Export the Portal DB Provider from PORTAL_SRC.

3. In the target database instance, use the Oracle IMP utility to import the PORTAL_SRC_DEMO schema.

4. Import the dump file (from step 2) into the newly created schema on the target database instance.

5. Import the Portal DB Provider into PORTAL_DST.

10.8.4.1 Migrating Your Portal DB Provider Schemas

The Portal DB Provider schemas must be exported and imported first. Follow the steps that follow:

- For database schema(s) used by Portal DB Providers that are marked for export

  - Use Oracle EXP/IMP utilities to migrate the database schema(s)

  - Grant privileges that were not covered by the migration specified earlier

10.8.4.2 Migrating Your Portal DB Providers and Components

Portal DB Providers (portlet builder components) may be exported as part of a transport set in much the same way as page groups, either independently or as part of the same transport set with page groups and shared objects.

> **Note:** The current release does not support exporting seeded providers, for example, 'people_app' or development in place portlets.

If the same schema name cannot be used for the Portal DB Provider in the target portal instance, then perform the following steps:

1. Create the schema with all the required objects for it to be a Portal DB Provider.

2. Manually register a Portal DB Provider with the same name as the export portal pointing to this schema before using Portal export/import.

The following Portlet Builder object types can be exported from Portal:

Table 10–8   Portlet Builder Object Types

Object Type	Description
Portal DB Provider	Exports the entire Portal DB Provider and the shared components it references.
Form	Exports the form and the shared components they reference.

Table 10–8   (Cont.) Portlet Builder Object Types

Object Type	Description
Report	Exports the report and the shared components they reference.
Chart	Exports the chart and the shared components they reference.
Calendar	Exports the calendar and the shared components they reference.
Dynamic Page	Exports the dynamic page and the shared components they reference.
XML Component	Exports the XML component and the shared components they reference.
Hierarchy	Exports the hierarchy and the shared components they reference.
Menu	Exports the menu and the shared components they reference.
URL	Exports the URL and the shared components they reference.
Frame Driver	Exports the frame driver and the shared components they reference.
Link	Exports the link and the shared components they reference.
List of Values	Exports the list of values and the shared components they reference.
Data Component	Exports the data component and the shared components they reference.

> **Note:** If a page group contains portlets from a Portal DB Provider, then the provider has to be explicitly included in the transport set you are exporting. Otherwise, export/import the provider beforehand.

10.8.5 Migrating Your Oracle Reports Components

Oracle Reports objects may be moved from development to production using the export/import utilities described in this document. Individual components can be moved to the target system, only if the first export migrates the entire page group from the source portal to the target portal instance. Subsequent transport sets can then export individual components.

Table 10–9   Oracle Reports Object Types

Object Type	Description
Server Access	Exports the Server Access file.
Combined Availability Calendar	Exports the Combined Availability Calendar.
Simple Availability Calendar	Exports the Simple Availability Calendar.
Report Definition File	Exports the Report Definition file.
Printer Access	Exports the Printer Access file.

The Portal DB Provider owner creates every Report Security Access component (`RWSVR`, `RWPRN`, `RWCAL`, `RWCPC`, `RWREP`). Once the owner is migrated, all of the dependent components, shared components, and report security access components are migrated as well.

Available options include:

- Remove the reports server integration from the pages being exported on the source system (this would also include removing page parameters associated with or dependent on the reports objects).

- Export/import the page group and verify that the pages imported on your target system display correctly.

- Import reports on the target (through the Portal DB Provider). Integrate the reports server objects with the imported page on your target system.

10.8.6 Migrating Your Search Components

There a number of options for adding search components to your pages. You can add a simple Basic Search to match search criteria entered into the Search field; an Advanced Search, and a Custom Search to create an automatically executed search.

10.8.6.1 Basic and Advanced Search Portlets

Basic Search portlets and Advanced Search portlets can be exported and imported. After import, the portlets should appear as they did in the source portal including the user preferences (if the user preferences were being imported).

10.8.6.2 Custom Search Portlets

Custom Search portlets can have many customizations which refer to other objects in the portal, such as page groups to search, attributes to search on, image on submission form, style for results, page for the results, attributes for the results, default values for category, perspective and item type attributes. These can be referred to as dependencies. When a custom search portlet is exported and imported its dependencies are not automatically exported and imported. Therefore, it is possible that a custom search portlet is customized in the source but the dependencies do not exist in the target.

Also, a custom search portlet in the source may have been customized and then the dependency is removed from the portal and the custom search portlet's customizations are not updated. In this case when the custom search portlet is used for a search the missing reference is ignored. When the custom search portlet is re customized and the customizations saved the missing reference is removed.

On export, all the custom search portlets that have been selected for export are checked and any missing references are removed. The customizations are then included in the transport set.

On import, a pre-check will determine if any dependencies are missing in the target after import. Messages are written to the log, for each custom search portlet that has missing dependencies, the log will show the reference path of the custom search portlet and the missing dependencies and what will happen on import.

The page on which the custom search portlet resides will be flagged with a warning. On the actual import the custom search portlet customizations are modified to have the correct ID's of all the same dependencies in the target and the customizations are copied into the target.

> **Note:** Search results saved using the Saved Searches portlet are not imported or exported. You should submit the same search in the new target and save the latest set of search results.

10.8.7 Migrating Your External Applications

Portal export/import does not migrate any data that is in the Single Sign-On schema, `ORASSO`. However, portal pages that are migrated may contain instances of the external applications portlet, which refers to external applications that are defined in the `ORASSO` schema, along with user credentials for these applications.

Pages may also contain portlets from providers that are defined to include an associated external application for automatically authenticating to an external application that the provider is integrated with. In these cases, the referenced external application needs to be migrated along with the provider information.

The external application information is treated as external dependencies by the portal export/import utility. See Section 10.4.1, "Creating Transport Sets" for more details on the types of objects. When migrating portal content that references external applications, the references are expected to be present on the target portal during the import. For this reason, you will need to migrate any external applications that may be referenced before completing your import into the target portal.

The portal export/import utility does not assume that the external application identifiers will be the same on the source portal and the target portal.

> **Note:** The portal export/import utility matches external applications by checking that the external application in the target portal's SSO server has the same name as the external application defined on the source portal's SSO server.

This association by name also enables you to manually synchronize external application definitions between the source and the target portal's SSO servers.

10.8.7.1 User Populations

If the user population is different between the source and the target portal, you may not want to manually migrate the external application definitions and credentials using the `ssomig` utility, see Section 10.8.7.2, "The Export and Import SSO Utility".

If the user population is the same on the source and the target, then the credentials can be transferred. Pages must be migrated with security. If the export is done without security and without preferences, the external application portlets are still migrated and loose-wired, but without any of their customizations. See Table 10–10 for more details.

Table 10–10   Dataset Options

Dataset Options Criteria	Migration of External Application Portlets	Loose Integration of External Application	External Application Customizations
Without security and without preferences	Yes	Yes	No
With security and without preferences	Yes	Yes	Yes
With security and with preferences	Yes	Yes	Yes

10.8.7.2 The Export and Import SSO Utility

The utility `ssomig` (`ssomig.bat` in Windows) uses Perl, Oracle SQL*Plus, and the tools EXP and IMP to move data between two version 9.0.4 servers. The two operational modes, export and import, must be run separately.

For more information on the SSO Export/Import utility (`ssomig`), refer to the *Oracle Application Server Single Sign-On Administrator's Guide*.

10.8.8 Migrating Your Portal Across Databases

Oracle EXP and IMP utilities can be useful when deploying identical content across multiple OracleAS Portal instances.

> **Note:**   In the following steps, the *ORACLE_HOME* is meant to reference the database Oracle home and not the Oracle Application Server Oracle home. It is important that when running database scripts that the correct version is used from the proper Oracle home corresponding to the actual database instance in which portal is being imported.

The migration is a multi-step process that involves:

1. **Listing the schemas to be exported**. It is necessary to identify all of the schemas that need to be exported, including the portal schema, the portal "Public" schema, and any schemas used for Database Providers or Portlet Builder components. To list all the schemas run the following query from SQL*Plus as the Portal schema owner:

```
SELECT USERNAME, DEFAULT_TABLESPACE, TEMPORARY_TABLESPACE FROM DBA_USERS
WHERE USERNAME IN (user, user||'_PUBLIC', user||'_DEMO', user||'_APP')
OR USERNAME IN (SELECT DISTINCT OWNER
FROM WWAPP_APPLICATION$
WHERE NAME != 'WWV_SYSTEM');
```

> **Note:** This will only list schemas that are directly related to Database Providers or Portlet Builder components registered in portal. If any of these schemas additionally reference objects in other schemas, then they should be added to the list of schemas to be exported.

2. **Listing the tablespaces in the source database.** To list the tablespaces used in the source database run the following query from SQL*Plus as the SYS user:

```
SELECT DISTINCT TABLESPACE_NAME FROM DBA_SEGMENTS WHERE OWNER IN (<list of
schemas>)
UNION
SELECT DISTINCT DEFAULT_TABLESPACE FROM DBA_USERS WHERE USERNAME IN (<list
of schemas>)
UNION
SELECT DISTINCT TEMPORARY_TABLESPACE FROM DBA_USERS WHERE USERNAME IN (<list
of schemas>)
```

3. **Running the Oracle EXP utility**.

```
EXP \'sys/<password of sys user>@<Connect String> as sysdba\'
FILE=portal.dmp OWNER=<List of Schemas> LOG=portal.log
```

The export should terminate without any errors. If there are any `ORA- 00942` errors reported in this step, run the following script from SQL*Plus as the SYS user:

```
ORACLE_HOME/rdbms/admin/catexp.sql
```

> **Note:** The difference in syntax between Unix and NT platforms, you should omit the '\'. For example, `'sys/<password of sys user>@<Connect String> as sysdba`.

4. **Creating a backup of the target database.** Backup the target database before proceeding to the next step.

5. **Preparing the target database for import.** Before importing the portal schemas into the target database, it is necessary to ensure that the necessary pre-requisite packages have been installed.

> **Note:** The target database must meet the same minimum requirements necessary as a database for an Oracle Application Server Metadata Repository.

- `PTLASST` must be run in `SYSOBJECTS` mode.

- Run `ORACLE_HOME`/rdbms/admin/catldap.sql.

- Recompile all invalid objects by running `ORACLE_HOME`/rdbms/admin/utlrp.sql

- Initialize the portal login trigger for import. Run as `SYS`

- `ORACLE_HOME`/portal/admin/plsql/wwhost/insttrig.sql SYS

6. **Creating or altering tablespaces in the target database.** Check that the required tablespaces exist in the target database. The tablespaces in the target database must be the same as the source tablespaces.

- Check that the list of tablespaces identified in Step 2 exists in the target database. To list all the tablespaces on the target, run the following script from SQL*Plus as the SYS user:

```
SELECT TABLESPACE_NAME FROM DBA_TABLESPACES;
```

- To create a new tablespace, use the `CREATE TABLESPACE` or `CREATE TEMPORARY TABLESPACE` commands. For example:

```
CREATE TABLESPACE <tablespace_name>
DATAFILE '<datafile_location>' SIZE 20M
DEFAULT STORAGE (INITIAL 1M NEXT 2M MINEXTENTS 2) AUTOEXTEND ON;
```

`<datafile_location>` is the file location for the dbf file. On UNIX, for example, the location may be: `/u02/oracle/data/tbsa01.dbf`.

For any tablespaces that already exist in the target database, it is recommended that they be set to autoextend or they must be sized large enough to hold the

imported portal schemas. The following script can be used to enable autoextend on all datafiles:

```
SET DEFI OFF
SPOOL DATAFILES.SQL
SELECT 'ALTER DATABASE DATAFILE '''||FILE_NAME||''' AUTOEXTEND ON;'
FROM DBA_DATA_FILES ;
SPOOL OFF
@DATAFILES.SQL
```

7. **Creating the portal schema**. Change directories to *ORACLE_HOME*/portal/admin/plsql/wwv and run the following script from SQL*Plus as the SYS user.

```
@wdbisys.sql <Portal Schema> <Portal Default Tablespace> <Portal Temporary
Tablespace> WDBISYS.LOG
```

This creates the portal schema and grants all of the necessary privileges. Use the results of the query from Step 1 to find the names of the default and temporary tablespaces for the portal schema.

8. **Creating the portal_public schema**. Change directories to *ORACLE_HOME*/portal/admin/plsql/wws run the following script as the SYS user.

```
@cruser.sql <Portal Schema> <Portal Default Tablespace> <Portal Temporary
Tablespace>
```

This creates the PORTAL_PUBLIC schema.

9. **Creating placeholders for the schemas**. Check that the list of schemas that will be imported from Step 1. If the schemas already exist in the target database, then it is recommended that they be dropped. Before dropping any schemas, ensure that those schemas are not in use by other applications. To create a new users, use the following syntax:

```
GRANT CONNECT, RESOURCE TO <user> IDENTIFIED BY <password>;
```

A user must be created for each user in the list from Step 1. Use the `ALTER USER` command to adjust any user properties as necessary. For instance, the default and temporary tablespaces should be set to the ones specified by the results from the query in Step 1.

10. **Running the Oracle IMP utility**.

```
IMP \'sys/<password of sys user>@<Connect String> as sysdba\' FROMUSER=<LIST
OF SCHEMAS> TOUSER=<LIST OF SCHEMAS> FILE=PORTAL.DMP LOG=PORTAL_IMP.LOG
```

The following Import error can be ignored as it is expected:

```
IMP-00041: Warning: object created with compilation warnings.
```

11. **Compiling all the invalid objects**. Compile all the invalid objects in all the imported schemas. Run the *ORACLE_HOME*/rdbms/admin/utlrp.sql as the SYS user.

12. **Dropping the temporary login trigger.** Change directories to *ORACLE_HOME*/portal/plsql/admin/wwhost and run the following script from SQL*Plus as the SYS user.

```
@droptrig.sql.
```

13. **Granting connect through portal**. Perform the following commands from SQL*Plus as the portal user:

```
SET HEAD OFF
SET LINES 4000
SPOOL DBUSERS.SQL
SELECT DISTINCT 'ALTER USER '||DB_USER ||' GRANT CONNECT THROUGH '|| WWCTX_
API.GET_PRODUCT_SCHEMA||';'
FROM WWSEC_PERSON$;
SPOOL OFF
```

Run DBUSERS.SQL in the target portal instance to grant connect through privilege to database users associated with portal users.

11

Syndicating Content Into OracleAS Portal

External content sources can be syndicated by Oracle Application Server Syndication Services into OracleAS Portal as a syndicated channel (subscription) by the Portal administrator using the Syndication Channel Administration portlet. A syndication channel is the means by which content configured to use a syndication offer can be subscribed to and placed in a Portal folder in the Oracle Application Server Metadata Repository. (A folder in this case refers to a Portal page containing a default region of type item.) Once a channel is established, subsequent updates received through administrative pull or automatic push operations will keep the destination Portal folder in sync with the offer content. A Portal administrator can then grant page designers access to the Portal folder containing the syndicated content.

This chapter contains the following sections:

- Registering Syndication Portlet Provider
- Configuring Portal for Content Syndication
- Using Syndication Channel Administration Portlet
- Advanced Configuration Parameters
- Syndication Channel Administration Error Messages

The Portal administrator using this Syndication Channel Administration portlet can:

- Create new syndication channels of content to subscribe to from a list of offers.
- Edit the properties of existing syndication channels.
- Delete a syndication channel.
- Perform a full update of the content available through a syndication channel.

- View a list of updates for a specific syndication channel.

- View a summary report for a single update event for a specific syndication channel.

Note that offers for a content provider must have already been created using Oracle Enterprise Manager Syndication Services Administration pages before you can create a Syndication Channel Administration portlet. See *Oracle Application Server Syndication Services Developer's and Administrator's Guide* for more information.

11.1 Registering Syndication Portlet Provider

Oracle Application Server comes with a set of Provider Groups, one of which contains the Syndication Portlet Provider. In order to use the Syndication Channel Administration portlet, the Syndication Portlet Provider must be registered.

To register the Syndication Portlet Provider, perform the following steps:

1. Login to OracleAS Portal as an administrator.

2. Go to Portal **Navigator**.

3. Select the **Providers** tab.

4. Navigate to **Providers Group** --> **OracleAS Providers**.

5. Click **Register** for the **Syndication Services** Web provider.

11.2 Configuring Portal for Content Syndication

The following sections describe how to build the Syndication Channel Administration page and set up the Portal privileges on destination folders.

11.2.1 Build the Syndication Channel Administration Home Page

To build the Syndication Channel Administration home page, you must perform the following steps:

1. Create a new page or decide on the page where you want the Syndication Channel Administration portlet to be installed.

> **Note:** Only one Syndication Channel Administration portlet instance can be used for each Portal installation.

2. As an administrator, edit the selected page and add the Syndication Channel Administration portlet to a region of your choice, by browsing the Portlet Repository to the Syndication provider (see Section 11.1 for more information).

11.2.2 Setup the Portal Privileges on Destination Folders

The channels created through the Syndication Channel Administration portlet transfer the syndicated content from external sources into Portal Pages. In order to accomplish this goal, some privileges need to be granted to the portal user used by the channel application. As an administrator, you must add `manage all` permission for the Syndication Channel Administration Portlet user (see Step 2 in Section 11.3, "Using Syndication Channel Administration Portlet") on the destination pages or page-groups that will be used by the channels.

11.3 Using Syndication Channel Administration Portlet

To access and use the Syndication Channel Administration portlet:

1. Navigate to the **Syndication Channel Administration** home page.
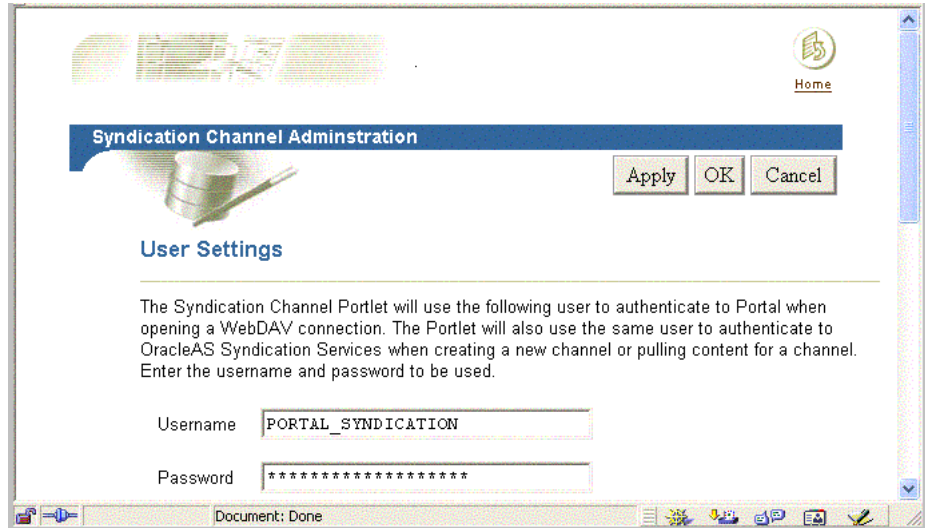
   At the **Syndication Channel Administration** home page, shown as follows, you can perform the following tasks:

   - Set user settings for the syndication channel administration portlet (click the **Edit Defaults** icon)

   - Create a syndication channel (click **Create a Syndication Channel**)

   - Edit the properties of an existing syndication channel (select a channel from the list provided and click **Edit**)

   - Delete a syndication channel (select a channel from the list provided and click **Delete**)

   - Immediately retrieve content (an incremental update) for the specified syndication channel (select a channel from the list provided and click **Pull Content**)
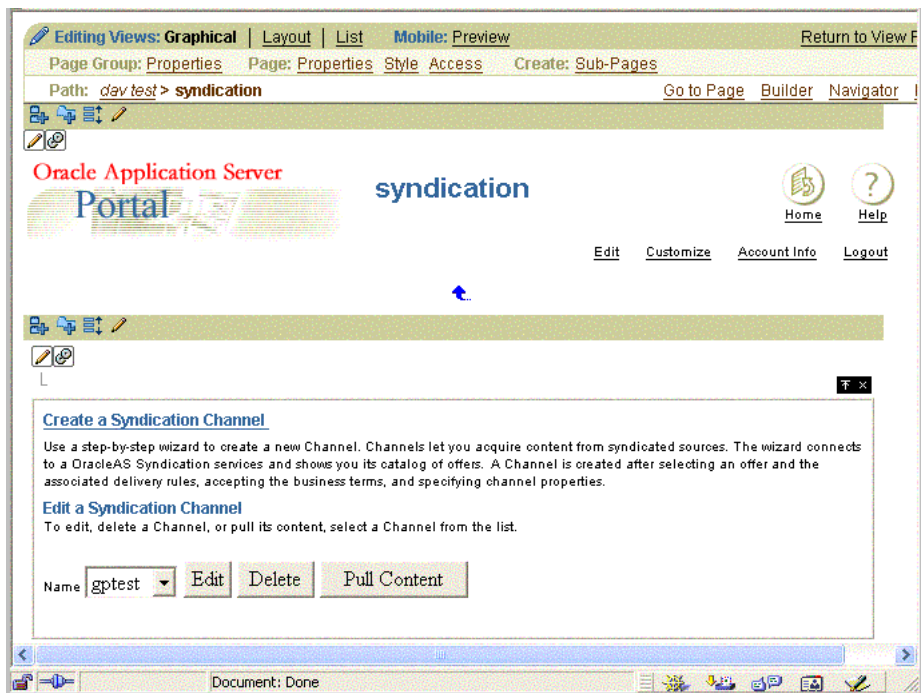
2. To set user settings, click the **Edit Defaults** icon. The **Syndication Channel Administration** configuration page is displayed as follows (the top half of the page followed by the bottom half of the page).

On this page:

- In the **User Settings** section, you can specify the user name and password to authenticate to Portal when opening a WebDAV connection. The portlet will use the same user to authenticate to OracleAS Syndication Services when creating a new channel or pulling content for a channel.

- In the **Portal WebDAV Settings** section, you can specify the URL to be used to connect to Portal using the WebDAV protocol.

- In the **Syndication Server Settings** section, you can specify the URL to be used to connect to OracleAS Syndication Services when creating new channels or pulling channel content.

  - Enter the user name and password to authenticate to Portal when opening a WebDAV connection.

  - Enter the URL to be used to connect to Portal using the WebDAV protocol.

  - Enter the URL to be used to connect to OracleAS Syndication Services when creating new channels or pulling channel content.

  - After you have entered this information, click **OK** to save this information. You will return to the **Syndication Channel Administration** home page.

3. To create a new syndication channel, click **Create a Syndication Channel** to launch the 5-step create syndication channel wizard.

   The first step of this wizard, the **Offers** page is displayed.

   > **Note:**   Use only the Back, Next, Cancel, Finish, or Home buttons on the create new syndication channel wizard pages to navigate through the wizard or to return to the Channel Syndication administration portlet. Do not use the Web browser's Forward or Back buttons to navigate through the wizard because the state of the wizard session is not validated and will result in an internal error being thrown at the next submit operation.

   a. At the **Offers** page, shown as follows, select one of the available offers by clicking its radio button, then click **Next** to continue to the next step.
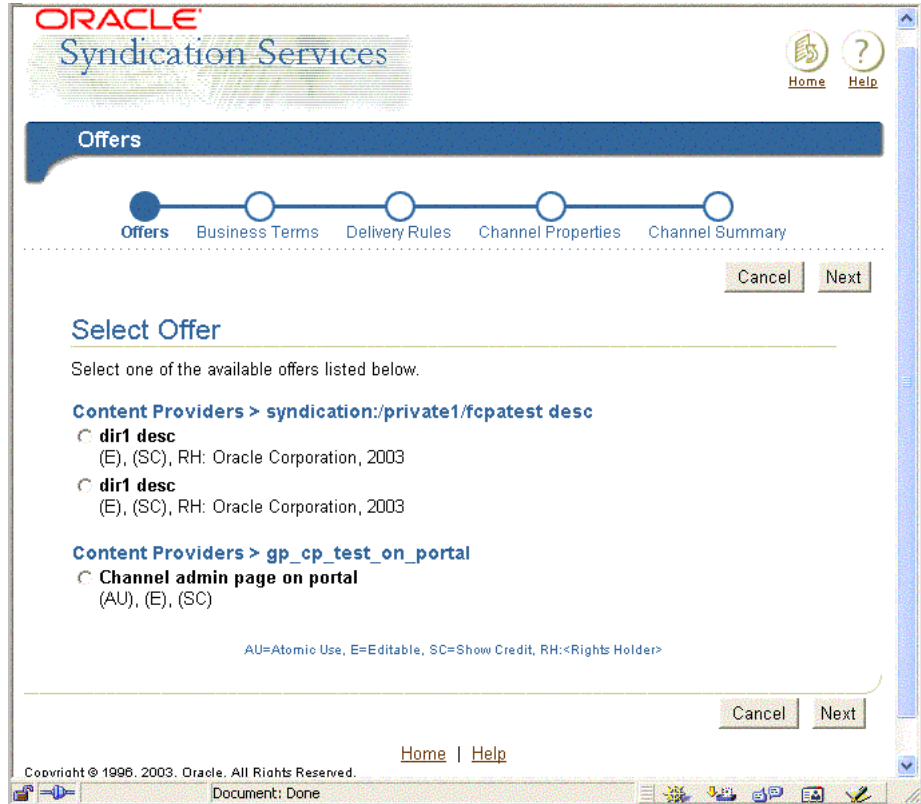
b. At the **Business Terms** page, shown as follows, in the **Accept Business Terms** section, review the business terms of the offer you selected.

If the business terms are acceptable, click the radio button **I Have Read and Accept**, then click **Next** to continue to the next step. If the business terms are not acceptable, click **Back** to return to the **Offers** page and find another offer whose business terms are acceptable.
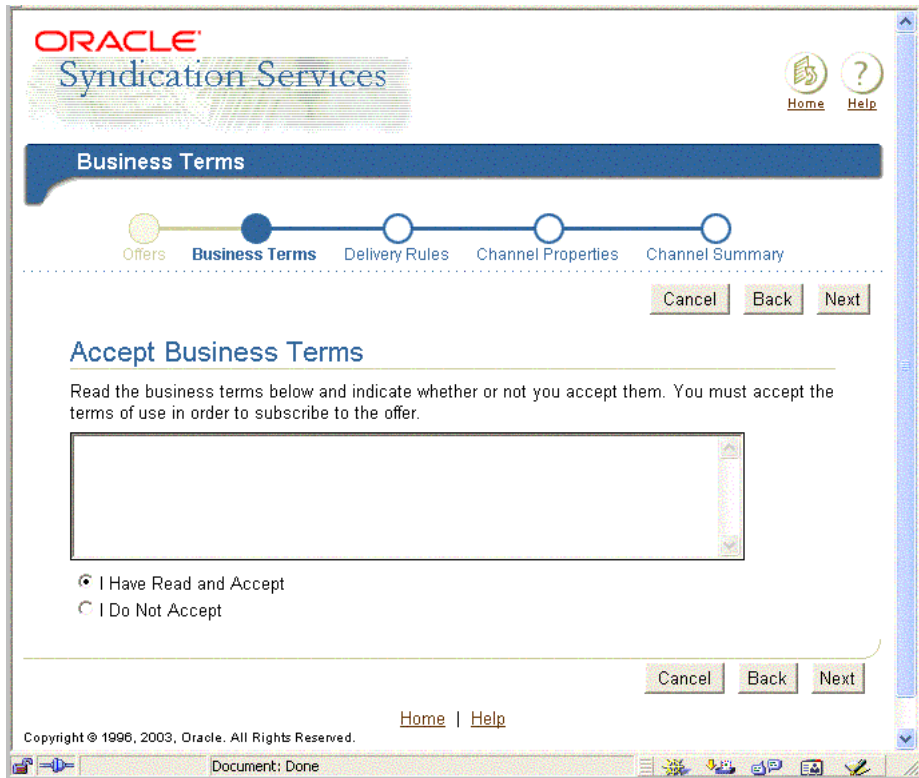
c. At the **Delivery Rules** page, shown as follows (the top half of the page followed by the bottom half of the page), in the **Expiration Policy** section, the expiration priority and expiration date information is displayed. In the **Pull** and **Push** sections, select the pull or push or both delivery rules to be used by OracleAS Syndication Services to deliver this offer to you by clicking its check box, then click **Next** to continue to the next step.
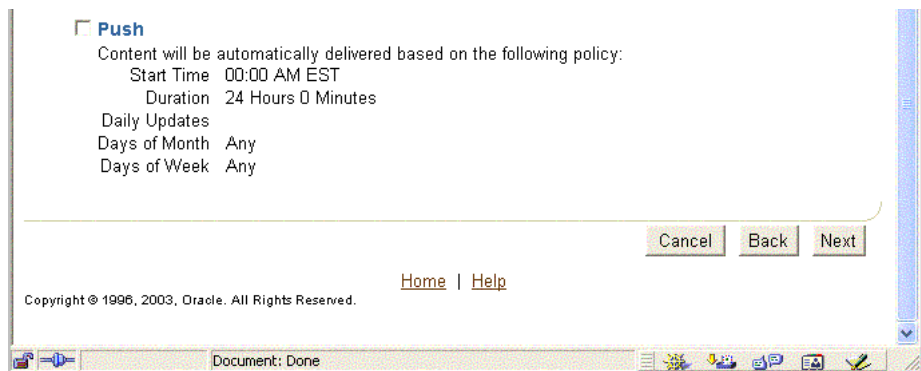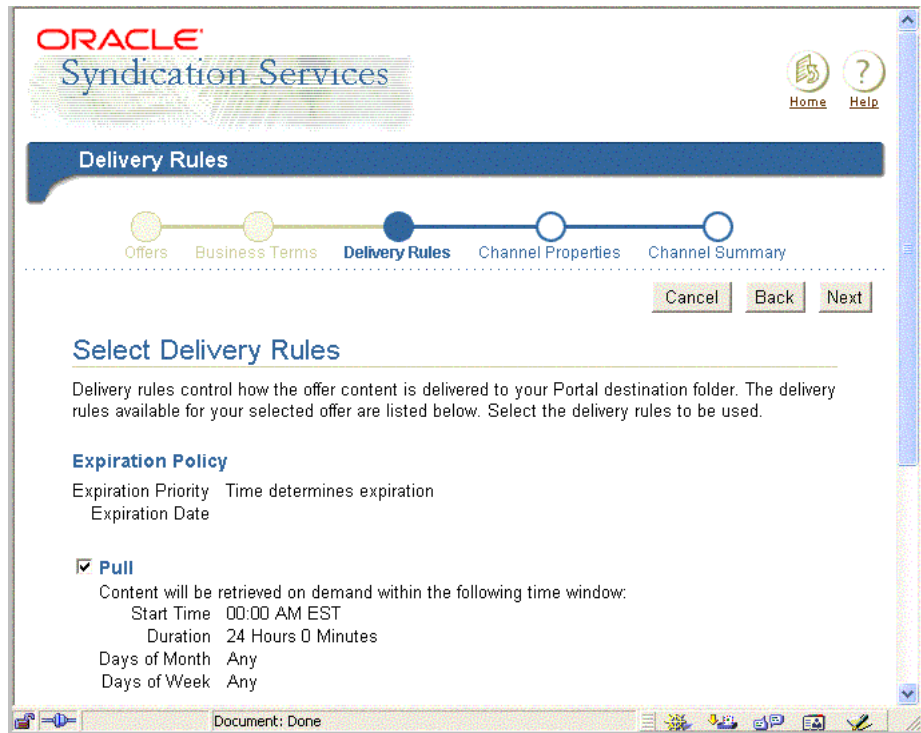
d. At the **Channel Properties** page, shown as follows (the top half of the page followed by the bottom half of the page), in the **Specify Channel Properties** section specify the syndication channel properties, including the syndication channel's name and description in the **Name** and **Description**

fields. Then, in the **Specify Channel Folder** section, specify the destination Portal folder name in the **Destination Folder** field, or click the flashlight icon, browse to the desired Portal folder, and select it; then click **Next** to continue to the next step.
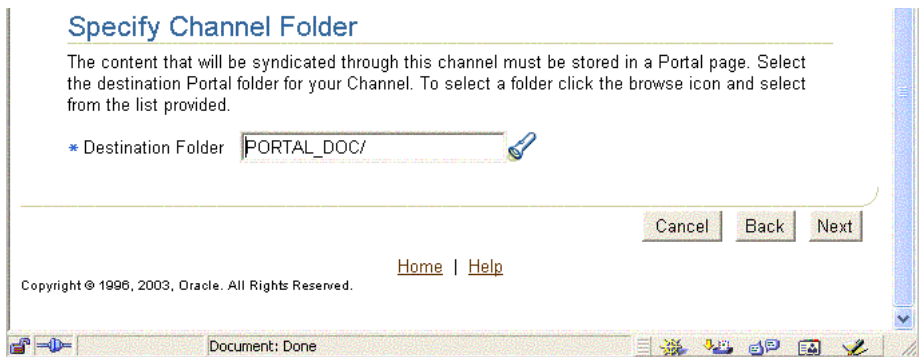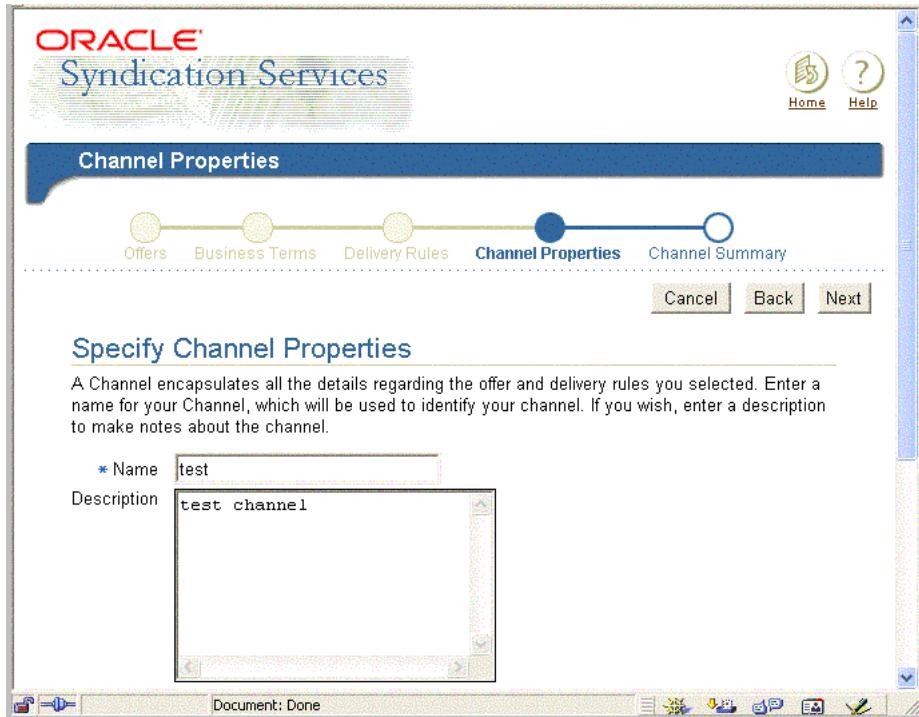
e. At the **Channel Summary** page, shown as follows (the top half of the page followed by the bottom half of the page), review and confirm the channel information. If the information is correct, click **Finish** to complete the syndication channel creation process. You will return to the **Syndication Channel Administration** home page and the newly created channel is listed in the drop down box in the bottom of the **Syndication Channel Administration** portlet.

If the information is not correct, click **Back** to return to the appropriate create syndication channel page where you can make the necessary change or changes, then click **Next** to return to this **Channel Summary** page to review a summary of the syndication channel information again.
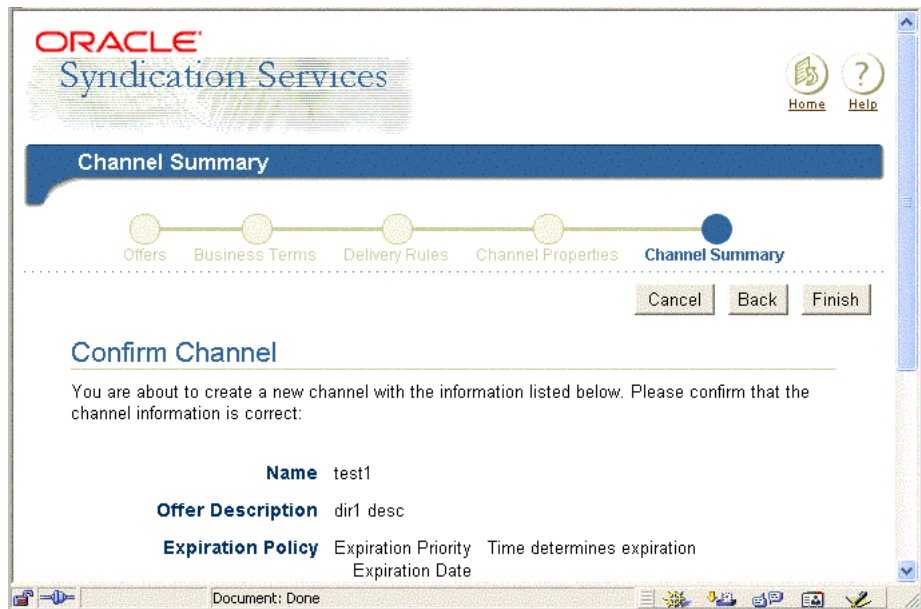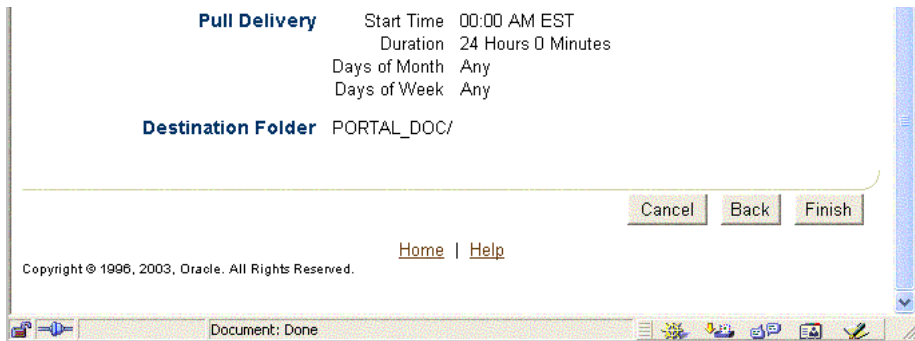
4. To edit the syndication channel properties for an existing syndication channel, at the **Syndication Channel Administration** home page, click the **Properties** tab to display the **Edit Channel:*<channel-name>*** page, shown as follows (the top half of the page followed by the bottom half of the page).

   In the **Channel Properties** section, you can edit the syndication channel name and its description. In the **Destination Folder** section, you can select another Portal destination folder as the destination folder. If no changes are necessary, click **Cancel**; if changes are necessary, make your changes, then click **OK**.
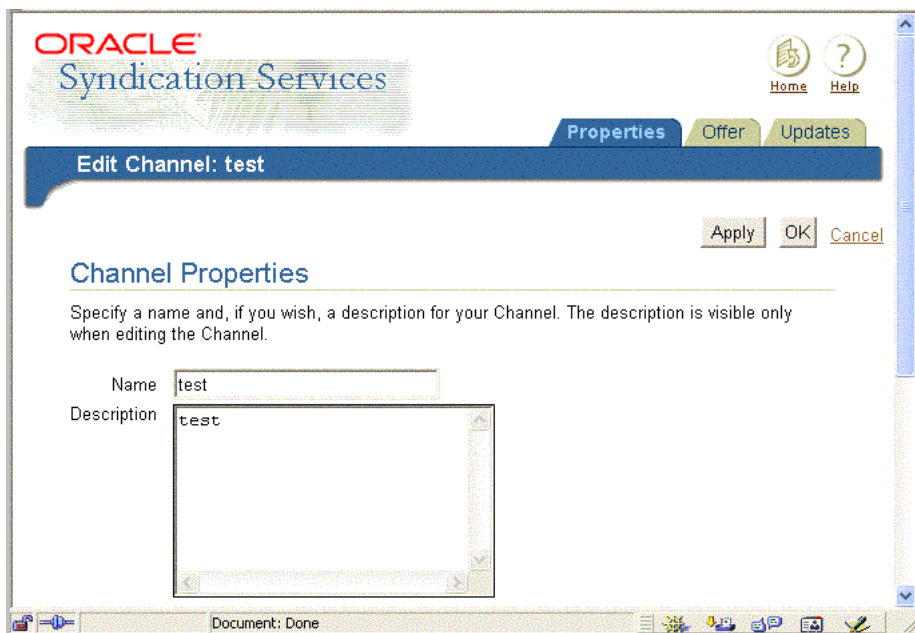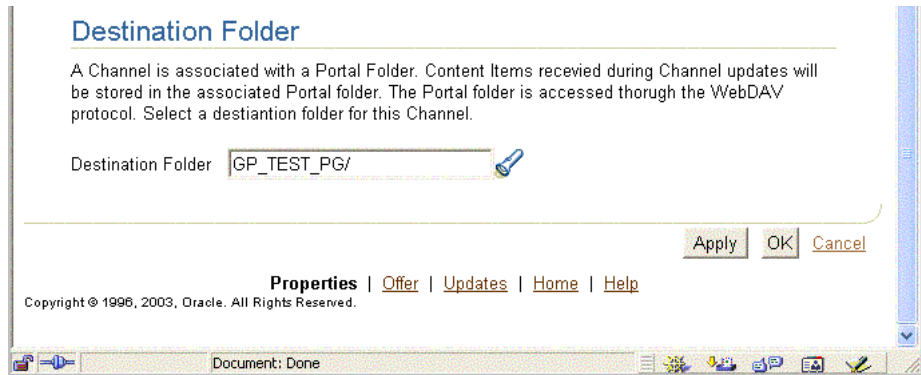
5. To view the offer details, at the **Syndication Channel Administration** home page, click the **Offer** tab to display the **Edit Channel:*<channel-name>*** page, shown as follows (the top half of the page followed by the bottom half of the page).

   The **Edit Channel:*<channel-name>*** page includes the **Offer Details** section, the **Delivery Rules** section, and the **Business Terms** section. Review the summary of properties for this offer configured to use this syndication channel. The subscription ID is the unique identifier assigned when the subscription is created. The subscription ID can be cross-referenced to the list of subscriptions in **Oracle Enterprise Manager Syndication Services Administration** --> **Subscriptions** page. Click **OK** when your review is complete.

6. To view full channel update information for this syndication channel, at the **Syndication Channel Administration** home page, click the **Updates** tab to display the **Edit Channel:<channel-name>** page, shown as follows. This page displays the **ChannelUpdates Log** section and the **Full Channel Update** section.

a. To view the list of updates performed for this syndication channel, browse the list of performed updates that display in the **Channel Updates Log** section.

b. To perform a full syndication channel update of the content available through this syndication channel, click **Perform Full Update**.

> **Note:** There is a known issue in Syndication Services with propagation of deleted items. If a content provider is registered using a content connector that cannot detect deleted items, removal of any item at the content source will never be revealed to the subscriber, and as an effect, such items will be retained on the subscriber site. See Section 4.1.4 "Content Packages Built" in *Oracle Application Server Syndication Services Developer's and Administrator's Guide* for more information.
>
> A second known issue is if you delete an item in the destination folder (that still exists in the source) the item is not re-created during a regular pull or push operation. A **Perform Full Update** operation is required for the item to be created in the destination folder.
>
> A third known issue is that using OracleAS Portal as a content source for a content provider (configured on the portal page using a WebDAV content connector), Portal item attributes are not retained. For example, item level security information, item versioning, expiration changes, deletion of time, and hiding of items do not change for an item after pulling.

c. To confirm the successful delivery of an update, select a report and click **Send Confirmation**. When the update gets confirmed the confirmation state changes to `confirmed`. If the confirmation state says `Not required`, the send confirmation operation is not necessary and will be ignored. If the delivery of a package requiring confirmation encounters an error the report will report the error description. Sending a confirmation for an update report having a code of `Error` will result in the update not getting confirmed and leaving the state as it is. This means that the package was received with an error, so at the next package delivery the same update will be sent again.

d. To view the details of a particular update report, select its radio button, then click **View Details**. The **Edit Channel:*<channel-name>* Updates** page is displayed as follows, showing the **Update Summary** section as the top half of the page followed by the **Update Details** section as the bottom half of the page). When you are through reviewing the contents of this page, click **back to Report list** to return to the **Edit Channel:*<channel-name>* Updates** page.

An update report can be deleted one at a time by clicking **Clear report**, or all reports can be deleted by clicking **Clear all reports**.

Note: Report deletion is only possible if there is no pending confirmation that is related to that report.

This completes the tasks that the Portal administrator can perform using the **Syndication Channel Administration** portlet.

11.4 Advanced Configuration Parameters

This section describes the advanced configuration parameters that can be used to support secure socket layer (SSL) encryption on the channel communication with OracleAS Syndication Services and with the Portal distributed authoring and versioning (DAV) entry point.

Table 11–1 shows the initialization parameters (`<init-params>`) that can be set on the channel Web application's `web.xml` file (`channel/WEB-INF/web.xml` in the syndprovider application on UNIX or `channel\WEB-INF\web.xml` in the syndprovider application on Windows). The value of **Parameter** is the `<param-name>` value.

Table 11–1   Initialization Parameters

Parameter	Description	Sample Value
`oracle.syndicate.ui.provider.ProviderSyndDelegator.walletLocation`	Specifies the location of the file containing all SSL certificates to establish HTTPS connections to DAV entry points or OracleAS Syndication Services.	`/tmp/mycert.db` on UNIX or `\tmp\mycert.db` on Windows
`oracle.syndicate.ui.provider.ProviderSyndDelegator.davPoolSize`	Specifies the size of the DAV context pool.	5
`oracle.syndicate.ui.provider.ProviderSyndDelegator.sycPoolSize`	Specifies the size of the syndicate connection pool.	5
oracle.syndicate.ui.provider.ProviderSyndDelegator.timeout	Specifies the timeout (in milliseconds) for requests sent to OracleAS Syndication Services by the syndication provider. It also controls the timeout when syndication provider writes content into Portal Content Repository.	60000

Table 11–2 shows the parameters that can be set for the listener Web application, the one responsible for incoming push deliveries of content, the `web.xml` file (`listener/WEB-INF/web.xml` in the syndprovider application on UNIX or

listener\WEB-INF\web.xml in the syndprovider application on Windows). The value of **Parameter** is the <param-name> value.

Table 11–2   Listener Web Application Parameters

Parameter	Description	Sample Value
oracle.syndicate.ui.provider.ProviderSyndRequestHandler.walletLocation	Specifies the location of the file containing all SSL certificates used to connect to DAV entry points.	/tmp/mycert.db on UNIX or \tmp\mycert.db on Windows
oracle.syndicate.ui.provider.ProviderSyndRequestHandler.davPoolSize	Specifies the size of the DAV context pool.	5
oracle.syndicate.ui.provider.ProviderSyndRequestHandler.timeout	Specifies the timeout (in milliseconds) to control access to the Portal Content Repository by the syndication provider.	60000

11.5 Syndication Channel Administration Error Messages

> **See:** *Oracle Application Server Portal Error Messages Guide*

12

Using the Federated Portal Adapter

This chapter provides information about the Federated Portal Adapter, previously known as the "PL/SQL HTTP Adapter". It describes how it can be used to share portlets with other OracleAS Portal instances.

This chapter contains the following sections:

- About the Federated Portal Adapter

- Setting Up the Environment to Use the Federated Portal Adapter

- Registering a Provider Using the Federated Portal Adapter

- Writing Custom Portlets Using the Federated Portal Adapter

- Troubleshooting Federated Portal Adapter

12.1 About the Federated Portal Adapter

In this section we will describe the following:

- Overview

- Differences Between Database Providers and Web Providers

- Use of the Federated Portal Adapter

- Security Issues

- Federated Portal Adapter Related Portlet Modifications

12.1.1 Overview

The Federated Portal Adapter is a component of OracleAS Portal that allows OracleAS Portal instances to share their database portlets through the Web portlet

interface. It is a tool that uses SOAP and HTTP to distribute database providers across database servers. The Federated Portal Adapter allows database providers to be accessed as though they were Web providers.

In earlier versions of Oracle Portal all database providers accessed from a portal instance had to be on the same physical database server that contained the portal instance.

In Oracle Portal version 3.0.9, it was possible to distribute database portlets across database servers. To do this the user had to register each portal 'node' with each other which created a database link between the 'nodes'. These portal nodes would not function beyond a firewall. Furthermore the registration of the portal nodes was symmetric, which made the registration of multiple nodes hard to manage

Portal already had the concept of Web providers where the communication between the portal and the provider is done with the open protocols HTTP and SOAP. The PDK-Java services allow users to easily develop providers in Java that receive SOAP messages and respond accordingly.

The Federated Portal Adapter is a module written in the portal instance (in both Java & PL/SQL) that receives the SOAP messages for a Web provider, parses the SOAP and then dispatches the messages to a database provider as PL/SQL procedure calls. In effect, the Federated Portal Adapter makes a database provider behave exactly the same way as a Web provider. This allows users to distribute their database providers across database servers. All remote providers can now be treated as Web providers, hiding their implementation from the user and effectively replacing the distributed Portal installations.

12.1.2 Differences Between Database Providers and Web Providers

The biggest difference between database providers and Web providers is that typically database providers use a portal session within the code, so that as part of the Federated Portal Adapter a portal session is created on the remote portal instance. The SOAP messages were extended to contain enough information to create a session on the remote portal instance, which means that the user in the remote portal must be the same user as in the local portal. For example, if 'UserA' is running in 'PortalA' and is using a provider on 'PortalB' through the Federated Portal Adapter then a session will be created in 'PortalB' for 'UserA'. Typically this means that 'PortalA' and 'PortalB' would share the same Oracle Application Server Single Sign-On, as partner applications. However an alternative arrangement could be that they have separate OracleAS Single Sign-Ons but the OracleAS Single Sign-Ons share the same name server. An example could be two OracleAS Single Sign-Ons sharing the same Oracle Internet Directory instance.

12.1.3 Use of the Federated Portal Adapter

The use of the Federated Portal Adapter can be divided into three categories:

Table 12–1   Use of the Federated Portal Adapter

Category	Description
OracleAS Portal Database Providers	Portal Database Providers created within OracleAS Portal will have the necessary code to be run through the Federated Portal Adapter. This means that applications created containing forms, charts, reports, and so on, can be shown on any other portal instance.
Pages	Pages exposed as portlets can also be run through the Federated Portal Adapter. Regions within pages can contain portlets or items. Using the Federated Portal Adapter these can now be accessed from any portal instance.
User Created Providers	Users may wish to create their own PL/SQL providers. You will be able to expose these providers through the Federated Portal Adapter as long as they are coded in accordance with the guidelines given in this chapter.

12.1.4 Security Issues

The Federated Portal Adapter creates a portal session in the remote portal based on the information passed in an initSession SOAP message. This introduces a security issue since it may be possible to replicate these SOAP messages and create sessions for any user on a portal and then access the portal as that user. To avoid this, an encryption key is shared between the two portals and part of the SOAP message is encrypted using that key. The requested private portal session can only be created if the previously shared key can decrypt it. Otherwise a PUBLIC session is created. The request to display a portlet is made with a Show message that is protected by the encrypted cookie which is created by the initSession SOAP message. The use of an encryption key means that the Federated Portal Adapter can safely trust the incoming SOAP message and create portal sessions in the portal instance without opening the portal to hackers.

> **See Also:** Section 12.2.2, "Federated Portal Adapter User Authentication Using HMAC"

If it is known that the portal instance will only be accessed through the Federated Portal Adapter from other portal instances, then security can be enhanced by

configuring the listener to restrict access from machines other than the known portal instances. This is done by using the 'Allow' directive in the `httpd.conf` file.

12.1.5 Federated Portal Adapter Related Portlet Modifications

It should be noted that database providers written before Oracle Application Server will not work when accessed through the Federated Portal Adapter if one of the following conditions is true:

- The portlet contains relative links.

- The portlet is customizable.

All links within a portlet should be absolute links, that is, `'http://host:port/images/foo.gif'` rather than relative, `'/images/foo.gif'` when using the Federated Portal Adapter. This is because the request is processed by the *Parallel Page Engine* on the local portal instance. Relative links will therefore be interpreted as relative to the local portal and not to the portal containing the portlet.

Customization is an issue because the processing of customization is different between database and Web providers. For Web providers the customization form is submitted to the *Parallel Page Engine* of the local portal, which in turn calls the portlet again and the customizations are saved and the page is redirected appropriately. Since database providers accessed through the Federated Portal Adapter are effectively Web providers then this method of customization should be undertaken for these providers. A public API is provided (WWPRO_API_ADAPTER) to do this.

Portal Database Portlet Providers developed in previous versions of OracleAS Portal will be upgraded automatically to work with the Federated Portal Adapter. Pages exposed as providers can also be accessed through the Federated Portal Adapter.

12.2 Setting Up the Environment to Use the Federated Portal Adapter

To use the Federated Portal Adapter there are a few administrative steps that must be undertaken. These steps are:

- Checking the PlsqlSessionCookieName Value

- Federated Portal Adapter User Authentication Using HMAC

- Setting the Cookie Domain

- Sharing an OracleAS Single Sign-On and an Oracle Internet Directory Server

12.2.1 Checking the PlsqlSessionCookieName Value

DADs must have a unique *PlsqlSessionCookieName* value for all the portals accessed by the Federated Portal Adapter.

For example,

- `portal1` can have the schema name `portal`, the DAD name `portal` and the PlsqlSessionCookieName value `portal1`.

- `portal2` can have the schema name `portal`, the DAD name `portal`, but must have a different PlsqlSessionCookieName value, like `portal2`.

> **Note:** In previous releases of OracleAS Portal, the DAD name had to be the same as the schema name, and the DAD name was always the same as the name of the session cookie created. This is no longer the case. You can now specify the name of the cookie created when portal is accessed by the DAD, and the schema name does not have to be the same as the DAD name.

Oracle Enterprise Manager can be used to update the Session Cookie Name. To do this:

1. Navigate to the Application Server Control.

    Typically, `http://<host>.<domain.com>:1812`. For more information, see Section 7.1, "Using the Application Server Control".

2. Navigate to the Application Server instance where you would like to add the DAD.

3. Select **HTTP Server** from the System Components table.

4. Click **Administration**.

5. Click **PL/SQL Properties**.

6. To edit an existing DAD, click the DAD name in the **DADs** section.

7. Click **Document, Alias and Session** in the navigation area on the left.

8. Enter a new value for **Session Cookie Name** in the page, and click **OK**.

9. Restart the Oracle HTTP Server.

You can also manually change the PlsqlSessionCookieName value in the `dads.conf` file. This file is located under:

ORACLE_HOME/Apache/modplsql/conf/dads.conf

A typical entry in this file looks like this:

```
<Location /pls/portal>
    SetHandler pls_handler
    Order allow,deny
    Allow from All
    AllowOverride None
    PlsqlDatabaseUsername portal
    PlsqlDatabasePassword SomePassword
    PlsqlDatabaseConnectString myhost.domain.com:1521:mySID
    PlsqlDefaultPage portal.home
    PlsqlAuthenticationMode SingleSignOn
    PlsqlSessionCookieName portal
    PlsqlMaxRequestsPerSession 500
    PlsqlDocumentTablename portal.wwdoc_document
    PlsqlDocumentPath docs
    PlsqlDocumentProcedure portal.wwdoc_process.process_download
    PlsqlPathAlias url
    PlsqlPathAliasProcedure portal.wwpth_api_alias.process_download
    PlsqlFetchBufferSize 128
</Location>
```

To edit a DAD entry, change the value of *PlsqlSessionCookieName* to, for example, `portal2`. After saving the file, update the Oracle HTTP Server configuration and restart the middle-tier as follows:

MID_TIER_ORACLE_HOME/dcm/bin/dcmctl updateconfig -ct ohs
MID_TIER_ORACLE_HOME/opmn/bin/opmnctl restartproc type=ohs

> **See Also:** Section 4.5.3, "Configuring a Portal DAD" for instructions on how to configure a DAD, using Application Server Control.

12.2.2 Federated Portal Adapter User Authentication Using HMAC

Federated Portal Adapter functionality will support the registering of remote Database providers between geographically dispersed portals. Database providers are registered as if they were Web providers residing at a special URL on the remote portal.

> **Note:** If you are only rendering public content in the remote portlets, you can ignore this section.

In order that more than just public content can be rendered in the remote portlets we require that in some way we can guarantee that user A on one portal is the same as user A on another portal. This will typically be achieved by using a shared Oracle Application Server Single Sign-On using the *partner application* feature, but may also be achieved with a shared name server (for example, Oracle Internet Directory), synchronized name servers or a manual process.

If this environment can be achieved, then using the *Hash Message Authentication Code* (HMAC) authentication mechanism, private sessions can be initiated on a remote portal to render private content of remote portlets.

Setting the HMAC Keys

If the administrator of portal A wishes to permit users of portal B to create private sessions on portal A, a private 'key' will have to be stored on each portal. This key is used to encode and decode portions of each SOAP request sent between them. If a key is missing or they are different on each portal, only PUBLIC sessions will be created.

A key must be at least 10 characters long, and one administrator should inform the other administrator of its value in a suitably secure way.

SQL scripts are provided to perform the task of maintaining the key store - all are found in the *ORACLE_HOME*/portal/admin/plsql/wwc directory.

Table 12–2   SQL Scripts for Maintaining the Key Store

Script	Description
proadsss.sql	Sets the key at the sending end (Portal instance on which the page with the remote portlets is created).
proadssr.sql	Sets the key at the receiving end (Portal instance on which the portlets are created).
proadsds.sql	Removes the key at the sending end (Portal instance on which the page with the remote portlets is created).
proadsdr.sql	Removes the key at the receiving end (Portal instance on which the portlets are created).

In each case, *sending* and *receiving* refer to the SOAP message.

Example 12–1   Setting the HMAC Keys:

In the example mentioned earlier, portal B is the sender (sending SOAP and show requests) and portal A is the receiver of those requests. The portal Administrator of portal B must connect to SQL*Plus as the portal owner and run:

```
SQL> @proadsss
Enter provider portal PL/SQL Adapter URL:
http://<portalA_hostname>:<port>/adapter/<portalA_DAD>
Enter shared key:<shared key>
exit;
```

The portal Administrator of portal A must connect to SQL*Plus as the portal owner and run:

```
SQL> @proadssr
Enter provider portal PL/SQL Adapter URL:
http://<portalB_hostname>:<port>/adapter/<portalB_DAD>
Enter shared key:<shared key>
exit;
```

If sharing of providers is required both ways, then this will need to be repeated the other way round, possibly with different shared keys. It should also be noted that a portal can expose its providers to several other portal instances (for example, 'Portal A' exposes providers to 'Portal B' and 'Portal C') and separate keys can be set up between each of the portal instances.

12.2.3 Setting the Cookie Domain

Normally cookie domains are restricted to a single machine. This can be widened by running a script on each portal, and then selecting the **Web provider in same cookie domain as the portal** option on provider registration. Once this is done, '*deep link*' functionality can be achieved. This means that when you click a link in a portlet rendered by the Federated Portal Adapter, the browser renders the referred page (typically from the remote portal). The session context that has already been established is also maintained.

Cookies received by a browser, or other HTTP client, are sent to servers if the domain of the cookie matches the server's host name. So cookies with the domain '.co.uk' and 'mycompany.co.uk' will be sent with a request to 'http://mycompany.co.uk/pls/etc/etc'. By default the scope of cookies created by portal is restricted to the host name of the middle-tier machine.

Since communication to the portlets is done in the middle-tier by the *Parallel Page Engine* (PPE) and not the browser, the session cookie for the remote portal will, by default, not be sent to the remote portal when links are followed within the portlet.

This can be solved by widening the scope of the cookies created by portal and making sure that the cookies received by the PPE are sent back to the browser Widening the scope of the cookies created by portal is achieved by running the SQL script `ctxckupd.sql` in the *ORACLE_HOME*/portal/admin/plsql/wwc directory.

For example, there are two portals:

- http://myhost1.mycompany.co.uk:3000/pls/portalA
- http://myhost2.mycompany.co.uk:4000/pls/portalB

and a provider is registered from 'Portal B' on 'Portal A'.

When showing a page on 'Portal A' that contained a portlet from 'Portal B' by default a portal session cookie for 'Portal B' whose domain is 'myhost2.mycompany.co.uk:4000' would be created, and sent to the PPE. If the '*Web provider in same cookie domain as the portal*' property is checked on the provider registration page then this cookie will be sent back to the browser, but the domain of the cookie will then be 'myhost1.mycompany.co.uk:3000' because that is where it is being sent from, because the PPE is at 'myhost1.mycompany.co.uk:3000').

If a link is followed from within the portlet the cookie is not sent with the request, because the domain of the cookie does not match with that of the host of the request.

To solve this, connect to SQL*Plus as the portal owner of each portal and run *ORACLE_HOME*/portal/admin/plsql/wwc/ctxckupd.sql and broaden the scope of the domain's cookies created by OracleAS Portal so each portal is in the same domain. Once this is done, the scope of the cookie domains created by any of the portals will be broad enough to be sent back to the browser. Links within the portlet will then work correctly.

> **See Also:** Section C.5, "Configuring the Portal Session Cookie"

12.2.4 Sharing an OracleAS Single Sign-On and an Oracle Internet Directory Server

The benefits of single sign-on can be maximized, by utilizing a common Identity Management server. Portal session information is passed to the remote portal, which uses the Federated Portal Adapter to create a session. It is recommended that

all portals on which you want to create private sessions, share the same Oracle Internet Directory server and the same OracleAS Single Sign-On.

For example, if a user 'JSMITH' displays a page on one portal and a portlet on that page is being sourced from the Federated Portal Adapter on a remote portal, then a session is created on the remote portal for user 'JSMITH'. If the two portals do not share OracleAS Single Sign-On then 'JSMITH' may be the username for 'John Smith' on one portal and 'Jane Smith' on the other. To avoid this sort of problem, ensure that all the portals participating in the Federated Portal are configured to use a single Oracle Identity Management. They should all use the same OracleAS Single Sign-On for authentication. However, if the portlets being shown are 'public' then there is no need to share the OracleAS Single Sign-On and a public portal session will be created at the remote portal instance.

If you currently have two portals using distinct OracleAS Single Sign-On servers, you may first need to consolidate the OracleAS Single Sign-On servers. To do this, refer to the section titled "Consolidating Multiple Servers" in the *Oracle Application Server Single Sign-On Administrator's Guide*.

Consolidating the servers means that you will be decommissioning one of the servers and identifying the other as the common server for both portals to use. Then you'll need to configure the portal that was configured to use the decommissioned OracleAS Single Sign-On to the consolidated one. In order to do this, you have to run the OracleAS Portal Configuration Assistant in the MIDTIER mode with -mode MIDTIER and -type SSO.

Example 12–2   Sharing an OracleAS Single Sign-On and an Oracle Internet Directory Server

You have two portals, **portal1** on database `portal1DB.domain.com:1521:portal1` on middle-tier `portal1.domain.com:7777` using SSO schema `portal1_sso` and **portal2** on database `portal2DB.domain.com:1521:portal2` on middle-tier `portal2.domain.com:7778` using SSO schema `portal2_sso`. You decide to decommission the SSO server for `portal2` and configure `portal2` to use the SSO server for `portal1`:

```
ptlasst.csh -mode MIDTIER -type SSO -host portal2.domain.com -port 7778 -i
custom -s portal2 -sp portal2 -sdad portal2 -c portal2DB.domain.com:1521:portal2
-o portal1_sso -op portal1_sso -odad portal1_sso -sso_c
portal1DB.domain.com:1521:portal1 -pa portal1_sso_pa -pap portal1_sso_pa -ps
portal1_sso_pp -pp portal1_sso_pp
```

> **Note:** Refer to Section B.2.2.2, "SSO Type" for information on how to use OPCA with -mode MIDTIER and -type SSO.

12.3 Registering a Provider Using the Federated Portal Adapter

Registering a provider through the Federated Portal Adapter is like registering any Web provider. You must perform the following steps:

1. On the first page of the **Register Provider** screen enter the **Name**, **Display Name**, **Timeout**, and **Timeout Message** as you would normally. Make sure the **Implementation Style** is set to **Web**. Although the provider is actually written in PL/SQL, all communication to it is as a Web provider and not a database provider so it is important to set the **Implementation Style** to **Web**.

2. On the second page enter the URL of the adapter service. The syntax for the URL should be:

   `http://host:port/adapter/dad/schema`

   If the DAD and the schema are the same you can just use:

   `http://host:port/adapter/dad`

   where the host, port, dad and schema locate the remote portal instance. You can verify that this is the correct URL by pasting it into a browser.

   If the URL is correct you should get to a page with the message "Congratulations - you got to the adapter test page"

3. Select the **Web provider in same cookie domain as the portal** option. This will ensure that cookies generated from the provider will be sent back to the browser. Note that it may be necessary to broaden the scope of the cookies created by portal as described earlier.

4. Enter the 'Service Id'. This should be in the form 'urn:<provider name>'. Where <provider name> is the name of the provider on the remote portal instance, this is case sensitive and will be upper case. This is the information that the Federated Portal Adapter uses to locate the specific provider at the remote portal.

   Note that for page groups exposed as providers, the name of the provider will be something like 'MYPAGE970D272EBE9D2D0FE034080020F7DA4B' it is important that you specify this 'Name' rather than the 'Display Name'. The name and display name can be accessed from the **Remote Providers** portlet,

available in the **Portlets** sub-tab under the **Administer** tab in Portal. Clicking the **Browse Providers** icon displays the names of all the providers.

5. In the **User/Session Information** section, select the **User** radio button and set the **Login Frequency** to be **Once Per User Session**. These settings make sure that information is sent with the request to allow a portal session to be created on the remote portal instance.

12.4 Writing Custom Portlets Using the Federated Portal Adapter

There are two main areas of code that need special attention when writing database providers that are accessed through the Federated Portal Adapter. They are:

- Relative Links
- Customization

12.4.1 Relative Links

Any links within portlets that are accessed through the Federated Portal Adapter should absolute rather than relative. If links are relative then they will not work since they will be relative to the local middle-tier rather than the remote middle-tier. For example, links should be of the form 'http://myhost.mycompany/etc/etc' rather than '/etc/etc'.

12.4.2 Customization

The way customizations work when accessing portlets through the Federated Portal Adapter is now very similar to the method used by JPDK portlets. There are two main areas of the portlet code that need to be changed to make customization work through the Federated Portal Adapter:

- The show call of the portlet needs additional logic to show the portlet in *edit_defaults* mode, or, if the parameter '*p_mode*' is null, in *customize* mode. If the '*p_mode*' is 'OK', 'APPLY' or 'RESET', then the customizations should be saved as appropriate.

- The <FORM> HTML tags generated for the customize page should be created using the procedure *wwpro_api_adapter.open_form*. This will ensure that the action for the form is correct, and that the correct parameters are passed upon page submission. The sequence of events when submitting the customization form is:

1. The page submits to the 'local' PPE. There are several standard parameters that need to be sent with this submission (for example, *_providerid*, *_dad*, *p_action*, and so on) as well as the parameters that are being customized. The procedure *wwpro_api_adapter.open_form* is supplied to make the generation of this submission as simple as possible.

2. The PPE then shows the customization page again. However the '*p_action*' parameter will now be set so that during the *show_portlet* call of the portlet it will be one of the following settings:

   '**OK**' - In this case the customizations should be saved and then there should be a re-direct to the page containing the portlets.

   '**APPLY**' - In this case the customizations should be saved and the customization page is shown.

   '**RESET**' - In this case the default values for parameters are queried and the customization page is shown.

More On
Portal Center

The *database services provider* is a sample provider in the Oracle Application Server Portal Developer Kit (PDK) that works with the Federated Portal Adapter. for more information, see the Portal Developer Kit on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

12.5 Troubleshooting Federated Portal Adapter

For issues related to the Federated Portal Adapter, see Section 13.9, "Troubleshooting Federated Portal Adapter".

13

Troubleshooting OracleAS Portal

This chapter shows you how to use various tools to diagnose problems, and lists possible causes and solutions to errors that you may encounter while installing or using OracleAS Portal.

More On
Portal Center

You can find the most up-to-date troubleshooting information on Portal Center, http://portalcenter.oracle.com. Click the **Search** icon in the upper right corner of any Portal Center page.

This chapter contains the following sections:

- Common Issues

- Miscellaneous Issues

- Verifying the Portal Dependency Settings File

- Diagnosing OracleAS Portal Problems

- Using the OracleAS Portal Diagnostics Assistant

- Using Application Server Control Log Viewer

- Troubleshooting Export and Import

- Troubleshooting Search Functionality

- Troubleshooting Federated Portal Adapter

- OracleAS Portal Errors

13.1 Common Issues

This section contains some troubleshooting information for common issues that you may experience while running OracleAS Portal. These are:

- OracleAS Portal is Not Accessible
- OracleAS Single Sign-On is Not Accessible
- Issues Creating Category/Perspective Pages
- Multi-language Support for Help

13.1.1 OracleAS Portal is Not Accessible

If you cannot access your OracleAS Portal instance through the internet/intranet, follow these steps to help diagnose the cause of the problem:

1. **Display the portal's target page in the Oracle Enterprise Manager Application Server Control.**

   See Section 7.1, "Using the Application Server Control".

2. **Check to see if Web Cache is up.**

   The Web Cache status is displayed in the portal's Component Status table.

   - If '**Up**', continue to next step.
   - If '**Down**', start Web Cache using the Application Server Control, or the command line.

     To access Web Cache monitoring and administration pages in the Application Server Control, click the Web Cache link in the:

        - Portal's Component Status table, or

        - Application Server's Component table.

     If Web Cache starts successfully, check to see if your portal is now accessible.

     If Web Cache fails to start, investigate Web Cache error log files and try to determine the problem. See Section 13.6, "Using Application Server Control Log Viewer". If you are not using Log Viewer, check the relevant error log files in *ORACLE_HOME*/opmn/logs .

3. **Check to see if the HTTP Server is up.**

   The HTTP Server status is displayed in the portal's Component Status table.

   - If '**Up**', continue to next step.
   - If '**Down**', start HTTP Server using the Application Server Control, or the command line.

To access HTTP Server monitoring and administration pages in the Application Server Control, click the HTTP Server link in the:

- Portal's Component Status table, or

- Application Server's Component table.

If HTTP Server starts successfully, check to see if your portal is now accessible.

If HTTP Server fails to start, investigate HTTP Server error log files and try to determine the problem. See Section 13.6, "Using Application Server Control Log Viewer". If you are not using Log Viewer, check the relevant error log files in the following directories:

- *ORACLE_HOME*/opmn/logs

- *ORACLE_HOME*/Apache/Apache/logs/error_log

4. **Check the status and configuration of the portal DAD.**

Check the DAD status using the DADs table, displayed on the mod_plsql Services page. Click the **mod_plsql Services** link in the portal's Component Status table to access this page. See also Section 4.5.3, "Configuring a Portal DAD".

- If ′**Up**′, continue to next step.

- If ′**Down**′, click the name of the DAD in the DADs table and verify that all properties are set correctly. Save any changes and restart HTTP Server for any change to take effect.

  Check to see if your portal is now accessible.

5. **Check to see if the portal's Metadata Repository database is running.**

The status is displayed on the portal's target page in the Enterprise Manager Application Server Control. Look under the section 'OracleAS Metadata Repository Used By Portal'.

- If ′**Up**′, continue to next step.

- If ′**Down**′, start the database using the Application Server Control (if this functionality is available), or use SQL*Plus.

  If the database starts successfully, check to see if your portal is now accessible.

  If the database fails to start, investigate further to establish what is wrong with this component.

6. **Check to see if the OC4J_Portal service is up.**

   The OC4J_Portal status is displayed in the portal's Component Status table.

   - If '**Up**', continue to next step.

   - If '**Down**', start OC4J_Portal using the Application Server Control, or the command line.

     To access OC4J_Portal monitoring and administration pages in the Application Server Control, click the OC4J_Portal link in the:

     - Parallel Page Engine Services page (available from the portal's Component Status table), or

     - Application Server's Component table.

     If OC4J_Portal starts successfully, check to see if your portal is now accessible. If you are not using Log Viewer, check the relevant error log files in *ORACLE_HOME*/opmn/logs.

     If OC4J_Portal fails to start, investigate OC4J_Portal error log files and try to determine the problem. See Section 13.6, "Using Application Server Control Log Viewer".

7. **Run the OracleAS Portal Diagnostic Assistant.**

   You can diagnose portal-related issues by reviewing the report generated from the OracleAS Portal Diagnostic Assistant. See also Section 13.5, "Using the OracleAS Portal Diagnostics Assistant".

8. **Contact Oracle Support.**

   If you are unable to establish why your portal is not accessible, contact Oracle Support. To help Oracle Support troubleshoot the problem, provide the following information:

   - ZIP file generated by the OracleAS Portal Diagnostic Assistant.

   - Details of any command line scripts you have run (for example, ptlasst.csh, orasso.cfg, ossoref.jar, and so on) including the full parameters used.

   - A rough network diagram, showing how your Oracle Application Server components are configured.

13.1.2 OracleAS Single Sign-On is Not Accessible

If the OracleAS Single Sign-On (SSO) is not accessible you cannot login to OracleAS Portal. Follow these steps to help diagnose the cause of this problem:

1. **Display the SSO's target page in the Oracle Enterprise Manager Application Server Control.**

   See Section 7.1, "Using the Application Server Control".

2. **Check to see if the HTTP Server is up.**

   Click the HTTP Server link, displayed in the Related Links section.

   - If '**Up**', continue to next step.

   - If '**Down**', start HTTP Server using the Application Server Control, or the command line.

     To access HTTP Server monitoring and administration pages in the Application Server Control, click the HTTP Server link in the:

       - OracleAS Single Sign-On target page, or

       - Application Server's Component table.

     If HTTP Server starts successfully, check to see if your OracleAS Single Sign-On is now accessible.

     If HTTP Server fails to start, investigate HTTP Server error log files and try to determine the problem. See Section 13.6, "Using Application Server Control Log Viewer". If you are not using Log Viewer, check the relevant error log files in the following directories:

       - *ORACLE_HOME*/opmn/logs

       - *ORACLE_HOME*/Apache/Apache/logs/error_log

3. **Check the status and configuration of the SSO DAD.**

   Check the DAD status using the DADs table, displayed on the mod_plsql Services page. Click the **mod_plsql Services** link in the portal's Component Status table to access this page. See also Section 4.5.3, "Configuring a Portal DAD".

   - If '**Up**', continue to next step.

   - If '**Down**', click the name of the DAD in the DADs table and verify that all properties are set correctly. Save any changes and restart HTTP Server for any change to take effect.

Check to see if your OracleAS Single Sign-On is now accessible.

4. Check to see if the database containing the SSO schema is running.

Database information is displayed on the OracleAS Single Sign-On's target page in the Enterprise Manager Application Server Control. Drill down for further information.

- If '**Up**', continue to next step.

- If '**Down**', start the database using the Application Server Control (if this functionality is available), or use SQL*Plus.

  If the database starts successfully, check to see if your OracleAS Single Sign-On is now accessible.

  If the database fails to start, investigate further to establish what is wrong with this component.

5. Check to see if the OC4J_Security service is up.

The OC4J_Security status is displayed in Application Server's page. Alternatively, you can establish its status using the following command line:

```
ORACLE_HOME/dcm/bin/dcmctl getstate
```

> **See Also:** *Oracle Application Server 10g Administrator's Guide* for more information on the Distributed Configuration Management (DCM) utility, dcmctl.

- If '**Up**', continue to next step.

- If '**Down**', start OC4J_Security using the Application Server Control, or the command line.

  To access OC4J_Security monitoring and administration pages in the Application Server Control, click the OC4J_Security link in the Application Server's Component table.

  If OC4J_Security starts successfully, check to see if your OracleAS Single Sign-On is now accessible.

  If OC4J_Security fails to start, investigate OC4J_Security error log files and try to determine the problem. See Section 13.6, "Using Application Server Control Log Viewer". If you are not using Log Viewer, check the relevant error log files in *ORACLE_HOME*/opmn/logs.

6. **Check to see if the Oracle Internet Directory service is up.**

   The Oracle Internet Directory status is displayed in Application Server's page.

   - If '**Up**', continue to next step.

   - If '**Down**', start Oracle Internet Directory using the Application Server Control, or the command line.

     To access Oracle Internet Directory monitoring and administration pages in the Application Server Control, click the Oracle Internet Directory link in the Application Server's Component table.

     If Oracle Internet Directory starts successfully, check to see if your OracleAS Single Sign-On is now accessible.

     If Oracle Internet Directory fails to start, investigate the Oracle Internet Directory error log files and try to determine the problem. See Section 13.6, "Using Application Server Control Log Viewer".

7. **Run the OracleAS Portal Diagnostic Assistant.**

   You can diagnose OracleAS Single Sign-On and portal-related issues by reviewing the report generated from the OracleAS Portal Diagnostic Assistant. See also Section 13.5, "Using the OracleAS Portal Diagnostics Assistant".

8. **Contact Oracle Support.**

   If you are unable to establish why you cannot login to OracleAS Portal, contact Oracle Support. To help Oracle Support troubleshoot the problem, provide the following information:

   - ZIP file generated by the OracleAS Portal Diagnostic Assistant.

   - Details of any command line scripts you have run (for example, `ptlasst.csh`, `orasso.cfg`, `ossoref.jar`, and so on) including the full parameters used.

   - A rough network diagram, showing how your Oracle Application Server components are configured.

13.1.3 Issues Creating Category/Perspective Pages

When you create a category in a page group, a category page is created based on the category template. Similarly, when you create a perspective, a perspective page is created based on the perspective template.

If changes are made to these underlying category/perspective templates, you may see one of these messages when you create a new category/perspective:

- **32022:catpagecreationerror** The category has been created but it was not possible to place the search portlets onto the category page. The category page will not show the items or pages in the category.

- **32023:persppagecreationerror** The perspective has been created but it was not possible to place the search portlets onto the perspective page. The perspective page will not show the items or pages in the perspective.

If either of these errors is displayed, you must first delete the current category/perspective template and then run scripts to:

- Replace the current category/perspective template with the original, shipped version.

- Re-create category/perspective pages that are based on current template. You can do this across all page groups, or for specific page groups.

This ensures that all new category/perspective pages are created without errors and that all existing category/perspective pages display their associated items and pages as expected.

The scripts required are available at:

```
ORACLE_HOME/portal/admin/plsql/wws/pstdefin.sql
ORACLE_HOME/portal/admin/plsql/wws/pstpgshw.sql
ORACLE_HOME/portal/admin/plsql/wws/pstundef.sql
ORACLE_HOME/portal/admin/plsql/wws/pstpgcre.sql
ORACLE_HOME/portal/admin/plsql/wws/pstprcpg.sql
```

To run these scripts:

1. Delete the current category or perspective templates.

2. Connect to OracleAS Portal using SQL*Plus as the Portal schema user.

3. Configure the `pstdefin.sql` file with:

   - Page group information. You can re-create the pages in a single page group, several page groups or all page groups.

   - Page information. You can re-create category pages only, perspective pages only, or both.

   Descriptions for these settings are in the file `pstdefin.sql`.

4. If necessary, use the script `pstpgshw.sql` to retrieve information from OracleAS Portal to configure the `pstdefin.sql` file.

5. Run the script `pstpgcre.sql` to apply the changes. For example:

```
SQL> @pstpgcre.sql
```

13.1.4 Multi-language Support for Help

In OracleAS Portal, there is multi-language support for the online help. However, only context sensitive Help topics are translated.

13.2 Miscellaneous Issues

This section contains some troubleshooting information for less common issues that you may experience while running OracleAS Portal. These are:

- Remote Web Providers Time Out in a Dynamic DNS Environment
- Memory Intense Operations Cause Problems

13.2.1 Remote Web Providers Time Out in a Dynamic DNS Environment

A remote Web provider that is located on a different machine from the OracleAS Portal middle-tier, works when the OC4J_Portal service is first started, but stops working after some time. After a long timeout, the message `Error: the portlet could not be contacted` is shown in the place of each portlet from the same provider. You can also find portlet timeout errors in the OC4J_Portal `application.log`. On restarting OC4J_Portal, the Web provider works again, but only for a limited period of time.

The possible cause for this problem can be that the Web provider is using dynamic DNS (DDNS) for its *Domain Name to IP Address* mapping. This means that the IP address that the Web provider's domain name resolves to, changes over time. Java's default caching policy caches IP addresses forever, once it has resolved them, which means it keeps using an outdated IP address of the Web provider if the IP address of the Web provider has changed because of using DDNS.

To resolve this problem, you need to perform additional configuration in OC4J_Portal to prevent remote Web providers from timing out. You must change the **sun.net.inetaddr.ttl** system property for OC4J_Portal. On JDK 1.3 and later, the **sun.net.inetaddr.ttl** system property can be used to specify the "time to live" (TTL) in seconds for cached IP addresses.

> **Note:** It is important that this system property is passed as a command line option to Oracle Application Server Containers for J2EE (OC4J). Setting the property in `oc4j.properties` will not help because the system property is read first before OC4J reads this file. Therefore, it is best to modify the `<java-option>` line in the OC4J_Portal section of *ORACLE_HOME*`/opmn/conf/opmn.xml`.

Usage Example

1. Edit `opmn.xml` as follows:

```
<java-option value="-server -Xincgc -Xnoclassgc -Xms256m -Xmx512m
-Dsun.net.inetaddr.ttl=120"/>
```

2. Shut down **opmn** and all its sub-processes and restart it for the latest configuration changes to take effect.

   To do this, run the following commands:

```
ORACLE_HOME/opmn/bin/opmnctl stopall
ORACLE_HOME/opmn/bin/opmnctl startall
```

13.2.2 Memory Intense Operations Cause Problems

By default, the *shared_pool_size* value in Oracle Application Server is 32 MB. This can cause problems if you are performing memory intense operations such as:

- Export/Import
- Creating Portal Forms/Reports

The typical error you will see is "**ORA-04031: unable to allocate 30192 bytes of shared memory.**" To facilitate memory intense operations, you must increase the value of the *shared_pool_size* parameter.

To change the value of the *shared_pool_size* parameter:

1. Edit the *shared_pool_size* parameter in the `init.ora` file for the database instance. The `init.ora` file can be found in your database's *ORACLE_HOME*.

> **Note:** This can be done only after the Infrastructure database is installed.

2. Change the value to **64**.

3. Restart the database for the changes to reflect.

13.3 Verifying the Portal Dependency Settings File

When troubleshooting OracleAS Portal, one of the first things to do is to review the contents of the Portal Dependency Settings file `iasconfig.xml`. This file stores configuration data from all the dependent components in a central place and the content of the file is updated when there are configuration changes. Therefore, the file should reflect the current configuration of OracleAS Portal with OracleAS Web Cache, Oracle Internet Directory, and Oracle Enterprise Manager. If the file does not accurately reflect your configuration settings, you must update the file and run the Portal Dependency Settings tool `ptlconfig` to update the Oracle Application Server Metadata Repository.

If you make configuration changes using the OracleAS Portal Configuration Assistant (OPCA) in MIDTIER mode (using the WEBCACHE, OHS, or OID type), the `iasconfig.xml` file is not updated to reflect these changes. This can cause your site to be misconfigured, and is therefore not recommended. Instead, the Portal Dependency Settings file and tool should be used to update the configuration, whenever possible. Refer to Appendix A, "Using the Portal Dependency Settings File" for more information about the Portal Dependency Settings file, and examples of the `iasconfig.xml` file.

13.4 Diagnosing OracleAS Portal Problems

OracleAS Portal consists of middle and database tiers each of which consist of numerous components. Not only can components be distributed across many machines, but they may also handle a large number of requests simultaneously.

To facilitate diagnosis, components can record information relating to the requests they receive, in log files. This section details how to configure and use various log files to diagnose problems. We will also see how an individual request can be traced from start to finish, using the Execution Context Identifier (ECID).

Execution Context Identifier

Because OracleAS Portal can satisfy a large number of requests simultaneously, tracing a single request through the various OracleAS Portal components can be difficult, as information relating to these requests is intermingled.

OracleAS Portal makes use of an ECID, a unique number that is assigned to a request and attached to information recorded for that request. As a request is passed from one component to another, the ECID can be incremented to form a sequence. This means that an individual request can be tracked through any number of components by following this ECID sequence.

An ECID is generated by the first Oracle Application Server component to receive a request without an ECID. We can observe this generation and propagation in Figure 13–1, where a solid arrow depicts a request with an ECID.

Figure 13–1   Request Flow with ECID Generation and Propagation
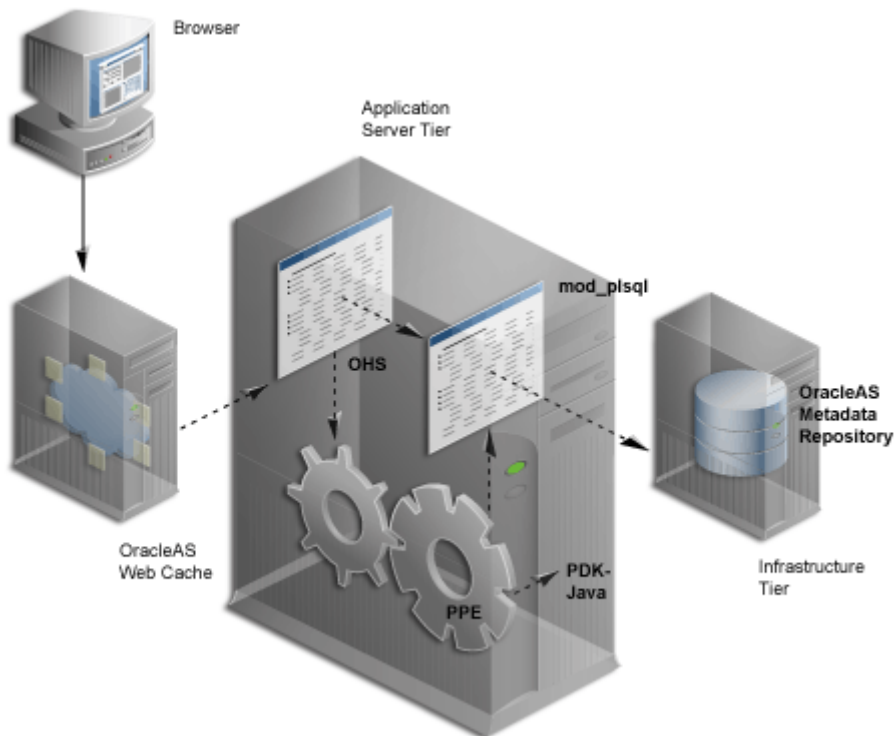
ECID generation is present in Web Cache, Oracle HTTP Server (OHS) and the Parallel Page Engine (PPE). An ECID is only generated if not already present.

Enabling ECID Logging

Oracle Application Server Containers for J2EE (OC4J) can include the ECID with each log entry it writes and this can be useful for debugging purposes. For more information about ECIDs and how they can help you to correlate messages from application server components, refer to the *Oracle Application Server 10g Administrator's Guide*.

If you want to log ECID information in OC4J logs, edit the file `opmn.xml` as follows:

1. In Oracle Enterprise Manager Application Server Control, navigate to the middle-tier Oracle Application Server target home page.

2. Click the **Process Management** link, to see the file `opmn.xml`.

3. Locate the "OC4J_Portal" entry.

4. Add the entry "-Doracle.dms.transtrace.ecidenabled=true" to the "java-options" property in the "start-parameters" category.

   Here is an example:

```
<process-type id="OC4J_Portal" module-id="OC4J">
   <environment>
      <variable id="DISPLAY" value="localhost:0"/>
      <variable id="LD_LIBRARY_PATH" value="/export/home/ias/pwhome/lib"/>
   </environment>
   <module-data>
      <category id="start-parameters">
      <data id="java-options" value="-server
      -Djava.security.policy=/export/home/ias/pwhome/j2ee/OC4J_Portal/conf
      ig/java2.policy -Djava.awt.headless=true
      -Doracle.dms.transtrace.ecidenabled=true -Xmx256m "/>
      <data id="oc4j-options" value="-properties"/>
      </category>
      <category id="stop-parameters">
      <data id="java-options"
      value="-Djava.security.policy=/export/home/ias/pwhome/j2ee/OC4J_Port
      al/config/java2.policy -Djava.awt.headless=true"/>
      </category>
   </module-data>
   <start timeout="900" retry="2"/>
   <stop timeout="120"/>
   <restart timeout="720" retry="2"/>
   <port id="ajp" range="3301-3400"/>
   <port id="rmi" range="3201-3300"/>
```

```
        <port id="jms" range="3701-3800"/>
        <process-set id="default_island" numprocs="1"/>
    </process-type>
```

5. Click **Apply**.

6. Navigate back to the Oracle Application Server target home page

7. Select the **OC4J_Portal** target check box.

8. Click the **Restart** button.

13.4.1 Components and Their Diagnostic Output

The various OracleAS Portal components can have their diagnostic output configured. The following are the components:

- Java Portal Developers Kit

- mod_plsql

- Parallel Page Engine

- Oracle Application Server Portal Developer Kit

- OracleAS Metadata Repository

- OracleAS Web Cache

13.4.1.1 Java Portal Developers Kit

The Java Portal Developers Kit (JPDK) provides a framework for the construction of Java-based portlets and portlet providers. A Java-based provider or Web provider is one that is written as a Web application. The JPDK includes a logging mechanism that is controlled based on each *Provider Adapter*.

The acceptable logging level values range from **1** to **7** and build incrementally. For example, at logging level **3**, the output for logging levels **1** and **2** are also recorded.

Table 13–1   Logging Levels

Logging Level	Description
1	Configuration
2	Severe Errors
3	Warnings
4	Exceptions

Table 13–1   (Cont.) Logging Levels

Logging Level	Description
5	Performance
6	Information
7	Debug

13.4.1.1.1   JPDK Log File Contents

A Provider Adapter's diagnostic information is recorded in the servlet context log file named `application.log`.

There are two types of JPDK messages:

- Standard JPDK Messages
- Performance JPDK Messages

Standard JPDK Messages

Here is an example of a standard JPDK message that you might find in a Provider Adapter's `application.log` file:

```
03/12/31 02:58:59 jpdk: [instance=1926_EXPIRESSAMPLE_886361,
id=1024597399815ApplicationServerThread-12,4] Beginning rendering of
portlet: 1926_EXPIRESSAMPLE_886361
```

Its content is as follows:

03/12/31 02:58:59 - Date and time

jpdk: - Web application

id=1024597399815ApplicationServerThread-12,4: - ECID, sequence number

instance=1926_EXPIRESSAMPLE_886361: - Portlet instance identifier

Beginning rendering of portlet: 1926_EXPIRESSAMPLE_886361: - Message

The portlet instance identifier, identifies a specific portlet instance on a specific page and can be broken down as follows:

1926: - Internal sequence number

EXPIRESSAMPLE: - Portlet name

886361: - Provider identifier

Additional details relating to some of these values are shown in Table 13–2.

Table 13–2   JPDK Standard Message Attributes

| Value	Detail
ECID	Some messages carry null ECID and portlet instance identifier values. These are typically SOAP messages from the repository.
Portlet instance identifier	Some messages carry null ECID and portlet instance identifier values. These are typically SOAP messages from the repository. The portlet instance identifier is **null** in this case, because the message does not relate to a particular portlet instance.

13.4.1.2 mod_plsql

mod_plsql is an Oracle HTTP Server module that enables a user to invoke PL/SQL applications over HTTP. Because mod_plsql is an Oracle HTTP Server module, its logging is performed through the Oracle HTTP Server.

Logging is controlled by the **LogLevel** parameter found in the configuration file httpd.conf, usually located at:

ORACLE_HOME/Apache/Apache/conf

The values for **LogLevel** are:

- emerg
- alert
- crit
- error
- warn
- notice
- info
- debug

The values build incrementally. For example, if **LogLevel** is set to **notice** then **notice**, **warn**, **error**, **crit**, **alert** and **emerg** messages are recorded.

If the value of **LogLevel** is altered, the Oracle HTTP Server must be restarted for this change to take effect.

13.4.1.2.1 mod_plsql Log File Contents

The location of mod_plsql's diagnostic information is dictated by the Oracle HTTP Server parameter **ErrorLog** found in the file `httpd.conf`. While this parameter is called **ErrorLog,** the file it describes can contain more than just error messages. A typical value for the Oracle HTTP Server parameter **ErrorLog** is:

```
ORACLE_HOME/Apache/Apache/logs/error_log
```

Two types of mod_plsql messages appear in the Oracle HTTP Server error log:

- Standard mod_plsql Messages
- Performance mod_plsql Messages

Standard mod_plsql Messages

Here is an example of a standard mod_plsql message found in the Oracle HTTP Server error log:

```
[Thu Aug 22 08:34:20 2002] [warn] mod_plsql: 'PlsqlCacheCleanupSize' is
deprecated.
```

The content is as follows:

Thu Aug 22 08:34:20 2002: - Date and time

warn: - Message level

mod_plsql: - Indicates this message comes from mod_plsql

'PlsqlCacheCleanupSize' is deprecated.: - Message text

13.4.1.3 Parallel Page Engine

The Parallel Page Engine (PPE) is a shared server process servlet that accepts data representing a page layout and then converts this data into a page containing portlets.

PPE logging can be controlled at the servlet and request level. If a request logging level is not specified then the servlet level is used for the request. If both servlet and request logging levels are specified, then the higher of the two is used for the request.

Servlet Level Logging

PPE servlet level logging is controlled by the **logmode** servlet initialization argument. The values for **logmode** are:

- none
- perf
- debug
- request
- content
- parsing
- all

The values build incrementally. For example, if **logmode** is set to **content** then **content**, **request**, **debug** and **perf** messages are also recorded. The default value is **none**. A value of **all** allows every logging message to be included.

As the PPE is a servlet, configuration varies with the Servlet container on which it is deployed. Under OracleAS Portal, the servlet container is OC4J and **logmode** can be found in the portal's web.xml file. This XML file contains properties for more than just the PPE and, consequently, **logmode** can appear more than once. It is important to modify the correct **logmode** value:

```
<init-param>
     <param-name>logmode</param-name>
     <param-value>perf</param-value>
</init-param>
```

This can be found inside the **page** servlet clause:

```
<servlet>
     <servlet-name>page</servlet-name>
     <servlet-class>oracle.webdb.page.ParallelServlet</servlet-class>
     .
     .
     <init-param>
          <param-name>logmode</param-name>
          <param-value>perf</param-value>
     </init-param>
     .
     .       .
</servlet>
```

If the value of **logmode** is altered, OC4J must be restarted for this change to take effect. The web.xml file can be found at:

```
ORACLE_HOME/j2ee/OC4J_Portal/applications/portal/portal/WEB-INF
```

Request Level Logging

PPE request level logging is controlled by the **_debug** URL parameter. For example, to specify request level logging for the following URL:

```
http://myserver.myplace.com:3000/portal/page?_pageid=111&_dad=myDAD&_
schema=mySchema
```

You must manually insert:

```
&_debug=3
```

To make:

```
http://myserver.myplace.com:3000/portal/page?_pageid=111&_dad=myDAD&_
schema=mySchema&_debug=3
```

The values for **_debug** are shown in Table 13–3.

Table 13–3   PPE Request Log Levels

Value	Detail
0	Activates page-debugging information.
1	Activates page-debugging information.
2	Log to page and set the request logmode to **debug**.
3	Log to page and set the request logmode to **request**.
4	Log to page and set the request logmode to **content**.
5	Log to page and set the request logmode to **parsing**.

Page Logging

With **_debug** set to **2**, **3**, **4**, or **5**, page logging is activated. This means that messages logged for the request are recorded in the PPE's log file as well as in the page returned.

Page logging is a simple means by which to obtain detailed information relating to a request. As a result, it is also a security issue, for which the **urlDebugMode** servlet initialization argument is provided.

urlDebugMode can be found alongside **logmode** in the portal's `web.xml` file:

```
<init-param>
     <param-name>urlDebugMode</param-name>
     <param-value>4</param-value>
</init-param>
```

Both **urlDebugMode** and **logmode** can be found inside the **page** servlet clause:

```
<servlet>
     <servlet-name>page</servlet-name>
     <servlet-class>oracle.webdb.page.ParallelServlet</servlet-class>
     .
     .
     <init-param>
          <param-name>urlDebugMode</param-name>
          <param-value>4</param-value>
     </init-param>
     .
     .         .
</servlet>
```

The values for **urlDebugMode** are shown in Table 13–4.

Table 13–4   PPE urlDebugMode Levels

Value	Detail
None	Ignore the **_debug** URL parameter.
0	Only allow **_debug** to be **0**.
1	Only allow **_debug** to be **0** or **1**.
2	Only allow **_debug** to be **0**, **1**, or **2**.
3	Only allow **_debug** to be **0**, **1**, **2**, or **3**.
4	Only allow **_debug** to be **0**, **1**, **2**, **3**, or **4**.
5	Only allow **_debug** to be **0**, **1**, **2**, **3**, **4**, or **5**.

13.4.1.3.1 PPE Log File Contents

PPE diagnostic messages are recorded in the servlet context `application.log` file. This file can be found at:

```
ORACLE_HOME/j2ee/OC4J_
Portal/application-deployments/portal/<island>/application.log
```

There are two types of PPE messages:

- Standard PPE Messages
- Performance PPE Messages

Standard PPE Messages

Here is an example of a standard PPE message found in its log file:

```
03/12/31 11:54:35 portal: id=22020914339,0 DEBUG: active=53  ContentFetcher
Unexpected Exception Request Failed:java.lang.IllegalArgumentException
name=content-fetcher52 label=dbPortlet url=https://abc.company.com:5001/pls/ptl_
9_0_4_0_87/!PTL_9_0_4_0_87.wwpro_app_provider.execute_portlet/391497559/4
time=38975ms timeout=15000ms process=ResponseHeaders
```

The content is as follows:

03/12/31 11:54:35: - Date and time

portal: - Web application

DEBUG: - logmode flag

active=53: - Active count

id=22020914339, 0: - ECID

ContentFetcher Unexpected Exception Request Failed: - Message

Additional details relating to some of these values are shown in Table 13–5.

Table 13–5   PPE Standard Message Attributes

Value	Detail
logmode flag	Indicates that logmode is **debug** or higher. If logmode is set to **perf** and is therefore lower than **debug**, the logmode flag is not included in the message.
Active count	The number of threads in the PPE's thread group. If **logmode** is set to **perf** and is therefore lower than **debug,** the active count is not included in the message.
ECID	The ECID value can be **null**. A message with such a value relates to a PPE background task (such as clearing pooled objects). Background tasks do not relate to a request and therefore do not have an ECID specified.

13.4.1.4 Oracle Application Server Portal Developer Kit

The Oracle Application Server Portal Developer Kit (PDK) provides a framework for the construction of portlets and portlet providers in a variety of Web languages including Java, Web Services, XML, ASP, PERL and PL/SQL. The PDK therefore encompasses the JPDK.

The PDK provides a core logging mechanism, which is augmented by logging in specific Developers Kits. PDK logging is controlled through a Web-based user interface as shown in Figure 13–2.

Figure 13–2   PDK Logging Page
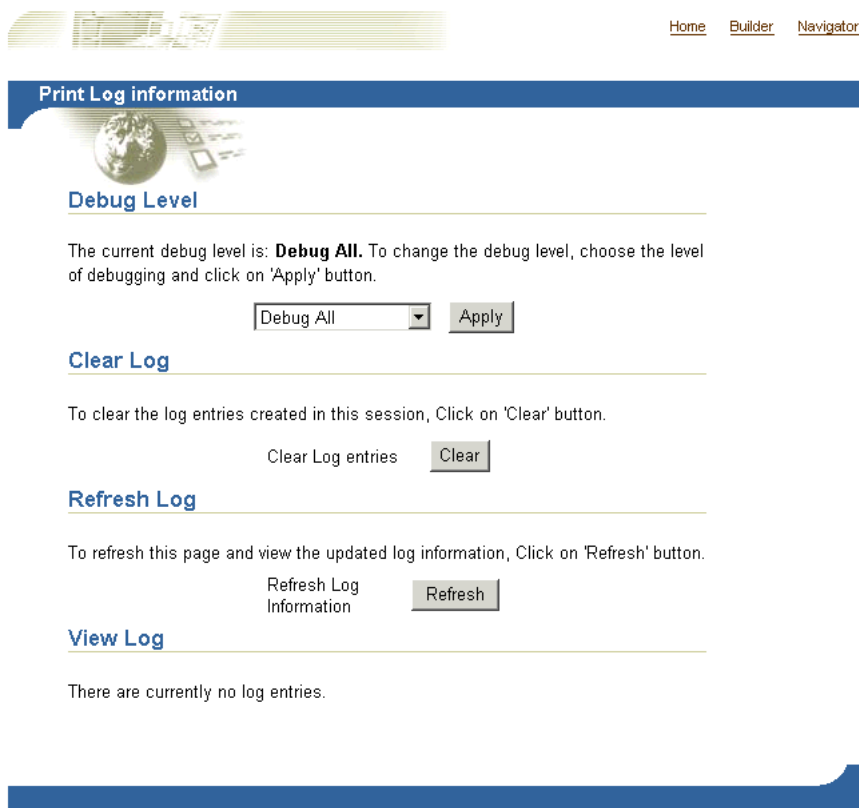
This can be found at:

```
http://<host>:<port>/pls/<dad>/<schema>.wwpro_log.render
```

For example:

```
http://myserver.myplace.com:3000/pls/portal/PORTAL.wwpro_log.render
```

From this page you can apply the following logging levels:

Table 13–6   PDK Log Levels

Level	Detail
No debugging	No logging.
PROHTTPJ	Provider framework logging only.
PROGRP	Provider logging only.
ADAPTER	Federated portal adapter logging only.
CACHE	Cache logging only.
FORCE	Internal to Oracle.
INVAL	Invalidation logging only.
PROREG	Provider registration logging only.
PROLOGIN	Page metadata generation, login and session initialization logging only.
PROPROV	Provider communication logging only.
PROPMR	Portlet repository metadata logging only.
PROHTTP	Web provider framework logging only.
All	Every logging level is activated.

13.4.1.4.1   PDK Log File Contents

You can view PDK log entries from the same page used to configure PDK logs, as shown in Figure 13–3.

Figure 13–3   Log Entries in the PDK Logging Page

13.4.1.5 OracleAS Metadata Repository

The OracleAS Metadata Repository consists of all the metadata, portal content and PL/SQL code that reside in the OracleAS Portal database schema. The PL/SQL code that executes in the OracleAS Portal schema also generates diagnostic output that can be correlated with diagnostics output generated from the other components of OracleAS Portal.

Since the log file is output from the OracleAS Metadata Repository, the database running OracleAS Portal needs to be configured to allow this. To do this, you must update the database's `init.ora` file, adding the following line:

```
UTL_FILE_DIR=<directory where you want to write the log file>
```

There can be many **UTL_FILE_DIR** entries, so if the directory you wish to write to is already defined, there is no need to modify this file.

> **Notes:**
>
> - On installation of OracleAS Metadata Repository, if the database you are installing into has the UTL_FILE_DIR parameter set, the OracleAS Portal installer will configure the OracleAS Metadata Repository such that it uses the first directory defined by the database parameter as the location for the OracleAS Metadata Repository log file. If UTL_FILE_DIR is not configured, OracleAS Metadata Repository logging is not set up on installation.
>
> - If you update the `init.ora` file, you must also create an **SPFILE** and restart the database. Refer to the Oracle9*i* Database Server documentation library for more information.

OracleAS Metadata Repository logging is performed through a logging package. This logging package is controlled using the script `logcfg.sql` which you must run from SQL*Plus.

The script `logcfg.sql` can be found at:

`ORACLE_HOME/portal/admin/plsql/wwc`

The `logcfg.sql` script can take five parameters in the following order: **log_level**, **log_state_level**, **log_format**, **log_file**, and **log_directory**. If less than five parameters are supplied, then one or more values are requested. If no value is received in response to this request, the current value is maintained.

Table 13–7 details `logcfg.sql` parameters.

Table 13–7   Repository Logging Package Parameters

Parameter	Detail
log_level	Describes the level of messages recorded. The values are: ■ 0 - None ■ 1 - Error ■ 2 - Warning ■ 3 - Information ■ 4 - Trace ■ 5 - Debug The values build incrementally. The default value is 1.
log_state_level	Describes the level of messages for which state information will automatically be logged. The values are: ■ 0 - None ■ 1 - Error ■ 2 - Warning ■ 3 - Information ■ 4 - Trace ■ 5 - Debug The values build incrementally.
log_format	Describes the format that automatically recorded context information, which is different from state information. The values are: ■ 0 - Simple ■ 1 - Detailed
log_file	The name of the log file to write to. An attempt is made to create this file if it does not already exist.
log_directory	The directory in which the log_file exists. This directory must be defined in the init.ora database file under the **UTL_FILE_DIR** property. For example: utl_file_dir=/export/home/oracle/iAS904/dblogs If the init.ora file is modified, the database must be restarted for this change to take effect.

For example, you can run the script logcfg.sql from SQL*Plus as follows:

```
@logcfg.sql 3 3 1 portal.log /export/home/oracle/iAS904/logs
```

On running `logcfg.sql`, the usage is displayed:

```
Configure Portal diagnostics
usage:
logcfg.sql <log_level> <log_state_level> <log_format> <log_file> <log_
directory>
If for any of the params a null value is specified the existing value will
be maintained.
Log levels:
0 - None (turn diagnostics off)
1 - Error
2 - Warning
3 - Information
4 - Trace
5 - Debug
Log formats:
0 - Simple
1 - Detailed
```

The current values are also displayed:

```
Current settings:
Log level:      3
Log state level:     3
Log format:     1
Log file:      portal.log
Log directory:    /export/home/oracle/iAS904/dblogs
```

To truncate the OracleAS Metadata Repository diagnostics log file, run the SQL script `logtrunc.sql` located at:

ORACLE_HOME/portal/admin/plsql/wwc

13.4.1.5.1 Repository Log File Contents

The location of the OracleAS Metadata Repository's diagnostic information is dictated by the Repository diagnostic package parameters **log_file** and **log_directory**.

Here is an example of a message found in the OracleAS Metadata Repository's log file:

```
[06-AUG-2002 15:02:15] [ERROR] id=(null) ctx=wwsrc_simple_edit.render_
simple_edit_prefs user=PORTAL subscriberId=1 language=us
```

```
userAgent="Mozilla/5.0" ip=192.0.0.1
ORA-30625: method dispatch on NULL SELF
[START-CALL-STACK]
----- PL/SQL Call Stack -----
object          line           object
handle          number         name
81b35e6c        350            package body PORTAL.WWLOG_API_DIAG
81b35e6c        443            package body PORTAL.WWLOG_API_DIAG
81b35e6c        526            package body PORTAL.WWLOG_API_DIAG
86765ac8        259            package body PORTAL.WWSRC_SIMPLE_EDIT
86765ac8        334            package body PORTAL.WWSRC_SIMPLE_EDIT
84317130        19             package body PORTAL.WWSBR_BASIC_SEARCH
88857980        713            package body PORTAL.WWSBR_SITEBUILDER_PROVIDER
8323ad18        1              anonymous block
87e53d5c        648            package body PORTAL.WWPRO_API_PROVIDER
81ae1e50        2644           package body PORTAL.WWPOB_PAGE
877a0d9c        12             anonymous block
[END-CALL-STACK]
[START-ERROR-STACK]
ORA-30625: method dispatch on NULL SELF
[END-ERROR-STACK]
[START-QUERY-STRING]
_providerid=102274117
_portletid=14
_mode=5
_title=Basic%20Search
_referencepath=1875_BASICSEARCH_102274117
_back_url=http%3A%2F%2Fmyserver.myplace.com%3A3000%2Fpls%2Fportal%
_portlet_reference=33_31293_33_1_1
[END-QUERY-STRING]
```

Its content is as follows:

ORA-30625: method dispatch on NULL SELF: - The message itself.

Along with its context and state information.

Context Information   Context information is produced in one of two formats, detailed or simple, as specified by **log_format**. In the given example, the format is detailed:

06-AUG-2002 15:02:15: - Date and time

ERROR: - Message level

id=(102733434, 1): - ECID

ctx=wwsrc_simple_edit.render_simple_edit_prefs: - Message context

user=PORTAL: - Database user

subscriberId=1: - Subscriber identifier

language=us: - Globalization Support language

userAgent="Mozilla/5.0": - User agent

ip=192.0.0.1: - Client IP address

The simple format is a subset of the detailed format and includes the following information:

06-AUG-2002 15:02:15: - Date and time

ERROR: - Message level

ctx=wwsrc_simple_edit.render_simple_edit_prefs: - Message context

Additional details relating to some of these values are included in Table 13–8.

Table 13–8   Repository Context Attributes

Value	Detail
Client IP address	Typically, this is the IP address of the client browser or HTTP proxy in use. Since the portal page assembly process makes use of loop back calls, the IP address can also represent the middle-tier itself.
Subscriber identifier	Identifies which subscriber has been accessed.
User agent	A description of the browser in use.

State Information   State information consists of the call stack:

```
[START-CALL-STACK]
----- PL/SQL Call Stack -----
object          line        object
handle          number      name
81b35e6c        350         package body PORTAL.WWLOG_API_DIAG
81b35e6c        443         package body PORTAL.WWLOG_API_DIAG
81b35e6c        526         package body PORTAL.WWLOG_API_DIAG
86765ac8        259         package body PORTAL.WWSRC_SIMPLE_EDIT
86765ac8        334         package body PORTAL.WWSRC_SIMPLE_EDIT
84317130        19          package body PORTAL.WWSBR_BASIC_SEARCH
```

```
88857980        713            package body PORTAL.WWSBR_SITEBUILDER_PROVIDER
8323ad18        1              anonymous block
87e53d5c        648            package body PORTAL.WWPRO_API_PROVIDER
81ae1e50        2644           package body PORTAL.WWPOB_PAGE
877a0d9c        12             anonymous block
[END-CALL-STACK]
```

Error stack:

```
[START-ERROR-STACK]
ORA-30625: method dispatch on NULL SELF
[END-ERROR-STACK]
```

And query string:

```
[START-QUERY-STRING]
_providerid=102274117
_portletid=14
_mode=5
_title=Basic%20Search
_referencepath=1875_BASICSEARCH_102274117
_back_url=http%3A%2F%2Fmyserver.myplace.com%3A3000%2Fpls%2Fportal%
_portlet_reference=33_31293_33_1_1
[END-QUERY-STRING]
```

13.4.1.5.2   Repository Diagnostics Log File Registration

Oracle Enterprise Manager provides a Log Reader and Log Viewer. The Log Reader allows administrators to upload log files to a file-based log repository. The Log Viewer allows administrators to view and query log entries loaded into the repository. For more information, see Section 13.6, "Using Application Server Control Log Viewer".

To load and view the Repository Diagnostics log file entries, you must first register the log file with Oracle Enterprise Manager. To do this, edit the following file:

ORACLE_HOME/diagnostics/config/registration/PORTAL.xml

In this file, there is a template entry that you can copy and expand to reflect details of your log file. The template is as follows:

```
<logs xmlns="http://www.oracle.com/iAS/EMComponent/ojdl" helpIDLogs="psm_cs_xml_
log_info">

<!--
<log path="<PATH>" componentId="PORTAL">
```

```
<logreader type="SimpleTextLog">
    <property name="ComponentId" value="PORTAL"/>
    <property name="ModuleId" value="Portal:<INSTANCE>"/>
    <property name="TimestampFormat" value="[dd-MMM-yyyy HH:mm:ss]"/>
    <property name="TimestampLocale" value="en_US"/>
</logreader>
<logviewer ComponentName="ID_VLOGS_PORTAL_REP@ResourceBundle"
        LogType="ERROR"
        LogName="Diagnostics for Portal instance <INSTANCE>"/>
</log>
-->

</logs>
```

Modify the following information in the copied template entry:

<PATH> - The absolute path and filename of the log file.

<INSTANCE> - The name of the OracleAS Portal target in Oracle Enterprise Manager, if one is defined. If there is no corresponding OracleAS Portal target in Oracle Enterprise Manager, use the name of the OracleAS Portal instance and database details. For example, `<portal schema name>-<db service name>`. This value is used to distinguish this log entry in the Log Viewer from other OracleAS Portal instance log entries.

Once you have saved the new `PORTAL.xml` entry, the Log Reader starts uploading the log file periodically, and you can use the Log Viewer to view and query this log file.

Since the OracleAS Metadata Repository can be accessed through many middle-tiers, you need to:

- Register the Repository diagnostics log file with one of the Oracle Enterprise Manager Application Server Control instances that is monitoring a OracleAS Portal middle-tier.

- Ensure that the log file is accessible over a network file system, if the OracleAS Portal database is on a machine other than the OracleAS Portal middle-tier.

- To perform log correlation in a multi middle-tier environment, you need to register the Repository diagnostics log file with each Oracle Enterprise Manager instance monitoring a OracleAS Portal middle-tier.

> **Note:** On changing the Infrastructure services that are used. using
> Oracle Enterprise Manager, you have to update the location of the
> Repository diagnostics log file in the `PORTAL.xml` file located at
> `ORACLE_HOME/diagnostics/config/registration/`.

13.4.1.6 OracleAS Web Cache

Oracle Application Server Web Cache events and errors are stored in an *event log*.
The event log can help you determine which documents or objects have been
inserted into the cache. It can also identify listening port conflicts or startup and
shutdown issues. By default, the event log has a file name of `event_log` and is
stored in `ORACLE_HOME/webcache/logs` on UNIX and `ORACLE_HOME\webcache\logs` on Windows.

> **See Also:** *Oracle Application Server Web Cache Administrator's Guide*

13.5 Using the OracleAS Portal Diagnostics Assistant

Use the OracleAS Portal Diagnostic Assistant to gather information if you are
troubleshooting issues after OracleAS Portal installation. Problems can vary from
accessing the portal, to users getting errors at different levels within the portal.

You can diagnose issues by reviewing the results from the OracleAS Portal
Diagnostic Assistant. Alternatively, you can upload the results to Oracle Support so
they can troubleshoot the problem for you.

The generated report includes the following:

- OracleAS Portal Repository database information

- OracleAS Single Sign-On database information

- Oracle Internet Directory diagnostics report

- Oracle Text diagnostic report

- Apache error log file analysis

In addition, all OracleAS Portal-related configuration files and log files are collected
and zipped for your convenience. For a detailed description of all the information
collected, refer to the readme file located in the directory `ORACLE_HOME/portal/admin/utils/tshoot`.

Each time you run the OracleAS Portal Diagnostic Assistant a new directory is
created for the generated files, under the directory `ORACLE_`

HOME/portal/admin/utils/tshoot. The directory names have a timestamp format, for example, 20030623132344 which means:

year - 2003

month - 06

day - 23

hour - 13

minutes -23

seconds - 44

After running the OracleAS Portal Diagnostic Assistant, locate the appropriate directory and open the HTML report named pda.htm in a Browser window. You can use the links provided to navigate through the report and review the diagnostic information.

If you want Oracle Support to troubleshoot the problem, upload the generated ZIP file named PDA<directory_name>.zip, for example, PDA20030623132344.zip.

Refer to readme.htm in the *ORACLE_HOME*/portal/admin/utils/tshoot directory for detailed information on using the OracleAS Portal Diagnostic Assistant.

Running the OracleAS Portal Diagnostic Assistant

To generate diagnostics information using the OracleAS Portal Diagnostic Assistant, follow these steps:

1. Check the **Support/Upgrade** section on Portal Center, http://portalcenter.oracle.com for the latest update/patch information for the OracleAS Portal Diagnostic Assistant.

Download the latest OracleAS Portal Diagnostic Assistant script. The **Support/Upgrade** link is located in the **Product Information** section.

2. Ensure that the ORACLE_HOME environment variable is set to the correct OracleAS Portal middle-tier Oracle home directory.

If you try to run the OracleAS Portal Diagnostic Assistant from a database *ORACLE_HOME* it fails and no diagnostics information is collected.

3. Go to the directory *ORACLE_HOME*/portal/admin/utils/tshoot and run the Perl script ptshoot.pl as follows:

ORACLE_HOME/perl/bin/perl ptshoot.pl

Run `ptlshoot.pl` without any arguments to get help information.

4. Open the latest HTML report (`pda.htm`) in a Browser window and use the information to help diagnose what is wrong with OracleAS Portal.

13.6 Using Application Server Control Log Viewer

You can use the Oracle Enterprise Manager Application Server Control to view and query entries from the following Oracle Application Server log files to diagnose issues relating to OracleAS Portal. The relevant Oracle Application Server component log files include:

- **Portal:<instance>** - displays a single, diagnostic error log file for each Portal instance named *<customer_specified_log_name>*. This log file is generated by the relevant OracleAS Metadata Repository.

- **HTTP_Server** - displays multiple error/access log files named `error_log` and `access_log`. This log file contains all relevant mod_plsql logging information.

- **OC4J_Portal** - displays multiple application log files named `application.log`. This log file contains all relevant PPE logging information.

- **JPDK** - For the JPDK sample providers in a standalone OC4J, the location is `j2ee/home/application-deployments/jpdk/application.log`. In an Oracle Application Server middle-tier, the location is similar with the addition of a directory for the default island.

- **Web Cache** - displays an error and access log files name `event_log` and `access_log`.

Before you can use the OracleAS Metadata Repository log file with the Application Server Control Log Viewer, you must complete a registration process. For instructions, see Section 13.4.1.5.2, "Repository Diagnostics Log File Registration".

If your JPDK OC4J instance is *not* located in the OracleAS Portal middle-tier Oracle home, then its log file may only be viewed through the local Application Server Control instance. If you want to perform diagnostic correlation (see subsequently), you will need to follow a similar remote registration process to that described for the OracleAS Metadata Repository log file when it is remotely located.

In addition to viewing the log file entries with the Application Server Control Log Viewer, you can also perform advanced diagnostics by correlating entries across log files using the ECID value discussed in Section 13.4, "Diagnosing OracleAS Portal

Problems". This drill-down correlation is automatically provided by the Application Server Control Log Viewer.

To view log file entries, click the **Logs** link in the Application Server Control. This link is located at the top and bottom of every Application Server Control component home page.

> **See Also:** For detailed instructions on how to use the Log Viewer, see *Oracle Application Server 10g Administrator's Guide*, chapter *Managing Diagnostic Log Files*. This chapter also describes how to perform advanced queries for diagnostic log file information, search through diagnostic messages (collected from selected Oracle Application Server components) in the Log Repository and correlate messages across log files and components.

Figure 13–4 shows an example of Oracle Application Server components selected in the View Logs page.

Figure 13–4   Application Server Control View Logs Page
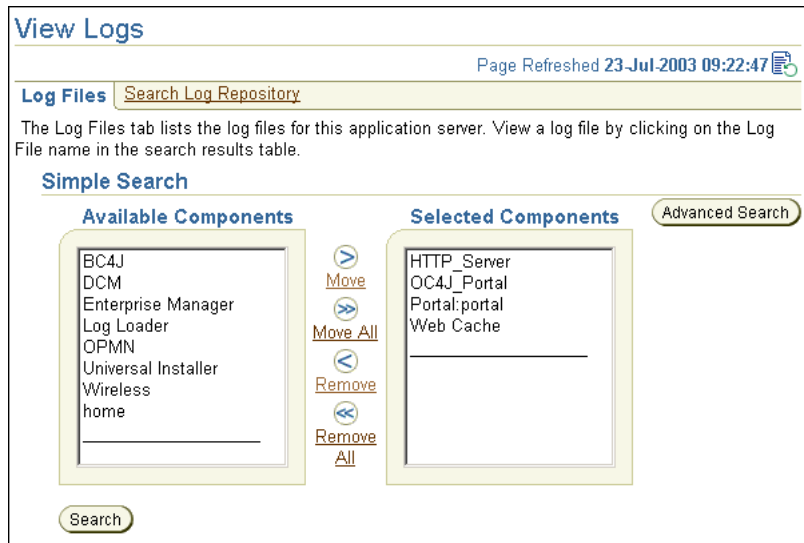
13.7 Troubleshooting Export and Import

For discussions on import and export, visit the Oracle Technology Network Discussion Forums site and look for export/import, accessible from the same URL.

For detailed error message descriptions, cause and actions, refer to the export/import error messages chapter of the *Oracle Application Server Portal Error Messages Guide*.

> **Note:** Look for documentation specific to OracleAS Portal 10*g* (9.0.4), as procedures and best practices for subsequent and prior releases of OracleAS Portal do vary.

13.8 Troubleshooting Search Functionality

This section provides information on the common problems encountered while using the Search functionality in OracleAS Portal, or Oracle Text.

13.8.1 Problems If Too Many Page Groups or Search Attributes Selected

Search functionality can become inconsistent if the search criteria includes a large number of attributes and the user can choose which page groups to search and the list of available page groups is very long. This problem is due to URL size constraints.

Here are examples of issues that can occur:

- Links do not work, for example, Saved Search, Bulk Action, Edit, and so on. Note that these links do work when fewer attributes/page groups are selected.

- Search results page can lose search criteria when changing between tabs.

- Some search results can be lost whilst saving a search. Also, some attributes may be lost when the search is run again.

A workaround is to reduce the number of attributes, or page groups, or both, available for user selection.

13.8.2 Cannot Search PL/SQL Attributes

If you define search criteria for a PL/SQL attribute, it is ignored and therefore search results are not returned as expected.

13.8.3 Troubleshooting Oracle Text Installation Problems

If you are experiencing Oracle Text-related problems, use the TEXTTEST utility to check that Oracle Text functionality is installed and setup correctly. See Appendix H, "Using TEXTTEST to Check Oracle Text Installation".

13.9 Troubleshooting Federated Portal Adapter

This section lists the issues you may encounter while working with the Federated Portal Adapter, and their workarounds.

Known Restrictions Showing Page Portlets with the Federated Portal Adapter

- The **Show Details** mode does not work, that is, the portlet name cannot be displayed as a link that shows additional information about the portlet.

- If the page portlet contains tabs, then clicking a tab is a 'deep link' and the rendered page takes over the whole page, that is, it is not shown within the original page as a portlet.

- The rendering of navigation pages, which includes the page banner, does not work properly when pages are displayed through the Federated Portal Adapter. For example, the **Customize** link in a *regular page portlet* displays customization options for the container page, but this is not the case in a *remote page portlet*. Also, page portlets shown through the Federated Portal Adapter do not display the banner of the container page, whereas the banner is displayed in the case of *regular page portlets*.

- If the page portlet has a navigation page portlet that has a sub page region in it, the sub page region will not be displayed on the page portlet when it gets rendered through the Federated Portal Adapter. For a non-remote page portlet, the region shows the sub pages of the container page holding the portlet.

13.10 OracleAS Portal Errors

Refer to the book *Oracle Application Server Portal Error Messages Guide*, for more information on error messages. This book contains the following sections:

- Installation Error Messages

- OracleAS Web Cache Error Messages

- Security Error Messages

- Portlet Development Error Messages
- Upgrade Error Messages
- Export/Import Error Messages
- Other Error Messages

Part IV

Appendices

Part four contains the following appendixes:

- Appendix A, "Using the Portal Dependency Settings File"
- Appendix B, "Using the OracleAS Portal Configuration Assistant Command Line Utility"
- Appendix C, "Using OracleAS Portal Installation and Configuration Scripts"
- Appendix D, "Configuring the Parallel Page Engine"
- Appendix E, "Using Oracle Application Server Configuration Files"
- Appendix F, "Integrating JavaServer Pages with OracleAS Portal"
- Appendix G, "Using the wwv_context APIs"
- Appendix H, "Using TEXTTEST to Check Oracle Text Installation"
- Appendix I, "Administering Web Clipping"

A

Using the Portal Dependency Settings File

OracleAS Portal is dependent on the components: Oracle Application Server Web Cache and Oracle Internet Directory. It is important that you understand these dependencies, as it may be necessary to fine tune or configure these components after Oracle Application Server is installed.

To simplify configuration changes, OracleAS Portal introduces the *Portal Dependency Settings File* (iasconfig.xml). This file stores configuration data from all the dependent components in a central place and the content of the file is updated when there are configuration changes.

You can use the Portal Dependency Settings file to:

- Check settings used by an OracleAS Portal instance
- Update settings in the Oracle Application Server Metadata Repository.

This appendix discusses the Portal Dependency Settings, and the Portal Dependency Settings tool in the following two sections:

- Portal Dependency Settings File Details
- Configuration Tools

A.1 Portal Dependency Settings File Details

The following sections describe the Portal Dependency Settings file in more detail:

- Name and Location
- Updating the Portal Dependency Settings File
- Configuration Elements
- Sample Portal Dependency Settings File

- Post-Installation Mapping in the Portal Dependency Setting File
- Common Configuration Mapping in the Portal Dependency Settings File

A.1.1 Name and Location

The name of the Portal Dependency Settings file is `iasconfig.xml`, and is located by default in *ORACLE_HOME*/portal/conf, where *ORACLE_HOME* is the OracleAS Portal and Oracle Application Server Wireless middle-tier home.

When using any of the tools that access the Portal Dependency Settings file, you can override the default location of the file by setting the environment variable IASCONFIG_LOC to the directory in which your file is stored, for example:

```
set IASCONFIG_LOC=/usr/local/ias904
```

A.1.2 Updating the Portal Dependency Settings File

If the Portal Dependency Settings file is accessible over a network file system, you can share the file across multiple hosts, avoiding the need to manually replicate it every time the file is modified. If the installation is running on an operating system which supports symbolic links, it is recommended that you use this mechanism to reference a shared file, instead of setting the IASCONFIG_LOC environment variable.

If, however, the Portal Dependency Settings file is not accessible over the network, you must ensure that the file is kept up-to-date with changes to your site topology. The Portal Dependency Settings file is used to configure the Portal Repository with details of OracleAS Web Cache, Oracle Internet Directory and Oracle Enterprise Manager that it is using. It is not required that it is copied into each individual middle-tier in your site, but you must ensure that any changes to the components modeled in the file that affect OracleAS Portal configuration are updated in the file.

To demonstrate how the Portal Dependency Settings file is kept up-to-date, let's use the configuration defined in Section 5.3, "Configuring Multiple Middle-Tiers with a Load Balancing Router".

1. The Portal Dependency Settings file gets first created in Section 5.3.1, "Step 1: Install a Single Portal and Wireless Middle-Tier (M1)", during the installation. It looks like Example 5–1, "iasconfig.xml After the First Middle-Tier Installation".

   This file will be located on machine `m1.abc.com`, typically in *ORACLE_HOME*/portal/conf of the middle-tier that has just been installed.

2. In Step 2: Configure OracleAS Portal on M1 to Be Accessed Through the LBR, the Portal Dependency Settings file is manually changed as shown in Example 5–2, "iasconfig.xml File Edited to Include Farm Element".

This file will be on machine `m1.abc.com`, typically in *ORACLE_ HOME*/portal/conf of the middle-tier installed in Step 1. You use the `ptlconfig` tool as shown in Section A.2.1, "Portal Dependency Settings Tool", after you make changes to the file.

Any future changes to the OracleAS Web Cache, Oracle Internet Directory, or Oracle Enterprise Manager settings in `iasconfig.xml` should be made using the Application Server Control, or by hand on `m1.abc.com`. You must also use the ptlconfig tool again after you make changes.

Note: Changes to OracleAS Portal's OracleAS Web Cache settings can also be made on the **Portal Web Cache Settings** page, as described in Section 7.2.3, "Portal Web Cache Settings Page".

Typically, the hostname and port number, by which OracleAS Portal is addressed, uses the OracleAS Web Cache hostname and port number. This is because, in a simple configuration, browser requests go directly to OracleAS Web Cache. However, in a configuration that has a load balancing router (LBR), or reverse proxy server front-ending OracleAS Web Cache, the hostname and port number defined on this page may need to reflect that of the LBR, or reverse proxy server.

In this configuration, you want OracleAS Web Cache invalidation messages to be sent directly to the OracleAS Web Cache host, as opposed to the LBR, or reverse proxy server. In the scenario where your published hostname is different from the hostname used for OracleAS Web Cache invalidation, you cannot use the **Portal Web Cache Settings** page in the Oracle Enterprise Manager Application Server Control, or the Portal Dependency Settings file `iasconfig.xml`, to establish these settings. Instead, you must use the OracleAS Portal Configuration Assistant (OPCA) in the MIDTIER mode with -type OHS., using the `host` parameter to specify the hostname of the LBR, or reverse proxy server, and the `-chost` parameter to define the OracleAS Web Cache hostname.

---

3. In Step 5: Configure the New Middle-Tier (M2) to Run Your Existing Portal, the Portal Dependency Settings file on m2.abc.com needs to be updated manually with the settings defined in the iasconfig.xml file on m2.abc.com.

A.1.3 Configuration Elements

The Portal Dependency Settings file is an XML file, that is made up of a number of elements that describe the settings of specific Oracle Application Server components and the dependencies Portal instances have on them. Figure A–1 shows all the elements that can be modeled in the Portal Dependency Settings file. The Portal Dependency Settings file definition is modeled in the schema file `iasconfig.xsd`, which is located in *ORACLE_HOME*/portal/conf.

Figure A–1 Elements in the Portal Dependency Settings file
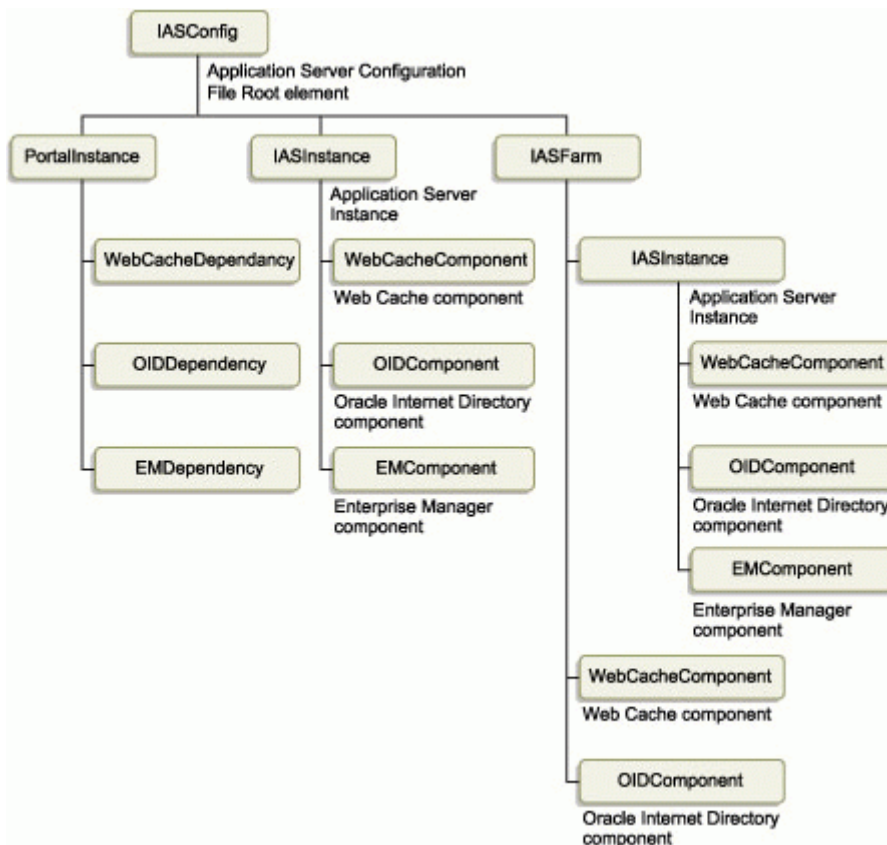
The individual elements are:

- IASFarm
- IASInstance
- PortalInstance
- WebCacheComponent
- OIDComponent
- EMComponent
- WebCacheDependency
- OIDDependency
- EMDependency

IASFarm

The *IASFarm* element represents a logical farm of Oracle Application Server instances.

Table A–1   Element IASFarm

Attribute Name	Type	Description
Name	String	Unique farm name
Host	String	Logical host machine

IASInstance

The *IASInstance* element represents a specific Oracle Application Server instance, which usually maps to an Oracle home.

Table A–2   Element IASInstance

Attribute Name	Type	Description
Name	String	Oracle Application Server instance name (for example, `ias904.host.domain`)
Host	String	Host machine
Version	String	Version of Oracle Application Server.

PortalInstance

These are the OracleAS Portal instance settings.

Table A–3   Element PortalInstance

Attribute Name	Type	Description
DADLocation	String	The name and location of the OracleAS Portal DAD (for example, /pls/portal).
ConnectString	String	OracleAS Metadata Repository connect string
SchemaUsername	Integer	OracleAS Portal schema username
SchemaPassword	String	OracleAS Portal schema password

WebCacheComponent

These are the OracleAS Web Cache settings.

Table A–4   Element WebCacheComponent

Attribute Name	Type	Description
ListenPort	Integer	Listening port
AdminPort	Integer	Administration port
InvalidationPort	Integer	Invalidation port
InvalidationUsername	String	Invalidation username
InvalidationPassword	String	Invalidation password
SSLEnabled	String	Flag to indicate whether the listening port is SSL enabled. The value can either be TRUE or FALSE.

OIDComponent

These are the Oracle Internet Directory settings.

Table A–5   Element OIDComponent

Attribute Name	Type	Description
AdminPassword	String	Oracle Internet Directory administration password

Table A–5   (Cont.) Element OIDComponent

Attribute Name	Type	Description
PortSSLEnabled	String	Flag to indicate whether the HTTP port is SSL enabled. The value can either be TRUE or FALSE.
LDAPPort	Integer	LDAP port that Oracle Internet Directory is running on.
AdminDN	String	Oracle Internet Directory administration distinguishing name

EMComponent

These are the Oracle Enterprise Manager Application Server Control settings.

Table A–6   Element EMComponent

Attribute Name	Type	Description
ConsoleHTTPPort	Integer	Listening port
SSLEnabled	String	Flag to indicate whether the listening port is SSL enabled. The value can either be TRUE or FALSE.

WebCacheDependency

This is the OracleAS Portal instance reference to the OracleAS Web Cache it is using.

Table A–7   Element WebCacheDependency

Attribute Name	Type	Description
ContainerType	String	The type of the container the OracleAS Web Cache component is running under. This can be either *IASInstance* or *IASFarm*.
Name	String	*IASInstance* name or the unique *IASFarm* name, depending on *ContainerType*.

OIDDependency

This is the OracleAS Portal instance reference to the Oracle Internet Directory it is using.

Table A–8   Element OIDDependency

Attribute Name	Type	Description
ContainerType	String	The type of the container the Oracle Internet Directory component is running under. This can be either *IASInstance* or *IASFarm*.
Name	String	*IASInstance* name or the unique *IASFarm* name, depending on *ContainerType*.

EMDependency

This is the Oracle Enterprise Manager Application Server Control managing this OracleAS Portal instance.

Table A–9   Element EMDependency

Attribute Name	Type	Description
ContainerType	String	The type of the container the Oracle Enterprise Manager Application Server Control is being managed by. This should be set to *IASInstance*.
Name	String	*IASInstance* name

A.1.4 Sample Portal Dependency Settings File

The following XML represents the contents of a sample Portal Dependency Settings file:

```
<IASConfig XSDVersion="1.0">

   <IASInstance Name="iAS-1" Host="abc.company.com" Version="9.0.4">
      <WebCacheComponent AdminPort="3001" ListenPort="3002"
InvalidationPort="3003" InvalidationUsername="orcladm"
InvalidationPassword="orcladm=" SSLEnabled="false"/>
   </IASInstance>

   <IASInstance Name="iAS-2" Host="xyz.company.com" Version="9.0.4">
      <OIDComponent AdminPassword="orcladm" PortSSLEnabled="false"
LDAPPort="3002" AdminDN="cn=orcladmin"/>
      <EMComponent ConsoleHTTPPort="1814" SSLEnabled="false"/>
   </IASInstance>

   <PortalInstance DADLocation="/pls/portal" SchemaUsername="portal"
```

```
SchemaPassword="welcome1" ConnectString="xyz.company.com:1521:s901dev3">
      <WebCacheDependency ContainerType="IASInstance" Name="iAS-1"/>
      <OIDDependency ContainerType="IASInstance" Name="iAS-2"/>
      <EMDependency ContainerType="IASInstance" Name="iAS-2"/>
   </PortalInstance>

</IASConfig>
```

In this example, the OracleAS Portal instance is:

- Accessed from the Database Access Descriptor (DAD) /pls/portal.

- Dependent on:

  - OracleAS Web Cache component running in Oracle Application Server instance **iAS-1**

  - Oracle Internet Directory component running in Oracle Application Server instance **iAS-2**

  - Oracle Enterprise Manager Application Server Control component running in Oracle Application Server instance **iAS-2**

A.1.5 Post-Installation Mapping in the Portal Dependency Setting File

When OracleAS Portal is installed, the OracleAS Portal Configuration Assistant (OPCA) creates appropriate entries in the Portal Dependency Settings file, based on what is installed.

In an Application Server installation, the dependencies of OracleAS Portal on Oracle Application Server Web Cache and Oracle Internet Directory are added to the Portal Dependency Settings file. Existing information is not updated if duplicate entries are encountered during the installation. Instead, a warning is output to the installation log file that the entries already exist.

> **See Also:** Chapter 3, "Installing OracleAS Portal" for more information about the different installation types.

> **Note:** By default, the Portal Dependency Settings file is accessed from *ORACLE_HOME*/portal/conf, where *ORACLE_HOME* is the OracleAS Portal and Oracle Application Server Wireless middle-tier home. However, if the IASCONFIG_LOC environment variable is set, the location defined by this variable is used.

- In a single machine OracleAS Portal and OracleAS Wireless installation, where OracleAS Web Cache and Oracle Internet Directory instances already reside on the same machine, entries to the Portal Dependency Settings file are created as follows:

```
<IASConfig XSDVersion="1.0">

   <IASInstance Name="iAS-1" Host="abc.company.com" Version="9.0.4">
      <OIDComponent AdminPassword="orcladm" PortSSLEnabled="false"
LDAPPort="3002" AdminDN="cn=orcladmin"/>
   </IASInstance>

   <IASInstance Name="iAS-2" Host="abc.company.com" Version="9.0.4">
      <WebCacheComponent AdminPort="3001" ListenPort="3002"
InvalidationPort="3003" InvalidationUsername="orcadm"
InvalidationPassword="orcladm=" SSLEnabled="false"/>
   </IASInstance>

   <PortalInstance DADLocation="/pls/portal" SchemaUsername="portal"
SchemaPassword="welcome1" ConnectString="xyz.company.com:1521:s901dev3">
      <WebCacheDependency ContainerType="IASInstance" Name="iAS-2"/>
      <OIDDependency ContainerType="IASInstance" Name="iAS-1"/>
   </PortalInstance>

</IASConfig>
```

A.1.6 Common Configuration Mapping in the Portal Dependency Settings File

This section shows what the Portal Dependency Settings file looks like in the recommended topologies.

OracleAS Portal and OracleAS Wireless Developer Configuration: Medium Sized Machines

The topology for this common configuration is seen in Figure A–2.

Figure A–2 OracleAS Portal and OracleAS Wireless Developer Configuration
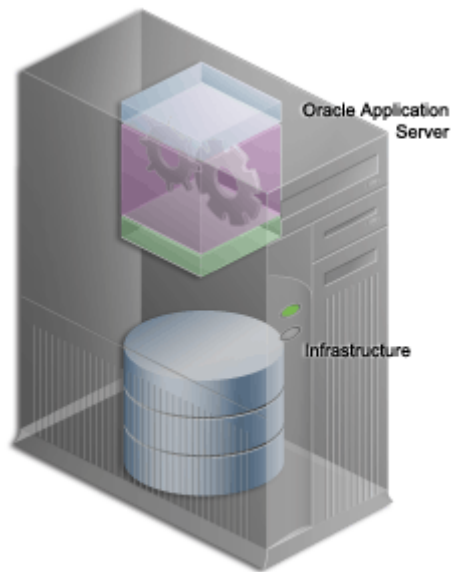
This configuration assumes that both the application server and the infrastructure are installed on the same machine, called **Host 1**.

When you install the Infrastructure on **Host 1** in Oracle home OH_2, no changes are made to the Portal Dependency Settings file.

When you install OracleAS Portal and OracleAS Wireless on **Host 1** in Oracle home OH_1, referencing the Oracle Internet Directory instance in OH_2, the Portal Dependency Settings file looks like this:

```
<IASConfig XSDVersion="1.0">

   <IASInstance Name="host1.OH_1" Host="host1.us.oracle.com" Version="9.0.4">
      <WebCacheComponent AdminPort="3001" ListenPort="7778"
InvalidationPort="3003" InvalidationUsername="orcadm"
InvalidationPassword="orcladm=" SSLEnabled="false"/>
   </IASInstance>

   <IASInstance Name="host1.OH_2" Host="host1.us.oracle.com" Version="9.0.4">
      <OIDComponent AdminPassword="orcladm" PortSSLEnabled="false"
LDAPPort="3002" AdminDN="cn=orcladmin"/>
   </IASInstance>
```

```
    <PortalInstance DADLocation="/pls/portal" SchemaUsername="portal"
SchemaPassword="welcome1" ConnectString="host1.us.oracle.com:1521:iasdb">
      <WebCacheDependency ContainerType="IASInstance" Name="host1.OH_1"/>
      <OIDDependency ContainerType="IASInstance" Name="host1.OH_2"/>
    </PortalInstance>

</IASConfig>
```

Enterprise Data Center Configuration: Multiple Departments Sharing the Same Data Center

The topology for this common configuration is seen in Figure A–3.

Figure A–3 Enterprise Data Center Configuration
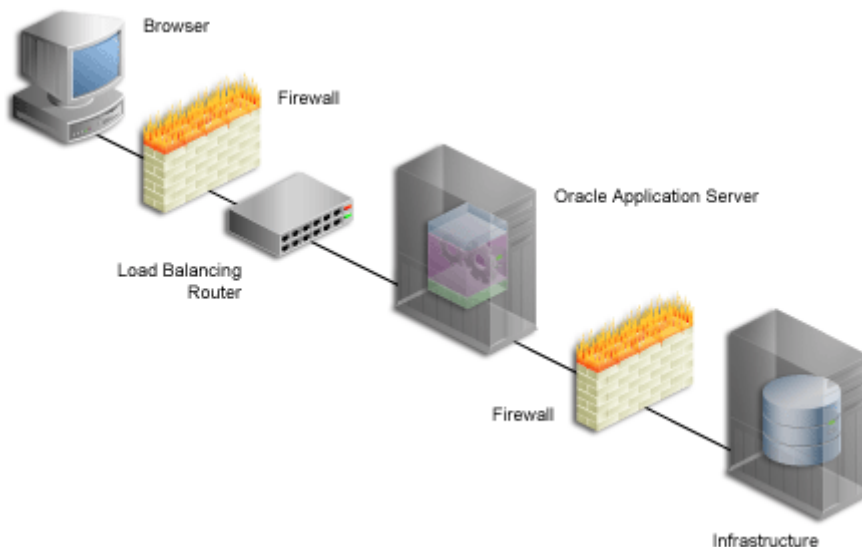
This configuration assumes that the application server and the infrastructure are installed on different machines.

When you install the entire infrastructure, no changes are made to the Portal Dependency Settings file.

As shown in Figure A–3, the OracleAS Web Cache cluster front-ending OracleAS Portal is not yet known. When you install the application server (Portal and

Wireless installation) on host **Host 1** in Oracle home `OH_1`, referencing the Oracle
Internet Directory on host **Host 2**, the configuration will look like this:

```
<IASConfig XSDVersion="1.0">

   <IASInstance Name="host2.OH_2" Host="host2.us.oracle.com" Version="9.0.4">
      <OIDComponent AdminPassword="orcladm" PortSSLEnabled="false"
LDAPPort="3002" AdminDN="cn=orcladmin"/>
   </IASInstance>

   <IASInstance Name="host1.OH_1" Host="host3.us.oracle.com" Version="9.0.4">
      <WebCacheComponent AdminPort="3001" ListenPort="7778"
InvalidationPort="3003" InvalidationUsername="orcladm"
InvalidationPassword="orcladm=" SSLEnabled="false"/>
   </IASInstance>

   <PortalInstance DADLocation="/pls/portal" SchemaUsername="portal"
SchemaPassword="welcome1" ConnectString="host1.us.oracle.com:1521:iasdb">
      <WebCacheDependency ContainerType="IASInstance" Name="host1.OH_1"/>
      <OIDDependency ContainerType="IASInstance" Name="host2.OH_2"/>
   </PortalInstance>

</IASConfig>
```

If you want the application server on **Host 1** to be front-ended by OracleAS Web
Cache (item 5 in the image), you need to manually edit the Portal Dependency
Settings file. First, remove the existing OracleAS Web Cache entry and then create
an OracleAS Web Cache entry that belongs to a farm. The modified Portal
Dependency Settings file will now look like this:

```
<IASConfig xmlns="http://www.oracle.com/iAS/iASConfigFile" XSDVersion="1.0">

   <IASInstance Name="host2.OH_2" Host="host2.us.oracle.com" Version="9.0.4">
      <OIDComponent AdminPassword="orcladm" PortSSLEnabled="false"
LDAPPPort="3002" AdminDN="cn=orcladmin"/>
   </IASInstance>

   <IASFarm name="Farm_1" host="frontend.us.oracle.com">
      <WebCacheComponent AdminPort="3001" ListenPort="7778"
InvalidationPort="3003" InvalidationUsername="orcladm"
InvalidationPassword="orcladm=" SSLEnabled="false"/>
   </IASFarm>

   <PortalInstance DADLocation="/pls/portal" SchemaUsername="portal"
SchemaPassword="welcome1" ConnectString="host1.us.oracle.com:1521:iasdb">
```

```
      <WebCacheDependency ContainerType="IASFarm" Name="Farm_1"/>
      <OIDDependency ContainerType="IASInstance" Name="host2.OH_2"/>
  </PortalInstance>

</IASConfig>
```

The OracleAS Portal instance now references the virtual OracleAS Web Cache front-ending it.

Finally, you must run `ptlconfig` to change the settings stored in the OracleAS Metadata Repository:

```
ptlconfig -encrypt
ptlconfig -all
```

A.2 Configuration Tools

The following sections describe the available configuration tools, and the state of the portal dependency settings file in various topologies, in more detail:

- Portal Dependency Settings Tool

- Oracle Enterprise Manager Application Server Control

A.2.1 Portal Dependency Settings Tool

To update the OracleAS Metadata Repository with configuration settings in `iasconfig.xml`, you must use the script `ptlconfig`. This script can:

- Update the OracleAS Metadata Repository for a *specific* Portal instance defined in the Portal Dependency Settings file.

- Update the OracleAS Metadata Repository for *all* Portal instances defined in the Portal Dependency Settings file.

- Encrypt all plain text passwords in the Portal Dependency Settings file.

- Update OracleAS Web Cache, Oracle Internet Directory, Oracle Enterprise Manager, and OracleAS Portal site data, as defined in the Portal Dependency Settings file.

The configuration script file is named `ptlconfig` (on UNIX) and `ptlconfig.bat` (on Windows). It is located in *ORACLE_HOME*/portal/conf, where *ORACLE_HOME* is the OracleAS Portal and OracleAS Wireless middle-tier home.

You can use this script as follows:

```
ptlconfig (-all | -dad <dad>) [-wc] [-oid] [-site] [-em] | -encrypt
```

Table A–10   ptlconfig Parameters

Parameter	Description	Example
-all	Updates all OracleAS Portal instances from the Portal Dependency Settings file.	`ptlconfig -all`
-dad	Portal DAD name. Used to update a specific OracleAS Portal instance from the Portal Dependency Settings file.	`ptlconfig -dad portal`
-encrypt	Encrypt any plain text passwords in the Portal Dependency Settings file.	`ptlconfig -encrypt`
-wc	Updates OracleAS Web Cache data as defined in the Portal Dependency Settings file.	`ptlconfig -dad portal -wc`
-oid	Updates Oracle Internet Directory data as defined in the Portal Dependency Settings file.	`ptlconfig -all -oid`
-site	Configures OracleAS Portal as a partner application for OracleAS Single Sign-On as defined in the Portal Dependency Settings file.	`ptlconfig -dad portal -site`
-em	Updates Oracle Enterprise Manager data as defined in the Portal Dependency Settings file.	`ptlconfig -dad portal -em`

When you run this script, the log file `ptlconfig.log` is created in the directory
ORACLE_HOME/portal/logs, which records operations performed on the
OracleAS Metadata Repository.

A.2.2 Oracle Enterprise Manager Application Server Control

Oracle Enterprise Manager Application Server Control also enables you to configure how an OracleAS Portal instance integrates with its dependent components. When you use Application Server Control to configure Portal Web Cache settings, the log file ptlemcfg.log is created in the directory *ORACLE_HOME*/portal/logs. This log file contains information about the operations performed on the OracleAS Metadata Repository. See Chapter 7, "Monitoring and Administering OracleAS Portal" for more information.

B

Using the OracleAS Portal Configuration Assistant Command Line Utility

OracleAS Portal Configuration Assistant (OPCA) is a Java-based configuration tool for installing and configuring the OracleAS Portal schema in the Oracle Application Server Metadata Repository.

In a typical Oracle Application Server installation, the Oracle Universal Installer (OUI) automatically invokes OPCA in the post-installation phase. OPCA can also be invoked standalone.

In an Oracle Application Server installation, the OracleAS Portal installation is performed in two phases:

- The OracleAS Metadata Repository installation option of the Oracle Application Server *Infrastructure* installation type includes a database with an OracleAS Portal schema. The OracleAS Portal schema can also be installed in an existing database through the Oracle Application Server Repository Creation Assistant (OracleAS RepCA).

  > **See Also:** Section 3.1, "How Does the Installation Process Work?" for more information on installing OracleAS Portal, and installing the portal repository in an existing database.

- The Oracle Application Server (middle-tier) installation type performs the required configuration for the middle-tier components, so they can use the OracleAS Metadata Repository.

OracleAS Portal provides the command line script `ptlasst` to invoke OPCA in standalone mode. This appendix describes the usage of `ptlasst`, and the configuration options available in standalone mode. Specific topics covered include:

- Using ptlasst

- ptlasst Modes

B.1 Using ptlasst

The scripts `ptlasst.csh` (UNIX) and `ptlasst.bat` (Windows NT/2000) are located in the *ORACLE_HOME*/`assistants` directory and can be used to run the OPCA standalone in different modes. To use `ptlasst`, perform the following steps:

On Windows NT/2000:

First, set the *ORACLE_HOME* environment variable to the Oracle Application Server Home. You can then start the OPCA from the command line by navigating to the *ORACLE_HOME*/`assistants/opca` directory and using the command:

```
ptlasst.bat -mode {PORTAL | SSO | MIDTIER | LANGUAGE | SYSOBJECTS | DEINSTALL }
{mode-dependent-parameters}
```

On UNIX:

First set the *ORACLE_HOME* environment variable to the Oracle Application Server Home. You can start the OPCA from the command line by navigating to the *ORACLE_HOME*/`assistants/opca` directory and using the command:

```
ptlasst.csh -mode {PORTAL | SSO | MIDTIER | LANGUAGE | SYSOBJECTS | DEINSTALL }
{mode-dependent-parameters}
```

Usage Notes:

- Mode names, like for example, MIDTIER, or LANGUAGE, must be passed to `ptlasst` in upper case.

- `ptlasst` generates a log file each time it is run. The log file generated is named *portal_schema_name*.`log`, and it is located in the directory *ORACLE_HOME*/`assistants/opca`.

- The file `ptlasst.README`, located in the directory *ORACLE_HOME*/`assistants/opca` contains the usage information for `ptlasst`, also listed in this appendix.

- To obtain schema password information, needed to run some of the `ptlasst` commands, like the portal, or OracleAS Single Sign-On partner application schema password, issue the following LDAP command:

```
ldapsearch -h directory_host_name -p directory_port -D directory_bind_dn -w
directory_bind_dn_password -b "orclReferenceName=infrastructure_database"
"orclresourcename=<schema_name>" orclpasswordattribute
```

Where `<schema_name>` is the name of the schema. For example, `PORTAL`, `ORASSO_PA`, or `ORASSO_PS`, as shown in Example B–1:

Example B–1   Obtaining the PORTAL Schema Password

```
ldapsearch -h m1.abc.com -p 389 -D "cn=orcladmin" -w welcome1 -b
"orclReferenceName=portal.abc.com,cn=IAS Infrastructure
Databases,cn=IAS,cn=Products,cn=oraclecontext" "orclresourcename=PORTAL"
orclpasswordattribute
```

Alternatively, you can use Oracle Directory Manager to obtain the password information. Drill down to **orclResourceName=schema_name**, where schema_name is the name of the schema for which you want to obtain the password. Click the entry and look for the `orclpasswordattribute` attribute value on the right panel. This value is the password for the selected schema.

> **See Also:**   *Oracle Internet Directory Administrator's Guide* for information on how to use Oracle Directory Manager.

B.2 ptlasst Modes

The following table contains a description of the different modes:

Table B–1   ptlasst Modes

Mode	Description
PORTAL	Installs the OracleAS Portal schema in the target database. You must run this mode from the Oracle Application Server Repository Creation Assistant CD.
MIDTIER	Configures the OracleAS Portal middle-tier to use an existing or newly installed OracleAS Portal schema.
LANGUAGE	Installs the strings for a specific language in the OracleAS Metadata Repository.
SYSOBJECTS	Installs the SYS schema dependencies, required for OracleAS Portal and OracleAS Single Sign-On. This mode only needs to be run once for every database.
DEINSTALL	Drops the OracleAS Portal schema, as well as the OracleAS Portal Oracle Internet Directory entries.

> **Usage Note:** This appendix uses the following conventions:
>
> **[]** indicates an optional parameter
>
> **{option1 | option2}** indicates a choice between option 1, or option 2 can be made.

B.2.1 PORTAL

Action

This mode installs the OracleAS Metadata Repository in the target database. No information from the middle-tier is required. It should be used for the OracleAS Portal seed database creation.

Environment

- OracleAS Portal sources for the repository installation are available only on the Oracle Application Server Repository Creation Assistant (RepCA) CD.

- The PORTAL mode can only be run from the OracleAS RepCA CD, and not from the *MID_TIER_ORACLE_HOME* in which OracleAS Portal is installed.

- Set the ORACLE_HOME environment variable to the Database Oracle home.

> **Note:** The prerequisites for running the OracleAS Metadata Repository installation, using the ptlasst script are:
>
> - PL/SQL LDAP packages must be installed in the database.
>
> - JVM option of the database must be configured.
>
> - Intermedia option of the database must be configured.
>
> - DB block size is at least 8 KB.
>
> - Shared pool size must be at least 100 MB.
>
> - Java pool size must be at least 64 MB.
>
> It is recommended to install the OracleAS Metadata Repository through the OracleAS RepCA tool, because the prerequisites are automatically checked during the installation.

Usage

```
ptlasst.csh -mode PORTAL -c connect_string -p sys_password -voh source_home -oh
oracle_home -log log_dir [-s portal_schema] [-u user_tablespace] [-t temp_
tablespace] [-d doc_tablespace] [-l logging_tablespace] [-in index_tablespace]
[-demo] [-owa]
```

Table B–2 lists and describes parameters supported for the PORTAL mode.

Table B–2   List of Supported Parameters for the PORTAL Mode

Parameter	Description
-s	Portal schema name (in the OracleAS Metadata Repository).
	Default: `portal`
-c	Connect string to the target database. The format should be `DbHostName:DbPortNumber:DbServiceName`.
-p	SYS password for the target database.
-u	User tablespace.
	The user tablespace selected should have at least 75 MB of free available space and should have *autoextend* on.
	Default: `users`
-t	Temporary tablespace.
	The temporary tablespace selected should have at least 20 MB of free available space and should have *autoextend* on.
	Default: `temp`
-d	Document tablespace.
	The document tablespace selected should have at least 4 MB of free available space and should have *autoextend* on.
	Default: Tablespace selected for the User tablespace
-l	Logging tablespace.
	The Logging tablespace selected should have at least 4 MB of free available space and should have *autoextend* on.
	DEFAULT: Tablespace selected for the User tablespace.
-in	Index tablespace.
	The Index tablespace selected should have at least 20 MB of free available space and should have *autoextend* on.
	Default: Tablespace selected for the User tablespace.

Table B–2 (Cont.) List of Supported Parameters for the PORTAL Mode

Parameter	Description
-demo	Installs the portlet builder demo components.
-voh	Oracle home location of the OracleAS Portal sources (OracleAS RepCA Home).
-oh	Oracle home of the database.
-log	Log directory location. This directory should have write permissions.
-owa	This parameter installs the PL/SQL Web Toolkit and other SYS schema packages. This is installed just once in the database.

Usage Example

```
ptlasst.csh -mode PORTAL -s portal -c myDBhost.domain.com:1521:dbServiceName -p
change_on_install -u users -t temp -d users -l users -in users -demo  -owa -voh
/private1/repca -log /private1/log -oh /private1/dbhome
```

> **Note:** Starting with the Oracle Application Server release 10*g*
> (9.0.4), the OracleAS Portal sources required for the repository
> installation will be available on the Oracle Application Server
> Repository Creation Assistant (RepCA) CD. The PORTAL OPCA
> mode can only be run directly from the OracleAS RepCA CD.

B.2.2 MIDTIER

Action

Configures OracleAS Portal middle-tier to use an existing OracleAS Portal schema.

The MIDTIER mode uses the option -type to determine which components to configure. The different *types* that are supported in the MIDTIER mode are:

- OID Type - Configures OracleAS Portal to work with Oracle Internet Directory. This type is used if configuration changes are required in OracleAS Portal due to changes in the Oracle Internet Directory component. For example, changes in the Oracle Internet Directory host, port, or protocol.

- SSO Type - Configures OracleAS Portal to work with OracleAS Single Sign-On. This type is used if configuration changes are required in OracleAS Portal due to changes in the OracleAS Portal published host, port, or protocol, as registered with Oracle Application Server Single Sign-On. Typically, the published host, port, and protocol, are that of OracleAS Web Cache, except in the case of a configuration where a load balancing router, or reverse proxy server is front-ending OracleAS Web Cache, or when using virtual hosts.

- WEBCACHE Type - Configures OracleAS Portal to work with Oracle Application Server Web Cache. This type is used if configuration changes are required in OracleAS Portal due to changes in the OracleAS Web Cache component. For example, changes in the OracleAS Web Cache hostname, listening port, invalidation port, invalidation password, or administration port.

- OHS Type - The OHS type is a superset of the SSO and WEBCACHE types. It configures OracleAS Portal to work with the Oracle HTTP Server, and to use the provider user interface and the provider group. This type is used if configuration changes are required in OracleAS Portal due to changes in the Oracle HTTP Server component. For example, changes in the HTTP server host, port, or protocol.

- ALL Type - Configures all of middle-tier components to work with the OracleAS Portal schema. The ALL type is a superset of the OID, SSO, WEBCACHE, and OHS types. This type is typically used to configure the OracleAS Portal middle-tier to use a new OracleAS Metadata Repository.

- DIPREG Type - Creates the provisioning profiles in Oracle Internet Directory.

- DIPUNREG Type - Deletes the provisioning profiles in Oracle Internet Directory.

> **Note:** If possible, use the Portal Dependency Settings file and tool to perform middle-tier configuration. If you use `ptlasst`, the Portal Dependency Settings file (`iasconfig.xml`) does not get updated, and using `iasconfig.xml` for subsequent configurations may cause your site to be misconfigured.
>
> After you update the `iasconfig.xml`, you must run the Portal Dependency Settings tool (`ptlconfig`) to update the OracleAS Metadata Repository with the configuration settings in `iasconfig.xml`.
>
> Shown subsequently is a mapping of various `ptlasst` MIDTIER types and their Portal Dependency Settings tool (`ptlconfig`) counterparts:
>
> - Instead of the WEBCACHE type, you can use `ptlconfig -dad <dad> -wc`.
>
> - Instead of the OID type, you can use `ptlconfig -dad <dad> -oid`.
>
> - Instead of the OHS type, you can use `ptlconfig -dad <dad> -site`.
>
> There are, however, some cases in which you do need to use OPCA in the MIDTIER mode. For example, in the scenario where your published hostname is different from the hostname used for OracleAS Web Cache invalidation, you cannot use the Portal Dependency Settings file, to establish this configuration.
>
> For more information about the Portal Dependency Settings file, and tool, refer to Appendix A, "Using the Portal Dependency Settings File".

Environment

- Set the ORACLE_HOME environment variable to the *MID_TIER_ORACLE_HOME* in which OracleAS Portal is installed.

- The MIDTIER mode should be run from *MID_TIER_ORACLE_HOME* in which OracleAS Portal is installed.

Assumptions

- OracleAS Portal and OracleAS Single Sign-On server are already installed.

■ Oracle Internet Directory, OracleAS Web Cache, and Oracle HTTP Server are up and running.

> **Note:** The MIDTIER mode is also used to synchronize or resynchronize OracleAS Portal users and groups with Oracle Internet Directory.

Usage

```
ptlasst.csh -mode MIDTIER [ -type {ALL | OID | SSO | OHS | WEBCACHE | DIPREG |
DIPUNREG} ] {type-dependent-parameters}
```

Table B–3 lists and describes parameters supported for the MIDTIER mode.

Table B–3   List of Supported Parameters for the MIDTIER Mode

Parameter	Description
-i	Installation type.
	This can be set to *typical* and *custom*. In the typical mode, the repository access APIs are used to get the details of Oracle Internet Directory, OracleAS Single Sign-On, and the OracleAS Portal schema in the configured OracleAS Metadata Repository.
	The repository access APIs provide infrastructure access details to the middle-tier component that it needs for its configuration.
	In the custom mode, the input provided on the command line is used for the configuration.
	Default: `typical`
-type	Middle-tier type. The available options are ALL, OID, SSO, WEBCACHE, DIPREG, and DIPUNREG.
	Default: `ALL`
-s	OracleAS Portal schema name.
	Default: `portal`
-sp	OracleAS Portal schema password.
-c	Connect string to the target OracleAS Portal database. The format should `DbHostName:DbPortNumber:DbServiceName`.

Table B–3   (Cont.) List of Supported Parameters for the MIDTIER Mode

Parameter	Description
`-sdad`	OracleAS Portal schema (in the OracleAS Metadata Repository) DAD name. Default: `portal`
`-o`	OracleAS Single Sign-On schema name. Default: `orasso`
`-op`	OracleAS Single Sign-On password. Default: `orasso`
`-odad`	OracleAS Single Sign-On DAD name. Default: `orasso`
`-host`	HTTP server hostname used for OracleAS Portal.
`-port`	HTTP server port number used for OracleAS Portal.
`-chost`	OracleAS Web Cache host. Default: HTTP Server hostname.
`-cport_i`	OracleAS Web Cache invalidation port.
`-cport_a`	OracleAS Web Cache administration port.
`-wc`	OracleAS Web Cache ON/OFF flag to enable or disable Web Cache within OracleAS Portal. Default: `ON` If set to `OFF`, OracleAS Portal would not use Web Cache though OracleAS Web Cache may be up and running.
`-ldap_h`	Hostname of the Oracle Internet Directory server.
`-ldap_p`	Port number of the Oracle Internet Directory server.
`-ldap_d`	Administration DN. Default: `cn=orcladmin`
`-ldap_w`	Password for DN.
`-pwd`	Initial password for OracleAS Portal seeded users (PORTAL and PORTAL_ADMIN) in Oracle Internet Directory.
`-sso_c`	Connect string for the OracleAS Single Sign-On database. The format should be `DbHostName:DbPortNumber:DbServiceName`.

Table B–3   (Cont.) List of Supported Parameters for the MIDTIER Mode

Parameter	Description
-ps	Password Store schema name.
	Default: `orasso_ps`
-pp	Password Store schema password.
-pa	Partner application schema name.
	Default: `orasso_pa`
-pap	Partner application password.
-wc_i_pwd	OracleAS Web Cache invalidator password.
	Default: Password of the Oracle Application Server instance (ias_admin password).
-ldaps	Flag to indicate that Oracle Internet Directory is SSL enabled.
-ultrasearch	Configure Oracle Ultra Search.
-syndication	Configure Syndication server and UDDI.
-ssl	Flag to indicate that OracleAS Portal needs to be SSL enabled.
-emport	Oracle Enterprise Manager Application Server Control port.
-iasname	Oracle Application Server instance name.

B.2.2.1 OID Type

```
ptlasst.csh -mode MIDTIER -type OID -ldap_w orcladmin_password -pwd ias_admin_
password [-i install_type] [ -s portal_schema] [-sp portal_schema_password] [-c
portal_db_connect_string] [-ldap_h oid_host_name] [-ldap_p oid_port_number]
[-ldap_d oid_admin_user] [-ldaps]
```

Usage example

Typical installation (-i typical)

Uses the repository access APIs to get the details of Oracle Internet Directory, and the OracleAS Portal schema in the configured OracleAS Metadata Repository.

```
ptlasst.csh -mode MIDTIER -type OID  -ldap_w welcome1 -pwd welcome1
```

Usage example

Custom installation (-i custom)

This example takes the input provided on the command line.

```
ptlasst.csh -mode MIDTIER -type ALL -i custom -ldap_w welcome1 -pwd welcome1 -s
portal -sp portal -c myDBhost.domain.com:1521:dbServiceName -ldap_h
myOID.domain.com -ldap_p 389
```

Note: If you are unsure whether to use Typical or Custom, use the Custom installation type.

B.2.2.2 SSO Type

```
ptlasst.csh -mode MIDTIER -type SSO -host portal_site_name -port portal_site_
port [-i install_type] [ -s portal_schema] [-sp portal_schema_password] [-sdad
portal_dad] [-c portal_db_connect_string] [-sso_c sso_db_connect_string] [-pap
partner_application_password] [-pp password_store_password]
```

OracleAS Portal is a partner application to the Oracle Application Server Single Sign-On. As such, it must be configured to use an OracleAS Single Sign-On for authentication services. When OracleAS Portal is installed, it is automatically configured to use the OracleAS Single Sign-On installed during the infrastructure installation.

Note: OracleAS Single Sign-On and OracleAS Portal from different versions cannot interoperate.

Due to the interdependency of the OracleAS Single Sign-On and OracleAS Portal with Oracle Internet Directory in Oracle Application Server 10*g* (9.0.4), you must not configure OracleAS Portal 10*g* (9.0.4) to use an OracleAS Single Sign-On Server (Login Server) from Oracle9*i*AS Release 1 (1.0.2.2) or earlier. Similarly, you must not configure Release 1-based versions of OracleAS Portal to use the current release of OracleAS Single Sign-On. The exception to this rule is that Portal version 3.0.9.8.4 and later can be configured to use a 9.0.x SSO server.

See Also:

- The Upgrades page on Portal Center,
  http://portalcenter.oracle.com/upgrades/.

- *Oracle Application Server 10g Migrating from Oracle Application Server*

What was called the ssodatax script, in versions 3.0.x of OracleAS Portal, has been obsoleted and replaced by running the OracleAS Portal Configuration Assistant in -mode MIDTIER -type SSO. When you install OracleAS Portal, the step previously done by ssodatan, is done automatically. However, after installation, there may be various reasons for configuring OracleAS Portal to use a different OracleAS Single Sign-On, or needing to re-run the configuration, because of a change in the hostname, port or protocol of the Oracle Application Server Single Sign-On.

Whereas the old ssodatax required you to set up the partner application entry in the SSO server and then invoke the script with the site_id, site_token, and encryption_key obtained from partner application registration, the -mode MIDTIER -type SSO mode of ptlasst.csh (OPCA) no longer requires partner application registration to be a two-step process.

The OracleAS Single Sign-On now provides a schema ORASSO_PA (default) for accessing the partner application registration procedure. You will need to get the password to this schema and an appropriate connect string to the OracleAS Single Sign-On instance to register the OracleAS Portal entry.

See Also: *Oracle Application Server Single Sign-On Administrator's Guide*

Usage example
Typical installation (-i typical)

Uses the repository access APIs to get the details of OracleAS Single Sign-On, and the OracleAS Portal schema in the configured OracleAS Metadata Repository.

```
ptlasst.csh -mode MIDTIER -type SSO -host mySite.domain.com -port 7777 -sdad
portal
```

Usage example
Custom installation (-i custom)

This example takes the input provided on the command line.

```
ptlasst.csh -mode MIDTIER -type SSO -i custom -host mySite.domain.com -port 7777
-s portal -sp portal -sdad portal -c myDBhost.domain.com:1521:dbServiceName
-sso_c myDBhost.domain.com:1521:dbServiceName -pap orasso_pa -pp orasso_ps
```

> **Note:** If you are unsure whether to use Typical or Custom, use the Custom installation type.

B.2.2.3 WEBCACHE Type

```
ptlasst.csh -mode MIDTIER -type WEBCACHE -host portal_site_name -port portal_
site_port -cport_i webcache_invalidation_port -cport_a webcache_administration_
port -wc_i_pwd webcache_invalidator_password [-chost webcache_hostname] [-i
install_type] [ -s portal_schema] [-sp portal_schema_password] [-sdad portal_
dad] [-c portal_db_connect_string] [-wc webcache_on_off_flag]
```

Usage example
Typical installation (-i typical)

Uses the repository access APIs to get the details of the OracleAS Portal schema in the configured OracleAS Metadata Repository.

```
ptlasst.csh -mode MIDTIER -type WEBCACHE -host mySite.domain.com -port 7777
-cport_i 4001 -cport_a 4001  -wc_i_pwd webcache_invalidator_password
```

Usage example
Custom installation (-i custom)

This example takes the input provided on the command line.

```
ptlasst.csh -mode MIDTIER -type WEBCACHE -i custom -host mySite.domain.com -port
7777 -cport_i 4001 -cport_a 4001 -s portal -sp portal -sdad portal -c
myDBhost.domain.com:1521:dbServiceName -wc_i_pwd webcache_invalidator_password
```

> **Note:** If you are unsure of the Typical/Custom categorization, use the Custom installation type.

B.2.2.4 OHS Type

```
ptlasst.csh -mode MIDTIER -type OHS -host portal_site_name -port portal_site_
```

```
port -cport_i webcache_invalidation_port -cport_a webcache_administration_port
-wc_i_pwd webcache_invalidator_password [-chost webcache_hostname] [-i install_
type] [ -s portal_schema] [-sp portal_schema_password] [-sdad portal_dad] [-c
portal_db_connect_string] [-sso_c sso_db_connect_string] [-pap partner_
application_password] [-pp password_store_password] [-wc webcache_on_off_flag]
[-ssl]
```

Usage example

Typical installation (-i typical)

Uses the repository access APIs to get the details of the OracleAS Portal schema in the configured OracleAS Metadata Repository.

```
ptlasst.csh -mode MIDTIER -type OHS -host mySite.domain.com -port 7777 -cport_i
4001 -cport_a 4001 -wc_i_pwd webcache_invalidator_password
```

Usage example

Custom installation (-i custom)

This example takes the input provided on the command line.

```
ptlasst.csh -mode MIDTIER -type OHS -i custom -host mySite.domain.com -port 7777
-cport_i 4001 -cport_a 4001  -s portal -sp portal -sdad portal -c
myDBhost.domain.com:1521:dbServiceName -sso_c
myDBhost.domain.com:1521:dbServiceName -pap orasso_pa -pp orasso_ps -wc_i_pwd
webcache_invalidator_password
```

> **Note:** If you are unsure whether to use Typical or Custom, use the Custom installation type.

B.2.2.5 ALL Type

```
ptlasst.csh -mode MIDTIER -type ALL -host portal_site_name -port portal_site_
port -cport_i webcache_invalidation_port -cport_a webcache_administration_port
-ldap_w orcladmin_password -pwd ias_admin_password  -emport em_port_number -wc_
i_pwd webcache_invalidator_password -iasname ias_instance [-chost webcache_
hostname] [-i install_type] [ -s portal_schema] [-sp portal_schema_password]
[-sdad portal_dad] [-c portal_db_connect_string] [-sso_c sso_db_connect_string]
[-pap partner_application_password] [-pp password_store_password] [-wc webcache_
on_off_flag] [-ldap_h oid_host_name] [-ldap_p oid_port_number] [-ldap_d oid_
admin_user] [-ldaps] [-ultrasearch] [-syndication] [-ssl]
```

> **Note:** The -ultrasearch and -syndication options can only be used with the **typical** install option (-i typical).

Usage example

Typical installation (-i typical)

Uses the repository access APIs to get the details of Oracle Internet Directory, OracleAS Single Sign-On, and the OracleAS Portal schema in the configured OracleAS Metadata Repository.

```
ptlasst.csh -mode MIDTIER -type ALL -host mySite.domain.com -port 7777 -cport_i
4001 -cport_a 4001 -ldap_w welcome1 -pwd welcome1 -wc_i_pwd webcache_
invalidator_password -emport 1812 -iasname as_midtier
```

Usage example

Custom installation (-i custom)

This example takes all the input provided on the command line. This should be used if the configuration is performed on an OracleAS Portal instance other than the out-of-the-box installation.

```
ptlasst.csh -mode MIDTIER -type ALL -i custom -host mySite.domain.com -port 7777
-cport_i 4001 -cport_a 4001 -ldap_w welcome1 -pwd welcome1  -s portal -sp portal
-sdad portal -c  myDBhost.domain.com:1521:dbServiceName -sso_c
myDBhost.domain.com:1521:dbServiceName -pap orasso_pa -pp orasso_ps -wc_i_pwd
webcache_invalidator_password -ldap_h myOID.domain.com -ldap_p 389 -emport 1812
-iasname as_midtier
```

> **Note:** If you are unsure whether to use Typical or Custom, use the Custom installation type.

B.2.2.6 DIPREG Type

This type should be used to create the provisioning profiles in Oracle Internet Directory. Refer to Section 6.1.6.3, "Relationship Between OracleAS Portal and Oracle Internet Directory" for more information about provisioning profiles.

```
ptlasst.csh -mode MIDTIER -type DIPREG -ldap_w orcladmin_password [-i install_
type] [ -s portal_schema] [-sp portal_schema_password] [-c portal_db_connect_
```

string] [**-ldap_h** oid_host_name] [**-ldap_p** oid_port_number] [**-ldap_d** oid_admin_
user]

Usage example
Typical installation (-i typical)

Uses the repository access APIs to get the details of Oracle Internet Directory, and
the OracleAS Portal schema in the configured OracleAS Metadata Repository.

```
ptlasst.csh -mode MIDTIER -type DIPREG  -ldap_w welcome1
```

Usage example
Custom installation (-i custom)

This example takes the input provided on the command line.

```
ptlasst.csh -mode MIDTIER -type DIPREG -i custom -ldap_w welcome1 -s portal -sp
portal -c myDBhost.domain.com:1521:dbServiceName -ldap_h myOID.domain.com -ldap_
p 389
```

> **Notes:**
>
> - If you are unsure whether to use Typical or Custom, use the
>   Custom installation type.
>
> - The DIPREG install type can also be used to create the
>   provisioning profile for a 9.0.2 repository.

B.2.2.7 DIPUNREG Type

This type should be used to delete the provisioning profiles in Oracle Internet
Directory of the OracleAS Portal instance. Refer to Section 6.1.6.3, "Relationship
Between OracleAS Portal and Oracle Internet Directory" for more information about
provisioning profiles.

```
ptlasst.csh -mode MIDTIER -type DIPUNREG -ldap_w orcladmin_password [-i install_
type] [ -s portal_schema] [-sp portal_schema_password] [-c portal_db_connect_
string] [-ldap_h oid_host_name] [-ldap_p oid_port_number] [-ldap_d oid_admin_
user]
```

Usage example

Typical installation (-i typical)

Uses the repository access APIs to get the details of Oracle Internet Directory, and the OracleAS Portal schema in the configured OracleAS Metadata Repository.

```
ptlasst.csh -mode MIDTIER -type DIPUNREG  -ldap_w welcome1
```

Usage example

Custom installation (-i custom)

This example takes the input provided on the command line.

```
ptlasst.csh -mode MIDTIER -type DIPUNREG -i custom -ldap_w welcome1 -s portal
-sp portal -c myDBhost.domain.com:1521:dbServiceName -ldap_h myOID.domain.com
-ldap_p 389
```

> **Note:** If you are unsure whether to use Typical or Custom, use the Custom installation type.

> **See Also:** *Oracle Application Server 10g Administrator's Guide*

B.2.3 LANGUAGE

Action

Installs the strings for a specific language in the OracleAS Portal schema.

OracleAS Portal is designed to allow application development and deployment in different languages. This allows developers to work in their own language when they build portals. In addition, the self-service content management supports multiple languages so that end users can provide documents and other content in different languages.

OracleAS Portal is configured with the languages that are selected in the Oracle Universal Installer (OUI) during the Oracle Application Server middle-tier installation. Languages that are configured show up in the **Set Language** portlet. You can use OracleAS Portal in the language that corresponds to the language setting in the browser, or to the language you have selected in the **Set Language** portlet. To configure additional languages after installation, the OracleAS Portal Configuration Assistant (OPCA) must be used in LANGUAGE mode.

Table B–4 shows the languages that are available for OracleAS Portal.

Table B–4   OracleAS Portal Languages

Language	Language Abbreviation
Arabic	ar (ARABIC_UNITED ARAB EMIRATES.AR8MSWIN1256)
Czech	cs (CZECH_CZECH REPUBLIC.EE8MSWIN1250)
German	d (GERMAN_GERMANY.WE8MSWIN1252)
Danish	dk (DANISH_DENMARK.WE8MSWIN1252)
Spanish	e (SPANISH_SPAIN.WE8MSWIN1252)
Greek	el (GREEK_GREECE.EL8MSWIN1253)
Latin American Spanish	esa (SPANISH_SPAIN.WE8MSWIN1252)
French	f (FRENCH_FRANCE.WE8MSWIN1252)
Canadian French	frc (FRENCH_FRANCE.WE8MSWIN1252)
Hebrew	iw (HEBREW_ISRAEL.IW8MSWIN1255)
Hungarian	hu (HUNGARIAN_HUNGARY.EE8MSWIN1250)
Italian	i (ITALIAN_ITALY.WE8MSWIN1252)
Japanese	ja (JAPANESE_JAPAN.JA16SJIS)

Table B–4   (Cont.) OracleAS Portal Languages

Language	Language Abbreviation
Korean	ko (KOREAN_KOREA.KO16KSC5601)
Norwegian	n (NORWEGIAN_NORWAY.WE8MSWIN1252)
Dutch	nl (DUTCH_THE NETHERLANDS.WE8MSWIN1252)
Polish	pl (POLISH_POLAND.EE8MSWIN1250)
Portuguese	pt (PORTUGUESE_PORTUGAL.WE8MSWIN1252)
Brazilian Portuguese	ptb (BRAZILIAN PORTUGUESE_BRAZIL.WE8MSWIN1252)
Romanian	ro (ROMANIAN_ROMANIA.EE8MSWIN1250)
Russian	ru (RUSSIAN_CIS.CL8MSWIN1251)
Swedish	s (SWEDISH_SWEDEN.WE8MSWIN1252)
Finnish	sf (FINNISH_FINLAND.WE8MSWIN1252)
Slovak	sk (SLOVAK_SLOVAKIA.EE8MSWIN1250)
Turkish	tr (TURKISH_TURKEY.TR8MSWIN1254)
Thai	th (THAI_THAILAND.TH8TISASCII)
Simplified Chinese	zhs (SIMPLIFIED CHINESE_CHINA.ZHS16GBK)

Table B–4   (Cont.) OracleAS Portal Languages

Language	Language Abbreviation
Traditional Chinese	zht (TRADITIONAL CHINESE_TAIWAN.ZHT16BIG5)

To install languages, after you have installed OracleAS Portal, run `ptlasst` in the LANGUAGE mode. You must run `ptlasst` with `-mode LANGUAGE` for each language that you want OracleAS Portal to support.

> **Caution:**   During login operations, information is sent to OracleAS Single Sign-On. The language used in the authentication request is sent back to OracleAS Portal. OracleAS Single Sign-On must have all languages installed that exist on the OracleAS Portal, so that the selected language is recognized. If OracleAS Single Sign-On does not have the selected language installed, it will default to **US English**. This is the language that would be asserted to any OracleAS Portal that requested authentication in a language that is not available on the OracleAS Single Sign-On server.
>
> The **Set Language** portlet in OracleAS Portal sets a language and a Persistent Language cookie on the OracleAS Single Sign-On server and OracleAS Portal.
>
> If there are multiple portals configured to use the same OracleAS Single Sign-On, and the portals have different languages installed, all the combined languages must exist on the OracleAS Single Sign-On to accommodate a Set Language request from any of the portals.

Environment

- Set the ORACLE_HOME environment variable to the *MID_TIER_ORACLE_HOME* in which OracleAS Portal is installed.

- The LANGUAGE mode must be run from the *MID_TIER_ORACLE_HOME* in which OracleAS Portal is installed.

Assumptions

OracleAS Metadata Repository is already installed, and the respective databases are up.

Usage

```
ptlasst.csh -mode LANGUAGE  -lang lang_code [-i install_type] [ -s portal_
schema] [-sp portal_schema_password] [-c portal_db_connect_string] [-available]
```

Table B–5 lists and describes parameters supported for the LANGUAGE mode.

Table B–5   List of Supported Parameters for the LANGUAGE Mode

Parameter	Definition
-i	Installation type.
	This can be set to *typical* and *custom*. In the typical mode, the repository access APIs are used to get the details of Oracle Internet Directory, OracleAS Single Sign-On, and the OracleAS Portal schema in the configured OracleAS Metadata Repository.
	The repository access APIs provide infrastructure access details to the middle-tier component that it needs for its configuration.
	In the custom mode, the input provided on the command line is used for the configuration.
	Default: typical
-s	OracleAS Portal schema name.
	Default: portal
-sp	OracleAS Portal schema password.
-c	Connect string to the target database where OracleAS Metadata Repository is installed. The format must be DbHostName:DbPortNumber:DbServiceName.
-lang	Abbreviation for the language to install. Refer to Table B–4, " OracleAS Portal Languages" for a list of all the supported abbreviations.
	Default: f
-available	Sets whether the language will be available for user translation.

Usage example

Typical installation (-i typical)

Uses the repository access APIs to get the details of the OracleAS Portal schema in the configured OracleAS Metadata Repository. The following example loads the Dutch language strings into the OracleAS Metadata Repository.

```
ptlasst.csh -mode LANGUAGE -lang nl -available
```

Usage example

Custom installation (-i custom)

This example passes in the input provided on the command line. The example loads the Dutch language strings into the OracleAS Metadata Repository.

```
ptlasst.csh -mode LANGUAGE -i custom -s portal -sp portal -c
myDBhost.domain.com:1521:dbServiceName  -lang nl -available
```

> **Note:** The character set for mod_plsql must be the same as the customer database character set. Refer to the *Oracle Application Server 10g mod_plsql User's Guide* for more information.

> **See Also:** *Oracle Application Server 10g Globalization Guide*

B.2.4 SYSOBJECTS

Action

Installs OracleAS Portal and OracleAS Single Sign-On required SYS schema dependencies as follows:

- Installs PL/SQL Web Toolkit (OWA) packages
- Installs VPD Context packages

> **Note:**
>
> - This mode has to be run only once for every database.
>
> - The -owa option in the PORTAL, and SSO modes provide the same functionality.
>
> - For information on obtaining the OWA package version, refer to the Frequently Asked Questions section of the *Oracle Application Server 10g mod_plsql User's Guide*.

Environment

- Sources for the SYSOBJECTS mode are only available in the OracleAS RepCA CD.

- The SYSOBJECTS mode can only be run from the OracleAS RepCA CD. This mode cannot be run from the *MID_TIER_ORACLE_HOME* on which OracleAS Portal is installed.

- Set the ORACLE_HOME environment variable to the Database *ORACLE_HOME*.

Usage

```
ptlasst.csh -mode SYSOBJECTS -c connect_string -p sys_password -voh source_home
```

Table B–6 lists and describes parameters supported for the SYSOBJECTS mode.

Table B–6   List of Supported Parameters for the SYSOBJECTS Mode

Parameter	Description
-c	Connect string to the target database. The format should be DbHostName:DbPortNumber:DbServiceName.
-p	SYS schema password for the target database.
-voh	Oracle home location of the sources (OracleAS RepCA Home).

Usage example

```
ptlasst.csh -mode SYSOBJECTS -c myDBhost.domain.com:1521:dbServiceName -p
change_on_install -voh /private1/repca
```

B.2.5 DEINSTALL

Action

Drops the OracleAS Portal schema, as well as the OracleAS Portal Oracle Internet Directory entries.

Environment

- The ORACLE_HOME environment variable is set to the *MID_TIER_ORACLE_ HOME* in which OracleAS Portal is installed.

- DEINSTALL mode should be run from the *MID_TIER_ORACLE_HOME* in which OracleAS Portal is installed.

- OracleAS Metadata Repository already exists and the respective databases are up.

- Oracle Internet Directory is up and running.

Usage

```
ptlasst.csh -mode DEINSTALL  -ldap_w orcladmin_password [-i install_type] [ -s
portal_schema] [-p portal_schema_password] [-c portal_db_connect_string]
[-ldap_h oid_host_name] [-ldap_p oid_port_number] [-ldap_d oid_admin_user]
```

Table B–7 lists and describes parameters supported for the DEINSTALL mode.

Table B–7   List of Supported Parameters for the DEINSTALL Mode

Parameter	Description
-i	Installation type.
	This can be set to *typical* and *custom*. In the typical mode, the repository access APIs are used to get the details of Oracle Internet Directory, OracleAS Single Sign-On, and the OracleAS Portal schema in the configured OracleAS Metadata Repository.
	The repository access APIs provide infrastructure access details to the middle-tier component that it needs for its configuration.
	In the custom mode, the input provided on the command line is used for the configuration.
	Default: typical.

Table B–7   (Cont.) List of Supported Parameters for the DEINSTALL Mode

Parameter	Description
-s	OracleAS Portal schema name. Default: `portal`
-c	Connect string to the target database. The format should be `DbHostName:DbPortNumber:DbServiceName`.
-p	SYS schema password of the target database.
-ldap_h	Hostname of the Oracle Internet Directory server.
-ldap_p	Port number of the Oracle Internet Directory server.
-ldap_w	Password of the Administration DN.

Usage example

Typical deinstall (-i typical)

Uses the repository access APIs to get the details of Oracle Internet Directory, and the OracleAS Portal schema in the configured OracleAS Metadata Repository.

```
ptlasst.csh -mode DEINSTALL -ldap_w welcome1
```

Usage example

Custom deinstall (-i custom)

This example takes the inputs provided on the command line.

```
ptlasst.csh -mode DEINSTALL -i custom -s portal -p change_on_install -c
myDBhost.domain.com:1521:dbServiceName -ldap_h myOID.domain.com -ldap_p 389
-ldap_w welcome1
```

C

Using OracleAS Portal Installation and Configuration Scripts

After installing OracleAS Portal as part of the Oracle Application Server installation, several scripts are available for post-installation configuration.

The specific topics covered in this appendix include:

- OracleAS Web Cache Configuration Scripts
- Disabling the IP Check of Cookie Validation
- Using the secupoid.sql Script
- Using the secjsdom.sql Script
- Configuring the Portal Session Cookie
- Managing the Session Cleanup Job
- Timing and Caching Statistics
- Using the cfgiasw Script to Configure Mobile Settings
- Using the ptlinvsw.sql Script to Invalidate Portal Container Pages

C.1 OracleAS Web Cache Configuration Scripts

This section shows how you can choose to run OracleAS Web Cache configuration scripts to configure OracleAS Portal to work with OracleAS Web Cache. You can use this method in lieu of running OPCA in the MIDTIER mode to adjust OracleAS Web Cache specific settings, such as the OracleAS Web Cache host, or OracleAS Web Cache invalidation port. Furthermore, this section describes how you can disable

OracleAS Web Cache and manage the invalidation message processing job using the script `cachjsub.sql`.

Specific topics covered in this section include:

- Using cachset.sql
- Managing the Invalidation Message Processing Job Using cachjsub.sql

C.1.1 Using cachset.sql

The script `cachset.sql` is used to turn on or off the use of OracleAS Web Cache. The script can be found in the *ORACLE_HOME*/portal/admin/plsql/wwc directory.

To use `cachset.sql`, connect to SQL*Plus as the schema owner and run `cachset.sql` as follows:

```
SQL>@cachset.sql
```

At the prompt, enter *on*, to enable the use of OracleAS Web Cache, and *off*, to disable it.

C.1.2 Managing the Invalidation Message Processing Job Using cachjsub.sql

OracleAS Portal uses caching to improve its performance. One type of caching it uses is the invalidation-based caching. In invalidation-based caching, OracleAS Portal caches various objects (pages, portlets, and so on) for a set amount of time. When these objects are requested, they are retrieved from the cache, if available; otherwise they are regenerated from the Oracle Application Server Metadata Repository. The cache for these objects will expire when the *maxcache* time has been reached, or when the objects are explicitly invalidated (expired) by invalidation messages.

OracleAS Portal uses invalidation messages when it needs to expire objects in the cache. Invalidation messages are categorized as hard and soft invalidations. Hard invalidations take effect immediately, that is, the objects that they intend to invalidate expire from cache immediately. Soft invalidations take effect when they are processed by the invalidation processing job. The frequency by which the invalidation job executes is configurable. This is done using the `cachjsub.sql` script.

To change the execution frequency of the invalidation processing job:

1. Locate the following directory:

ORACLE_HOME/portal/admin/plsql/wwc

2. On the database where the Portal schema is installed, log on to SQL*Plus with the appropriate user name and password for that schema.

For example:

```
sqlplus portal/portal
```

3. Enter the following command to update the execution frequency of the invalidation job:

```
SQL> @cachjsub.sql <start_time> <start_time_fmt> <interval_mins>
```

`cachjsub.sql` takes three parameters:

- *start_time* is either when the first job should be run or START.
- *start_time_fmt* is the date format to be applied to the value of *start_time*.
- *interval_mins* is how many minutes each run is scheduled apart.

> **Note:** If START is provided for the first parameter, the second parameter is ignored, and it will default the start time to the current time.

Example 1:

```
SQL> @cachjsub.sql START null 120
```

Example 2:

```
SQL> @cachjsub.sql '02-22-2003 7:30' 'MM-DD-YYYY HH:MI' 1440
```

C.2 Disabling the IP Check of Cookie Validation

As part of the process of validating the session cookie of a user's request (even if that user is PUBLIC), Portal performs a comparison between the IP address stored in the cookie with the IP address of the current client. Only if the two values are the same will OracleAS Portal consider the request legitimate.

When a proxy exists between the user's client and the portal, the IP address stored in the session cookie is that of the proxy, and not that of the client.

Some proxy systems make use of multiple servers, each with different IP addresses. In these circumstances, it is conceivable that the original request from a user's client (the request that causes the session cookie to be created) is routed through one proxy server and that a subsequent request is routed through another, separate, proxy server. In these cases, the IP addresses compared by OracleAS Portal will differ. The request will raise a security violation during the IP checking step. And access to the page will be denied.

Depending on the network configuration into which the Oracle Application Server is installed, it may be necessary to disable IP checking in cookie validation.

To change the state of IP checking in cookie validation, you must use SQL*Plus to update data in both the portal schema and the SSO schema as detailed in Table C–1.

Table C–1   Enabling and Disabling the IP Check

	Portal Schema	SSO Schema
Enable IP Checking	update wwsec_enabler_ config_info$ set url_cookie_ip_check = 'Y'; commit;	update wwsec_enabler_ config_info$ set url_cookie_ip_check ='Y'; update wwsso_ls_ configuration_info$ set cookie_ip_check = 'Y'; commit;
Disable IP Checking	update wwsec_enabler_ config_info$ set url_cookie_ip_check = 'N'; commit;	update wwsec_enabler_ config_info$ set url_cookie_ip_check ='N'; update wwsso_ls_ configuration_info$ set cookie_ip_check = 'N'; commit;

C.3 Using the secupoid.sql Script

By default, OracleAS Portal connects to Oracle Internet Directory using LDAP without SSL. If the Oracle Internet Directory server is configured for an SSL port, though, OracleAS Portal can be configured to use LDAP over SSL, also known as LDAPS.

See Also: *Oracle Internet Directory Administrator's Guide*

To configure OracleAS Portal to use SSL to connect to Oracle Internet Directory, you must run the `secupoid.sql` script. This script enables you to change the following OracleAS Portal configuration parameters related to Oracle Internet Directory:

- Oracle Internet Directory host name

- Oracle Internet Directory port

- application Oracle Internet Directory password

- SSL setting

When you install OracleAS Portal, it is automatically configured to use an Oracle Internet Directory server. However, you may want to change some settings, such as whether to use SSL, after installation. To change to an SSL connection for Oracle Internet Directory, simply run the `secupoid.sql` script in the PORTAL schema to specify the LDAPS port instead of the LDAP port, and indicate that you want to use SSL.

Running the secupoid.sql Script

This section illustrates a sample execution of `secupoid.sql` from SQL*Plus.

In the example, Oracle Internet Directory was initially configured to run LDAP on port 389. Later, an LDAPS port was activated on 636. Since the server name does not change, we retain the old value, update the port, and indicate that we want to use SSL by setting the `Use SSL?` value to `Y`. When you run the script, it displays the current configuration and lets you replace any of the configurable settings. The script also enables you to update OracleAS Portal's Oracle Internet Directory cache after running it. Since activating SSL does not change any of the Oracle Internet Directory information cached by OracleAS Portal, it is not usually necessary to refresh the cache in this case.

```
SQL> @secupoid
Current Configuration
--------------------
OID Host: oid.domain.com
OID Port: 389
Application DN:
orclApplicationCommonName=PORTAL,cn=Portal,cn=Products,cn=OracleContext
Application Password: 3E8C2D1B87CB61011757239C5AA9B390
Use SSL? N
```

```
PL/SQL procedure successfully completed.

Updating OID Configuration Entries
Press [Enter] to retain the current value for each parameter
For SSL Connection to LDAP, specify "Y"es or "N"o
----------------------------------------------
Enter value for oid_host:
Enter value for oid_port: 636
Enter value for app_password:
Enter value for use_ssl_to_connect_to_ldap: Y
Enter value for refresh_with_new_settings: N

PL/SQL procedure successfully completed.

No errors.
```

After executing the script, OracleAS Portal is configured for LDAPS access of Oracle Internet Directory.

> **See Also:**   *Oracle Application Server 10g Security Guide*

C.4 Using the secjsdom.sql Script

If you have your Oracle Internet Directory and OracleAS Portal servers residing in different domains, you must explicitly set the JavaScript domain for OracleAS Portal such that it can resolve user and group lists of values. To do this, you must use the secjsdom.sql script located in the directory *ORACLE_HOME*/portal/admin/plsql/wwc.

Suppose your installation has OracleAS Portal configured to use an Oracle HTTP Server other than DAS. In this situation, you must have a common domain, so that the values can be transferred from the list of values displayed by DAS to the page displayed by OracleAS Portal.

To create a single domain in this case, do the following:

1. Login to SQL*Plus as PORTAL.

2. Run the following SQL script:

    ```
    SQL> @secjsdom.sql <domain_name>
    ```

If, in the preceding example, the DAS servlet is running on a machine infra.acme.com and OracleAS Portal is running on a machine portal.acme.com, then the secjsdom.sql script should be invoked like this:

```
@ SQL> @secjsdom.sql acme.com
```

Performing this procedure enables you to run Oracle Internet Directory lists of values from OracleAS Portal in either Netscape, or Microsoft Internet Explorer. When using lists of values, a transit window is displayed in addition to the list of values itself. The transit window is required to pass values to OracleAS Portal without forcing pages to reset their domain.

> **See Also:** *Oracle Application Server 10g Security Guide*

C.5 Configuring the Portal Session Cookie

OracleAS Portal uses a session cookie to maintain session state for portal applications. For portal to work correctly, the client browser must be configured to accept cookies from the server. Upon installation, the portal session cookie has a default name, scope, and security that are set appropriately for most installations. This section describes these defaults, and how they can be changed if needed.

C.5.1 Configuring the Cookie Name

By default the portal's session cookie is named `portal` after the default Database Access Descriptor (DAD) used to access the Portal schema. You can use Oracle Enterprise Manager to change the cookie name, if it needs to explicitly be set to something else. To do this, you must access the **DAD Edit** page in the Oracle Enterprise Manager Application Server Control. This page is located under **mod_plsql services** of the Portal middle-tier component. The cookie name can be set on the **Document Alias and Session Parameter** page. To change the name of the cookie, provide the desired name in the **Session Cookie Name** field of the Session Cookie section.

C.5.2 Configuring the Scope of the Cookie

> **Note:** You should make these changes when there is no traffic on the portal, otherwise existing sessions will experience session errors (ORA-20000) after you change the session cookie name.

In cases where you want access to the same portal from two middle-tiers at the same time, or if you want to open the portal cookie domain as required by the PL/SQL Adapter functionality, you must define the scope of the OracleAS Portal

session cookie to be sent to all the middle-tier servers involved in the architecture. By default, the session cookie's domain is scoped to the host from which it was generated. The path for the cookie is set to "/".

For example, if the cookie was generated from `www.oracle.com`, then the cookie domain is `www.oracle.com`. However, let's say that another server, `portal.oracle.com` is also a middle-tier server that needs access to that session cookie. Then the cookie domain would need to be widened so that the `portal.oracle.com` server can also see the cookie.

Follow these steps to modify the scope of the portal session cookie:

1. Locate the following directory:

   `ORACLE_HOME/portal/admin/plsql/wwc`

2. On the database where your OracleAS Portal schema is installed, log on to SQL*Plus as the portal schema. For example:

   `sqlplus portal/portal_pwd`

3. Enter the following command:

   ```
   SQL> @ctxckupd
   OracleAS Portal
   Current Settings for Portal Session Cookie:
   Cookie Domain : Only send cookie back to originating host:port
   Set Cookie as Secure: Y
   Enter the domain for the session cookie: .oracle.com
   Should cookie be flagged as secure for HTTPS sessions? (Y/N): N
   Settings changed to
   Cookie Domain : .oracle.com
   Do not set cookie as secure. (N)
   SQL>
   ```

   This enables you to set the cookie domain for the session cookie. In this example, the cookie domain is set to `.oracle.com`.

> **Note:** If you want to use different listeners or keep the session
> cookie throughout different domains, specify a Cookie Domain to
> be the host name only. For example, if you access OracleAS Portal
> from two machines:
>
> - `machine1.us.oracle.com:3000`
>
> - `machine2.us.oracle.com:4000`
>
> When running `ctxckupd.sql`, set the cookie domain to
> `.us.oracle.com`.

> **Note:** The cookie domain also determines the scope of the NLS_
> LANGUAGE cookie, which is a persistent cookie that determines
> the user's preferred language. This NLS_LANGUAGE cookie is set
> when selecting languages in the set language portlet.

C.5.3 Securing the Cookie

In this release of OracleAS Portal, the script `ctxckupd.sql` contains an additional
option, *Set Cookie as Secure*.

The default location for this script is *ORACLE_HOME*/portal/admin/plsql/wwc.
When you run this script, you see the following output:

```
SQL> @ctxckupd
OracleAS Portal
Current Settings for Portal Session Cookie:
Cookie Domain : Only send cookie back to originating host:port
Set Cookie as Secure: Y
Enter the domain for the session cookie...
Leave blank to scope to originating host:
Should cookie be flagged as secure for HTTPS sessions? (Y/N): N
Settings changed to
Cookie Domain : Only send cookie back to originating host:port
Do not set cookie as secure. (N)
SQL>
```

Set Cookie as Secure indicates that the cookie should be sent back to the server if the
request is over an **HTTPS** connection only. This setting ensures that the session
cookie is not transmitted over an insecure connection when it needs to be protected.
By default, this option is set to *Yes* and is sufficient for most deployments.

In some cases, you may need to set the *Set Cookie as Secure* option to *No*. For example, if your portal is accessed over both **HTTP** and **HTTPS** and you want the session cookie to be shared across both protocols (possible if they are running on the default ports 80 (http) and 443 (https)). In this instance, when *Set Cookie as Secure* is set to *No*, the same cookie produced over an https request, is sent over any subsequent http requests.

C.6 Managing the Session Cleanup Job

OracleAS Portal and OracleAS Single Sign-On perform session management similar to other Web-based applications. Sessions are tracked with cookies. Session information is stored in a table in the Portal and OracleAS Single Sign-On schema. When a user logs out, the session information is marked inactive. A DBMS job subsequently cleans up the inactive rows.

The session table accumulates a number of rows that are flagged as active. When a user shuts down the browser instead of logging out, the row is "active", even though it is not actually in use. The cleanup job cleans up the active rows that are older than a specified duration.

When OracleAS Portal is installed, a DBMS job is installed to perform session cleanup of the session table, `WWCTX_SSO_SESSION$`. The cleanup job is set to run every 24 hours. The first scheduled cleanup occurs 24 hours after the installation of the job.

When the job runs, it deletes all inactive sessions and all sessions marked active (`WWCTX_SSO_SESSION$.ACTIVE = 1`), that are older than 7 days (`WWCTX_SSO_SESSION$.SESSION_START_TIME < sysdate - 7`).

These default settings can be modified by running some job management scripts in the Portal schema to manage Portal sessions, or in the OracleAS Single Sign-On schema to manage OracleAS Single Sign-On sessions. They utilize the same session management infrastructure.

Follow these steps to obtain the current cleanup job information:

1. Locate the following directory:

    ORACLE_HOME/portal/admin/plsql/wwc

2. On the database where the Portal or OracleAS Single Sign-On schema is installed, log on to SQL*Plus with the appropriate user name and password for that schema.

    For example:

```
sqlplus portal/portal
```

3. Enter the following command to get the current job information:

```
SQL> @ctxjget
```

The command results in the display of the currently installed job information, as returned by the DBMS_JOB package:

```
The session cleanup job is job ID 7381
dbms_job.isubmit(job=>7381,what=>'begin execute immediate''begin
wwctx_sso.cleanup_sessions(p_hours_old => 168); end;''; exception when
others then null; end;',next_date=>to_date('2001-04-17:14:07:20',
'YYYY-MM-DD:HH24:MI:SS'),interval=>'SYSDATE + 24/24',no_parse=>TRUE);

PL/SQL procedure successfully completed.
```

The results indicate which procedure is executed, what parameters are passed to it, and when the next invocation is to occur. This particular example indicates that the job is to cleanup active sessions that are a week old (168 hours). It also indicates that the next scheduled job execution is on 4/17/2001 at 5:14 pm, and the job should run every 24 hours thereafter.

If the job execution must be modified, either to adjust the age of sessions that should be deleted, or to increase or decrease the frequency of cleanup, you can run the ctxjsub.sql script to submit modified execution parameters.

Follow these steps to submit modified job execution parameters:

1. Locate the following directory:

```
ORACLE_HOME/portal/admin/plsql/wwc
```

2. On the database where the Portal or OracleAS Single Sign-On schema is installed, log on to SQL*Plus with the appropriate user name and password for that schema. For example:

```
sqlplus portal/portal
```

3. Enter the following command to submit new cleanup job information:

```
@ctxjsub <hours_old> <start_time> <time_format> <interval_hours>
```

Table C–2 lists the ctxjsub parameters.

Table C–2   ctxjsub Parameters

Parameter	Description
hours_old	The age of an active session that should be deleted.
start_time	The time that the next job should run.
time_format	The time format string that specifies how start_time is formatted.
interval_hours	The amount of time, in hours, between runs of the cleanup job.

For example:

```
SQL> @ctxjsub 200 '04/17/2001 10:00' 'MM/DD/YYYY HH24:MI' 12
```

The job information is displayed, similar to:

```
Created path for job id.
DBMS_JOB id = 7381
Cleanup job updated. Job ID = 7381

PL/SQL procedure successfully completed.
```

The cleanup job submission script can be run any number of times to modify the execution parameters. Each invocation updates the job information associated with the job ID for the cleanup job. This job ID is maintained in the preference store so that the job information is updated instead of submitting multiple jobs.

You can also specify a start_time of START, in which case, the time_format parameter is ignored, but you still need to pass it a value (such as NOW). The result is to run the job <interval_hours> hours from now:

```
SQL> @ctxjsub 168 START NOW 24
```

This submits the job as it does in the installation.

If you want the cleanup job to execute immediately, then obtain the job ID by calling ctxjget.sql. Once you know the job ID, you can execute the job by issuing the following command in the product schema:

```
SQL> exec dbms_job.run(7381);
```

In the preceding example, 7381 is the job ID returned by the call to ctxjget.sql. When you execute a job in this manner, the next automated invocation of the job occurs at interval_hours after this manual invocation. To run the job on the original schedule, resubmit the start_time desired using ctxjsub.sql.

C.7 Timing and Caching Statistics

All OracleAS Portal pages can be run in a special mode in which timing and caching information is displayed. If you want to see this debug information on every page you can set the Parallel Page Engine Parameter showPageDebug to true in the web.xml file.

> **See Also:**   Appendix D, "Configuring the Parallel Page Engine".

If you want to see the debug information for just a few select pages and portlets, you can control the logging level by the _debug URL parameter. For example, to see the timing statistics for the following OracleAS Portal page:

```
http://abc.com/servlet/page?_pageid=21
```

You can manually insert &_debug=3

To make:

```
http://abc.com/servlet/page?_pageid=21&_debug=3
```

Possible values for _debug are *0*, *1*, *2*, *3*, *4*, and *5*.

Values greater than *1* will potentially raise the **logmode** value for the duration of the request, and trigger all request log messages to be echoed into the page response.

> **Note:**   All values greater than *0* cause _debug=1 to be propagated in back end requests.

Table C–3 shows the results of _debug values:

Table C–3   _debug Values for Timing and Caching Statistics

Value	Timing and Caching Statistics?	Flag Forwarded to Providers? (as value 1)	logmode Raised to a Minimum of	Log Messages Written to Page Response?
0	Yes	-	-	-
1	Yes	Yes	-	-
2	Yes	Yes	debug	Yes
3	Yes	Yes	request	Yes

Table C–3   (Cont.) _debug Values for Timing and Caching Statistics

Value	Timing and Caching Statistics?	Flag Forwarded to Providers? (as value 1)	logmode Raised to a Minimum of	Log Messages Written to Page Response?
4	Yes	Yes	content	Yes
5	Yes	Yes	parsing	Yes

`urlDebugMode` and `urlDebugUsers` are additional parameters that can be used to restrict the use of _debug on a URL. See Appendix D, "Configuring the Parallel Page Engine" for more information.

The following statistics are available when the portal page is run in debug mode:

- Portlet Statistics

- Page Statistics

- Additional Summary Statistics

The following image shows a page that is running in the `_debug=0` mode:

Figure C–1   Portal Page Running in Debug Mode

C.7.1 Portlet Statistics

In Figure C–1, you can see a number of Portlet related statistics listed under each portlet. Each Portlet has a unique internal reference identification number. This number is used in the "Information for Portlet" summary. For the portlet in the top left corner of Figure C–1, you can see that this number is 6256.

For each portlet the following statistics are listed:

C.7.1.1 Portlet Timing Information

■   **Portlet Timing** (msecs) (wait msecs)

Indicates how many milliseconds it took to retrieve the portlet, and how long the request was queued, also in milliseconds.

■   **Timing Status**

This is deprecated and no longer in use.

- **XSLT Timing** (msecs)

  Displays the number of milliseconds that were needed to retrieve the XSL style sheet, in case the portlet is an XML portlet.

C.7.1.2 Portlet Caching Information

- **Portlet Cache status** Web Cache (values) File System Cache (values)

  This is the Cache status from both OracleAS Web Cache and the mod_plsql file cache.

  Valid values for OracleAS Web Cache are:

  - MISS, or NEW [M] indicating a cache miss in OracleAS Web Cache and that the content that is generated by the portlet is new.

  - MISS, or STALE [G] indicating a cache miss, due to stale content in OracleAS Web Cache.

  - HIT [H] indicating a OracleAS Web Cache hit.

  Valid values for File System Cache are:

  - HIT_PING indicating a cache hit for a validation-based portlet.

  - HIT_EXPIRES indicating a cache hit for an expiry-based portlet.

  - MISS_STALE indicating a cache miss due to stale content in the Cache. This applies to both expiry, as well as validation-based portlets.

  - MISS_NEW indicating a cache miss and that the content that is generated by the portlet is new. This applies to both expiry, as well as validation-based portlets.

  If a portlet uses the File System Cache, then the information mentioned in the preceding text will be listed. Otherwise it will be null.

  If there is a hit on OracleAS Web Cache, no details about File System Cache will be displayed because the content is served directly out of OracleAS Web Cache. Additionally, if a portlet does not use OracleAS Web Cache, then no Web Cache information will be printed.

- **From Cache:Web Cache** Cache Expires (seconds), Age in Cache (secs), File System Cache (values).

  Information from both OracleAS Web Cache and File System Cache will be printed here based on the type of caching that the portlet uses.

> **See Also:** *Oracle Application Server Web Cache Administrator's Guide*

"Cache Expires" lists the number of seconds after which the portlet content in OracleAS Web Cache will expire.

"Age in Cache" lists the number of seconds that the portlet content has been Cached in OracleAS Web Cache.

"File System Cache" displays the information obtained from the File System Cache about Cache Key, Cache Expiry and about the Cache Level in case of a cache hit, with the Cache Status of either HIT_PING, or HIT_EXPIRES.

In case of a cache hit, the Cache Key and Cache Level (for Validation-based portlets) and Cache Expires and Cache Level (for expiry-based portlets) are displayed, with the Cache Status value of either HIT_PING or HIT_EXPIRES.

For Validation-based and Expires-based portlets, "None" is printed when there is a cache miss due to the portlet content being new. (Cache Status: MISS_NEW) The portlet is contacted to get the new Cache Key, Cache Expiry and Cache Level.

For Validation-based portlets, if the content in the Cache has become stale resulting in a cache miss, the current values in the cache for Cache Key and Cache Level are displayed. In this case, the portlet is contacted to get the updated Cache Key and the level (Cache Status: MISS_STALE).

For Expires-based portlets, when the content in the cache has become stale resulting in a cache miss, a value of INVALID in the Expires field and Cache Level are displayed. In this case, the portlet is contacted to get the updated Cache Expiry and Cache Level (Cache Status: MISS_STALE).

- **From Portlet**: (Cache Key) (Cache Level)

  This is the information obtained from the portlet about File System Cache Key, Cache Expiry, and Cache Level when there is a cache miss and when portlet is contacted for the updated, or new values (Cache Status: MISS_NEW, or MISS_ STALE). Note that there is no OracleAS Web Cache related information displayed in this section.

  For Validation-based portlets, when there is a cache hit and if the ping is successful, meaning the content in the Cache is still valid, then the portlet does not return a new Cache Key and Cache Level; instead it will indicate that the cache is still valid. In this case, "Ping Success" is displayed (Cache Status: HIT_ PING).

For Expires-based portlets, when there is a cache hit and if the content has not expired, then the portlet is not contacted for the content. In this case, "Not contacted" is displayed (Cache Status: HIT_EXPIRES).

Following are a few examples that show different caching scenarios and the resulting output. Note that the other page and portlet related output is not shown here.

Example Caching Information Debug Output 1

- **Portlet Cache:** File System Cache, **Caching Type:** Validation-based, **Status:** MISS, STALE.

```
Caching information for portlet:
Portlet Cache status: File System Cache:- MISS,STALE
From Cache: File System Cache:- Cache Key: 42, Cache Level: USER
From Portlet: Cache Key: 44, Cache Level: USER
```

Example Caching Information Debug Output 2

- **Portlet Cache:** File System Cache, **Caching Type:** Expires-based, **Status:** MISS, NEW.

```
Caching information for portlet:
Portlet Cache status:File System Cache:-  MISS,NEW
From Cache: File System Cache:-None
From Portlet: Cache Expires: 1, Cache Level: USER
```

Example Caching Information Debug Output 3

- **Portlet Cache:** File System Cache, Web Cache, **Caching Type:** Validation and Invalidation-based, **Status:** MISS, NEW in File System Cache and Web Cache.

```
Caching information for portlet:
Portlet Cache status: Web Cache:- MISS,NEW [M], File System Cache:-
MISS,NEW
From Cache: Web Cache:- Cache Expires: 86400 secs, Age in Cache: 0 secs
, File System Cache:- None
From Portlet: Cache Key: 9.0.2.2.1502:04:18:09:19:56, Cache Level:
SYSTEM
```

Example Caching Information Debug Output 4

- **Portlet Cache:** Web Cache, **Caching Type:** Invalidation-based, **Status:** HIT in Web Cache.

```
Caching information for portlet:
```

```
Portlet Cache status: Web Cache:- HIT [H]
From Cache: Web Cache:- Cache Expires: 86400 secs, Age in Cache: 58 secs
From Portlet: -
```

C.7.2 Page Statistics

Every page has a unique internal reference identification number, similar to the portlets on the page, shown in Figure C–1.

For the page, the following statistics are listed:

- **Elapsed Time** (msecs)

  This is the total amount of time required to generate the page calculated in the Parallel Page Engine (PPE). The actual generation time in the browser can be higher, due to network overhead.

  Elapsed time is made up of page meta WAIT time and Stream time. Page meta WAIT time is the time taken to wait on content through an HTTP connection. Stream time is the time taken streaming and assembling the content pieces. Stream time is in turn composed of the following elements:

  - Page meta time

  - Time waiting for portlets to complete

  - Time taken streaming content to the browser

  Effectively, elapsed time is the total amount of time (in milliseconds) that it takes to put the page together, from the time the request was received to the last byte being written to the browser.

- **Page meta-time** (msecs) (wait = msecs)

  Displays the time that it takes to retrieve the page meta data. The wait time (msecs) represents how long the request was queued.

- **Page meta Cache Status** (Web Cache values), (Cache Expires msecs), (Age in Cache msecs), (File System Cache values)

  Represents the cache status from both OracleAS Web Cache and mod_plsql file cache. Valid values for OracleAS Web Cache are MISS, or NEW and HIT. Valid values for file Cache are HIT, or PING, and MISS, or STALE. The Web Cache Expires value and the Age in Cache are both measured in milliseconds.

- **Login meta-time** (msecs) (wait msecs)

Displays the time (in milliseconds) that it takes to retrieve the login meta data. The wait time represents the total amount of time (in milliseconds) that the request spends in the request queue.

- **Login meta Cache Status**

  Similar to **Page meta Cache Status** mentioned earlier, represents the cache status for the login meta data from both Web Cache and mod_plsql file cache.

C.7.3 Additional Summary Statistics

- **Stream info** (msecs)

  Represents (in milliseconds) how long it takes for the page to stream to the browser.

- **processing** (msecs)

  Processing time (in milliseconds) for streaming.

- **write** (msecs)

  The write lines can repeat several times. The lines represent each physical buffer write to the stream itself. This are one set for each buffer write.

- **flush** (msecs)

  The flush logs indicate that the writing stream was flushed. This is logged to keep track of the number of network round trips.

C.8 Using the cfgiasw Script to Configure Mobile Settings

If you want to change Portal's references to OracleAS Portal or Oracle Application Server Wireless' Portal service URLs, you need to use the script `cfgiasw.csh` (UNIX) or `cfgiasw.cmd` (Windows) to manually update the references. The script files are located here:

```
ORACLE_HOME/assistants/opca/
```

Running the script without parameters will print its usage to the screen, which is shown next:

Usage:

```
cfgiasw.csh -s portal_schema -sp portal_schema_password
    -w ias wireless url
    -h portal home page url
```

```
-c connect_string
```

Table C–4   Oracle Application Server Wireless Configuration Parameters

Parameter	Description
-s	Oracle Database schema for OracleAS Portal database objects.
	Default = `PORTAL`
-sp	Password for the OracleAS Portal schema.
	Default = `portal_schema`
-w	The URL of the Oracle Application Server Wireless gateway for mobile requests to OracleAS Portal. This parameter is not mandatory (no default).
-h	The URL of the OracleAS Portal home page. This is used within Portal to determine the character set of the Portal middle-tier. This information is required when creating an Oracle Application Server Wireless service This parameter is not mandatory (no default).
-c	Connect string for database (no default).

For non-hosted Portals, the Oracle Application Server Wireless' Portal service URL reference can be set in the **Mobile** tab of the **Global Settings** page, except the URL of the OracleAS Portal home page, which can only be set using the `cfgiasw` script.

This script is used to set references to both the Oracle Application Server Wireless Portal Service URL and the OracleAS Portal home page URL, in OracleAS Portal. It can be used in a hosted environment to set the URL references, and will affect all subscribers, because this information is not configured separately for each subscriber.

For example:

```
cfgiasw.csh -s portal -c portal_db -w
'http://iaswhost:port/ptg/rm?PAoid=%%wireless_service_id%%'
```

In the preceding example, if a mobile device makes a request to the OracleAS Portal directly without being mediated by an Oracle Application Server Wireless server, OracleAS Portal redirects the client to the URL specified here. This URL should be the OracleAS Portal's service URL on the Oracle Application Server Wireless server, in the form:

```
http://<host>:<port>/ptg/rm?PAoid=<service_id>
```

If this setting is blank, then mobile client requests made directly to OracleAS Portal receive an HTTP status indicating that their request is not supported.

For configuring other mobile settings in OracleAS Portal, see Section 14, "Click OK.".

C.9 Using the ptlinvsw.sql Script to Invalidate Portal Container Pages

If a user navigates to a sub-page within a page portlet and edits it, the changes are not visible immediately unless the page containing the portlet is invalidated by other means.

Session store lookup helps solve this issue. When a page is edited, the session store is first looked up to determine all the pages that have a portlet which is currently displaying the edited page, and then those pages are invalidated.

Since the session store lookup affects performance, this feature is not enabled by default.

To enable Portal container page invalidation:

1. On the database where the Portal schema is installed, navigate to the *ORACLE_HOME*/portal/admin/plsql/wws directory.

2. Log on to SQL*Plus with the appropriate user name and password for the Portal schema.

    For example:

    ```
    sqlplus portal/portal
    ```

3. Enter the following command:

    ```
    SQL> @ptlinvsw.sql TRUE
    ```

    To disable this invalidation option, enter the following command:

    ```
    SQL> @ptlinvsw.sql FALSE
    ```

D

Configuring the Parallel Page Engine

The Oracle Application Server Portal architecture is designed around a three-tier architecture that allows any browser to connect to it. This flexible architecture allows each component (browser, Oracle HTTP Server listener, Oracle9*i* Database Server, and OracleAS Portal) to be upgraded individually as required.

A part of the OracleAS Portal middle-tier, the Parallel Page Engine (PPE) is a shared server process servlet engine that runs in the Oracle Application Server Containers for J2EE and services page requests. The PPE reads page metadata, calls providers for portlet content, accepts provider responses, and assembles the requested page in the specified page layout.

D.1 Configuring Parallel Page Engine Parameters

When a page is requested from OracleAS Portal, the request is made from the browser to the Oracle HTTP Server listener. The returned page is comprised of many types of portlets. A portlet is an area on a portal page that contains data from a particular data source.

The Parallel Page Engine (PPE) obtains the page metadata from the Oracle Application Server Metadata Repository and is responsible for assembling the portlets on the page.

D.1.1 Setting PPE Configuration Parameters

With the release of Oracle9*i*AS version 9.0.2 and later, all of the servlets are installed under OC4J, based upon the application deployment. All of the configuration parameters for PPE are entered in the web.xml file, in a section related to the PPE Deployment. In the default installation, this file can be found at the following location:

```
MID_TIER_ORACLE_HOME/j2ee/OC4J_Portal/applications/portal/portal/WEB-INF/
```

D.1.2 Parallel Page Engine Configuration Settings

The following table describes each of the different configuration parameters available for use with the Parallel Page Engine (PPE). Each parameter affects the operation of the PPE in a different manner. Some are simply for logging, while others can affect the performance of the engine or OracleAS Portal itself. In most cases, the default values should be sufficient; however, there may be configurations where this is not the case. Each parameter is described with its syntax, description, and default.

Table D–1   Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value

poolSize	`<init-param>`
	`<param-name>poolSize</param-name>`
	`<param-value>25</param-value>`
	`</init-param>`
		This represents the number of connections that the Parallel Page Engine is capable of making at any one time. This value can be raised or lowered based upon performance needs. Setting the number higher makes more threads and connections available for use; however, this uses more resources.	25

requesttime	`<init-param>`
	`<param-name>requesttime</param-name>`
	`<param-value>30</param-value>`
	`</init-param>`
		This is the default time out assigned to portlet requests that do not have their own time out value specified. It is applied as the amount of time (in seconds) allowed before response headers are returned by the server. Time outs are weighted by where they originate. If the portlet sets its own time out value, then that is the time out that is used. If no portlet time out is available, then the provider registration time out is used. If neither of these is present, then the `requesttime` is used.
		
		Note that the upper limit of this parameter should be set to a response time acceptable by a Web user (typically a few seconds).
			30 sec

minTimeout	`<init-param>`
	`<param-name>minTimeout</param-name>`
	`<param-value>5</param-value>`
	`</init-param>`
		This is the minimum timeout allowed to be used by a Portlet. Thus, if the `minTimeout` is set to `5`, and a portlet sends a timeout of `2`, the `minTimeout` value of `5` would be applied to that portlet.	5 sec

Table D–1 (Cont.) Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value

stall	```<init-param>```
	 ```<param-name>stall</param-name>```
	 ```<param-value>120</param-value>```
	 ```</init-param>```
		If the response headers are returned, but the data itself lags behind, then a stall comes into affect. This value keeps the Parallel Page Engine from holding on to connections forever. Once the response headers are received, the PPE makes every effort to wait as long as is feasible to retrieve all of the data. Set this value appropriately if the portlets being requested are large, or running over a slow network.
		
		 Note that the upper limit of this parameter should be set to a response time acceptable by a Web user (typically a few seconds).
			120 sec

prefix	```<init-param>```
	 ```<param-name>prefix</param-name>```
	 ```<param-value>/pls</param-value>```
	 ```</init-param>```
		The string used to indicate to where mod_plsql is located. The default matches the default Oracle Application Server installation configuration, but it must be changed if the Oracle Application Server configuration has changed.	/pls

proxyHost
 proxyPort
	```<init-param>```
	 ```<param-name>proxyHost</param-name>```
	 ```<param-value>ph.comp.com</param-value>```
	 ```</init-param>```
	 ```<init-param>```
	 ```<param-name>proxyPort</param-name>```
	 ```<param-value>8888</param-value>```
	 ```</init-param>```
		This is the host name and port number of a proxy server that may be required to request data from the Oracle Application Server. These parameters are only required if a proxy server is in use between PPE and the Oracle Application Server listener.	N/a

Table D–1   (Cont.) Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value

offlinePathHtml
offlinePathMxml
	`<init-param>`
	`<param-name>offlinePath</param-name>`
	`>`
	`<param-value>/path/offline.html</param-value>`
	`ram-value>`
	`</init-param>`
	`<init-param>`
	`<param-name>offlinePathMxml</param-name>`
	`name>`
	`<param-value>/path/offline.xml</param-value>`
	`am-value>`
	`</init-param>`
		By setting either of these, the PPE is set to display the desired off-line message. There are two available messages: one for an HTML browser and one for a mobile enabled device.	null

showError	`<init-param>`
	`<param-name>showError</param-name>`
	`<param-value>true</param-value>`
	`</init-param>`
		When a portlet times out, or something within the Parallel Page Engine goes wrong with a particular portlet request, an error is displayed to the user. The messages tend to be generic, but do give the user some information and an indication that the page did not display as expected. If you set this to `false`, no messages are displayed to the user.	true

cacheBuffer	`<init-param>`
	`<param-name>cacheBuffer</param-name>`
	`>`
	`<param-value>32768</param-value>`
	`</init-param>`
		This parameter sets the number of bytes to use for buffering when reading a completed page from the cache. By determining the size of pages generally used in a portal, this value can be adjusted to fit the portal configuration. By setting the value higher, a larger page can be read quickly, but more resources are needed. If the value is set low, then reading the cache file is slower.	32768 Bytes

cacheEncryptionKey	`<init-param>`
	`<param-name>cacheEncryptionKey</param-name>`
	`am-name>`
	`<param-value>KEY</param-value>`
	`</init-param>`
		This key is used to obscure the headers used for caching using OracleAS Web Cache. This allows for a more secure cache key, and makes retrieving a cached object more difficult for unwanted requests.	Server Context information

Table D–1   (Cont.) Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value

enableWebCacheStaticRules	`<init-param>`
	`<param-name>enableWebCacheStaticRul`
	`es</param-name>`
	`<param-value>false</param-value>`
	`</init-param>`
		If set to `false`, PPE includes the `no-store` directive in the surrogate control response header of an assembled page. This overrides any static cacheability rule defined in OracleAS Web Cache, and ensures that the assembled page is not cached in the Web Cache.
		
		If set to `true`, PPE does not include the `no-store` directive in the surrogate control response header of an assembled page. This allows the use of static cacheability rules for caching the assembled page in OracleAS Web Cache.
			`false`

showPageDebug	`<init-param>`
	`<param-name>showPageDebug</param-na`
	`me>`
	`<param-value>false</param-value>`
	`</init-param>`
		If you set `showPageDebug` to `true`, the Page timing information is shown on every request.
		
		Refer to Section C.7, "Timing and Caching Statistics" for a description of the timing and caching statistics.
			false

dmsLogging	`<init-param>`
	`<param-name>dmsLogging</param-name>`
	`<param-value>false</param-value>`
	`</init-param>`
		If you set `dmsLogging` to `true`, the PPE outputs data for DMS Logging.	true

queueTimeout	`<init-param>`
	`<param-name>queueTimeout</param-nam`
	`e>`
	`<param-value>10</param-value>`
	`</init-param>`
		The amount of time a request should stay in the queue before being timed out. This parameter can be used if requests for portlets are timing out, but the requests are never being sent. Although this points to other performance problems that could be solved by alternative configurations, this option is available to allow requests to stay in the queue for longer or shorter periods of time.	10 sec

Table D–1   (Cont.) Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value
cacheDir	`<init-param>` `<param-name>cacheDir</param-name>` `<param-value>c:\iAS904\Apache\modpl sql\cache\</param-value>` `</init-param>`	The `cacheDir` parameter value points to a directory in the file system where cache files are stored and retrieved. The PPE caches portlet contents and fully assembled page contents into this directory. Note that if this is not specified, it will default to the default deployment location: *ORACLE_ HOME*`/Apache/modplsql/cac he` If a value is specified, it should be an absolute path rather than the relative path. Since this directory is shared by mod_plsql and the PPE, make sure that the directory has adequate privileges for the OHS and OC4J processes to write into it. If you want to change the location of the cache files, it is recommended that you use symbolic links so that the location of the cache directory is still *ORACLE_ HOME*`/Apache/modplsql/cac he`. If your operating system does not support symbolic links, make sure that corresponding changes are done to the mod_ plsql cache configuration file (*ORACLE_ HOME*`/Apache/modplsql/con f/cache.conf`).	`Default deployment location`
jspRoot	`<init-param>` `<param-name>jspRoot</param-name>` `<param-value>/JSP PATH/</param-value>` `</init-param>`	The relative path where JSP files for JSP Pages can be found.	`jsp`

Table D–1   (Cont.) Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value

jspSrcAlias	`<init-param>`
	`<param-name>jspSrcAlias</param-name`
	`>`
	`<param-value>/PATH</param-value>`
	`</init-param>`
		The Alias for the jsp engine, like `/portal/jsp` or some other path.	`/jsp/`

urlDebugMode	`<init-param>`
	`<param-name>urlDebugMode</param-nam`
	`e>`
	`<param-value>1</param-value>`
	`</init-param>`
		Specifies the highest value of the *_debug* URL parameter that the PPE should honor. Possible values for *_debug* are:
		
		`none`, `0`, `1`, `2`, `3`, `4`, and `5`
		
		If a value higher than that allowed is received by the PPE, it is reduced to the highest value permitted, or ignored if no value is allowed.
		
		The values build incrementally. For example, at debug value `2`, values for debug level `1` and `0` are also recorded.
			1

urlDebugUsers	`<init-param>`
	`<param-name>urlDebugUsers</param-na`
	`me>`
	`<param-value>fred,bill,ben</param-v`
	`alue>`
	`</init-param>`
		This is specified to indicate the list of users allowed to use the *_debug* URL parameter, subject to the value restriction in the `urlDebugMode` parameter. If this is not specified, all users can use it subject to the value restriction.
		
		The format is a comma-delimited list of portal user names, with leading and trailing spaces being ignored.
			`none required`

useWebCache	`<init-param>`
	`<param-name>useWebCache</param-name`
	`>`
	`<param-value>true</param-value>`
	`</init-param>`
		Indicates whether Web Cache is being used with the PPE. Allowed values are `true` and `false`.	`true`

dadFilePath	`<init-param>`
	`<param-name>dadFilePath</param-name`
	`>`
	`<param-value>c:\iAS904\Apache\modpl`
	`sql\conf\dads.conf</param-value>`
	`</init-param>`
		The absolute filename and path of the `dads.conf` file used by mod_plsql and read by the PPE. If not specified, it defaults to:
		
		ORACLE_HOME/Apache/modplsql/conf/dads.conf
			ORACLE_HOME/Apache/modplsql/conf/dads.conf

Table D–1   (Cont.) Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value

logmode	```<init-param>```
	```<param-name>logmode</param-name>```
	```<param-value>debug</param-value>```
	```</init-param>```
		Enables the Parallel Page Engine to run in debug mode. This mode writes debug information to the Parallel Page Engine log file. This mode does cause some degradation in performance because large amounts of information are being written to disk. The Parallel Page Engine log file (`application.log`) by default is located at:
		
		ORACLE_HOME`/j2ee/OC4J_ Portal/application-deplo yments/portal/`
		
		Allowed values are:
		
		`none` - No debug messages
		
		`perf` - Performance messages only
		
		`debug` - General debug messages
		
		`request` - Details of requests made by the PPE
		
		`content` - Details of the content of requests made by the PPE
		
		`parsing` - Details of metadata parsing
		
		`all` - All debug messages
		
		The values build incrementally. For example, at logging level `request`, the output for logging levels `debug` and `perf` will also be recorded.
			none - no debug messages

maxParallelPortlets	```<init-param>```
	```<param-name>maxParallelPortlets</pa ram-name>```
	```<param-value>20</param-value>```
	```</init-param>```
		Used to specify the maximum number of portlet requests for a given page, that should be allowed, to execute at the same time. Allowed values are:
		
		`0` - Indicates no restriction (beyond the number of fetchers available).
		
		Any positive integer - Indicates a restriction on simultaneous requests.
			20

Table D–1   (Cont.) Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value

httpsports	`<init-param>`
	`<param-name>httpsports</param-name>`
	`<param-value>433:444</param-value>`
	`</init-param>`
		This is a colon (':') separated list of ports on which the PPE should use SSL to communicate with the Portlet Repository.
		
		The **Event** servlet, which is also configured through `web.xml`, has the **httpsports** setting that should be set to the same value as the `httpsports` parameter in PPE. This is the only configuration for the **Event** servlet, and is only applicable if SSL is being used.
		
		If you change the parameter in the PPE settings, you must make the same change in the **httpsports** setting of the **Event** servlet.
			null

Table D–1   (Cont.) Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value

useScheme	`<init-param>`
	`<param-name>useScheme</param-name>`
	`<param-value>http</param-value>`
	`</init-param>`
		Overrides the scheme (http or https) used when the PPE makes requests to the portal. The default, if not specified, is to always use the page request scheme. Note that you must set the `useScheme` and `usePort` parameters.
		
		You need to specify these in scenarios where public access is through **https** on **port A**, and you want to set PPE requests to use a faster **http** connection on **port B**.
			`Use page request scheme`

usePort	`<init-param>`
	`<param-name>usePort</param-name>`
	`<param-value>8888</param-value>`
	`</init-param>`
		Overrides the port used when the PPE makes requests to the portal. The default, if not specified, is to always use the page request port. Note that you must set the `useScheme` and `usePort` parameters.
		
		You need to specify these in scenarios where public access is through **https** on **port A**, and you want to set PPE requests to use a faster **http** connection on **port B**.
			`Use page request port`

Table D–1 (Cont.) Parallel Page Engine (PPE) Parameters

PPE Setting	Syntax	Description	Default Value

x509certfile	`<init-param>`
	`<param-name>x509certfile</param-name>`
	`<param-value>c:\certificates\trustedcerts.txt</param-value>`
	`</init-param>`
		Specifies a file containing a list of certificates to be implicitly trusted by HTTPClient. These certificates are added as trust points to all connections made by HTTPClient using SSL. Once this setting is in use, all SSL connections must be trusted. Otherwise, HTTPClient will throw an exception in the PPE.
		
		Note that SSL connections are made from the PPE for two reasons, and this configuration affects both:
		
		loopback requests to the portal, for example, for PMD.
		
		show calls to Providers.
		
		Note that the file specified here can be obtained from a wallet by exporting all trusted certificates, but the comments in the resultant file must be removed. Alternatively, it can be created manually.
			`trust points not used`

E

Using Oracle Application Server Configuration Files

This appendix provides information about the configuration files and tables that can affect the connection to and the behavior of the Oracle Application Server and its components in the middle-tier as well as on other machines to which it is connecting.

Specific topics covered include:

- Oracle HTTP Server Configuration File (httpd.conf)

- Oracle Database Connection File (tnsnames.ora)

- Web Cache Configuration Files

- OracleAS Single Sign-On Configuration Table

- OracleAS Single Sign-On's Partner Application Table

- Local HOSTS File

- Using Oracle Enterprise Manager

E.1 Oracle HTTP Server Configuration File (httpd.conf)

The Oracle HTTP Server configuration file, `httpd.conf`, contains configuration information for running the Oracle HTTP Server. The content of this file includes information about listening ports, server names, virtual hosts, proxy configurations, and the like. This file also configures Secure Sockets Layer (SSL) support by defining information such as certificates and other HTTPS configuration directives.

ORACLE_HOME/Apache/Apache/conf/httpd.conf

E.2 Oracle Database Connection File (tnsnames.ora)

The `tnsnames.ora` file defines the entries that can be used as connect strings in the DADs.

Also, the `tnsnames.ora` file in the Oracle home location containing your Oracle Application Server must have a connect string entry pointing to the database where your Oracle Portal installation is located.

In the C shell, for example, type the following at a command line prompt:

```
setenv TNS_ADMIN path
```

`path` points to the `tnsnames.ora` file. This command differs depending on the shell used.

E.3 Web Cache Configuration Files

The following OracleAS Web Cache configuration files can be found in the `ORACLE_HOME`/webcache directory:

- `webcache.xml`

- `internal.xml`

- `internal_admin.xml`

> **See Also:** *Oracle Application Server Web Cache Administrator's Guide*

E.4 OracleAS Single Sign-On Configuration Table

The `WWSEC_ENABLER_CONFIG_INFO$` table is the configuration table for the Single Sign-On enabler stack. Typically, modifications to this table are handled by running the Portal Dependency Settings tool, or OracleAS Portal Configuration Assistant (OPCA), in the case of advanced configurations. This section is provided for additional information about the SSO configuration table. Modifications are not to be made directly, but instead by using the Portal Dependency Settings tool, (`ptlconfig`) described in Appendix A, "Using the Portal Dependency Settings File", or the OracleAS Portal Configuration Assistant, described in Appendix B, "Using the OracleAS Portal Configuration Assistant Command Line Utility".

Each partner application to the OracleAS Single Sign-On has such a table for configuration information. One such table exists in the OracleAS Portal schema as well as the OracleAS Single Sign-On schema, since the OracleAS Single Sign-On

application is also a partner application. This table defines the login URL for the OracleAS Single Sign-On that this partner Application is configured to use.

It is important to understand how the `LSNR_TOKEN` is used in the enabler configuration table, to help you plan what entries are required depending on your configuration.

This table may have more than one entry. There is one entry for each way the application's server is addressed. Understanding this requires a review of the authentication sequence. For the purpose of this discussion, the main flows include:

- Initial request to the requested URL

- Redirect to the OracleAS Single Sign-On for authentication

- Redirect to OracleAS Portal's success URL (`wwsec_app_priv.process_signon`)

- Redirect back to the requested URL

The OracleAS Single Sign-On (SSO) partner enabler APIs read the `WWSEC_ENABLER_CONFIG_INFO$` table for configuration information. Similarly, in the OracleAS Single Sign-On, the OracleAS Single Sign-On's private APIs read the `WWSSO_PAPP_CONFIGURATION_INFO$` table. In the latter table, the URL should be redirected to each partner application.

Since each partner application's success URL is stored in the OracleAS Single Sign-On's partner application configuration table, to support multiple host names for the partner application, each distinct host name requires its own partner application entry on the OracleAS Single Sign-On. This is so that each one can specify a success URL that has the same hostname as the partner application, so that the session cookie can be scoped appropriately. Furthermore, the domain to which cookies are scoped includes the server name (ServerName) and port, so server.domain.com:80 is treated as a different cookie domain from server.domain.com:8080.

Each entry in the enabler configuration table is then selected based on the host name and port that was used by the partner application.

For example, let's say that you wanted OracleAS Portal to be accessible from `http://www.xyz.com` as well as `http://www.abc.com`. In this case, two partner applications must be registered in the OracleAS Single Sign-On. One is defined for the www.xyz.com host and the other for the www.abc.com host. Each one specifies a success URL that is appropriate:

- `http://www.xyz.com/pls/portal/portal.wwsec_app_priv.process_signon` for the www.xyz.com partner

- `http://www.abc.com/pls/portal/portal.wwsec_app_`
  `priv.process_signon` for the www.abc.com application

Each of these partner application entries on the OracleAS Single Sign-On would have a distinct site ID, site token, and encryption key. OracleAS Portal's enabler configuration table has one row for each partner application, for example:

```
LSNR_TOKEN    SITE_ID  LS_LOGIN_URL …

www.xyz.com   1321      https://www.login.com/pls/…

www.abc.com   1322      https://www.login.com/pls/…
```

> **See Also:** *Oracle Application Server Single Sign-On Application Developer's Guide*

E.5 OracleAS Single Sign-On's Partner Application Table

The configuration table on the OracleAS Single Sign-On's side is the partner application Table, WWSSO_PAPP_CONFIGURATION_INFO$. Maintenance of this table is typically done using the OracleAS Single Sign-On application's user interface for adding or editing partner applications.

For an initial installation on a single database instance, running the OPCA with -mode MIDTIER -type SSO, or -type OHS, populates both the OracleAS Single Sign-On's partner configuration table as well as OracleAS Portal's enabler configuration table.

E.6 Local HOSTS File

The HOSTS file on a network host defines mappings of IP names to IP addresses. Normally, a Domain Name Server (DNS) provides the mapping of IP name to IP address. In some of the configurations described in Chapter 4, "Performing Basic Configuration and Administration", a host may need to be addressed in an internal network with a domain name that is not defined within the internal network. In these cases, the server's HOSTS file can provide the necessary name resolution.

E.7 Using Oracle Enterprise Manager

You can use Oracle Enterprise Manager Application Server Control for administering OracleAS Portal. Application Server Control is a Web-based tool that enables you to perform some of the management tasks described in this book. Refer

to Chapter 7, "Monitoring and Administering OracleAS Portal" for more information about using Oracle Enterprise Manager.

> **See Also:** *Oracle Application Server 10g Administrator's Guide*

F

Integrating JavaServer Pages with OracleAS Portal

OracleAS Portal gives you the ability to create various kinds of Web pages. You can supplement this ability with JavaServer Pages (JSPs).

This appendix describes how you can secure OracleAS Portal to allow access to only approved JSPs, and prevent unauthorized access by JSPs to portlet content. It also describes the steps required to allow access for protected external JSPs that require login.

The following topics are covered in this appendix:

- Using the JavaServer Page Configuration File
- Setting Up a JAZN File for External Communication

F.1 Using the JavaServer Page Configuration File

Because almost any JSP using the tag library can request OracleAS Portal portlet content, there is a need for a secure way to ensure that only approved JSPs obtain access. You can control this through two mechanisms:

- The <portal:usePortal> tag in the JSP
- An external JSP configuration file

The configuration file identifies the OracleAS Portal instances, and page groups within those instances, to which an external JSP is allowed access.

The specific coding requirements of the configuration file are explained in Section F.1.1, "Contents of Your JavaServer Page Configuration File".

Your completed configuration file must then be identified to OracleAS Portal. This step is explained in Section F.1.3, "Location of Your JavaServer Page Configuration File".

This section contains the following sub-sections:

- Contents of Your JavaServer Page Configuration File

- Example JavaServer Page Configuration File

- Location of Your JavaServer Page Configuration File

- External JavaServer Page Login

F.1.1 Contents of Your JavaServer Page Configuration File

The required tags are:

- <jps>

- <portal>

- <database>

- <url>

- <cookie>

- <pageGroups>

- <pageGroup>

F.1.1.1 The <jps> Tag

The <jps> tag is a container tag that provides a list of OracleAS Portal instances to which external JSPs can have access.

Opening tag
<jps version="1.0">

Version must be set to 1.0 for the current OracleAS Portal release.

Closing tag
</jps>

F.1.1.2 The <portal> Tag

The <portal> tag describes an individual OracleAS Portal instance.

Opening tag
<portal name="MyPortal" default="true">

Closing tag
</portal>

Table F–1   The <portal> Tag's Attributes

Attribute	Value
name	Any descriptive name given to an OracleAS Portal instance. The name must be unique within the configuration file.
default	A `true` or `false` flag indicating whether this portal is the default instance that is used if a *usePortal* tag does not specify a portal name. If you provide no value, default is set to `false`.

Only **one** default portal is allowed for each configuration file.

F.1.1.3 The <database> Tag

The <database> tag provides database connection information about a given OracleAS Portal instance. For example:

```
<database data-source="jdbc/MyPortal"/>
```

The data-source attribute value is the name of the data source, which must be specified in the `data-sources.xml` file located in the *J2EE_HOME*/config directory.

Here is an example of a data-source definition:

```
<data-source
    class="com.evermind.sql.DriverManagerDataSource"
    name="MyPortal"
    location="jdbc/MyPortal"
    xa-location="jdbc/xa/MyPortal"
    ejb-location="jdbc/MyPortal"
    connection-driver="oracle.jdbc.driver.OracleDriver"
    username="portal_app"
    password="portal_app"
    url="jdbc:oracle:thin:@xyz.oracle.com:1521:orcl"
    inactivity-timeout="30"
/>
```

The username and password attributes must be set to the OracleAS Portal application schema user name and password.

F.1.1.4 The <url> Tag

The <url> tag provides connection information to the OracleAS Portal instance. For example:

```
<url protocol="http" host="defg.oracle.com" port="7500" path="/pls/portal"/>
```

Table F–2   The <url> Tag's Attributes

Attribute	Value
protocol	The name of the protocol used to connect to the OracleAS Portal instance. Currently, only http and https protocols are supported. If you do not specify a protocol attribute, the default will be http.
host	The machine name for the OracleAS Portal middle-tier.
port	Port number. If no port is specified, the default number will be 80.
path	For this release, path must be set to /pls/<PORTAL-DAD-NAME>.

F.1.1.5 The <cookie> Tag

The <cookie> tag describes the OracleAS Portal cookie. For example:

```
<cookie name="portal" maxAge="-1" path="/" domain=".oracle.com"/>
```

Table F–3   The <cookie> Tag's Attributes

Attribute	Value
name	The name of the cookie. This must be the same as the OracleAS Portal instance cookie name. *name* is a required attribute of the cookie tag.
maxAge	The maximum age of the cookie, specified in seconds. Specify a value of -1 if you want the cookie to persist until browser shutdown. *maxAge* is a required attribute of the cookie tag.
path	The path on the server to which the browser returns this cookie. *path* is a required attribute of the cookie tag.

Table F–3   (Cont.) The <cookie> Tag's Attributes

Attribute	Value
`domain`	This attribute should be specified only if changes were made to the SSO portlet cookie configuration. See the SSO documentation.

F.1.1.6 The <pageGroups> Tag

The <pageGroups> tag forms a container for the pageGroup tags. This tag has no attributes.

Opening tag
<pageGroups>

Closing tag
</pageGroups>

F.1.1.7 The <pageGroup> Tag

The <pageGroup> tag describes each individual page group's properties. For example:

```
<pageGroup name="JPSDemo" key="welcome" default="true"/>
```

Table F–4   The <pageGroup> Tag's Attributes

Attribute	Value
`name`	The page group name. This must be the name given to the page group when it was created in OracleAS Portal.
`key`	The page group's key. The value must match the Access Key value that was assigned to the page group in OracleAS Portal. (Note that a page group identified here must have JSP Access enabled.)
`default`	A flag set to `true` or `false` indicating whether or not this page group is the default page group within this OracleAS Portal instance. A default page group is the one used in the *usePortal* tag if no page group name is supplied. If no value provided for default in this pageGroup tag, it will be set to `false`.

Only **one** default page group is allowed for each portal instance.

F.1.2 Example JavaServer Page Configuration File

The following is an example of a JSP configuration file:

Example F–1   Example JavaServer Page Configuration File

```
<jps version="1.0">
   <portal name="MyPortal" default="true">
      <database data-source="jdbc/MyPortal"/>
      <url host="xyz.oracle.com" port="7500" path="/pls/portal"/>
      <cookie name="portal" maxAge="-1" path="/" />
      <pageGroups>
         <pageGroup name="JPSDemo"  key="welcome" default="true"/>
         <pageGroup name="JPSDemo2" key="welcome" default="false"/>
      </pageGroups>
   </portal>
   <portal name="AnotherPortal">
      <database data-source="jdbc/AnotherPortal"/>
      <url protocol="http" host="abc.oracle.com" port="8888"
         path="/pls/portal90"/>
      <cookie name="portal90" maxAge="-1" path="/" />
      <pageGroups>
         <pageGroup name="JPSDemo"  key="welcome"/>
         <pageGroup name="JPSDemo1"  key="welcome1"/>
         <pageGroup name="JPSDemo2"  key="welcome2"/>
         <pageGroup name="JPSDemo3"  key="welcome3"/>
         <pageGroup name="JPSDemo4"  key="welcome4"/>
      </pageGroups>
   </portal>
</jps>
```

F.1.3 Location of Your JavaServer Page Configuration File

By default, the name of the configuration file is assumed to be wwjps.xml, and the default location of the file is:

J2EE_HOME/applications/portal/portal/WEB-INF

However, your configuration file can have any other name, and can be located anywhere in the file system.

You specify the location using a context parameter in the web.xml file, which is located in the directory *J2EE_HOME*/applications/portal/portal/WEB-INF.

The context parameter in the web.xml file is:

```
<context-param>
   <param-name>oracle.webdb.service.ConfigLoader</param-name>
   <param-value>/WEB-INF/wwjps.xml</param-value>
      <description>This parameter specifies the location of the JPS
         configuration file</description>
</context-param>
```

F.1.4 External JavaServer Page Login

External JSPs can be categorized by their login requirements:

- Public JSPs, which do not require login (or to which users login through the OracleAS Portal login link)

- Protected JSPs, which do require login

Protected external JSPs have additional setup requirements. These are explained in the next section.

F.2 Setting Up a JAZN File for External Communication

The following steps are required only for protected external JSPs. That is, external JSPs that require login.

In the external JSPs, if you need to log in to the portal, you need to use the following tag syntax:

```
<portal:usePortal id="AnyPortal" pagegroup="AnyPageGroup" login="true" />
```

When you execute this JSP, you will be redirected to the OracleAS Single Sign-On server if you are not already logged on. In order to make this work, look at the following sections:

- Setting Up mod_osso (if not already set up)

- Setting Up JAZN with LDAP

F.2.1 Setting Up mod_osso

By default, your Oracle HTTP Server is registered with the OracleAS Single Sign-On server. The following steps are given here only if that has been changed, and re-registration is necessary.

F.2.1.1 Register Oracle HTTP Server with OracleAS Single Sign-On Server

You need to register your Oracle HTTP Server with the OracleAS Single Sign-On server as a partner application. To do so:

1. Go to your OracleAS Single Sign-On server home and login, for example:

   ```
   http://abc.company.com:3000/pls/portal_sso/
   ```

2. Go to **SSO Server administration**: **Administer Partner applications**: **Add Partner Application**.

3. Enter the following at the prompts.

   - Name: Oracle HTTP Server (any name)

   - Home URL: `http://abc.company.com:3000/`

   - Success URL: `http://abc.company.com:3000/osso_login_success`

   - Logout URL: `http://abc.company.com:3000/osso_logout_success`

4. Click **Apply**. Keep a record of the values that are displayed. You will need these values in the next step.

F.2.1.2 Create a Directory File

Create a file (for example, `portal.clr`) based on Example F–2. Replace the values in this example file with the values you got in the previous step (when you registered your application).

Example F–2   Example Directory File

```
sso_server_version=v1.2
cipher_key=95CFC0004E594CB3
site_id=1325
site_token=NCZ4UZMV1325
login_url=http://abc.company.com:3000/pls/portal_sso/portal_sso.wwsso_a
pp_admin.ls_login
logout_url=http://abc.company..com:3000/pls/portal_sso/portal_sso.wwsso_app_
admin.ls_logout
cancel_url=http://abc.company..com:3000/pls/portal_sso/
```

After making your changes in the file, place the file in the following directory:

```
<Oracle HTTPhome>/Oracle HTTP/Oracle HTTP/conf/osso/
```

F.2.1.3 Run Oracle HTTP osso.conf

Go to `<Oracle HTTPhome>/Oracle HTTP/Oracle HTTP/conf/osso/` and run:

```
<Oracle HTTPhome>/Oracle HTTP/Oracle HTTP/bin/apobfuscate portal.clr osso.conf
```

F.2.1.4 Remove Comments from the httpd.conf File

In your `httpd.conf` file, make sure the following line is not commented out:

```
include "<Oracle HTTPhome>/Oracle HTTP/Oracle HTTP/conf/mod_osso.conf"
```

> **See Also:** *Oracle Application Server Single Sign-On Administrator's Guide*

F.2.1.5 Restart the Oracle HTTP Server

After completing the earlier steps, restart the Oracle HTTP Server.

F.2.2 Setting Up JAZN with LDAP

JAZN is the internal name for a Java Authentication and Authorization Service (JAAS) provider. JAAS is a Java package that enables applications to authenticate and enforce access controls upon users. The use of JAZN in OracleAS Portal is limited to the authentication of external JSPs

Confirm that the JAZN is working with the LDAP. (You can use the demo provided by the JAZN.)

Do the following additional step:

- Go to *J2EE_HOME*`/application-deployments/portal/orion-application.xml` and add the following:

```
<jazn provider="LDAP" location="ldap://<OIDHOST>:389"
default-realm="oracle">
<jazn-web-app auth-method="SSO" />
</jazn>
```

Port number 389 is a default port for LDAP servers. However, any other port can be assigned. Contact your Oracle Internet Directory Administrator to obtain <host> and <port> information.

> **See Also:** For more information:
>
> - Section F.2, "Setting Up a JAZN File for External Communication"
>
> - *Oracle Application Server Containers for J2EE Services Guide*

G

Using the wwv_context APIs

The `wwv_context` package contains procedures to create and maintain Oracle Text indexes used by OracleAS Portal. This appendix describes the content of this package in the following sections:

- Procedures

- Constants

- Exceptions

> **Note:** For more information about Oracle Text indexes and their use in OracleAS Portal, see Chapter 8, "Configuring the Search Features in OracleAS Portal".

G.1 Procedures

The `wwv_context` package contains these procedures:

add_attribute_section

create_index

create_missing_indexes

create_prefs

createindex

drop_all_indexes

drop_index

drop_invalid_indexes

drop_prefs

dropindex

optimize

sync

touch_index(p_indexes wwsbr_array)

touch_index

G.1.1 add_attribute_section

```
procedure add_attribute_section(
    p_attributeid    in number,
    p_attributesiteid in number
)
```

Adds a new section to the section groups used by the Item and Page indexes. The section group corresponds to an attribute. This changes the index metadata only, it does not update the index data itself. The new sections can be searched but the indexes themselves are not changed.

The indexes are only changed if they exist, that is, if they do not exist this procedure does not do anything.

Parameters:

p_attributeId - ID of the attribute section to add.

p_attributeSiteId - Site ID of the attribute section to add.

G.1.2 create_index

```
procedure create_index(
    p_index in varchar2
)
```

Creates a specific, named Oracle Text index. For more information, see Section 8.3.3, "Oracle Text Indexes".

Use this procedure for troubleshooting purposes only. Under normal circumstances, use create_missing_indexes to create *all* of the indexes that are absent, or createindex to drop invalid indexes and then re-create the preferences and missing indexes.

For this procedure, use one of the following Constants to name the index:

- **Page index -** wwv_context.PAGE_TEXT_INDEX

- **Document index -** wwv_context.DOC_TEXT_INDEX

- **Perspective index -** wwv_context.PERSPECTIVE_TEXT_INDEX

- **Item index -** wwv_context.ITEM_TEXT_INDEX

- **Category index -** wwv_context.CATEGPRY_TEXT_INDEX

- **URL content index -** wwv_context.URL_TEXT_INDEX

Parameters:
p_index - The name of the index to create, that is, one of the constants listed earlier.

Exceptions:
INVALID_INDEX - The name of the index was not recognized.

G.1.3 create_missing_indexes

```
procedure create_missing_indexes(
    p_indexes out wwsbr_array
)
```

Creates all of the Oracle Text indexes that are absent. An index is considered to be present if it exists according to the view ctx_user_indexes.

This procedure does not check to see if the existing indexes are valid. Use the procedure drop_invalid_indexes to drop any indexes that are not entirely valid.

This procedure creates empty indexes. To populate the indexes, you must first mark them as 'requiring re-indexing' and then you must synchronize the indexes. To do this, use the procedure touch_index(p_indexes wwsbr_array).

This procedure does not create Oracle Text preferences as these must already exist. Use the procedure create_prefs to create the preferences, if they do not exist.

Parameters:
p_indexes - returns an array containing the list of indexes created.

G.1.4 create_prefs

```
procedure create_prefs
```

Creates the Oracle Text preferences, which are used when creating Oracle Text indexes, that is, both Datastore and Filter preferences. For more information, see Section 8.3.3.2, "Oracle Text Index Preferences".

This procedure does not create any of the Lexer preferences. Use the script sbrimtlx.sql located in the directory *ORACLE_HOME*/portal/admin/plsql/wws to create Lexer preferences. For more information, see Section 8.3.3.5, "Multilingual Functionality (Multilexer)".

G.1.5 createindex

```
procedure createindex(
    p_language in  varchar2 default wwnls_api.nls_default_language,
    p_message  out varchar2
)
```

Creates Oracle Text indexes used by OracleAS Portal. For more information, see Section 8.3.3, "Oracle Text Indexes".

This high level procedure performs the following tasks:

- Drops all existing preference objects.

- Drops any invalid indexes.

- Re-creates Oracle Text preferences.

- Creates indexes that are missing (initially empty).

- Marks all indexable OracleAS Portal content as requiring re-indexing, for all new indexes.

- Synchronizes indexes, that is, first populates and then optimizes the indexes.

This procedure is logically equivalent to:

```
wwv_context.drop_prefs;
wwv_context.drop_invalid_indexes;
wwv_context.create_prefs;
wwv_context.create_missing_indexes(l_indexes);
wwv_context.touch_index(l_indexes);
wwv_context.sync;
wwv_context.optimize;
```

G.1.6 drop_all_indexes

```
procedure drop_all_indexes
```

Drops *all* the Oracle Text indexes used by OracleAS Portal.

This procedure does not drop the Oracle Text preferences. Use the procedure drop_prefs to do this.

G.1.7 drop_index

```
procedure drop_index(
    p_index in varchar2
)
```

Drops a specific, named Oracle Text index. This procedure does not validate that the index exists.

Use one of the following Constants to name the index:

- **Page index -** wwv_context.PAGE_TEXT_INDEX

- **Document index -** wwv_context.DOC_TEXT_INDEX

- **Perspective index -** wwv_context.PERSPECTIVE_TEXT_INDEX

- **Item index -** wwv_context.ITEM_TEXT_INDEX

- **Category index -** wwv_context.CATEGPRY_TEXT_INDEX

- **URL content index -** wwv_context.URL_TEXT_INDEX

Parameters:

`p_index` - The name of the index to drop, that is, one of the constants listed earlier.

Exceptions:

`INVALID_INDEX` - The name of the index was not recognized.

G.1.8 drop_invalid_indexes

```
procedure drop_invalid_indexes
```

Drops invalid Oracle Text indexes only, that is, valid Oracle Text indexes are not dropped.

An index is considered to be valid, if the following status columns, in the following views, are all set to 'VALID':

- user_indexes.status

- user_indexes.domidx_status

- user_indexes.domidx_optstatus

- ctx_user_indexes.idx_status

If any status column is not valid or, if the index does not have an entry in both views, it is considered to be invalid and will be dropped. For more information, see Section 8.3.7, "Viewing the Status of Oracle Text Indexes".

G.1.9 drop_prefs

```
procedure drop_prefs
```

Drops the Oracle Text datastore and filter preferences. For more information, see Section 8.3.3.2, "Oracle Text Index Preferences".

Datastore and Filter preferences are used when creating the Oracle Text indexes. This procedure does not drop any of the Lexer preferences that are created using the script sbrimtlx.sql. The script is located in the directory *ORACLE_ HOME*/portal/admin/plsql/wws.

G.1.10 dropindex

```
procedure dropindex(
    p_language in varchar2 default wwnls_api.nls_default_language,
    p_message  out varchar2
)
```

Drops all existing Oracle Text indexes used by OracleAS Portal. For more information, see Section 8.3.3, "Oracle Text Indexes".

This procedure is equivalent to:

```
wwv_context.drop_prefs;
wwv_context.drop_all_indexes;
```

G.1.11 optimize

```
procedure optimize(
    p_optlevel in varchar2 default ctx_ddl.optlevel_full,
```

```
    p_maxtime  in number   default null,
    p_token    in varchar2 default null
)
```

Optimizes all *existing* Oracle Text indexes used by OracleAS Portal. Each index is optimized by calling the Oracle Text procedure `ctx_ddl.optimize_index()`.

Indexes are only optimized if they are sufficiently fragmented to make optimization worthwhile. This is measured by the average number of times that a token occurs more than once. If this value is greater than 10, the index is optimized. For more information, see Section 8.3.5.5, "Optimizing Oracle Text Indexes".

The parameters for this procedure are the same as those required by the Oracle Text procedure `ctx_ddl.optimize_index`.

Parameters:

`p_optlevel` - Optimization level, one of 'FULL','FAST' or 'TOKEN'.

`p_maxtime` - Maximum time for full optimization, in minutes.

`p_token` - Token to optimize (when doing TOKEN optimization).

You'll find additional information in the Oracle Text Reference documentation on the Oracle Technology Network, http://otn.oracle.com/products/text/content.html.

G.1.12 sync

```
procedure sync
```

Synchronizes *all* Oracle Text indexes used by OracleAS Portal. Each index is synchronized by calling the Oracle Text procedure `ctx_ddl.sync_index()`. This procedure re-indexes any rows that have been updated since the last synchronization. After synchronization, newly added or updated content can be searched. For more information, see Section 8.3.5.1, "Synchronizing Oracle Text Indexes".

Before synchronization, the pending queue is updated for the table `wwsbr_url$` which contains all URLs attribute values stored in OracleAS Portal. Rows from this queue are removed when the URL value is equal to the value of the constant `wwv_context_util.g_noindex`. These rows are set to this value to indicate that the original URL was not indexable by Oracle Text, for example, URLs such as those beginning with `https://` or `javascript:`.

You'll find additional information on `ctx_ddl.sync_index` in Oracle Text Reference documentation on the Oracle Technology Network, http://otn.oracle.com/products/text/content.html.

G.1.13 touch_index(p_indexes wwsbr_array)

```
procedure touch_index(
    p_indexes in wwsbr_array
)
```

Touches content for one or more indexes, that is, marks all index content (or indexes) as requiring synchronization by updating the column on which the index is created for every row in the table. For more information, see Section 8.3.5.4, "Synchronizing All the Index Content".

Once index content is marked in this way, use the procedure sync to re index the marked content.

Note that this procedure operates across multiple virtual private portal subscribers, it is not confined to the current subscriber. The procedure switches to each subscriber in turn and returns back to the original subscriber when complete.

Parameters:

`p_indexes` - An array containing index names to touch.

G.1.14 touch_index

```
procedure touch_index(
    p_index in varchar2 default null
)
```

Touches content for a single index, or for all indexes. This procedure is a convenient way to touch a single, named index. Alternatively, it can be used to touch all indexes, by passing the value *null*.

This procedure calls `touch_index(p_indexes wwsbr_array)` mentioned earlier.

For more information, see Section 8.3.5.4, "Synchronizing All the Index Content".

Parameters:

`p_index` - The name of the index to touch, or null to touch all indexes.

For more information, refer to `touch_index(p_indexes wwsbr_array)` described in Section G.1.13.

G.2 Constants

G.2.1 Index Name Constants

The following constants can be used in to specify the six Oracle Text indexes used by OracleAS Portal:

- **Page index -** wwv_context.PAGE_TEXT_INDEX

- **Document index -** wwv_context.DOC_TEXT_INDEX

- **Perspective index -** wwv_context.PERSPECTIVE_TEXT_INDEX

- **Item index -** wwv_context.ITEM_TEXT_INDEX

- **Category index -** wwv_context.CATEGORY_TEXT_INDEX

- **URL content index -** wwv_context.URL_TEXT_INDEX

PAGE_TEXT_INDEX

```
PAGE_TEXT_INDEX constant varchar2(30) := 'WWSBR_CORNER_CTX_INDX'
```

DOC_TEXT_INDEX

```
DOC_TEXT_INDEX constant varchar2(30) := 'WWSBR_DOC_CTX_INDX'
```

PERSPECTIVE_TEXT_INDEX

```
PERSPECTIVE_TEXT_INDEX constant varchar2(30) := 'WWSBR_PERSP_CTX_INDX'
```

ITEM_TEXT_INDEX

```
ITEM_TEXT_INDEX constant varchar2(30) := 'WWSBR_THING_CTX_INDX'
```

CATEGORY_TEXT_INDEX

```
CATEGORY_TEXT_INDEX constant varchar2(30) := 'WWSBR_TOPIC_CTX_INDX'
```

URL_TEXT_INDEX

```
URL_TEXT_INDEX constant varchar2(30) := 'WWSBR_URL_CTX_INDX'
```

G.2.2 URL Unsuitable for Indexing Constant

The absolute URL value used to indicate that a row should not be indexed. The URL index is created on the `wwsbr_url.absolute_url` column and this column is populated by a trigger.

If a URL is not suitable for indexing, such as URLs beginning with `javascript:`, this constant value is used. For more information, see Section 8.3.6.2, "Unsupported URLs".

G_NOINDEX

```
G_NOINDEX constant varchar2(15) := 'wwsbr_noindex'
```

G.3 Exceptions

INVALID_INDEX

The name of the index was not recognized.

```
INVALID_INDEX exception
```

H

Using TEXTTEST to Check Oracle Text Installation

OracleAS Portal uses the Oracle Text functionality to extend its search capabilities. If you want to check that Oracle Text functionality is working correctly, you can use the utility TEXTTEST. This utility is located at *MID_TIER_ORACLE_HOME*/portal/admin/texttest/textest.

This appendix contains the following sections:

- When to Use TEXTTEST
- Before Running TEXTTEST
- Running TEXTTEST
- Understanding TEXTTEST Results
- Configuring TEXTTEST
- Descriptions of TEXTTEST Tests

> **Note:** This utility only checks Oracle Text functionality that is specifically required by OracleAS Portal.

H.1 When to Use TEXTTEST

Oracle Text functionality is now enabled in OracleAS Portal by default and therefore all new OracleAS Portal installations expect Oracle Text to be present and functioning correctly. The TEXTTEST utility is useful if you want to:

- Check that Oracle Text functionality is working correctly prior to installing an Oracle Text enabled portal.

- Determine whether a problem with Oracle Text searching functionality within OracleAS Portal is due to an Oracle Text installation issues.

If you choose to disable Oracle Text searching functionality in OracleAS Portal, you do not need to run this utility.

H.2 Before Running TEXTTEST

1. You need to run the TEXTTEST utility from an Oracle Application Server Oracle home and it requires access to:

   - A working Perl installation (TEXTTEST has been tested with Perl 5.6.1).

   - Perl DBI and DBD::Oracle modules. The DBD::Oracle modules themselves require the Oracle database client libraries.

     To ensure access, set the path PATH $ORACLE_HOME/perl/bin:$PATH and set the Perl library path setenv PATH $ORACLE_HOME/perl/lib/5.6.1:$PATH.

   All of these are found in an Oracle Application Server Oracle home.

2. Ensure the correct Oracle home is selected.

   - For UNIX platforms, ensure the ORACLE_HOME environment variable is set and that the library path used by ld includes *ORACLE_HOME*/ctx/lib. The library path environment variable for the different UNIX platforms are as follows:

     Solaris, Tru64 UNIX, Linux: $LD_LIBRARY_PATH

     HP/UX: $SHLIB_PATH and $LD_LIBRARY_PATH

     IBM AIX: $LIBPATH

     For more, detailed information, see *About Inso Filtering Technology* in the *Oracle Text Reference*.

   - On Windows, use the Oracle home selector to choose the correct Oracle home.

   This is necessary so that the Perl DBD::Oracle module can find the correct Oracle client libraries. TEXTTEST also makes reference to the Oracle home environment variable. The Oracle home selected must be the Oracle Application Server Oracle home from where you intend to run the TEXTTEST utility.

3. Ensure that Perl can resolve the Perl module Portal::Text::Test.

This module resides at:

```
ORACLE_HOME/perl/lib/site_perl/5.6.1/Portal/Text/Test.pm
```

If you are using the Perl installation from the Oracle Application Server Oracle home, this is automatically included on the @INC path and no action is necessary. However, if you are using another Perl installation to run the utility, you may need to take steps to ensure that this location is included in the @INC path before running TEXTTEST. One way to do this, is to set the PERL5LIB environment variable to include:

```
ORACLE_HOME/perl/lib/site_perl/5.6.1
ORACLE_HOME/perl/lib/5.6.1:$PERL5LIB
```

4. If necessary, configure some of the tests that TEXTTEST will run.

For example, if your Oracle Application Server installation is behind a firewall and you perform URL tests that access content on the Internet. See Section H.5, "Configuring TEXTTEST".

H.3 Running TEXTTEST

The TEXTTEST utility is located at *MID_TIER_ORACLE_HOME*/portal/admin/texttest/textest. The default document directory is *ORACLE_HOME*/Portal/admin/texttest/doc.

You can run the TEXTTEST utility from the command line or DOS prompt. If you run the utility with no arguments, usage information is displayed. The command line arguments are detailed subsequently:

```
ORACLE_HOME/perl/bin/perl texttest -c sys_connect_string [-v] [-k] [-d document_
directory] [-t textcase_schema] [-p proxy] [-n noproxy]
```

Table H–1   TEXTTEST Parameters

Parameter	Description
-c	Connect string for the schema to connect as with DBA privileges in order to create the test schema. For example, sys/change_on_install@orcl as sysdba
-v	Show verbose output.
-k	Keep test schema after tests.

Table H–1 (Cont.) TEXTTEST Parameters

Parameter	Description
-d	Document directory containing documents to upload. The document indexing tests use these uploaded documents. If not specified, TEXTTEST looks for a directory called 'doc' in the same location as this script.
-t	Name of the test schema. This is the schema that is created and in which the tests are run. Default is TEXTCASE. The password will be the same as the schema name. If it already exists, the existing schema will be used. However without the -k option it will still be dropped at the end of the test, so be careful.
-p	Proxy to use for the URL indexing tests, For example, global.uk.mycompany.com:80. The port is optional. The same proxy is used for both HTTP and FTP URLs.
-n	No proxy domains, comma separated list of up to 16 domains that the proxy will not be used for. For example, uk.mycompany.com,us.mycompany.com
-u	URL indexing test datafile location.

The only mandatory argument is -c (the database connection information) and this must be a SQL*Plus style connect string. The schema specified is the one that is used to connect to the database. A separate schema will be created for running the tests.

The schema specified in the -c argument is not the schema used to run the tests. This schema needs DBA privileges. If you need to connect with a particular role, such as SYSDBA, when connecting to the SYS schema, specify this in the normal SQL*Plus format.

Note that if the -c argument contains spaces, you must add quotes. For example,

```
texttest -c 'sys/change_on_install@orcl as sysdba'
```

The -t argument specifies the name of the schema in which the tests are run. The default schema name is TEXTCASE. This schema is created in the early stages of the tests and is normally dropped at the end of the tests. You must ensure that this schema does not already exist in the database. If the test schema already exists, it is used but is dropped at the end of the testing.

H.4 Understanding TEXTTEST Results

By default, the output of the TEXTTEST is a simple statement of whether each test passed or failed, that is, OK or Not OK. For more detailed information about the tests and what causes them to fail, run TEXTTEST in verbose mode, that is, specifying the -v command line flag. The information displayed when the verbose mode is enabled is shown in more detail later.

For details on why a test fails, see Section H.6, "Descriptions of TEXTTEST Tests". Remember that if some of the tests fail, it may cause other tests to fail later on. For example, if the first connect to the database fails, then all subsequent tests also fail. Therefore, it is recommended that you investigate failures in the order they occur.

H.5 Configuring TEXTTEST

Use the file *ORACLE_HOME*/perl/lib/site_perl/5.6.1/Portal/Text/Config.pm to customize the default behavior of TEXTTEST. This file contains a Perl hash definition which itself contains definitions for various default values.

In most cases these values can be overridden by specifying command line arguments, as described in Section H.3, "Running TEXTTEST". If there is a default value defined in Config.pm and no command line value is specified, then the value from Config.pm is used.

Edit Config.pm if you want to change the default values permanently. This may be useful, for example, you always want to have proxy settings defined and you don't want to specify them every time on the command line. However, it is possible to successfully run TEXTTEST without modifying this configuration file.

H.5.1 Configuring Document Tests

OracleAS Portal uses Oracle Text functionality to search document content that is uploaded into the portal. When content is uploaded it is stored within OracleAS Portal database tables. Before it can be searched, the content must be indexed. During the indexing process Oracle Text processes each of the uploaded documents in turn. If the document is in a binary format (for example, a Word Document, or a Powerpoint document) it must be filtered and converted to plain text before it is indexed.

To test this functionality TEXTTEST creates a document table, uploads a number of files and attempts to filter them. The files that are uploaded are taken from a

document directory. The default location is configured in `Config.pm` as *ORACLE_HOME*`/Portal/admin/texttest/doc`.

Oracle Text cannot filter all documents. Therefore, some documents that are expected to fail the indexing test can be placed into the document directory, with a specific error reported. Since the error is expected, the test should still pass when the error occurs.

In order to test this behavior, you can configure a list of expected exceptions in an exceptions file. This file lists the filename and the expected error. You must enter one file name in each line followed by the expected error, separated by a space. If the filename contains a space, it should be escaped using \ as an escape character.

The error is treated as a Perl regular expression so it does not need to contain the whole error message. At the simplest level, you can specify part of the error string and this will match. This enables you to specify just the error code, for example. More complicated Perl regular expressions are also permissible. Refer to perldoc (on perlre page) for more information on Perl regular expressions. If the expected error is simply *, any exception is expected, and no failure whilst indexing this document will cause a test failure.

For example, the file might contain these four lines:

```
searchnotes.zip DRG-11207: user filter command exited with status 1
# The following PDF has security and cannot be filtered
my\ secured\ pdf.pdf DRG-11207
search.jar *
```

The first line includes the entire error. The second line is a comment that is ignored. The third line treats any DRG-11207 error as a expected. The fourth line can fail with any error and the test still passes.

By default, the document indexing exceptions file is called `index_exceptions` and it is located in the document indexing directory (configured in `Config.pm`). If the location is specified as a relative path, it is relative to the document directory.

Note that due to limitations in the Perl DBD::Oracle module it's not possible to stream the documents from the file system to the database. Instead the entire document is loaded into memory before being uploaded to the database. This means that it is necessary to have enough memory to contain the entire document. Only enough space for each document at a time is required.

H.5.2 Configuring URL Tests

OracleAS Portal uses Oracle Text functionality to fetch URLs that are listed as URL attributes, either on URL items, other items or pages. Once the fetched content is indexed and becomes searchable.

TEXTTEST tests this functionality by creating a similar URL index. The test data for URL testing consists of a list of URLs. TEXTTEST loads the URLs from a URL datafile. Each line in the data file contains a URL to attempt to index. It may also optionally contain an error message. If that error is found while indexing the corresponding URL, it is accepted as an expected error and does not cause the test to fail.

The expected error message is taken as a Perl regular expression that is matched against the error obtained from indexing. You must separate the expected error from the URL by a space character. If the expected error is specified as * then any error is treated as expected and does not cause the test to fail. For example

```
http://www.oracle.com
http://www.google.com DRG-11614: URL store: communication with host specified in
http://www.google.com timed out
http://www.notarealurl.com DRG-11612: URL store: unknown host specified in
http://www.notarealurl.com
http://www.anotherimaginaryurl.com DRG-11612
http://www.expectederror.com *
```

The first URL is expected to be found. An error is reported if it cannot be indexed.

`http://www.google.com` is expected to timeout (perhaps because the portal is behind a firewall and no proxies are specified). If this failure occurs the test will still pass.

`http://www.notarealurl.com` is expected to fail with an unknown host error.

`http://www.anotherimaginaryurl.com` is also expected to fail with an unknown host error. Note that it's not necessary to specify the whole error string. Because it's treated as a regular expression just the error code will match. If it fails with this error the test will still pass.

`http://www.expectederror.com` will never cause the test to fail. We have said that regardless of any errors that occur, we should still pass the test.

Section 8.3.9, "Viewing Indexing Errors" describes some of the most common Oracle Text URL error messages.

Expected and unexpected errors are reported when TEXTTEST is run in verbose mode (-v command line flag). When TEXTTEST is run it opens the URL data file

and uses it to populate the URL test table. This enables you to amend and augment the list of URLs used for testing by changing the contents of the file.

The default location for the URL datafile is specified in the file `Config.pm`. Alternatively you can specify a URL test datafile using the `-u` command line argument when running TEXTTEST. For example,:

```
texttest -c 'sys/change_on_install@orcl as sysdba' -u ORACLE_
HOME/Portal/texttest/url
```

ORACLE_HOME`/Portal/texttest/url` is the default location for the URL datafile, within the Oracle Application Server Oracle home.

You may change the URL details if you think a specific URL is causing problems in your portal installation. Or perhaps, your Oracle Application Server installation resides behind a firewall and you wish to change the URL test data to include URLs that are local to your intranet, rather than public URLs on the Internet.

H.5.3 URL Tests and Proxies

If your Portal installation resides behind a firewall it may be necessary to configure Oracle Text to use a proxy before it can fetch URLs that reside beyond the firewall.

If you run TEXTTEST in these circumstances without setting proxies, the URL indexing tests fail. In this case you have three choices:

- Remove the failing URLs from the test dataset. Simply remove the line from the URL data file.

- Mark the offending tests as expected to fail. Do this by placing the URL followed by the expected error message in the URL data file.

- Specify a proxy to use. See Section H.5.4, "Specifying Proxies for Use with URL Indexing Tests".

H.5.4 Specifying Proxies for Use with URL Indexing Tests

You can specify a proxy to use in two locations:

- In the file `Config.pm` that contains separate settings for *ftp_proxy* and *http_proxy*.

- Using the `-p` parameter for the TEXTTEST script. In this case, the same proxy is used for both HTTP and FTP proxies.

In both cases the form of the proxy should be `<hostname>.<domain>:<port>`. The port is optional. For example,

```
www-proxy.us.abc.com:80
emeacache.abc.com
```

The `-n` command line argument and the no_proxy `Config.pm` setting can both be used to specify a list of domains for which the proxy should not be used. The list should be comma separated. For example,

```
uk.abc.com,us.abc.com,abc.com
```

H.6 Descriptions of TEXTTEST Tests

This section describes each of the tests that TEXTTEST performs and outlines some of the common causes for failure of each test.

H.6.1 Connect to Database as User sys

Description:
Connects to the database as the privileged user used to create the test schema. This is referred to the *sys* user or the *sys* schema. However, it does not have to be the user *sys*, any sufficiently privileged user will suffice.

Possible cause of failure:
- Incorrect schema name or password.
- If the user, such as *sys*, needs to connect with a specific role then the roles must be specified in the in the connect string in the usual format, that is, `sys/change_on_install as sysdba`.

When this test fails, it causes other tests to fail.

H.6.2 Create textcase Schema

Description:
Creates the schema into which test objects are installed. By default, this schema is called `textcase` and it is referred to as the *test* schema.

Possible cause of failure:
- The user with which TEXTTEST is connected, does not have privileges to create other users.

There are several other reasons why it might not be possible to create a new schema, for example, there may be insufficient space in the database.

When this test fails, it causes other tests to fail.

H.6.3 Grant DBA Role to textcase Schema

Description:

Grants the DBA role to the `test` schema. This allows it to directly create and remove objects from the `ctxsys` schema.

Possible cause of failure:

- The user with which TEXTTEST is connected, does not have the necessary privileges to grant the DBA role to another user. It must have the DBA role itself to do this.

H.6.4 Grant CTXAPP Role to textcase Schema

Description:

Grants the CTXAPP role to the *test* schema. This is required when using Oracle Text features.

Possible cause of failure:

- The user with which TEXTTEST is connected, does not have the necessary privileges to grant CTXAPP to another user. It must have the DBA role itself to do this.

- The CTXAPP role is missing. This indicates an incomplete, corrupt or missing Oracle Text installation.

H.6.5 Disconnect From sys

Description:

TEXTTEST disconnects from the *sys* schema in order to reconnect to the *test* schema.

Possible cause of failure:

- No obvious cause of failure.

H.6.6 Connect to textcase Schema

Description:

TEXTTEST reconnects to the *test* schema in order to begin creating schema objects and running Oracle Text tests.

Possible cause of failure:
- No obvious cause of failure.

H.6.7 Create textcase Item Related Tables

Description:

Creates the tables used for testing item indexing with a user datastore.

Possible causes of failure:
- No obvious cause of failure.
- General database problems such as insufficient free tablespace to complete the operation.

H.6.8 Populate Item Tables

Description:

Populates the tables used for item indexing tests. They are populated using data held within the TEXTTEST script itself.

Possible cause of failure:
- No obvious cause of failure.

H.6.9 Create Document Table

Description:

Creates the table used for document filtering and indexing tests.

Possible cause of failure:
- No obvious cause of failure.

H.6.10 Populate Document Table

Description:

Populates the document table from a specified document directory.

Possible cause of failure:

- The specified document directory cannot be found or is not readable. The files within the document directory must be readable.

- Insufficient memory on the machine where TEXTTEST is running to hold any one of the documents in memory.

H.6.11 Create URL Table

Description:

Creates the table used for URL indexing tests.

Possible cause of failure:

- No obvious cause of failure.

H.6.12 Populate URL Table

Description:

Populates the tables used for URL indexing tests. They are populated from the URL datafile. See Section H.6.11, "Create URL Table".

Possible cause of failure:

- The URL indexing datafile cannot be found, or is not readable.

- Data within the URL datafile is in an incorrect format.

H.6.13 Create Oracle Text Datastore Procedure

Description:

Creates a datastore procedure in the ctxsys schema. The *test* user has DBA privileges and this procedure is created or replaced, so if the ctxsys schema is installed, there should not be a problem.

Possible cause of failure:

- The `ctxsys` schema is not present, which also implies that Oracle Text is not installed in the database.

H.6.14 Create Oracle Text Preferences

Description:

Creates the Oracle Text preferences (not including the Lexer preferences). Any existing preferences are dropped to avoid clashes.

Possible cause of failure:

- Problems with the Oracle Text installation.

- Problems with the compatibility of the preferences that TEXTTEST is attempting to create with this Oracle Text version, that is, preference version is not as expected.

H.6.15 Create Lexer Preferences

Description:

Creates the Oracle Text lexer preferences. Any existing preferences are dropped to avoid clashes.

Possible cause of failure:

- Problems with the Oracle Text installation.

- Problems with the compatibility of the preferences that TEXTTEST is attempting to create with this Oracle Text version, that is, preference version is not as expected.

H.6.16 Create Section Group and Zone Sections

Description:

Creates the section groups and zone sections for the item indexing tests.

Possible cause of failure:

- No obvious cause of failure.

- Possibly a problem with the Oracle Text installation, or one of the previous test having failed.

H.6.17 Create Oracle Text Item Index

Description:

Creates the Oracle Text index for testing item indexing with a user datastore. This test does **not** populate the index.

Possible cause of failure:

- No obvious cause of failure.
- Possibly a problem with the Oracle Text installation, or one of the previous test having failed.

H.6.18 Create Oracle Text Document Index

Description:

Creates the Oracle Text index for testing document indexing. This test does **not** populate the index.

Possible cause of failure:

- No obvious cause of failure.
- Possibly a problem with the Oracle Text installation, or one of the previous test having failed.

H.6.19 Create Oracle Text URL Index

Description:

Creates the Oracle Text index for testing URL indexing. This test does **not** populate the index.

Possible cause of failure:

- No obvious cause of failure.
- Possibly a problem with the Oracle Text installation, or one of the previous test having failed.

H.6.20 Touch All Item Content So That Pending

Description:

Updates all of the rows in the items test table so that they are placed in the Oracle Text pending queue.

Possible cause of failure:
- No obvious cause of failure.
- Possibly a problem with the Oracle Text installation, or one of the previous test having failed.

H.6.21 Touch All Document Content So That Pending

Description:

Updates all of the rows in the document test table so that they are placed in the Oracle Text pending queue.

Possible cause of failure:
- No obvious cause of failure.
- Possibly a problem with the Oracle Text installation, or one of the previous test having failed.

H.6.22 Touch All URL Content So That Pending

Description:

Updates all of the rows in the URL test table so that they are placed in the Oracle Text pending queue.

Possible cause of failure:
- No obvious cause of failure.
- Possibly a problem with the Oracle Text installation, or one of the previous test having failed.

H.6.23 Synchronize Item Index

Description:

Synchronizes the Oracle Text index on the item indexing test tables. This causes the content to be indexed.

Since the dataset used for the item indexing is controlled and internal to the TEXTTEST script, this test is always expected to pass.

Possible cause of failure:

- A previous test has failed.

- Possibly a problem with the Oracle Text installation. Verify the Oracle Text installation and reinstall if necessary. Ensure that you complete *all* manual steps for any database upgrades, as these often contain Oracle Text related steps.

H.6.24 Synchronize Document Index

Description:

Synchronizes the Oracle Text index on the document indexing test table. This causes the content to be indexed.

Possible cause of failure:

- One of the documents uploaded for the test could not be filtered. This is not necessarily a problem as the document might not be in one of the formats that are filterable by Oracle Text.

  Consult the *Oracle Text Reference* (see chapter on supported formats). Either remove the document, or mark it as an expected failure (see Section H.5.1, "Configuring Document Tests").

- An unexpected indexing failure, either caused by a bug in the filtering software or by incorrect configuration.

  Consult the *Oracle Text Reference* and Chapter 8, "Configuring the Search Features in OracleAS Portal". If the Oracle Text installation is configured correctly and the document format is a supported one but it still cannot be filtered, please contact Oracle Support.

H.6.25 Synchronize URL Index

Description:

Synchronizes the Oracle Text index on the URL indexing test tables. This causes the content to be indexed.

Possible cause of failure:

- One of the URLs specified in the URL indexing test data may not be returning HTML or plain text that can be indexed by Oracle Text. This can happen for a number of reasons. The URL may be incorrect or the site might be unavailable.

- If the database instance is behind a firewall and the URL is beyond the firewall, then it might be necessary to configure the tests to use a proxy server. See Section H.5.2, "Configuring URL Tests". If the URL is expected to fail, you can marked it as such in the URL test data so that this test will pass.

H.6.26 Drop Datastore Procedure from ctxsys

Description:

Drops the datastore procedure created in the `ctxsys` schema.

This test is not carried out if the `-k` option is used to keep the *test* schema once the tests are completed. See Section H.3, "Running TEXTTEST".

Possible cause of failure:

- No obvious cause of failure.

H.6.27 Disconnect From textcase Schema

Description:

Disconnects from the *test* schema.

Possible cause of failure:

- No obvious cause of failure.

H.6.28 Connect As User sys

Description:

Reconnects to the sys schema in order to drop the test schema.

This test is not carried out if the -k option is used to keep the *test* schema once the tests are completed. See Section H.3, "Running TEXTTEST".

Possible cause of failure:

■   No obvious cause of failure.

H.6.29 Drop textcase Schema

Description:

Drops the *test* schema.

This test is not carried out if the -k option is used to keep the *test* schema once the tests are completed. See Section H.3, "Running TEXTTEST".

Possible cause of failure:

■   No obvious cause of failure.

H.6.30 Disconnect From Database

Description:

Disconnects from the sys schema.

Possible cause of failure:

■   No obvious cause of failure.

I

Administering Web Clipping

The Web Clipping provider is a provider to Oracle Application Server Portal. It provides the Web Clipping portlet that renders clipped Web content as a portlet. The Web Clipping portlet enables you to collect Web content into a single, centralized portlet. You can use it to consolidate content from hundreds of Web sites scattered throughout a large organization.

Before you use the Web Clipping portlet, you must perform some administrative tasks that include:

- Configuring the Web Clipping Repository

- Configuring HTTP or HTTPS Proxy Settings

- Configuring Caching

    This section describes configuring caching and using OracleAS Web Cache. See Section 1.3, "Understanding Caching in OracleAS Portal" and Section 5.7, "Configuring OracleAS Web Cache Caching in OracleAS Portal" for more information about caching.

- Adding Certificates for Trusted Sites

    This section in Chapter 6, "Securing OracleAS Portal" describes how to configure or extend the trusted certificate file. A trusted server certificate file, ca-bundle.txt, generated from Oracle Wallet Manager is shipped with the Web Clipping Portlet feature. This file, located in *ORACLE_HOME*/portal/conf on UNIX or in *ORACLE_HOME*\portal\conf on Windows, contains an initial list of trusted server certificates that might be used for navigating to some secure servers using HTTPS. However, this is not a complete list of all possible server certificates that exist on the Web. Therefore, this file must be configured or extended to recognize any additional trusted server certificates for any new trusted sites that are visited. See Section 6.1.9,

"Securing the Web Clipping Provider" for more information about how to configure or extend this trusted certificate file.

- Configuring Oracle Advanced Security for the Web Clipping Provider

  This section in Chapter 6, "Securing OracleAS Portal" describes configuring Oracle Advanced Security Option (ASO) to secure and encrypt the channel between itself (at the middle-tier) and the database, which hosts the Web Clipping Repository.

I.1 Configuring the Web Clipping Repository

Web clippings have definitions that must be stored persistently in the Web Clipping Repository hosted by an Oracle9*i* Database Server. The Web Clipping Provider test page automatically detects whether the Web Clipping Provider is configured to access a database. If it is not configured to do so, the Web Clipping Provider displays an **Edit** link next to the status of the **Web Clipping Repository** field to let you configure or reconfigure the database connection parameters.

Configuring the Web Clipping Repository Using the Web Clipping Provider Test Page

As Portal Administrator, you can configure the Web Clipping Repository using the Web Clipping Provider test page at

```
http://<host>:<port>/portalTools/webClipping/providers/webClipping
```

The provider Test Page will first automatically detect whether the Web Clipping Provider is configured to access the database. If not, it will display an **Edit** link next to the status of the Web Clipping Repository. This enables you to reconfigure the database connection parameters in the Edit Provider Page.

If this is the first time you are installing the Web Clipping Provider, you will need to request a database user account from your database administrator before you begin configuring. For more information, click **Learn More** in the **Provider Configuration** section of the Web Clipping Provider test page and see **Web Clipping Provider Test Page** section.

Under the **Provider Configuration** section, in the **Setting** column, there is a **Web Clipping Repository** field. Click its corresponding **Edit** link in the **Actions** column. In the **Repository Settings** section of the **Edit Provider: webClipping** page, you can specify the database connection information for the Web Clipping Provider, then select **OK** to save the settings and return to the Web Clipping Provider test page.

The database connection information consists of first selecting the repository target database, of which there are two choices: OracleAS Infrastructure database (default) or Other 9i Database. If you select the option **Oracle Application Server Infrastructure Database (default)**, no other connection parameters need be specified.

If you need to connect to another Oracle9*i* Database Server, in the **Repository Target** field, select the **Other 9i Database** option, then specify the following connection parameters for the following fields: **Database Server Host**, **Database Listener Port**, **Database SID**, **Database Username**, and **Database Password**.

If you have specified for your Repository Target the same database as the one used for a PDK 9.0.2.4.0 installation before, as part of your PDK upgrade, you will be notified that a Repository Upgrade also needs to take place. Upon entering the Web Clipping Provider Test Page, you will see a **Upgrade (from 9.0.2.4.0)** link that enables you to do a one-click upgrade for installing new tables as well as migrating existing Clipping Definitions to the latest versions.

> **Note:** After the upgrade, the Clipping Definitions stored in the Web Clipping Repository can no longer work with PDK 9.0.2.4.0.

Finally, if you require a secure database connection, in the **Advanced Security Option** field, select the **enable (secure database connections)** option. See Section 6.1.9.2, "Configuring Oracle Advanced Security for the Web Clipping Provider" for more information about configuring the Advanced Security Option.

For more information, click **Learn More** in the **Provider Configuration** section of the Web Clipping Provider test page.

I.2 Configuring HTTP or HTTPS Proxy Settings

Your HTTP or HTTPS proxy settings must be set to allow the Web Clipping Studio to go through firewalls for HTTP requests.

I.2.1 Configuring Proxy Settings Using the Web Clipping Provider Test Page

As Portal Administrator, you can configure proxy settings using the Web Clipping Provider test page at

```
http://<host>:<port>/portalTools/webClipping/providers/webClipping
```

Under the **Provider Configuration** section, in the **Setting** column, there is an **HTTP Proxy** field. Click its corresponding **Edit** link in the **Actions** column. In the **Proxy Settings** section of the **Edit Provider: webClipping** page, enter your proxy settings for the Web Clipping Provider, then select **OK** to save the settings and return to the Web Clipping Provider test page. For more information, click **Learn More** in the **Provider Configuration** section of the Web Clipping Provider test page.

I.2.2 Setting Proxy Settings Manually

As the Portal Administrator, you can also set proxy settings manually according to your HTTP or HTTPS configuration. Edit the appropriate entries in the `provider.xml` file located on UNIX and Windows in the following directory:

On UNIX

```
ORACLE_HOME/j2ee/OC4J_Portal/applications/
portalTools/webClipping/WEB-INF/providers/webClipping
```

On Windows

```
ORACLE_HOME\j2ee\OC4J_Portal\applications\
portalTools\webClipping\WEB-INF\providers\webClipping
```

After modifying the proxy settings manually, use `opmnctl` to restart the OC4J instances, which includes the OC4J_Portal instance, in order for these proxy changes to take effect.

I.2.3 Restricting Users from Clipping Content from Unauthorized External Web Sites

The Web Clipping Provider provides a basic mechanism for restricting users from clipping content from unauthorized external Web sites, using the proxy exception list. This is only available for environments that utilize a proxy server to reach external Web sites. The proxy exception list is listed under `Proxy Settings` in the Edit Provider page, which is linked from the Web Clipping Provider test page. By setting up your proxy host and port as before, you can now include a list of domains in the proxy exception field for which you want to restrict users from clipping. Users attempting to reach a Web site in one of the listed domains, from the Web Clipping Studio, will see an HTTP timeout error.

I.3 Configuring Caching

By default, validation-based caching is used through OracleAS Portal for all Web clipping portlets. With validation-based caching, the Parallel Page Engine (PPE) contacts the Portal Provider to determine if the cached item is still valid.

If you have Oracle Application Server Web Cache installed, you can elect to use invalidation-based caching through Web Cache. Note that each type of caching is mutually exclusive, that is, you can choose to use only one or the other, but not both.

With invalidation-based caching, an item remains in the cache until the cache receives notification that the item needs to be refreshed. For example, if the Web clipping portlet contains content that is updated on a regular basis, the cache will be invalidated. Invalidation-based caching as shown in Figure I–1 decreases the number of requests the Web Clipping Provider must entertain while maintaining the same network traffic for each round trip involving PPE. Depending on your deployment scenario, you may prefer using one caching method over the other. For more information about caching, refer to Section 1.3, "Understanding Caching in OracleAS Portal".

Section I.3.1, "Configuring Caching Using the Web Clipping Provider Test Page" and Section I.3.2, "Configuring Web Cache Manually" describe how to configure caching either using the Web Clipping Provider test page or enabling Web Cache manually. By default, the Web Clipping Provider uses Portal caching (validation-based caching). If you want to use Web Cache (invalidation-based caching), see Section I.3.1 or Section I.3.2.

If you decide to use Web Cache to cache Web clipping content, as a final step, you must use the Portal Navigator and change the connect string for the provider URL to point to a URL with the Web Cache port (`portalTools/builder/providerui/Navigator?event=init`). Usually the port is 7778 with Web Cache. Check the Oracle Application Server port usage page to verify this value. See *Oracle Application Server 10g Administrator's Guide* for a list of default port numbers used by OracleAS components and services.

In this configuration, Web Cache caches Web clipping content between the OracleAS Portal instance and the Web Clipping Provider.

Figure I–1   Invalidation-Based Caching Provided by Oracle Web Cache
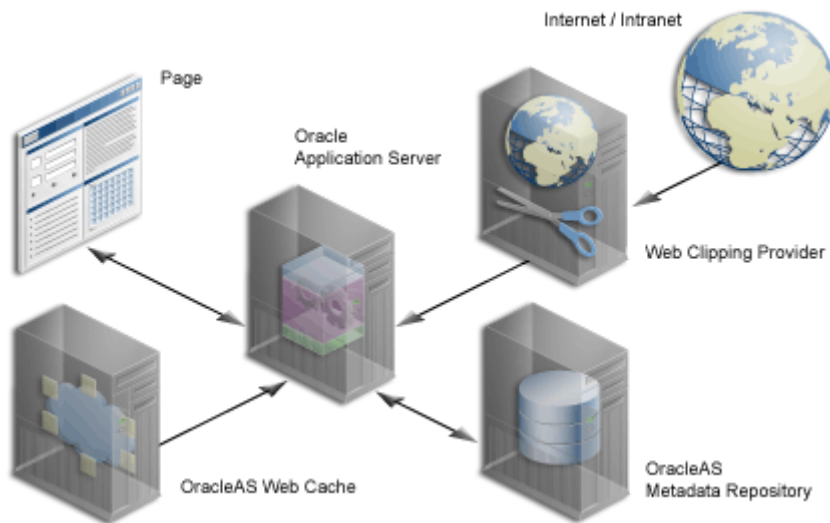
I.3.1 Configuring Caching Using the Web Clipping Provider Test Page

As Portal Administrator, you can configure caching using the Web Clipping Provider test page at

```
http://<host>:<port>/portalTools/webClipping/providers/webClipping
```

Under the **Provider Configuration** section, there is a Portlet Caching setting entry. Click its corresponding **Edit** link in the **Actions** column. In the **Caching Parameters** section of the **Web Clipping Portlet** page, select the caching scheme in the pull-down menu (either `validation` or `invalidation (requires OracleAS Web Cache)`) in the **Caching Scheme** field, then specify a cache expires value in the **Cache Expires** field. The default value for cache expires is 30 minutes. After configuring caching for the Web Clipping Provider, select **OK** to save the settings and return to Web Clipping Provider test page. For more information, click **Learn More** in the **Provider Configuration** section of the Web Clipping Provider test page.

I.3.2 Configuring Web Cache Manually

To enable Web Cache, you must first check the `cache.xml` file in the following directory on UNIX and on Windows to verify the accurate values of the invalidation host and port number:

On UNIX

`ORACLE_HOME/portal/conf`

On Windows

`ORACLE_HOME\portal\conf`

Next, you must manually update the `provider.xml` file located in the following directory on UNIX and on Windows:

On UNIX

```
ORACLE_HOME/j2ee/OC4J_Portal/applications/
portalTools/webClipping/WEB-INF/providers/webClipping
```

On Windows

```
ORACLE_HOME\j2ee\OC4J_Portal\applications\
portalTools\webClipping\WEB-INF\providers\webClipping
```

Perform the following steps:

1. Search for the **useInvalidationCaching** tag and set its value to true to enable Web Cache invalidation-based caching.

2. Search for the **cacheExpires** tag and set your default value if you wish to modify that value. This value is in minutes.

Part V

Index

Part five contains the Index.

Index

R