Oracle® HTTP Server

Administrator's Guide

10*g* Release 2 (10.1.2)

B14007-04

March 2006

ORACLE®

Oracle HTTP Server Administrator's Guide, 10*g* Release 2 (10.1.2)

B14007-04

Primary Author: Harry Schaefer

Contributing Author: Julia Pond, Sanket Atal, Warren Briese, Olivier Caudron, Kevin Clark, Priscila Darakjian, Sander Goudswaard, Helen Grembowicz, Mathew Joy, Pushkar Kapasi, Keith Kelleman, Eric Kienle, John Lang, Bruce Lowenthal, Li Ma, Chuck Murray, Mark Nelson, Carol Orange, Bert Rich, Jon Richards, Shankar Raman, Baogang Song, Kevin Wang, Karen Wilson

Contents

4 Managing Server Processes

5 Managing the Network Connections

6 Configuring and Using Server Logs

7 Application Server Control Console Management

8 Understanding Modules

10 Managing Security

11 Enabling SSL for Oracle HTTP Server

12 Using PHP with Oracle Application Server

A Using Oracle Application Server Proxy Plug-in

B Using Oracle Application Server SSO Plug-in

C Using Oracle Application Server Containers for J2EE Plug-in

D Load Balancing Using mod_oc4j

E Configuration Files

F Frequently Asked Questions

G Troubleshooting Oracle HTTP Server

H Third Party Licenses

Glossary

Index

Preface

This guide describes how to administer Oracle HTTP Server.

Audience

Oracle HTTP Server Administrator's Guide is intended for application server administrators, security managers, and managers of databases used by application servers.

Documentation Accessibility

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

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

Accessibility of Code Examples in Documentation

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

Accessibility of Links to External Web Sites in Documentation

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

TTY Access to Oracle Support Services

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

Related Documents

For more information, see the Oracle Application Server Documentation Library.

Conventions

The following text conventions are used in this document:

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

1

Overview

This chapter describes the Oracle HTTP Server, highlighting the differences between the Oracle distribution and the open source Apache product on which it is based. It also explains how to start, stop, and restart the server.

Topics discussed are:

- Oracle HTTP Server Features
- Oracle HTTP Server Components
- Oracle HTTP Server Support
- Oracle HTTP Server Management
- Starting, Stopping, and Restarting Oracle HTTP Server

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Oracle HTTP Server Features

Oracle HTTP Server is the Web server component of Oracle Application Server. Based on the **Apache** infrastructure, Oracle HTTP Server allows developers to program their site in a variety of languages and technologies - Perl (through `mod_perl` and CGI), C (through CGI, and FastCGI), C++ (through FastCGI), PHP, and Oracle's PL/SQL. It can also be a proxy server, both forward and reverse. In addition, the features of single sign-on, clustered deployment, and high availability, enhance the operations of the Oracle HTTP Server.

Based on Apache - HTTP v1.1 Support

Oracle HTTP Server code is based on Apache 1.3 Web Server (`http://www.apache.org`). With such a proven code base, Oracle HTTP Server provides Oracle Application Server customers with the stability, flexibility, and scalability required of a Web server.

Security - Encryption with SSL

Secure Sockets Layer is required to run any Web site securely. Oracle HTTP Server supports SSL encryption based on industry standard, patented, algorithms. The SSL

works seamlessly with both Internet Explorer and Netscape browsers. In addition, the infrastructure has been upgraded to share the same wallet information as the database users. Features include:

- **SSL HW Acceleration Support**: SSL encryption is slower when done in software. Dedicated hardware support for this purpose is now supported, specifically with nCipher.
- **Variable Security per Directory**: This feature allows individual directories to be protected by different strength encryption.
- **Oracle HTTP Server to OC4J SSL Support**: Oracle HTTP Server and OC4J can communicate using AJP protocol over SSL. Previously, Oracle HTTP Server and OC4J used the AJP 1.3 protocol unencrypted, without support for authentication. Now, Oracle HTTP Server has been modified to extend support to the AJP 1.3 protocol over SSL providing both encryption and authentication.

See Also:

- *Oracle Application Server Security Guide*
- Chapter 10, "Managing Security"
- Chapter 11, "Enabling SSL for Oracle HTTP Server"

Security - Single Sign On

Oracle HTTP Server supports the standard basic authentication features of HTTP servers. The source for the username and password used here is a flat file (with encrypted passwords). In addition, a module, `mod_osso`, is included to support single sign on across sites and across applications. This provides for a much better end user experience (they have to login only once), and a much easier development cycle (most of the security is declarative).

See Also:

- *Oracle Application Server Single Sign-On Administrator's Guide*
- "mod_osso" on page 8-19

Virtual Hosts

The virtual host facility allows an HTTP server to service multiple domain names over one IP address. Thus, virtual hosts `www.north.com` might have the same IP address as `www.south.com`. Oracle HTTP Server provides a "container" environment for a virtual host, thus providing a virtual host with its own set of security and other configuration directives, in addition to locations from which the files are served. This allows an ISP to save on hardware and administrative costs by enabling hundreds to thousands of sites to be served from a single runtime instance of Oracle HTTP Server. Only one virtual host on a single IP address can accommodate SSL. Oracle HTTP Server can support multiple IP addresses and each one of them can have one, but only one virtual host.

Distributed Authoring and Versioning Support

WebDAV, an IETF standard, is an HTTP based protocol that allows DAV enabled clients, such as MS Office, Windows Explorer, to edit files on a server. The Apache Software Foundation provides a module, `mod_oradav`, that provides support for file based storage on the server. In addition to providing this functionality, Oracle HTTP Server enables the server side store to be a database or other repository.

See Also: Chapter 9, "Configuring and Using mod_oradav"

Proxy Server and URL Rewriting

Any Web site that is "alive" changes often. Along with that, the directory structure and the URLs change. Oracle HTTP Server makes it easy to accommodate these changes by including an engine that support URL rewriting so that the end users do not have to change their bookmarks. It also supports reverse proxy capabilities, thus making it easier to make content served by different servers to appear from one single server.

Oracle Application Server Proxy Plug-in

A separately available component which enables IIS and Sun ONE Web servers to route requests to Oracle Application Server. Users can benefit from all of the Oracle Application Server features even if their corporate standard requires them to use IIS or Sun ONE Web servers. The proxy plug-in provides Oracle Application Server features, such as single sign-on, load balancing, and AJP port tunneling, to be accessed when using IIS or Sun ONE Web servers.

> **See Also:** Chapter A, "Using Oracle Application Server Proxy Plug-in"

Oracle Application Server SSO Plug-in

A separately available component which enables IIS and Sun ONE Web servers to be integrated with Oracle Application Server Single Sign-on. IIS and Sun ONE listener applications can now be protected by using the single sign-on infrastructure. You can now be authenticated to these listeners using only one single sign-on password. This functionality is similar to what `mod_osso` provides to Oracle HTTP Server.

> **See Also:**
>
> - Appendix B, "Using Oracle Application Server SSO Plug-in"
> - "mod_osso" on page 8-19

Oracle Application Server Containers for J2EE Plug-in

A separately available component which enables IIS, Sun ONE and Apache 1.3.x Web servers to route requests directly to OC4J. This functionality is similar to what `mod_oc4j` provides for Oracle HTTP Server.

> **See Also:**
>
> - Chapter C, "Using Oracle Application Server Containers for J2EE Plug-in"
> - "mod_oc4j" on page 8-9

PL/SQL Stored Procedures

This feature allows access to PL/SQL code stored in the Oracle database.

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

PL/SQL Server Pages

Similar in concept to the Java Server Pages, this module allows PL/SQL to be used as the scripting language within an HTML page. It gets translated into a stored procedure, which then uses the module described earlier (for PL/SQL stored procedure) to send the output to the browser.

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

Server Side Include

Server Side Includes provide an easy way of adding some dynamic, or uniform static content, across all the site's pages. It is typically used for header/footer information. Oracle HTTP Server supports special directives to enable these only for certain types of files, or for given virtual hosts.

Perl

Perl is a scripting language often used to provide dynamic content. Perl can either be called as a CGI program, or directly through `mod_perl`. Oracle Application Server uses Perl version 5.6.1.

See Also: "mod_perl" on page 8-19

PHP

PHP is an open source, widely-used, general-purpose, client-side scripting language, that is embedded in standard HTML. It is used to generate dynamic HTML pages.

See Also:

Chapter 12, "Using PHP with Oracle Application Server"

"mod_php" on page 8-22

C/C++ (CGI and FastCGI)

CGI programs have been commonly used to program Web applications. Oracle HTTP Server enhances them by providing a mechanism to keep them alive beyond the request lifecycle, thus improving the performance tremendously.

Dynamic Monitoring Service

Dynamic Monitoring Services (DMS) metrics give runtime performance statistics for both Oracle HTTP Server and OC4J processes. As applications run, DMS collects detailed performance statistics. This data enables you to monitor the duration of important request processing phases and status information. With this information, you can locate performance bottlenecks and tune the application server to maximize throughput and minimize response time. Process metrics as well as event information is available and can be viewed through Oracle Enterprise Manager.

See Also: *Oracle Application Server Performance Guide*

Oracle Process Manager and Notification Server

Oracle Application Server provides a high availability infrastructure integration with Oracle Process Manager and Notification Server (OPMN), for process management, death detection, and failover for OC4J and Oracle HTTP Server processes.

See Also:

- *Oracle Application Server High Availability Guide*
- *Oracle Process Manager and Notification Server Administrator's Guide*

Distributed Configuration Management

Distributed Configuration Manager (DCM) enables cluster wide deployments in a very simple way. J2EE applications for OC4J can now be deployed to a cluster, or a new node can be added to the cluster with ease.

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

Load Balancing

Oracle HTTP Server includes a module called `mod_oc4j` that routes requests from the OC4J instances in a cluster. OPMN helps ensure that `mod_oc4j` instances knows of all the OC4J in the system without requiring a system administrator to do any configuration.

See Also: Appendix D, "Load Balancing Using mod_oc4j"

Oracle HTTP Server Components

Oracle HTTP Server consists of several components that run within the same process. These components provide the extensive list of features that Oracle HTTP Server offers when handling client requests. The following are the major components:

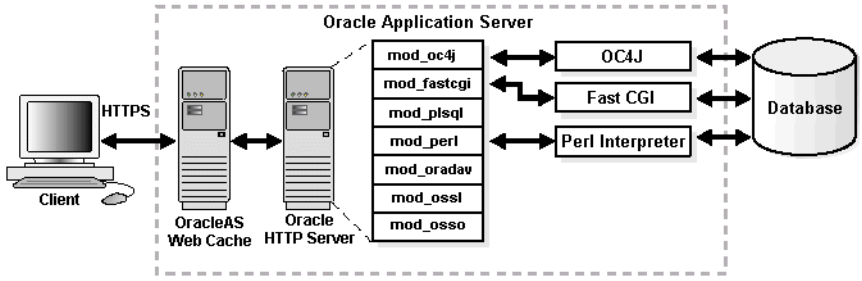
- **HTTP Listener**: Oracle HTTP Server is based on an Apache HTTP listener to serve client requests. An HTTP server listener handles incoming requests and routes them to the appropriate processing utility.
- **Modules (mods)**: Modules both implement and extend the basic functionality of Oracle HTTP Server. Many of the standard Apache modules are included with Oracle HTTP Server. Oracle also includes several internal modules that are specific to Oracle Application Server components.

  See Also: "Oracle HTTP Server Modules" on page 1-5

- **Perl Interpreter**: A persistent Perl runtime environment embedded in Oracle HTTP Server through mod_perl.

Figure 1–1 shows the path of various requests through Oracle HTTP Server components, where a client machine connects to Oracle Application Server Web Cache, which in turn connects to Oracle HTTP Server. Oracle HTTP Server, using various modules, connects to the database through OC4J, FastCGI, or the Perl interpreter.

Figure 1–1 Oracle HTTP Server Request Flow



See Also: *Oracle Application Server Concepts*

Oracle HTTP Server Modules

Table 1–1 identifies the modules shipped with Oracle HTTP Server. Modules extend the basic functionality of the Web server, and support integration between Oracle HTTP Server and other Oracle Application Server components. Note that the list differs from the Apache open source distribution (given the inclusion of Oracle modules).

Table 1–1 Oracle HTTP Server Modules

Module	Notes	Module	Notes
mod_access		mod_log_referer	Deprecated.
mod_actions		mod_mime	
mod_alias		mod_mime_magic	
mod_asis		mod_mmap_static	
mod_auth		mod_negotiation	
mod_auth_anon		mod_oc4j	Oracle module.
mod_auth_dbm		mod_onsint	Oracle module.
mod_autoindex		mod_oradav	Oracle module.
mod_cern_meta		mod_ossl	Oracle module.
mod_certheaders	Oracle module	mod_osso	Oracle module.
mod_cgi		mod_perl	
mod_define	UNIX systems only.	mod_php	
mod_digest		mod_plsql	Oracle module.
mod_dir		mod_proxy	
mod_dms	Oracle module.	mod_rewrite	
mod_env		mod_security	
mod_example		mod_setenvif	
mod_expires		mod_speling	
mod_fastcgi		mod_status	
mod_headers		mod_unique_id	UNIX systems only.
mod_imap		mod_userdir	
mod_include		mod_usertrack	
mod_info		mod_vhost_alias	
mod_log_agent	Deprecated.	mod_wchandshake	Oracle module.
mod_log_config			

See Also: Chapter 8, "Understanding Modules"

Oracle HTTP Server Support

Oracle provides technical support for the following Oracle HTTP Server features and conditions:

- Modules included in the Oracle distribution. Oracle does not support modules obtained from any other source, including the Apache Software Foundation. Oracle HTTP Server will still be supported when non-Oracle provided modules are included. If it is suspected that the non-Oracle provided modules are contributing to reported problems, customers may be requested to reproduce the problems without those modules being included.
- Problems that can be reproduced within an Apache configuration consisting only of supported Oracle Apache modules.

- Use of the included Perl interpreter within the supported Apache configuration.

Oracle HTTP Server Management

Oracle HTTP Server can be managed using the following two methods:

- Application Server Control Console
- Oracle Application Server Command-line Tools

Application Server Control Console

You can manage Oracle HTTP Server using Oracle Enterprise Manager 10*g*. Oracle Enterprise Manager 10*g* enables you to manage your server from a Web browser using **Oracle Enterprise Manager 10g Application Server Control Console** (Application Server Control Console).

See Also:

- Chapter 7, "Application Server Control Console Management"
- *Oracle Enterprise Manager Concepts*

Oracle Application Server Command-line Tools

You can manage Oracle HTTP Server using the following command line tools:

- opmnctl
- dcmctl

opmnctl

Provides a command-line utility for Oracle Process Manager and Notification Server (OPMN) for process management. It is located in

- UNIX: *ORACLE_HOME*`/opmn/bin`
- Windows: *ORACLE_HOME*`\opmn\bin`

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

dcmctl

Provides a command-line utility for **Distributed Configuration Management** (DCM) for configuration management and application deployment. It is located in

- UNIX: *ORACLE_HOME*`/dcm/bin`
- Windows: *ORACLE_HOME*`\dcm\bin`

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

Using dcmctl You must use the DCM utility `dcmctl` in circumstances such as:

- Managing clusters and farms of Oracle Application Server instances.
- Managing the configuration of individual components, such as OC4J, Oracle HTTP Server instances, and Oracle Process Manager and Notification Server, or Java Authentication and Authorization Service.
- Performing cluster-wide OC4J application deployment.

- Managing versions of configuration with archive, save and restore, and import and export functions.

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

Starting, Stopping, and Restarting Oracle HTTP Server

Oracle HTTP Server is managed by Oracle Process Manager and Notification Server (OPMN). You can use Oracle Enterprise Manager 10*g* Application Server Control Console to start, stop, and restart the server.

See Also: Chapter 7, "Application Server Control Console Management"

For command-line management, you can use the opmnctl utility to start, stop, and restart the server.

You must always use OPMN to start, stop and restart Oracle HTTP Server. Otherwise, the configuration management infrastructure cannot detect or communicate with the Oracle HTTP Server processes, and problems may occur.

Note: Do not use the `apachectl` utility to manage Oracle HTTP Server.

To determine the state of Oracle HTTP Server, use the following command:

```
opmnctl status
```

The processes are listed with their current state (Up, Down, and so on.)

Starting Oracle HTTP Server

To start Oracle HTTP Server, use the `startproc` command:

- UNIX: *ORACLE_HOME*`/opmn/bin> opmnctl [verbose] startproc ias-component=HTTP_Server`
- Windows: *ORACLE_HOME*`\opmn\bin> opmnctl [verbose] startproc ias-component=HTTP_Server`

Stopping Oracle HTTP Server

To stop Oracle HTTP Server, use the `stopproc` command:

- UNIX: *ORACLE_HOME*`/opmn/bin> opmnctl [verbose] stopproc ias-component=HTTP_Server`
- Windows: *ORACLE_HOME*`\opmn\bin> opmnctl [verbose] stopproc ias-component=HTTP_Server`

Restarting Oracle HTTP Server

Restarting Oracle HTTP Server performs a graceful restart, which is invisible to clients. In a graceful restart, on UNIX, a `USR1` signal is sent. When the process receives this signal, it tells the children to exit after processing the current request. (Children that are not servicing requests exit immediately.)

The parent re-reads the configuration files and re-opens the log files, replacing the children with new children in accordance with the settings it finds when re-reading the configuration files. It always observes the process creation settings (`MaxClients`, `MaxSpareServers`, `MinSpareServers`) specified, and takes the current server load into account.

To restart Oracle HTTP Server, use the `restartproc` command:

- UNIX: *`ORACLE_HOME`*`/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`
- Windows: *`ORACLE_HOME`*`\opmn\bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`

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

2

Concepts

This chapter introduces you to the Oracle HTTP Server directory structure, configuration files, configuration file syntax, modules, and directives.

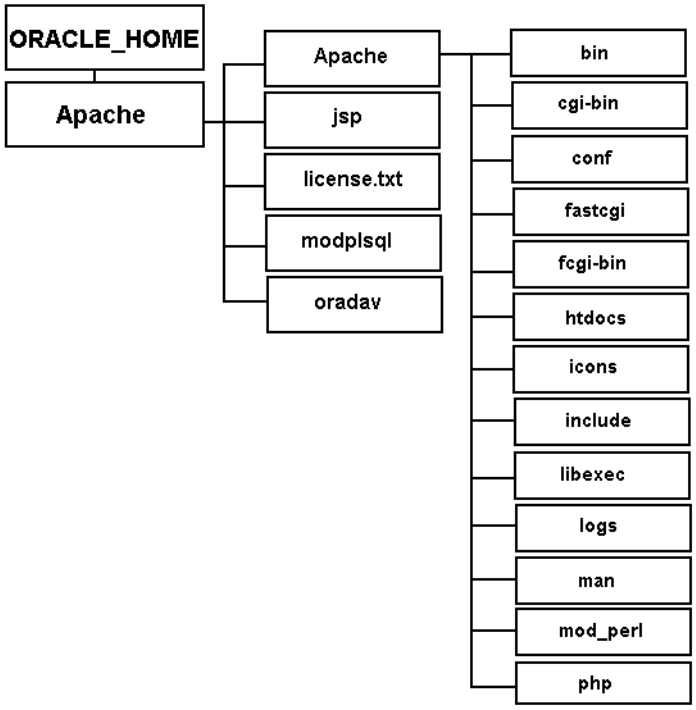
Topics discussed are:

Understanding Oracle HTTP Server Directory Structure

Oracle HTTP Server is installed in the *ORACLE_HOME*/Apache directory on UNIX or *ORACLE_HOME*\Apache directory on Windows.

Figure 2–1 illustrates Oracle HTTP Server directory structure.

Figure 2–1 Oracle HTTP Server Directory Structure



The `Apache` directory is located at the top level under the *`ORACLE_HOME`*. It contains subdirectories for configuring modules such as `mod_plsql`, and `mod_oradav`. It also contains another directory called `Apache`, which is the base directory of Oracle HTTP Server. Table 2–1 contains information about the subdirectories within the *`ORACLE_HOME`*`/Apache/Apache` directory.

Table 2–1 Apache Subdirectories

Directory Name	Contents
`bin`	Contains Oracle HTTP Server executables.
`cgi-bin`	Contains the CGI scripts. These are programs or shell scripts that can be executed by Oracle HTTP Server on the behalf of its clients.
`conf`	Contains the configuration files.
`fastcgi`	Contains the fastcgi runtime libraries, the necessary bits that you need to build your own fastcgi applications.
`fcgi-bin`	Contains FastCGI scripts.
`htdocs`	Contains the HTML scripts. The `htdoc` directory and its subdirectories are accessible to anyone on the Web, and therefore pose a severe security risk if used for anything other than publicly available data.
`icons`	Contains the icons that Oracle HTTP Server uses when displaying information or error messages.
`include`	Contains header files for building custom modules.
`libexec`	Contains the shared library files for modules.
`log`	Contains the log data, for both access and errors.
`man`	Contains the `man` page for Oracle HTTP Server

Table 2–1 (Cont.) Apache Subdirectories

Directory Name	Contents
mod_perl	Contains sample code for mod_perl. It also contains mod_perl specific libraries, and man pages.
php	Contains sample code for mod_php. On UNIX, the php directory also contains the PHP CLI (PHP executables), man pages, and headers. The headers enable you to build your own PHP extensions. On Windows, it contains the PHP CLI, and OCI extension to PHP (on UNIX, the OCI extension is compiled into mod_php).

Accessing Configuration Files

The main configuration file for Oracle HTTP Server is `httpd.conf`. This file, along with other configuration files used by the server are located in:

- UNIX: *ORACLE_HOME*`/Apache/Apache/conf`
- Windows: *ORACLE_HOME*`\Apache\Apache\conf`

Some of these files are read only once when the server starts or is reloaded, whereas some files are read every time a related file or directory is requested.

The configuration files which are read only once are called *server-wide* configuration files.

See Also: Appendix E, "Configuration Files"

Configuration Files Syntax

Directives are configuration instructions for Oracle HTTP Server. Directives are placed in `httpd.conf` and other configuration files to determine the behavior of the server.

Oracle HTTP Server configuration files should contain one directive per line. The back-slash "\" can be used as the last character on a line to indicate that the directive continues onto the next line. There must be no other characters or white space between the back-slash and the end of the line.

Directives in the configuration files are case-insensitive, but arguments to directives are often case-sensitive. Lines which begin with the character "#" are considered comments, and are ignored. Comments may not be included on a line after a configuration directive. Blank lines and white space occurring before a directive are ignored, so you may indent directives for clarity.

For example:

```
#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/private1/oracle/Apache/Apache/htdocs"

#
# Each directory to which Apache has access, can be configured with respect
# to which services and features are allowed and/or disabled in that
# directory (and its subdirectories).
#
# First, we configure the "default" to be a very restrictive set of
# permissions.
```

```
#
<Directory />
    Options FollowSymLinks MultiViews
    AllowOverride None
</Directory>
```

Classes of Directives

Table 2–2 classifies directives according to the context in which they can be used: global, per-server, and per-directory.

Table 2–2 Classes and Directives

Class	Context	Where Used
global	server configuration	Inside server configuration files, but only outside of container directives (directives such as `VirtualHost` that have a start and end directive).
per-server	server configuration, virtual host	Inside server configuration files, both outside (for the main server) and inside `VirtualHost` directives.
per-directory	server configuration, virtual host, directory	Everywhere; particularly inside the server configuration files.

Note: In Table 2–2, each class is a subset of the class preceding it. For example, directives from the per-directory class can also be used in the per-server and global contexts, and directives from the per-server class can be used in the global context.

Scope of Directives

Directives placed in the main configuration files apply to the entire server. If you wish to change the configuration for only a part of the server, you can scope your directives by placing them in specific sections.

There are two types of directives:

- Container Directives
- Block Directives

Container Directives

Container directives specify the scope within which directives take effect. The following container directives are discussed in detail in subsequent sections:

- <Directory>
- <DirectoryMatch>
- <Files>
- <FilesMatch>
- <Limit>
- <LimitExcept>
- <Location>

- <LocationMatch>
- <VirtualHost>

<Directory>

Encloses a group of directives that apply only to the named directory and subdirectories of that directory. Any directory that is allowed in a directory context may be used. The directory is either the full path to a directory, or a wildcard string. In a wildcard string, ? matches any single character, and * matches any sequence of characters. It is important to note that `<Directory />` operates on the whole file system, where as `<Directory dir>` refers to absolute directories. `<Directory>` containers cannot be nested inside each other, but can refer to directories in the document root that are nested.

<DirectoryMatch>

Specifies regular expressions, instead of using the tilde form of `<Directory>` with wildcards in the directory specification. The following two examples have the same result, matching directories starting with `web` and ending with a number from 1 to 9:

```
<Directory ~/web[1-9]/>
<DirectoryMatch "/web[1-9]/">
```

<Files>

The `<Files file>` and `</Files>` directives support access control by filename. It is comparable to the <Directory> and <Location> directives. The directives given within this section can be applied to any object within a base name (the last component of the filename) matching the specified file name. `<Files>` sections are processed in the order that they appear in the configuration file, after the `<Directory>` sections, and `.htaccess` files are read, but before `<Location>` sections. Note that the `<Files>` directives can be nested inside `<Directory>` sections to restrict the portion of the file system to which they apply.

<FilesMatch>

Provides access control by filename, just as the <Files> directive does. However, it accepts regular expressions.

<Limit>

`<Limit method>` defines a block according to the HTTP method of the incoming request. The following example limits the application of the directives that follow scripts that use the specified method:

```
<Limit POST PUT OPTIONS>
  order deny, allow
  deny from all
  allow from 127.0.0.192
</Limit>
```

Generally, `<Limit>` should not be used unless needed. It is useful only for restricting directives to particular methods. `<Limit>` is frequently used with other containers, and it is contained in any of them.

<LimitExcept>

Restricts access controls to all HTTP methods except the named ones.

<Location>

Limits the application of the directives within a block to those URLs specified, rather than to the physical file location like the <Directory> directive. <Location> sections are processed in the order that they appear in the configuration file, after the <Directory> sections, and .htaccess files are read, and after the <Files> sections. <Location> accepts wildcard directories and regular expressions with the tilde character.

<LocationMatch>

Functions in an identical manner to <Location>. You should use it for specifying regular expressions instead of the tilde form of <Location> with wildcards in the location specification.

For example:

```
<LocationMatch "/(extra|special)/data">
```

matches the URLs that contained the /extra/data or /special/data sub string.

<VirtualHost>

Oracle HTTP Server has the capabilities to serve many different Web sites simultaneously. Directives can also be scoped by placing them inside <VirtualHost> sections, so that they will only apply to requests for a particular Web site.

Virtual host refers to the practice of maintaining more than one server on one machine, as differentiated by their apparent hostname. For example, it is often desirable for companies sharing a Web server to have their own domain, and Web servers accessible, for example, www.oracle1.com and www.oracle2.com, without requiring you to know any extra path information.

Oracle HTTP Server supports both IP-based virtual hosts and name-based virtual hosts. The latter variant is sometimes also called host-based or non-IP virtual hosts.

Each virtual host can have its own name, IP address, and error and access logs. Within a <VirtualHost> container, you can set up a large number of individual servers run by a single invocation of the Oracle HTTP Server. With virtual hosting, you can specify a replacement set of the server-level configuration directives that define the main host, and are not allowed in any other container.

Block Directives

Specify a condition which must be true in order for directives within to take effect.

<IfModule> and <IfDefine> are block directives rather than container directives because they do not limit the scope of the directives they contain. They define whether Oracle HTTP Server parses the directives inside the block into its configuration, and the directives are ignored once the server is running.

Understanding Modules

Oracle HTTP Server is a modular server. Modules extend the basic functionality of the Web server, and support integration between Oracle HTTP Server and other Oracle Application Server components. Oracle HTTP Server includes Apache modules as well as Oracle HTTP Server modules.

You can add modules using the `LoadModule` directive. Here is an example of `LoadModule` usage:

```
LoadModule status_module modules/mod_status.so
```

See Also: Chapter 8, "Understanding Modules"

About .htaccess Files

Oracle HTTP Server allows for decentralized management of configuration through special files places inside the Web tree. The special files are usually called `.htaccess`, but can be specified in the `AccessFileName` directive. Directives placed in `.htaccess` files apply to the directory where you place the file, and all subdirectories. The `.htaccess` files follow the same syntax as the main configuration files. Since `.htaccess` files are read on every request, changes made in these files take immediate effect.

The server administrator further controls what directives may be placed in `.htaccess` files by configuring the `AllowOverride` directive in the main configuration files.

3

Specifying Server and File Locations

This chapter explains how to set Oracle HTTP Server and server administrator options, and specify file locations.

Topics discussed are:

- Setting Server and Administrator Functions
- Specifying File Locations

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Setting Server and Administrator Functions

The following directives set basic Oracle HTTP Server and administrator functions. They are located in the "Main Server Configuration" portion of the `httpd.conf` file.

> **See Also:** "httpd.conf File Structure" on page E-2

- ServerName
- UseCanonicalName
- ServerAdmin
- ServerSignature
- ServerTokens
- ServerAlias

ServerName

Enables the server to set a hostname that can be used to create redirection URLs, through which you can access directories without having to use a "/" at the end.

For example, `ServerName www.company.com` would be used if the main name of the actual machine was `main.company.com`.

> **See Also:** "`ServerName` directive" in the Apache Server documentation.

UseCanonicalName

Determines which hostname and port to use when redirecting the URL to the same server.

- `On`: Server uses the hostname and port values set in ServerName and Port. This is the default setting.
- `Off`: Server uses the hostname and port that you specify in the request.

For example: `UseCanonicalName On`.

> **See Also:** "`UseCanonicalName` directive" in the Apache Server documentation.

ServerAdmin

Creates an email address that is included with every default error message that clients encounter. It is useful to create a separate email address for this.

For example: `ServerAdmin you@your.emailaddress`.

> **See Also:** "`ServerAdmin` directive" in the Apache Server documentation.

ServerSignature

Enables the server to recognize which server, among the various proxies, created the returned response, such as an error message.

- `on`: Server creates a footer to the returned document that includes information such as ServerName and server version number. This is the default setting.
- `email`: Server creates an additional "mailto:" reference to the ServerAdmin of the document.
- `off`: Footer and "mailto:" reference are not created.

For example: `ServerSignature On`

> **See Also:** "`ServerSignature` directive" in the Apache Server documentation.

ServerTokens

Controls server information which is returned to clients, such as in error messages. This information includes a description of the generic operating system-type of the server, and compiled-in modules.

- `min(imal)`: provides information such as server name and version.

  For example, `Server: Apache/1.3.0`
- `OS`: provides information such as server name, version and operating system.

  For example, `Server: Apache/1.3.0 (UNIX)`
- `full`: provides information such as server name, version, operating system, and complied modules.

For example: `Server: Apache/1.3.0 (UNIX) PHP/3.0 MyMod/1.2`

> **See Also:** "`ServerTokens` directive" in the Apache Server documentation.

ServerAlias

Sets alternate names for the current virtual host.

For example:

```
<VirtualHost *>
ServerName server.domain.com
ServerAlias server server2.domain.com server2
...
</VirtualHost>
```

> **See Also:** "`ServerAlias` directive" in the Apache Server documentation.

Specifying File Locations

The following directives control the location of various server files. They are located in the "Global Environment" of the `httpd.conf` file.

> **See Also:** "httpd.conf File Structure" on page E-2

- CoreDumpDirectory
- DocumentRoot
- ErrorLog
- LockFile
- PidFile
- ScoreBoardFile
- ServerRoot

CoreDumpDirectory

Specifies the directory in which the server dumps core. The default is the ServerRoot directory. This directive is applicable to UNIX only.

For example: `CoreDumpDirectory /tmp`

> **See Also:** "`CoreDumpDirectory` directive" in the Apache Server documentation.

DocumentRoot

Sets the directory from which httpd serves files. Unless matched by a directive like `Alias`, the server appends the path from the requested URL to the document root to make the path to the document for static content.

For example: `DocumentRoot "/oracle/Apache/Apache/htdocs"`

> **See Also:** "`DocumentRoot` directive" in the Apache Server documentation.

ErrorLog

Sets the name of the file to which the server notes any errors it encounters. If the name of the file does not begin with a slash (/), then it is assumed to be relative to the

ServerRoot. If the name of the file begins with a pipe (|), then it is assumed to be a command to spawn to handle the error log.

For example: `ErrorLog "|/private1/oracle/Apache/Apache/bin/rotatelogs /private1/oracle/Apache/Apache/logs/error_log 43200"`

> **See Also:** "`ErrorLog` directive" in the Apache Server documentation.

LockFile

Sets the path to the lockfile used when Oracle HTTP Server is compiled with either `USE_FCNTL_SERIALIZED_ACCEPT` or `USE_FLOCK_SERIALIZED_ACCEPT`. It is recommended that default value be used. The main reason for changing it is if the logs directory is NFS mounted, since the lockfile must be stored on a local disk.

For example: `LockFile /oracle/Apache/Apache/logs/httpd.lock`

> **See Also:** "`LockFile` directive" in the Apache Server documentation.

PidFile

Enables you to set and change the location of the `PID` file to which the server records the process identification number. If the filename does not begin with a slash (/), then it is assumed to be relative to the ServerRoot.

For example: `PidFile /oracle/Apache/Apache/logs/httpd.lock`

> **See Also:** "`PidFile` directive" in the Apache Server documentation.

ScoreBoardFile

Required in some architectures to set a file that the server uses to communicate between the parent and children processes. To verify if your architecture requires a scoreboard file, run Oracle HTTP Server and see if it creates the file named by the directive. If your architecture requires it, then you must ensure that this file is not used at the same time by more than one invocation of the server.

For example: `/oracle/Apache/Apache/logs/httpd.scoreboard`

> **See Also:** "`ScoreBoardFile` directive" in the Apache Server documentation.

ServerRoot

Specifies the directory that contains the `conf` and `logs` subdirectories. If the server is started with the `-f` option, then you will have to specify ServerRoot.

For example: `"/oracle/Apache/Apache"`

> **See Also:** "`ServerRoot` directive" in the Apache Server documentation.

4

Managing Server Processes

This chapter provides an overview of the Oracle HTTP Server processes, and provides information on how to regulate, and monitor these processes.

Topics discussed are:

- Oracle HTTP Server Processing Model
- Handling Server Processes
- Configuring the Number of Processes and Connections
- Running Oracle HTTP Server as Root
- Security Considerations
- Getting Information about Processes

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Oracle HTTP Server Processing Model

Once Oracle HTTP Server is started, the system is ready to listen for and respond to http(s) requests. The request processing model on UNIX differs from that on Windows.

On UNIX, there is a single parent process that manages multiple child processes. The child processes are responsible for handling requests. The parent process brings up additional child processes as necessary, based on configuration. Although the server has the ability to dynamically bring up additional child processes, it is best to configure the server to start enough children initially so that requests can be handled without having to spawn more child processes.

On Windows, there is a single parent process and a single child process. The child process creates threads that are responsible for handling client requests. The number of threads created is static and can be configured.

Handling Server Processes

By default, on UNIX, the main httpd parent process and child processes are configured to run as the user who installed Oracle Application Server. The User and Group

directives are used to set the privileges for the child processes. These directives are ignored if you are not running as `root`. The child processes must be able to read all the content that will be served.

Use the following directives to manage the server processes:

- ServerType
- Group
- User

ServerType

Provides the following two options, both being applicable to UNIX only:

inetd: Starts up a new child process every time a request comes in. The program exits once the request is dealt with. This setting eliminates the option of having several child processes in waiting, making it slower and expensive, but more secure. This option should be avoided, if possible.

standalone: Enables several waiting child processes, and requires the server to be started only once. It is the default and recommended setting for a busy Web site.

You must specify the User and Group under which the servers answer requests.

For example: `ServerType standalone`

> **See Also:** "`ServerType` directive" in the Apache Server documentation.

Group

Specifies the group under which the server answers requests. Run the standalone server as `root` to use this directive. It is recommended that you create a new group for running the server. This is applicable to UNIX only.

For example: `Group myorg`

> **See Also:** "`Group` directive" in the Apache Server documentation.

User

Specifies the user ID to which the server answers requests. Run the standalone server as `root` to use this directive.You should have privileges to access files that are available for everyone, and should not be able to execute code which is not meant for httpd requests. It is recommended that you set up a new user for running the server. This is applicable to UNIX only.

For example: `User jdoe`

> **See Also:** "`User` directive" in the Apache Server documentation.

Configuring the Number of Processes and Connections

The following directives tune the performance of Oracle HTTP Server by configuring how clients requests are processed. They are located in the "Global Environment" of the `httpd.conf` file.

> **See Also:** "httpd.conf File Structure" on page E-2

- StartServers
- ThreadsPerChild
- MaxClients
- MaxRequestsPerChild
- MaxSpareServers
- MinSpareServers

StartServers

Sets the number of child server processes created when Oracle HTTP Server is started. The default is 5. This is applicable to UNIX only.

Usage: `StartServers 5`

> **See Also:** "`StartServers` directive" in the Apache Server documentation.

ThreadsPerChild

Controls the maximum number of child threads handling requests. The default is 50. This is applicable to Windows only.

Usage: `ThreadsPerChild 50`

> **See Also:** "`ThreadsPerChild` directive" in the Apache Server documentation.

MaxClients

Limits the number of requests that can be dealt with at one time. The default and recommended value is 150. This is applicable to UNIX only.

Usage: `MaxClients 150`

> **See Also:** "`MaxClients` directive" in the Apache Server documentation.

MaxRequestsPerChild

Controls the number of requests a child process handles before it dies. If you set the value to 0, which is the default, then the process will never die.

On Windows, it is recommended that this be set to 0. If it is set to a non-zero value, when the request count is reached, the child process exits, and is respawned, at which time it re-reads the configuration file. This can lead to unexpected behavior if you have modified a configuration file, but are not expecting the changes to be applied yet.

Usage: `MaxRequestsPerChild 0`

> **See Also:** "`MaxRequestsPerChild` directive" in the Apache Server documentation.

MaxSpareServers

Sets the maximum number of idle child server processes. An idle process is one which is running, but not handling a request. The parent process kills off idle child processes

that exceed the value set for this directive. The default is 20. This is applicable to UNIX only.

Usage: `MaxSpareServers 20`

> **See Also:** "`MaxSpareServers` directive" in the Apache Server documentation.

MinSpareServers

Sets the minimum number of idle child server processes. An idle process is one which is running but not handling a request. The parent process will create new children at the maximum rate of one process per second if there are fewer processes running. The default is 5. This is applicable to UNIX only.

Usage: `MinSpareServers 5`

> **See Also:** "`MinSpareServers` directive" in the Apache Server documentation.

Running Oracle HTTP Server as Root

On UNIX, if you want to run on ports less than 1024, then you will have to run as `root`.

In order to run Oracle HTTP Server as `root`, perform the following steps:

1. Stop Oracle HTTP Server using Application Server Control Console, or with the following command:

   ```
   ORACLE_HOME/opmn/bin> opmnctl [verbose] stopproc ias-component=HTTP_Server
   ```

   > **See Also:** "Starting, Restarting, and Stopping Oracle HTTP Server" on page 7-3

2. Change to `root` user.
3. Navigate to *`ORACLE_HOME`*`/Apache/Apache/bin` and execute the following command:

   ```
   chown root .apachectl
   chmod 6750 .apachectl
   ```

4. Exit `root`.
5. Restart Oracle HTTP Server using Application Server Control Console, or with the following command:

   ```
   ORACLE_HOME/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server
   ```

   > **See Also:** "Starting, Restarting, and Stopping Oracle HTTP Server" on page 7-3

Security Considerations

For additional security on UNIX, you can change the user to "nobody". Be sure that the child processes can accomplish their tasks as the user "nobody". Change all static content, such as the *`ORACLE_HOME`*`/Apache/Apache/htdocs` directory, so that all

the files are readable, but ideally not writable by the user "nobody". Also, verify that all the CGI and FastCGI programs can be run by user "nobody".

If your PL/SQL application is using the file-system caching functionality in `mod_plsql`, then the httpd processes should have read and write privileges to the cache directory through the parameter PlsqlCacheDirectory in *`ORACLE_HOME`*`/Apache/modplsql/conf/cache.conf` on UNIX or *`ORACLE_HOME`*`\Apache\modplsql\conf\cache.conf` on Windows. By default, this parameter points to *`ORACLE_HOME`*`/Apache/modplsql/cache` on UNIX or *`ORACLE_HOME`*`\Apache\modplsql\cache` on Windows.

For Oracle Application Server Portal, the content cached by `mod_plsql` is used, or updated by the Parallel Page Engine running under OC4J Portal. This means that the cache directory is readable and writable by the OC4J Portal process as well. If Oracle HTTP Server is configured to run as "nobody", then `OC4J_Portal` should also run as the same user.

Finally, given that the cached content might contain sensitive data, the final contents of the file-system cache should be protected. So, although Oracle HTTP Server might run as "nobody", access to the system as this user should be well-protected.

> **See Also:** "mod_plsql" on page 8-22

Getting Information about Processes

There are several ways to monitor Oracle HTTP Server processes.

1. Use Oracle Enterprise Manager 10*g* Application Server Control Console to monitor Oracle HTTP Server processes.

   > **See Also:** Chapter 7, "Application Server Control Console Management"

   If a network error occurs on a device such as a router or firewall between the application server and the database, JDBC connections may stop responding. In this situation, you must stop Oracle HTTP Server manually, and there may be a delay in stopping the processes.

2. Use the performance monitor on Windows, or the `ps` utility on UNIX.

   > **See Also:** *Oracle Application Server Performance Guide* and your operating system documentation for more information.

3. Use mod_status for server status. By default, it is available from localhost only.

5

Managing the Network Connections

This chapter provides information about specifying IP addresses and ports, and managing server interaction, and network connection persistence.

Topics discussed are:

- Specifying Listener Ports and Addresses
- Managing Interaction Between Server and Network
- Managing Connection Persistence
- Obtaining Client IP Address
- Configuring Reverse Proxies and Load Balancers

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Specifying Listener Ports and Addresses

The port that Oracle HTTP Server listens on when it is started depends on your installation type.

Table 5–1 contains information about Oracle HTTP Server ports.

Table 5–1 Oracle HTTP Server Ports

Platform	Middle Tier Installation	Infrastructure Installation
Solaris	Non-SSL: 7777 (7777-7877 range) SSL: 4443 (4443-4543 range)	Non-SSL: 7777 (7777-7877 range) SSL: 4443 (4443-4543 range)
Windows	Non-SSL: 80 (7777-7877 range) SSL: 443 (4443-4543 range)	Non-SSL: 7777 (7777-7877 range) SSL: 4443 (4443-4543 range)

If ports 7777 or 80, for example, are occupied, Oracle HTTP Server listens on the next available port number between a range of 7777-7877. Accordingly, it would listen on port 7778, and so on.

Note: SSL is disabled by default. For information on enabling SSL, refer to Chapter 11, "Enabling SSL for Oracle HTTP Server".

A file named `setupinfo.txt` is automatically generated in *ORACLE_HOME*`/install` on UNIX, or *ORACLE_HOME*`\install` on Windows. It contains port information for Oracle HTTP Server. This file is generated at install time, and is not updated thereafter. If you restart Oracle HTTP Server, the information in this file becomes inaccurate.

You can change the Oracle HTTP Server listener port (SSL and non-SSL) after installation. If you make a port change, then you have to also update other components to use the new port number.

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

You can specify that the server listens on more than one port, selected addresses, or a combination. The following directives, located in the "Global Environment" of the `httpd.conf` file, specify listener ports and addresses. Note that `BindAddress` and `Port` can be used only once. Apache group recommends the use of `Listen` instead.

- BindAddress
- Port
- Listen

See Also: "httpd.conf File Structure" on page E-2

BindAddress

Restricts the server to listen to a single IP address. If the argument to this directive is *, then it listens to all IP addresses. This directive has been deprecated. Listen offers similar functionality.

For example: `BindAddress *`

See Also: "`BindAddress` directive" in the Apache Server documentation.

Port

Specifies the **port** of the listener, if no Listen or BindAddress are present. If `Listen` is present, the `Port` value becomes the default port value that is used when Oracle HTTP Server builds URLs, or other references to itself. Usually, the values of `Port` and `Listen` should match, unless Oracle HTTP Server is fronted by a caching, or proxy server. Then, you can set `Port` to be the port that is being used by the front end server, and `Listen` to the port that Oracle HTTP Server is actually listening to. By doing this, redirects or other URLs generated by Oracle HTTP Server point to the front-end server rather than directly to Oracle HTTP Server.

For example: `Port 7779`

See Also: "`Port` directive" in the Apache Server documentation.

Listen

Specifies an IP port that Oracle HTTP Server listens on. Multiple `Listen` directives can be used to listen on multiple ports. If present, this value will override the value of

`Port`. Accordingly, if you have a `Port` value of 7777 and a `Listen` value of 7778, then Oracle HTTP Server only listens on one port, 7778.

For example:

- `Listen 7778`
- `Listen 12.34.56.78:80`

For platforms that support IPv4 and IPv6 (such as UNIX), setting Listen to `localhost:`*<portnumber>* and having IPv6 disabled can result in the following error:

```
..(126)Cannot assign requested address: make_sock: could not bind to address
[::1]:8000
no listening sockets available, shutting down
```

To resolve the error, either enable IPv6 or do not set Listen to `locahost`.

> **See Also:** "`Listen` directive" in the Apache Server documentation.

Managing Interaction Between Server and Network

The following directives are used to specify how the server interacts with the network. They are located in the "Global Environment" of the `httpd.conf` file.

- ListenBackLog
- SendBufferSize
- TimeOut

> **See Also:** "httpd.conf File Structure" on page E-2

ListenBackLog

Specifies the maximum length of the queue of pending connections. This is useful if the server is experiencing a `TCP SYN` overload, which causes numerous new connections that open up, but do not complete the task.

> **See Also:** "`ListenBackLog` directive" in the Apache Server documentation.

SendBufferSize

Increases the `TCP` buffer size to the number of bytes specified, thereby improving performance.

> **See Also:** "`SendBufferSize` directive" in the Apache Server documentation.

TimeOut

Sets the maximum time, in seconds, that the server waits for the following:

- The total amount of time it takes to receive a `GET` request.
- The amount of time between receipt of `TCP` packets on a `POST` or `PUT` request.
- The amount of time between `ACKs` on transmissions of `TCP` packets in responses.

The default is 300 seconds.

See Also: "`TimeOut` directive" in the Apache Server documentation.

Managing Connection Persistence

The following directives determine how the server handles persistent connections. They are located in the "Global Environment" of the `httpd.conf` file.

- KeepAlive
- KeepAliveTimeout
- MaxKeepAliveRequests

See Also:

- *Oracle Application Server Performance Guide*
- "httpd.conf File Structure" on page E-2

KeepAlive

Enables HTTP 1.1 keep-alive support, allowing reuse of the same TCP connection for multiple HTTP requests from a single client, when set to "On". The default is "On".

See Also: "`KeepAlive` directive" in the Apache Server documentation.

KeepAliveTimeout

Sets the number of seconds the server waits for a subsequent request before closing a KeepAlive connection. Once a request has been received, the timeout value specified by the TimeOut directive applies. The default is 15 seconds.

See Also: "`KeepAliveTimeout` directive" in the Apache Server documentation.

MaxKeepAliveRequests

Limits the number of requests allowed per connection when KeepAlive is on. If it is set to "0", unlimited requests will be allowed. The default is 100.

See Also: "`MaxKeepAliveRequests` directive" in the Apache Server documentation.

Obtaining Client IP Address

`UseWebCacheIp` is a global directive that enables Oracle HTTP Server to obtain IP address of a client. It can be set to "On" or "Off", and defaults to "Off". It is not set to "On" by default because it can open a security hole in some circumstances.

When OracleAS Web Cache acts as a reverse proxy in front of Oracle HTTP Server, the TCP connection from the client is terminated at OracleAS Web Cache. The TCP connection that Oracle HTTP Server sees actually originates at OracleAS Web Cache. Oracle HTTP Server gets the IP address of the client and uses it for various purposes, such as:

- Populating the `REMOTE_ADDR` CGI variable that can be used by applications in and behind Oracle HTTP Server to identify where the client came from.

- Evaluating `mod_access` allow/deny rules that allow the administrator to restrict access based on IP address.

Without the `UseWebCacheIp` directive, this functionality fails when OracleAS Web Cache is used in front of Oracle HTTP Server. This is because Oracle HTTP Server sees all connections coming from the same place - the IP address where OracleAS Web Cache is running.

With every request that OracleAS Web Cache forwards to Oracle HTTP Server, it sends a header that contains the IP address of the client connection that it received. If `UseWebCacheIp` is set to "On", then it directs Oracle HTTP Server to use the IP value from this header, instead of the value from the TCP connection as the client's IP address. This enables `REMOTE_ADDR` CGI variable to have the correct value, and allows `mod_access` to function correctly.

You should set this directive only if you are sure that the clients can only connect to Oracle HTTP Server through OracleAS Web Cache. If clients can connect directly to Oracle HTTP Server, then they have to find out the header that is used to transfer the client IP, and set it so that it would seem to have come from any IP address you want. In a typical set up, with a firewall and OracleAS Web Cache, the only port open through the firewall is the OracleAS Web Cache port. Hence, the only path from the client to Oracle HTTP Server goes through OracleAS Web Cache. In this case, it is safe to turn on `UseWebCacheIp`.

Configuring Reverse Proxies and Load Balancers

By default, Oracle Application Server installs using the local hostname as set up by ServerName directive in Oracle HTTP Server. Most Web sites tend to have a specific hostname or domain name for their Web or application server. However, this is not possible out of the box because with the `ServerName` directive, Oracle HTTP Server is instantiated with the local host.

Example 5–1 Using Reverse Proxies and Load Balancers with Oracle HTTP Server

Domain Name: `www.oracle.com:80 123.456.7.8` (hosted on a reverse proxy, load balancer, or firewall)

Host Name of Oracle Application Server Host: `server.oracle.com 123.456.7.9`

ServerName and Port of Oracle Application Server Host: `server.oracle.com:7777`

Make the following changes in the `httpd.conf` file:

```
Port 80
Listen 7777
Listen 80
# Virtual Hosts
# This section is mandatory for URLs that are generated by
# the PL/SQL packages of the Oracle Portal and various other components
# These entries dictate that the server should listen on port
# 7777, but will assert that it is using port 80, so that
# self-referential URLs generated specify www.oracle.com:80
# This will create URLs that are valid for the browser since
# the browser does not directly see the host server.oracle.com.
NameVirtualHost 123.456.7.9:7777
<VirtualHost server.oracle.com:7777>
ServerName www.oracle.com
Port 80
```

```
</VirtualHost>
# Since the previous virtual host entry will cause all links
# generated by the Oracle Portal to use port 80, the server.company.com
# server needs to listen on 80 as well since the Parallel Page
# Engine will make connection requests to Port 80 to request the
# portlets.
NameVirtualHost 123.456.7.9:80
<VirtualHost server.oracle.com:80>
ServerName www.oracle.com
Port 80
<VirtualHost>
```

See Also: *Oracle Application Server High Availability Guide*

6

Configuring and Using Server Logs

This chapter discusses Oracle Diagnostic Logging, log formats, and describes various log files and their locations.

Topics discussed are:

- Using Oracle Diagnostic Logging
- Specifying Log Level
- Specifying Log Files

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Using Oracle Diagnostic Logging

Oracle offers a new method for reporting diagnostic messages. This new method, Oracle Diagnostic Logging (ODL), presents a common format for diagnostic messages and log files, and a mechanism for correlating all diagnostic messages from various components across Oracle Application Server. Using ODL, each component logs messages to its own private local repository. A tool called `LogLoader` collects messages from each repository and loads them into a common repository where messages can be viewed as a single log stream, or analyzed in different ways.

You can view Oracle Application Server diagnostic log files using either Application Server Control Console, or a text editor.

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

ODL is further discussed in the following sections:

- Overview
- Configuring Oracle HTTP Server

Overview

Oracle HTTP Server enables you to choose the format in which you want to generate log messages. You can either continue to generate log messages in the legacy Apache

message format, or generate log messages using ODL, which complies with the new Oracle-wide standards for generating log messages.

Configuring Oracle HTTP Server

To enable Oracle HTTP Server to use ODL, enter the following directives in the `httpd.conf` file:

- OraLogMode oracle | odl | apache
- OraLogSeverity module_name <msg_type>{:msg_level]
- OraLogDir <bus stop dir>

Oracle recommends that you enter the directives before any modules are loaded (`LoadModule` directive) in the `httpd.conf` file so that module-specific logging severities are in effect before modules have the opportunity to perform any logging.

OraLogMode oracle | odl | apache

Enables you to switch between the Oracle logging format, the legacy Apache logging format, and the ODL logging format. Logging formats are defined as follows:

- **oracle**: Fully conformant, multi-line log records in XML format. Provides the most information.
- **odl**: Standard Apache log format and ECID information for log records specifically associated with a request. This is the default setting.
- **apache**: Standard Apache log format. Provides the least information.

OraLogSeverity module_name <msg_type>{:msg_level]

Enables you to set message severity. The message severity specified with this directives is interpreted as the lowest message severity that is desired, and all messages of that severity level and higher will be logged.

`OraLogSeverity` may be specified multiple times. It can be specified globally (no module_name) and once for each module for which a module-specific logging severity is desired.

This directive is only used when `OraLogMode` is set to "oracle". This directive can be used in place of the `LogLevel` directive, but is not required. If `OraLogSeverity` is present and `OraLogMode` is set to "oracle", then `LogLevel` will be ignored.

module_name This argument is the internal name of a module, as it appears in the module structure. The `<IfModule>` directive also makes use of this internal name. The module structure derives the module name from the value of the `_FILE_` macro, without path prefix, of the file which defines the module structure. If a module name is not supplied, the `OraLogSeverity` directive is applied globally.

If the module name is specified, then the directive overrides the global directive value of all the messages originating from the specified module. Specifying a module name for a module that does not get loaded generates an error.

msg_type Message types may be specified in upper or lower case, but will appear in the message output in upper case. This parameter must be of one of the following values:

- `INTERNAL_ERROR`
- `ERROR`

- WARNING
- NOTIFICATION
- TRACE

msg_level This parameter must be an integer in the range of 1-32. 1 is most severe, 32 is least severe. Using level 1 will result in fewer messages than using level 32.

Table 6–1 lists some examples of `OraLogSeverity`.

Table 6–1 Examples of OraLogSeverity

OraLogSeverity Example	Action Taken
`OraLogSeverity INTERNAL_ERROR:10`	Logs all messages of type "internal error" of levels 1-10
`OraLogSeverity WARNING:7`	Logs all messages of type "internal error" of all levels Logs all messages of type "error" of all levels Logs all messages of type "warning" of levels 1-7
`OraLogSeverity WARNING` `OraLogSeverity mod_oc4j.c NOTIFICATION:4`	If message source is `mod_oc4j`, then ■ Logs all messages of type "internal error" of all levels ■ Logs all messages of type "error" of all levels ■ Logs all messages of type "warning" of all levels ■ Logs all messages of type "notification" of levels 1-4 For messages from all other sources: ■ Logs all messages of type "internal error" of all levels ■ Logs all messages of type "error" of all levels ■ Logs all messages of type "warning" of all levels

Default If a message level is not specified, then the level defaults to the lowest severity. If the entire directive is omitted, then the value of the global Apache `LogLevel` directive is used and translated to the corresponding Oracle message type and the lowest level within the corresponding range, as listed in Table 6–2:

Table 6–2 Apache Log Level and Corresponding Oracle Message Type

Apache Log Level	Oracle Message Type
`emerg`	`INTERNAL_ERROR:16`
`alert`	`INTERNAL_ERROR:32`
`crit`	`ERROR:16`
`error`	`ERROR:32`
`warn`	`WARNING:32`
`notice`	`NOTIFICATION:16`
`info`	`NOTIFICATION:32`
`debug`	`TRACE:32`

See Also: "Specifying Log Level" on page 6-4

OraLogDir <bus stop dir>

Specifies the path to the directory which contains all log files. This directory must exist.

Default:

- UNIX: *ORACLE_HOME*/Apache/Apache/logs/oracle
- Windows: *ORACLE_HOME*\Apache\Apache\logs\oracle

Specifying Log Level

Table 6–3 lists all the different logging levels, their descriptions, and, example messages for LogLevel directive:

Table 6–3 Logging Level

Logging Level	Description	Example Message
emerg	Emergencies- system is unusable.	"Child cannot open lock file. Exiting."
alert	Action must be taken immediately.	"getpwuid: couldn't determine user name from uid"
crit	Critical conditions.	"socket: Failed to get a socket, exiting child"
error	Error conditions.	"Premature end of script headers"
warn	Warning conditions.	"child process 1234 did not exit, sending another SIGHUP"
notice	Normal but significant condition.	"httpd: caught SIGBUS, attempting to dump core in..."
info	Informational.	"Server seems busy, (you may need to increase StartServers, or Min/MaxSpareServers)..."
debug	Debug-level messages.	"Opening config file..."

Note: LogLevel directive may be omitted when OraLogMode is "oracle' and OraLogSeverity is set.

Specifying Log Files

The following log files are described in subsequent sections:

- Access Log
- CustomLog
- Error Log
- PID File
- Piped Log
- Rewrite Log
- Script Log
- SSL Log
- Transfer Log

It is important to periodically rotate the log files by moving or deleting existing logs on a moderately busy server. For this, the server must be restarted after the log files are moved or deleted so that new log files are opened.

See Also: "Log Rotation" in the Apache Server documentation.

Access Log

Records all requests processed by the server. The location and content of the access log is controlled by the CustomLog directive. The `LogFormat` directive can be used to simplify the selection of the contents of the logs.

Specifying LogFormat

`LogFormat` specifies the information included in the log file, and the manner in which it is written. The default format is the Common Log Format (CLF). The CLF format is:
`host ident authuser date request status bytes`

- `host`: This is the client domain name or its IP number.
- `ident`: If `IdentityCheck` is enabled and the client machine runs `identd`, then this is the client identity information.
- `authuser`: This is the user ID for authorized user.
- `date`: This is the date and time of the request in the `<day/month/year:hour:minute:second>` format.
- `request`: This is the request line, in double quotes, from the client.
- `status`: This is the three-digit status code returned to the client.
- `bytes`: This is the number of bytes, excluding headers, returned to the client.

See Also: "Access Log" in the Apache Server documentation.

CustomLog

Log requests to the server. A log format is specified, and the logging can optionally be made conditional on request characteristics using environment variables.

See Also: "`CustomLog` directive" in the Apache Server documentation.

Error Log

The server sends diagnostic information and records error messages to a log file located, by default, in:

- UNIX: *`ORACLE_HOME`*`/Apache/Apache/logs/error_log`
- Windows: *`ORACLE_HOME`*`\Apache\Apache\logs\error_log`

The file name can be set using the ErrorLog directive.

See Also: "ErrorLog directive" in the Apache Server documentation.

PID File

When the server is started, it notes the process ID of the parent httpd process to the PID file located, by default, in

- UNIX: *ORACLE_HOME*/Apache/Apache/logs/httpd.pid
- Windows: *ORACLE_HOME*\Apache\Apache\logs\httpd.pid

This filename can be changed with the PidFile directive. The process ID is for use by the administrator for restarting and terminating the daemon. If the process dies (or is killed) abnormally, then it is necessary to kill the children httpd processes.

See Also: "Pid File" in the Apache Server documentation.

Piped Log

Oracle HTTP Server is capable of writing error and access log files through a pipe to another process, rather than directly to file. This increases the flexibility of logging, without adding code to the main server. In order to write logs to a pipe, replace the file name with the pipe character "|", followed by the name of the executable which should accept log entries on its standard input. Oracle HTTP Server starts the piped-log process when the server starts, and restarts it if it crashes while the server is running.

Piped log processes are spawned by the parent Oracle HTTP Server httpd process, and inherit the user ID of that process. This means that piped log programs usually run as `root` so it is important to keep the programs simple and secure.

See Also: "Piped Log" in the Apache Server documentation.

Rewrite Log

Necessary for debugging when mod_rewrite is used. This log file produces a detailed analysis of how the rewriting engine transforms requests. The level of detail is controlled by the `RewriteLogLevel` directive.

See Also: "Rewrite Log" in the Apache Server documentation.

Script Log

Enables you to record the input to and output from the CGI scripts. This should only be used in testing, and not for live servers.

See Also: "Script Log" in the Apache Server documentation.

SSL Log

When Oracle HTTP Server starts in SSL mode, it creates `ssl_engine_log` and `ssl_request_log` in

- UNIX: *ORACLE_HOME*/Apache/Apache/logs
- Windows: *ORACLE_HOME*\Apache\Apache\logs

`ssl_engine_log` tracks SSL and protocol issues, where as `ssl_request_log` records user activity. Use the `SSLLogFile` directive to control output.

See Also: Chapter 11, "Enabling SSL for Oracle HTTP Server"

Transfer Log

Specifies the file in which to store the log of accesses to the site. If it is not explicitly included in the `conf` file, then no log is generated. The server typically logs each request to a transfer file located, by default, in

- UNIX:*ORACLE_HOME*/Apache/Apache/logs/access_log
- Windows: *ORACLE_HOME*\Apache\Apache\logs\access_log

The filename can be set using a `CustomLog` directive.

7

Application Server Control Console Management

This chapter provides information for managing Oracle HTTP Server using Oracle Enterprise Manager 10*g* Application Server Control Console (Application Server Control Console).

Topics discussed are:

Overview

You can manage Oracle HTTP Server in two ways: using Oracle Enterprise Manager 10*g*, or using command-line utilities such as opmnctl and dcmctl. The subsequent sections provide information on managing Oracle HTTP Server using Oracle Enterprise Manager 10*g*.

Oracle Enterprise Manager 10*g* enables you to manage Oracle HTTP Server from a Web browser using Application Server Control Console. Application Server Control Console is installed with each instance of Oracle Application Server, enabling you to administer and monitor a single Oracle Application Server instance. You can access and manage the Oracle HTTP Server from the Application Server Control Console, as described in subsequent sections.

See Also: *Oracle Enterprise Manager Concepts*

Accessing Application Server Control Console

After installation, you can access the Application Server Control Console from the URL specified in `setupinfo.txt` file. It is located in *`ORACLE_HOME`*`/install` on UNIX or *`ORACLE_HOME`*`\install` on Windows.

`setupinfo.txt` also contains the URL for Oracle Application Server Welcome page. You can access the Application Server Control Console by clicking on the "Login to Oracle Enterprise Manager 10*g*" link on the Oracle Application Server Welcome page. Enter the username, which is `ias_admin`, and the password, which is specified during the installation process, to access the Application Server Control Console.

Note: During installation, Oracle Universal Installer's "End of Installation" screen also contains the location of the Application Server Control Console and Oracle Application Server Welcome page.

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

Accessing Oracle HTTP Server Home Page

Oracle HTTP Server Home page enables you to perform tasks such as monitor the status and performance of your server, start and stop the server, create virtual hosts, modify configuration files, change log properties, manage client requests, and specify a port for a listener.

You can access the Oracle HTTP Server Home page by clicking on the "HTTP Server" link in the Name column of the "System Components" table on the Application Server Control Console.

Figure 7–1 displays the Oracle HTTP Server Home page.

Figure 7–1 Oracle HTTP Server Home Page

Managing Oracle HTTP Server

The Oracle HTTP Server Home page is divided into three sections: Home, Virtual Hosts, and Administration, where you can perform tasks such as:

- Performing Basic Administration
- Managing Virtual Hosts

- Administering Oracle HTTP Server

> **See Also:** "About the Enterprise Manager Application Server Control" in the Enterprise Manager Online Help.

Performing Basic Administration

You can perform the following basic administration tasks under the "Home" tab of the Oracle HTTP Server Home page:

- Starting, Restarting, and Stopping Oracle HTTP Server
- Managing Default Server Configuration
- Monitoring Status
- Monitoring Response and Load
- Monitoring Performance

Starting, Restarting, and Stopping Oracle HTTP Server

You can start, restart, or stop the server under the "Home" tab of Oracle HTTP Server Home page. To do so, click the appropriate button in the "General" section. You can also see the status, and the start time in this section.

Managing Default Server Configuration

You can verify the name of the server, the path of the document root, and the time the server was last modified on the "Default Server Configuration" section under the "Home" tab of the Oracle HTTP Server Home page.

> **See Also:** "Modifying the DocumentRoot, Administrator E-mail, and Group Settings" on page 7-10

Monitoring Status

You can monitor the heap usage, CPU usage, memory usage, error rate, number of active connections, and the time the connections have been open, on the "Status" section under the "Home" tab of the Oracle HTTP Server Home page.

Monitoring Response and Load

You can monitor the number of active requests, the request throughput time, request processing time, data throughput, and data processed on the "Response and Load" section under the "Home" tab of the Oracle HTTP Server Home page.

Monitoring Performance

You can view the general server status, and the response and load information under the "Home" tab on Oracle HTTP Server Home page.

Status Metrics The "Status" section provides information such as heap usage, CPU usage, memory usage, error rate, number of active connections, and time the connections have been open.

Click "Status Metrics" under the "Performance" section to view detailed status details.

Response and Load Metrics The "Response and Load" section provides information such as number of active requests, how many requests were submitted, and how long it

took for the server to respond to your request. It also provides information about how many bytes of data were processed with the requests.

Click "Response and Load Metrics" under the "Performance" section to view detailed response and load information.

Module Metrics The "Module Metrics" section enables you to view the status of the modules being used by clicking on "Module Metrics" under the "Performance" section. It provides information such as the number of active requests, number of requests processed since startup, number of current requests throughput, and the current request processing time.

Error Log You can view the last 2000 lines of the httpds error log by clicking "Error Log" under the "Performance" section.

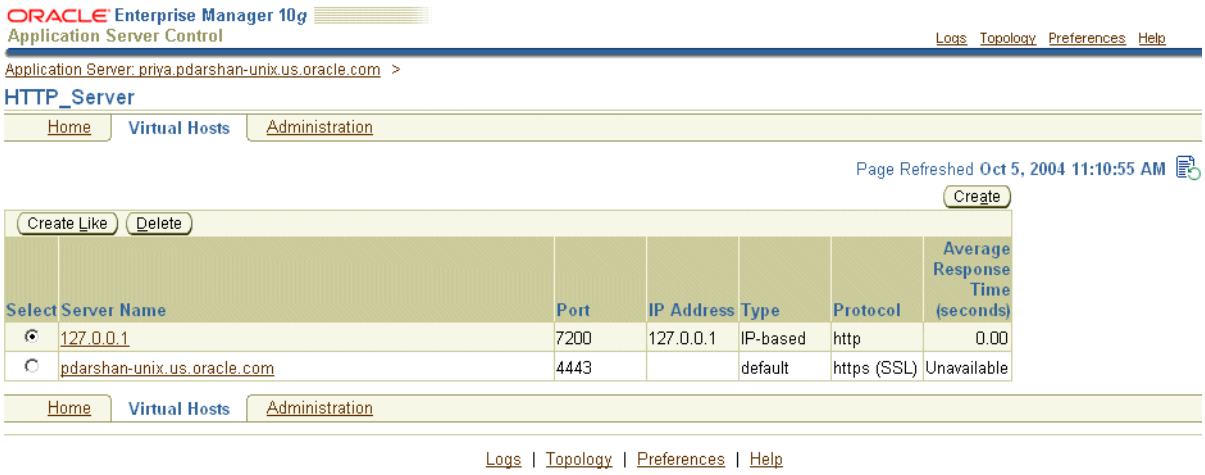
Managing Virtual Hosts

Figure 7–2 displays the Virtual Hosts page. You can view the Virtual Hosts page by clicking on the "Virtual Hosts" tab on the Oracle HTTP Server Home page. The following topics are discussed in this section:

- Requirements for Managing Virtual Hosts
- Performing Basic Tasks on Virtual Hosts Page
- Modifying Virtual Hosts
- Administering Virtual Hosts

> **See Also:** "About the Enterprise Manager Application Server Control" in the Enterprise Manager Online Help.

Figure 7–2 Virtual Hosts Page



Requirements for Managing Virtual Hosts

Virtual hosts that meet the following requirements can be managed by the Application Server Control Console:

- A `ServerName` directive is specified for each virtual host.

> **See Also:** "ServerName" on page 3-1

- Only a single <*IP listen address*>:<*port*> pair that meets the following requirements can be specified for the virtual host:
  - The IP listen address is either a numeric IP address, * for all addresses, or the keyword `_default_`.
  - The port is either a port number or * for all the ports that Oracle HTTP Server is using. Alternatively, if :<*port*> is omitted, the main server's default port will be used.
- The virtual host must be specified in a particular section of the configuration file, as follows:
  - Non-SSL virtual hosts must be specified at the top nesting level of the configuration file.
  - SSL virtual hosts must be specified just inside an `<IfDefine SSL>` directive, and that directive must be at the top nesting level of the configuration file.

See Also: "Block Directives" on page 2-6

Performing Basic Tasks on Virtual Hosts Page

You can do the following on the Virtual Hosts page:

- View settings for a virtual host.
- Create a new virtual host using the Create Virtual Host wizard. To do so, click **Create**.
- Create a new virtual host by modifying a copy of the settings of an existing virtual host. To do so, click **Create Like**. The existing virtual host is left unchanged.
- Delete a virtual host.
- Modify the settings for an existing virtual host. To do so, click the link for the host and access modification features.

Modifying Virtual Hosts

The following sections provide information on modifying or monitoring an existing virtual host. You can monitor the following for specific virtual hosts by clicking on their link on the "Virtual Hosts" page:

- Configuration
- Request Throughput
- Load
- Request Process Time

Configuration You can verify the type, the IP address, port number, protocol, and path of the document root of the virtual host, in the "Configuration" section of the virtual host page of the virtual host you selected.

Request Throughput You can monitor the number of active requests, the current throughput, the throughput since startup, and the total number of requests processed since startup, in the "Request Throughput" section of the virtual host page of the virtual host you selected.

Load You can monitor the current data throughput, the data throughput since startup, current response size, average response size since startup, and the total data since startup, in the "Load" section of the virtual host page of the virtual host you selected.

Request Process Time You can monitor the current processing time, and the average processing time since startup, in the "Request Processing Time" section of the virtual host page of the virtual host you selected.

Administering Virtual Hosts

This section contains information about administering virtual hosts. You can perform the following administrative tasks for specific virtual hosts by clicking on their link on the "Virtual Hosts" page:

- Virtual Hosts Properties
- Virtual Host MIME Languages
- Virtual Host MIME Encoding
- Virtual Host MIME Types

Virtual Hosts Properties You can view or modify the following settings on General section of the Virtual Hosts Properties page:

- **Virtual Host Type**: Displays the type of virtual host. The possible types are name-based, IP-based, or default.
- **Server Name**: Displays the server name for the virtual host.
- **Document Root**: Displays the path of the directory from which the server *serves* files. Note that the document root directory is different from the server root directory, which is only used to *store* the server files. You can specify the directory using the DocumentRoot directive.
- **Directory Index**: Specifies the resource or resources that Oracle HTTP Server will look for when the client requests the index of a directory by specifying a slash (/) at the end of the directory name.
- **Administrator Email**: Displays the server's main contact. This address receives notifications if the server experiences error conditions.
- **IP Address**: Specifies the IP address or addresses on which you want the virtual host to listen. The IP address you specify for the virtual host must already exist for Oracle HTTP Server.
- **Listening Ports**: Specifies the port or ports on which you want the virtual host to listen. Any ports you specify for the virtual host must already exist for Oracle HTTP Server. Ports for Oracle HTTP Server appear in the Listening Addresses/Ports section of the Server Properties page.

  See Also: "Specifying a Port for a Listener" on page 7-11

- **Protocol**: Displays the protocol settings.

  You can also modify the SSL Wallet path if the virtual host is using the HTTPS protocol, which uses SSL for secure connections. Note that SSL is supported for default virtual hosts and IP-based virtual hosts, but not for name-based virtual hosts.

The value of the SSL Wallet field corresponds to the `SSLWallet` entry in httpd.conf file. The path to the SSL Wallet must be in the form of a valid **Wallet Resource Locator**.

See Also: "SSLWallet" on page 11-14

- **Logging**: Provides access to the server's error log files and access log files.

  The error log file is an important source of information for maintaining a well-performing server. The error log records all of the information about problem situations so that the system administrator can easily diagnose and fix the problems.

  To provide access to the error log file--without providing access to all of the other configuration files--you may need to move the error log file to a shared directory.

  The access log file contains basic information about every HTTP transaction that the server handles. This information can be used to generate statistical reports about the server's usage patterns.

  In addition to viewing error log files and access log files in the Logging section, you can also perform these tasks for the virtual host:

  Choose a logging level for the error log file.

  See Also: "Specifying Log Level" on page 6-4

  Setting the error logging level to `Notice`, `Informational`, or `Debug` tends to flood the error log with unimportant informational messages.

  - Change the error log file name or location
  - Remove an access log file
  - Change an access log file name or location
  - Change the log format of an access log file

    See Also: "Specifying LogFormat" on page 6-5

  - Add an access log file (click Add Another Row) and specify a log format and location for it

  When you specify a location for an error log file or access log file, you can enter an absolute path and file name or a relative path and file name for the file. A relative path will be relative to the Server Root directory specified during initial configuration. The Server Root directory is displayed in the General section.

Virtual Host MIME Languages The Multipurpose Internet Mail Extension (MIME) Language setting maps the given file extensions to a particular language. This directive is used most commonly for content negotiation, where the Oracle HTTP Server returns the document that most closely matched the preferences set by the client.

To add a new MIME Language:

1. Select "Virtual Hosts MIME Languages" under the Administration section. This opens the Virtual Hosts MIME Languages page.
2. Enter the new language code in the Standard Language code field. Examples include `en` for English, `fr` for French, and `es` for Spanish.

3. Enter the types of files that should be opened with the language code in the File Extension(s) field. The extension argument is case-insensitive, and can be specified with or without a leading period. Examples include `.en`, `.fr`, and `.es`.
4. In the Default Language Code field, enter the default language type that should be used if no language is specified.
5. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
6. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

To remove a MIME language, select it and click **Remove**.

Virtual Host MIME Encoding The Multimedia Internet Mail Extension (MIME) mapping allows the Oracle HTTP Server to determine the type of file from the given extension. As part of its MIME support, Oracle HTTP Server enables you to add or remove MIME encodings. The Encoding directive maps the given filename extensions to the specified encoding type.

To add a new MIME encoding:

1. Select "Virtual Hosts MIME Encoding" under the Administration section. This opens the Virtual Hosts MIME Encoding page.
2. Click **Add Another Row**.
3. Enter the new encoding type in the Encoding field. Examples include `x-gzip`, and `x-compress`.
4. Enter the types of files that should be opened with the encoding type in the File Extension(s) field.The extension argument is case-insensitive, and can be specified with or without a leading period.
5. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
6. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

To remove a MIME encoding, select it and click **Remove**.

Virtual Host MIME Types The Multipurpose Internet Mail Extension (MIME) type maps the given filename extensions onto the specified content type. The MIME type is used for filenames containing an extension. This mapping is added to any extension already in use, overriding any mappings that already exist for the same extension.

To add a new MIME type:

1. Select "Virtual Hosts MIME Types" under the Administration section. This opens the Virtual Hosts MIME Types page.
2. Click **Add Another Row**.

3. Enter the new MIME type in the MIME type field. Examples include: text/plain, text/`.html`, and image/`.gif`.
4. Enter the types of files in the File Extension(s) field that should be opened with the MIME type. The extension argument is case-insensitive, and can be specified with or without a leading period. Examples include `.txt`, `.html`, and `.gif`.
5. In the Default MIME Type field, enter the default MIME type that should be used for unknown file types.
6. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
7. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

To remove a MIME type, select it and click **Remove**.

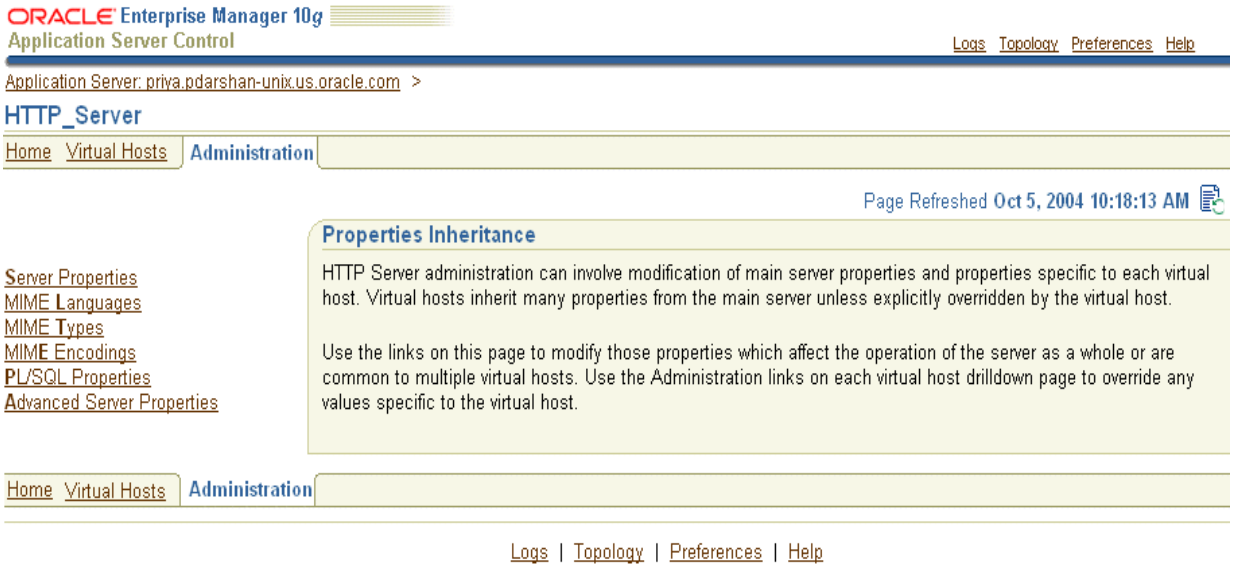
Administering Oracle HTTP Server

Figure 7–3 displays the "Administration" page. You can view the "Administration" page by clicking on the "Administration" tab on the Oracle HTTP Server Application Server Control Home page. The following topics are discussed in this section:

- Server Properties
- MIME Languages
- MIME Types
- MIME Encoding
- PL/SQL Properties
- Advanced Server Properties

See Also: "About the Enterprise Manager Application Server Control" in the Enterprise Manager Online Help.

Figure 7–3 Administration Page



Server Properties

You can view and modify the following basic settings for your Oracle HTTP Server on the "Server Properties" page.

- Modifying the DocumentRoot, Administrator E-mail, and Group Settings
- Specifying a Port for a Listener
- Changing the Error Log Properties
- Adding an Access Log File
- Changing the Access Log Properties
- Managing the Client Request and Connection Handling

Modifying the DocumentRoot, Administrator E-mail, and Group Settings After you start Oracle HTTP Server, the system is ready to listen for and respond to requests. You may need to make modifications to the document root, administrator email, User, and Group settings in order to process requests efficiently.

- **Document Root**: The directory from which the server *serves* files. Note that the document root directory is different from the server root directory, which is only used to *store* the server files. You can specify the directory using the DocumentRoot directive.
- **Administrator Email Address**: The server's main contact. This address receives notifications if the server experiences error conditions.
- **User**: Specifies the user ID to which the server answers requests. This directive is only used on UNIX systems. You should have privileges to access files that are available for everyone, and should be able to execute code which is not meant for HTTP requests. It is recommended that you set up a new user for running the server.
- **Group**: Specifies the group under which the server answers requests. This directive is only used on UNIX systems. It is recommended that you create a new group for running the server.

To modify these settings:

1. Select "Server Properties" under the "Administration" page. This opens the Oracle HTTP Server Properties page.
2. Type a new path in the "Document Root" field to change the document root directory. The path should be relative to the Server Root directory specified during initial configuration.
3. Type the appropriate email address in the "Administrator Email" field. Oracle HTTP Server uses this email address to issue notices and warnings. The administrator should have full privileges.
4. Add or change the `User` identifier by typing a new user name in the fields provided.
5. Add or change the `Group` identifier by typing a new group name in the fields provided.
6. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
7. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

Specifying a Port for a Listener When you start Oracle HTTP Server, it connects to a port and awaits client requests. Oracle HTTP Server automatically attempts to listen on port 7777.

See Also: "Specifying Listener Ports and Addresses" on page 5-1

To specify a listener port:

1. Select "Server Properties" in the "Administration" page. This opens the Server Properties page.
2. Scroll down to the Listening Addresses/Ports table.

   The first row in the Listening Addresses/Ports table identifies the default listener port. To edit the default listener port, edit the number in the Listening Port column.

   To add port settings, click **Add Another Row** to add a new row to the table. Enter the IP address and/or port number to the new row.
3. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
4. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

Changing the Error Log Properties You can change Error Log properties from the Oracle HTTP Server Home page. The Error Log file is an important source of information for

maintaining a well-performing server. The Error Log records all of the information about problem situations so that you can easily diagnose and fix the problems.

> **See Also:** "Error Log" on page 6-5

To customize the error log properties:

1. Select "Server Properties" in the "Administration" page. This opens the Server Properties page.
2. Scroll to the "Logging" section of the Server Properties page.
3. Type the full path name of the directory where you want to keep the error log file in the Error Log Filename field. You can also type the relative path name. A relative path is assumed to be relative to the Server Root directory.
4. Select the logging level from the Error Logging Level drop-down menu. The logging level indicates the severity of the error being reported.

   Note: Setting the log level to notice, info, or debug tends to flood the error log with informational messages. Use these options only if you need to perform a very detailed analysis or to debug a specific performance problem.

   > **See Also:** "Using Oracle Diagnostic Logging" on page 6-1

5. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
6. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

Adding an Access Log File You can change Access Log properties from the Oracle HTTP Server Home page. The Access Log contains basic information about every HTTP transaction that the server handles. Specifically, the access log file contains hostname, remote logname, remote user, time, request, response code, and bytes transferred. This information can be used to generate statistical reports about the server's usage patterns.

> **See Also:** "Access Log" on page 6-5

Note: At installation time, an access log with the `common LogFormat` is created.

To create an access log file:

1. Select "Server Properties" in the "Administration" page. This opens the Server Properties page.
2. Scroll to the Logging section of the Server Properties page.

3. Click **Add Another Row** in the Select Access Log table to add a new row. When the page reloads, scroll back to the Logging section.
4. Type the full path and filename of the access log file you want to create in the empty field. For example, you can type the following location:
   - UNIX: *ORACLE_HOME*/Apache/Apache/logs/access_log
   - Windows: *ORACLE_HOME*\Apache\Apache\logs\access_log

   You can enter an absolute path or a relative path. A relative path will be relative to the Server Root directory specified during initial configuration.
5. Set the log format by typing a new format name. The default is common. For information on creating custom log formats, go to "Adding an Access Log File" from the online help for the Server Properties page.

   Note: For a full description of the available log formats, click **Help** at the top of the Server Properties page.

6. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
7. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

Changing the Access Log Properties To change access log properties:

1. Select "Server Properties" in the "Administration" page. This opens the Server Properties page.
2. Scroll to the Logging section of the Server Properties page.
3. Select the Client Access Log file you want to relocate in the Select Access Log section.
4. Type the new destination in the Client Access Log Filename field. The destination can be the full path and filename, or a relative path and filename. A relative path is assumed to be relative to the Server Root directory.
5. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
6. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

Managing the Client Request and Connection Handling You can specify how the child processes on UNIX, child threads on Windows, and connections should initialize resources during the server's processing phase through the Oracle HTTP Server Home page. The Child Process and Connection settings impact the ability of the server to

process requests. You may need to modify these settings as the number of requests increases or decreases to maintain a well-performing server.

See Also: "Oracle HTTP Server Processing Model" on page 4-1

To modify child process and connection settings:

1. Select "Server Properties" in the "Administration" page. This opens the Server Properties page.
2. Scroll to the Client Request Handling or Client Connection Handling sections of the Server Properties page.
3. Modify the Client Request Handling and Client Connections Handling directives by changing the default values in the appropriate fields.

   For help on individual settings, click **Help** at the top of the Server Properties page.
4. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
5. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

MIME Languages

The Multipurpose Internet Mail Extension (MIME) Language setting maps the given file extensions to a particular language. This directive is used most commonly for content negotiation, where the Oracle HTTP Server returns the document that most closely matched the preferences set by the client.

To add a new MIME Language:

1. Select "MIME Language" in the "Administration" page. This opens the MIME Languages page.
2. Enter the new language code in the Standard Language code field. Examples include `en` for English, `fr` for French, and `es` for Spanish.
3. Enter the types of files that should be opened with the language code in the File Extension(s) field. The extension argument is case-insensitive, and can be specified with or without a leading period. Examples include `.en`, `.fr`, and `.es`.
4. In the Default Language Code field, enter the default language type that should be used if no language is specified.
5. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
6. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

To remove a MIME language, select it and click **Remove**.

MIME Types

The Multipurpose Internet Mail Extension (MIME) type maps the given filename extensions onto the specified content type. The MIME type is used for filenames containing an extension. This mapping is added to any extension already in use, overriding any mappings that already exist for the same extension.

To add a new MIME type:

1. Select "MIME Types" in the "Administration" page. This opens the MIME Types page.
2. Click **Add Another Row**.
3. Enter the new MIME type in the MIME type field. Examples include: text/plain, text/`.html`, and image/`.gif`.
4. Enter the types of files in the File Extension(s) field that should be opened with the MIME type. The extension argument is case-insensitive, and can be specified with or without a leading period. Examples include `.txt`, `.html`, and `.gif`.
5. In the Default MIME Type field, enter the default MIME type that should be used for unknown file types.
6. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
7. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

To remove a MIME type, select it and click **Remove**.

MIME Encoding

The Multimedia Internet Mail Extension (MIME) mapping allows the Oracle HTTP Server to determine the type of file from the given extension. As part of its MIME support, Oracle HTTP Server enables you to add or remove MIME encodings. The Encoding directive maps the given filename extensions to the specified encoding type.

To add a new MIME encoding:

1. Select "MIME Encoding" in the "Administration" page. This opens the MIME Encoding page.
2. Click **Add Another Row**.
3. Enter the new encoding type in the Encoding field. Examples include `x-gzip`, and `x-compress`.
4. Enter the types of files that should be opened with the encoding type in the File Extension(s) field.The extension argument is case-insensitive, and can be specified with or without a leading period.
5. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.

6. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

To remove a MIME encoding, select it and click **Remove**.

PL/SQL Properties

Oracle HTTP Server contains the mod_plsql module, which provides support for building PL/SQL-based applications on the Web. PL/SQL stored procedures retrieve data from a database, and generate HTTP responses containing data and code to display in a Web browser.

In order to use `mod_plsql`, you must install the PL/SQL Web Toolkit into a database, and create a Database Access Descriptor (DAD) which provides `mod_plsql` with connection information for the database.

See Also: "mod_plsql" on page 8-22

Creating a Database Access Descriptor (DAD) for mod_plsql You can create a DAD using Oracle HTTP Server Home page:

1. Select "PL/SQL Properties" in the "Administration" page. This opens the `mod_plsql` Services page.
2. On the `mod_plsql` Services page, scroll to the DAD Status section. Click **Create**. This opens the DAD Type page.
3. If you intend to use `mod_plsql` with Oracle Application Server Portal or Oracle Login Server, select the Portal radio button. Otherwise, select the General radio button. The subsequent screens are populated with default values based on your selection. Click **Next**. This opens the Database Connection page.
4. Type a unique name in the DAD Name field. Enter the database account, password, and connection information in the Database Connectivity Information section. In the Default page field, type the name of the PL/SQL procedure that should be invoked when one is not specified. In the NLS Language field, type the Oracle Language and Character Set for the back-end database. Choose an Authentication Mode in the Authentication Mode section. Click **Next**. This opens the Document, Alias, and Session page.
5. On the Document, Alias, and Session page, fill in the fields that are required for your DAD configuration. Click **Next**. This opens the Advanced page.
6. On the Advanced page, fill in the fields that are required for your DAD configuration. These fields are typically not configured. Refer to the online help for more information. Click **Finish**. This opens the Confirmation page. Click **OK**.
7. Restart Oracle HTTP Server.

See Also: "mod_plsql" on page 8-22

Deleting a Database Access Descriptor (DAD) for mod_plsql You can delete a DAD using the Oracle HTTP Server Home page:

1. Select "PL/SQL Properties" in the "Administration" page. This opens the `mod_plsql` Services page.
2. On the `mod_plsql` Services page, scroll to the DAD Status section. Select the radio button in the Select column for the DAD you would like to delete. Click **Delete**.
3. Restart Oracle HTTP Server.

Advanced Server Properties

You can access the Oracle HTTP Server configuration files directly on the Advanced Server Properties page. Use these files to customize the features of your server.

Editing the Server Configuration Files Perform the following steps to modify the Oracle HTTP Server configuration files:

1. Select "Advanced Server Properties" in the "Administration" section. This opens the Advanced Server Properties page.
2. Select the configuration file you want to edit. A text editor appears where you can make the appropriate changes.
3. Click **Apply** at the bottom of the page to accept the changes. If you do not click **Apply**, you will lose your changes. If you make a mistake or want to undo any changes, click **Revert**.

   Oracle Enterprise Manager 10*g* Application Server Control Console displays a confirmation page, which confirms that the appropriate configuration files have been updated.
4. Click **Yes** to restart the Oracle HTTP Server so the changes will take effect. Click **No** to restart the server later.

8

Understanding Modules

This chapter describes the modules (mods) included in the Oracle HTTP Server. The modules extend the basic functionality of the Web server, and support integration between Oracle HTTP Server and other Oracle Application Server components.

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

List of Modules

Table 8–1 lists all the Oracle HTTP Server modules discussed in this chapter.

Table 8–1 Oracle HTTP Server Modules

Oracle HTTP Server Modules			
mod_access	mod_actions	mod_alias	mod_asis
mod_auth	mod_auth_anon	mod_auth_dbm	mod_autoindex
mod_cern_meta	mod_certheaders	mod_cgi	mod_define
mod_digest	mod_dir	mod_dms	mod_env
mod_example	mod_expires	mod_fastcgi	mod_headers
mod_imap	mod_include	mod_info	mod_log_agent
mod_log_config	mod_log_referer	mod_mime	mod_mime_magic
mod_mmap_static	mod_negotiation	mod_oc4j	mod_onsint
mod_oradav	mod_ossl	mod_osso	mod_perl
mod_php	mod_plsql	mod_proxy	mod_rewrite
mod_security	mod_setenvif	mod_speling	mod_status
mod_unique_id	mod_userdir	mod_usertrack	mod_vhost_alias
mod_wchandshake			

mod_access

Controls access to the server based on characteristics of a request, such as hostname or IP address.

> **See Also:** Module `mod_access` in the Apache Server documentation.

mod_actions

Enables execution of CGI scripts based on file type or request method.

> **See Also:** Module `mod_actions` in the Apache Server documentation.

mod_alias

Enables manipulation of URLs in processing requests. It provides mapping between URLs and file system paths, and URL redirection capabilities.

> **See Also:** Module `mod_alias` in the Apache Server documentation.

mod_asis

Enables sending files that contain their own HTTP headers.

> **See Also:** Module `mod_asis` in the Apache Server documentation.

mod_auth

Enables user authentication with files based user lists.

> **See Also:** Module `mod_auth` in the Apache Server documentation.

mod_auth_anon

Enables anonymous user access to protected areas (similar to anonymous FTP, where the email addresses can be logged).

> **See Also:** Module `mod_auth_anon` in the Apache Server documentation.

mod_auth_dbm

Uses DBM files to provide user authentication.

mod_autoindex

Generates directory indexes automatically.

> **See Also:** Module `mod_autoindex` in the Apache Server documentation.

mod_cern_meta

Emulates CERN (Conseil Europeen pour le Recherche Nucleaire) HTTPD metafile semantics. Metafiles are additional HTTP headers that can be produced for each file the server accesses, in addition to the typical set.

mod_certheaders

Allows reverse proxies that terminate SSL connections in front of Oracle HTTP Server, such as OracleAS Web Cache, to transfer information regarding SSL connection, such as SSL client certificate information, to Oracle HTTP Server, and applications running behind Oracle HTTP Server. This information is transferred from the reverse proxy to Oracle HTTP Server using HTTP headers. The information is transferred from the headers to the standard CGI environment variable, which `mod_ossl` or `mod_ssl` populates if the SSL connection is terminated by Oracle HTTP Server. It is an Oracle module.

It also allows certain requests to be treated as HTTPS requests even though they are received through HTTP. This is done using the `SimulateHttps` and `AddCertHeader` directives.

`SimulateHttps` takes the container it is contained within, such as <VirtualHost>, <Location>, and so on, and treats all requests received for this container as if they were received through HTTPS, regardless of the real protocol that the request was received through.

`AddCertHeader` is specifically for use with OracleAS Web Cache. For OracleAS Web Cache, it adds a special header that indicates to Oracle HTTP Server which requests OracleAS Web Cache received through HTTPS. `mod_certheaders` triggers Oracle HTTP Server to only treat those cases where OracleAS Web Cache received the request as HTTPS as if Oracle HTTP Server received it through HTTPS.

Perform the following steps to configure `mod_certheaders`:

1. Configure Oracle HTTP Server to load `mod_certheaders`. To do this, add a `LoadModule` directive to `httpd.conf` file:
   - UNIX: `LoadModule certheaders_module libexec/mod_certheaders.so`
   - Windows: `LoadModule certheaders_module modules/ApacheModuleCertHeaders.dll`
2. Specify which headers should be translated to CGI environment variables. This can be achieved by using the `AddCertHeader` directive. This directive takes a single argument, which is the CGI environment variable that should be populated from a HTTP header on incoming requests. For example, to populate the `SSL_CLIENT_CERT` CGI environment variable, add the following lines to `httpd.conf`:

   ```
   AddCertHeader SSL_CLIENT_CERT
   ```

   The `AddCertHeader` directive can be a global setting if it is placed in the base virtual server section of `httpd.conf`. It can be specific to a single virtual host by placing it within a virtual host container, or it can be specific to a set of URIs by placing it within a <Directory> or <Location> container directive within `httpd.conf`. The combination of these directives are additive, so that for a given URI, all directives that are specific to that URI will be added to any that are specific to that request's virtual host, which will be added to any that is defined for that base virtual host.

Table 8–2 lists all the supported CGI environment variables with their corresponding HTTP header names.

Table 8–2 CGI Environment Variables with Corresponding Header Names

CGI Variable	Header Name	CGI Variable	Header Name
SSL_PROTOCOL	SSL-Protocol	SSL_SESSION_ID	SSL-Session_Id
SSL_CIPHER	SSL-Cipher	SSL_CIPHER_EXPORT	SSL-Cipher-Export
SSL_CIPHER_ ALGKEYSIZE	SSL-Cipher-Algkeysi ze	SSL_VERSION_LIBRARY	SSL-Version-Library
SSL_CLIENT_CERT	SSL-Client-Cert	SSL_VERSION_ INTERFACE	SSL-Version-Interfa ce
SSL_CLIENT_CERT_ CHAIN_n	SSL-Client-Cert-Cha in-n	SSL_CIPHER_ USEKEYSIZE	SSL-Cipher-Usekeysi ze
SSL_CLIENT_VERIFY	SSL-Client-Verify	SSL_SERVER_CERT	SSL-Server-Cert
SSL_CLIENT_M_ VERSION	SSL-Client-M-Versio n	SSL_SERVER_M_ VERSION	SSL-Server-M-Versio n
SSL_CLIENT_M_SERIAL	SSL-Client-M-Serial	SSL_SERVER_M_SERIAL	SSL-Server-M-Serial
SSL_CLIENT_V_START	SSL-Client-V-Start	SSL_SERVER_V_END	SSL-Server-V-End
SSL_CLIENT_V_END	SSL-Client-V-End	SSL_SERVER_V_END	SSL-Server-V-End
SSL_CLIENT_S_DN	SSL-Client-S-DN	SSL_SERVER_S_DN	SSL-Server-S-DN
SSL_CLIENT_S_DN_C	SSL-Client-S-DN-C	SSL_SERVER_S_DN_C	SSL-Server-S-DN-C
SSL_CLIENT_S_DN_ST	SSL-Client-S-DN-ST	SSL_SERVER_S_DN_ST	SSL-Server-S-DN-ST
SSL_CLIENT_S_DN_L	SSL-Client-S-DN-L	SSL_SERVER_S_DN_L	SSL-Server-S-DN-L
SSL_CLIENT_S_DN_O	SSL-Client-S-DN-O	SSL_SERVER_S_DN_O	SSL-Server-S-DN-O
SSL_CLIENT_S_DN_OU	SSL-Client-S-DN-OU	SSL_SERVER_S_DN_OU	SSL-Server-S-DN-OU
SSL_CLIENT_S_DN_CN	SSL-Client-S-DN-CN	SSL_SERVER_S_DN_CN	SSL-Server-S-DN-CN
SSL_CLIENT_S_DN_T	SSL-Client-S-DN-T	SSL_SERVER_S_DN_T	SSL-Server-S-DN-T
SSL_CLIENT_S_DN_I	SSL-Client-S-DN-I	SSL_SERVER_S_DN_I	SSL-Server-S-DN-I
SSL_CLIENT_S_DN_G	SSL-Client-S-DN-G	SSL_SERVER_S_DN_G	SSL-Server-S-DN-G
SSL_CLIENT_S_DN_S	SSL-Client-S-DN-S	SSL_SERVER_S_DN_S	SSL-Server-S-DN-S
SSL_CLIENT_S_DN_D	SSL-Client-S-DN-D	SSL_SERVER_S_DN_D	SSL-Server-S-DN-D
SSL_CLIENT_S_DN_UID	SSL-Client-S-DN-Uid	SSL_SERVER_S_DN_UID	SSL-Server-S-DN-Uid
SSL_CLIENT_S_DN_ Email	SSL-Client-S-DN-Ema il	SSL_SERVER_S_DN_ Email	SSL-Server-S-DN-Ema il
SSL_CLIENT_I_DN	SSL-Client-I-DN	SSL_SERVER_I_DN	SSL-Server-I-DN
SSL_CLIENT_I_DN_C	SSL-Client-I-DN-C	SSL_SERVER_I_DN_C	SSL-Server-I-DN-C
SSL_CLIENT_I_DN_ST	SSL-Client-I-DN-ST	SSL_SERVER_I_DN_ST	SSL-Server-I-DN-ST
SSL_CLIENT_I_DN_L	SSL-Client-I-DN-L	SSL_SERVER_I_DN_L	SSL-Server-I-DN-L
SSL_CLIENT_I_DN_O	SSL-Client-I-DN-O	SSL_SERVER_I_DN_O	SSL-Server-I-DN-O
SSL_CLIENT_I_DN_OU	SSL-Client-I-DN-OU	SSL_SERVER_I_DN_OU	SSL-Server-I-DN-OU
SSL_CLIENT_I_DN_CN	SSL-Client-I-DN-CN	SSL_SERVER_I_DN_CN	SSL-Server-I-DN-CN
SSL_CLIENT_I_DN_T	SSL-Client-I-DN-T	SSL_SERVER_I_DN_T	SSL-Server-I-DN-T

Table 8–2 (Cont.) CGI Environment Variables with Corresponding Header Names

CGI Variable	Header Name	CGI Variable	Header Name
SSL_CLIENT_I_DN_I	SSL-Client-I-DN-I	SSL_SERVER_I_DN_I	SSL-Server-I-DN-I
SSL_CLIENT_I_DN_G	SSL-Client-I-DN-G	SSL_SERVER_I_DN_G	SSL-Server-I-DN-G
SSL_CLIENT_I_DN_S	SSL-Client-I-DN-S	SSL_SERVER_I_DN_S	SSL-Server-I-DN-S
SSL_CLIENT_I_DN_D	SSL-Client-I-DN-D	SSL_SERVER_I_DN_D	SSL-Server-I-DN-D
SSL_CLIENT_I_DN_UID	SSL-Client-I-DN-Uid	SSL_SERVER_I_DN_UID	SSL-Server-I-DN-Uid
SSL_CLIENT_I_DN_ Email	SSL-Client-I-DN-Ema il	SSL_SERVER_I_DN_ Email	SSL-Server-I-DN-Ema il
SSL_CLIENT_A_SIG	SSL-Client-A-Sig	SSL_SERVER_A_SIG	SSL-Server-A-Sig
SSL_CLIENT_A_KEY	SSL-Client-A-Key	SSL_SERVER_A_KEY	SSL-Server-A-Key

3. `mod_certheaders` can be used to instruct Oracle HTTP Server to treat certain requests as if they were received through HTTPS even though they were received through HTTP. This is useful when Oracle HTTP Server is front-ended by a reverse proxy or load balancer, which acts as a termination point for SSL requests, and forwards the requests to Oracle HTTP Server through HTTPS.

   If OracleAS Web Cache is being used as the load balancer, it sends an HTTP header that identifies all requests it received through HTTPS. This means that `mod_certheaders` automatically detects which requests should be treated as HTTPS requests by simply looking for this header. To enable this, add the following directive to `httpd.conf`:

   ```
   AddCertHeader HTTPS
   ```

   This affects all URLs processed by Oracle HTTP Server.

   For other load balancers, `mod_certheaders` must be explicitly configured to determine which requests should be treated as HTTPS requests. To do this, use the following directive:

   ```
   SimulateHttps on
   ```

   `SimulateHttps` can be embedded within a virtual host, such as:

   ```
   <VirtualHost localhost:7777>
       SimulateHttps on
       .
       .
       .
   </VirtualHost>
   ```

   This tells `mod_certheaders` to treat every request handled by this virtual host as HTTPS, or the directive can be placed within a <LocationMatch>, <Directory>, or <DirectoryMatch> directive container such as:

   ```
   <Location /foo/>
       SimulateHttps on
   </Location>
   ```

   This limits it to URLs starting with `/foo/`.

4. Edit the $ORACLE_HOME/sso/conf/sso_apache.conf, and comment out the following line:

```
#SSLOptions +ExportCertData +StdEnvVars
```

5. Runthe following command:

```
dcmctl updateconfig -ct ohs
```

6. Run the following command:

```
opmnctl restartproc type=ohs
```

7. Test that the SSO server can be logged into with client authentication.

mod_cgi

Enables the server to run CGI scripts.

See Also: Module `mod_cgi` in the Apache Server documentation.

mod_define

Enables the `Define` directive, which defines a variable that can be expanded on any configuration line. The `Define` directive has the status `Extension`, which means that it is not compiled into the server by default.

This module requires the Extended API (EAPI). Oracle HTTP Server always has EAPI-enabled.

This module is available on UNIX systems only.

mod_digest

Uses an older version of the MD5 Digest Authentication specification than that used in `mod_auth_digest` to provide user authentication. `mod_digest` probably only works with older browsers.

See Also: Module `mod_digest` in the Apache Server documentation.

mod_dir

Enables the server to perform slash (/) redirects. Directories must contain a trailing slash. If a request for a URL without a trailing slash is received, `mod_dir` redirects the request to the same URL followed by a trailing slash. For example:

```
http://myserver/documents/mydirectory
```

is redirected to

```
http://myserver/documents/mydirectory/
```

See Also: Module `mod_dir` in the Apache Server documentation.

mod_dms

Enables you to monitor performance of site components with Oracle's Dynamic Monitoring Service (DMS). It is an Oracle module.

See Also: *Oracle Application Server Performance Guide*

mod_env

Enables you to control the environment for CGI scripts and SSI (Server Side Includes) pages by passing, setting, and unsetting environment variables.

`ModifyEnv` appends or prepends a value to an existing `ENV` variable's value, and passes it into the Oracle HTTP Server environment. The following is the usage:

Let `$FOO = "foo"`:

```
ModifyEnv FOO "bar" modifies the value of $FOO from "foo" to "foo:bar"

ModifyEnv FOO "+bar" modifies the value of $FOO from "foo" to "bar:foo"
```

Let `$FOO` be undefined:

```
Modify Foo "bar" sets the value of $FOO to "bar"
```

See Also: Module `mod_env` in the Apache Server documentation.

mod_example

Provides examples and guidance on how to write modules using the Apache API. When implemented, it demonstrates module callbacks triggered by the server.

mod_expires

Enables the server to generate Expires HTTP headers, which provide information to the client about document validity. Documents are served from the source if, based on the expiration criteria, the cached copy has expired.

See Also: Module `mod_expires` in the Apache Server documentation.

mod_fastcgi

Supports the FastCGI protocol, which enables you to maintain a pool of running servers for CGI applications, thereby eliminating start-up and initialization overhead.

See Also: Module `mod_fastcgi` in the Apache Server documentation.

mod_headers

Enables you to merge, replace, or remove HTTP response headers.

See Also: Module `mod_headers` in the Apache Server documentation.

mod_imap

Enables server-side image map processing.

mod_include

Provides a filter that processes documents for SSI (Server Side Includes) directives.

See Also: Module `mod_include` in the Apache Server documentation.

mod_info

Summarizes the entire server configuration, including all installed modules and directive settings.

See Also: Module `mod_info` in the Apache Server documentation.

mod_log_agent

Enables logging of client user agents. It is deprecated; you should use mod_log_config instead of `mod_log_agent`.

mod_log_config

Provides configurable, customizable logging of server activities. You can choose the log format, and select or exclude individual requests for logging, based on characteristics of the requests.

See Also: Module `mod_log_config` in the Apache Server documentation.

mod_log_referer

Enables logging of documents that reference documents on the server. It is deprecated; you should use mod_log_config instead of `mod_log_referer`.

See Also: Module `mod_log_referer` in the Apache Server documentation.

mod_mime

Enables the server to determine the type of a file from its filename, and associate files with handlers for processing.

See Also: Module `mod_mime` in the Apache Server documentation.

mod_mime_magic

Enables the server to determine the MIME type of a file by examining a few bytes of its content. It is used in cases when mod_mime cannot determine a file type. Make sure that `mod_mime` appears before `mod_mime_magic` in the configuration file, so that `mod_mime` processes the files first.

See Also: Module `mod_mime_magic` in the Apache Server documentation.

mod_mmap_static

Maps a list of files into memory, useful for frequently requested files that are not changed often.

mod_negotiation

Enables the server for content negotiation (selection of documents based on the client's capabilities).

> **See Also:** Module `mod_negotiation` in the Apache Server documentation.

mod_oc4j

Routes requests from the Oracle HTTP Server to Oracle Application Server Containers for J2EE (OC4J), through the AJP 1.3 protocol. It is an Oracle module.

`mod_oc4j` is enabled by default. During installation, the `oc4j_deploy_tool.jar` adds mount points to mod_oc4j.conf for applications deployed into OC4J instances. Requests that come in for specific mount points in `mod_oc4j` are routed to the OC4J instance for that mount point.

OC4J instances are started and managed by Oracle Process Manager and Notification Server (OPMN).

> **See Also:**
>
> - *Oracle Application Server Containers for J2EE User's Guide*
> - *Oracle Process Manager and Notification Server Administrator's Guide*
> - Appendix C, "Using Oracle Application Server Containers for J2EE Plug-in"

This section discusses the following topics:

- Configuring mod_oc4j
- Load Balancing Using mod_oc4j
- Enabling SSL between mod_oc4j and OC4J
- Integrating Generic Apache with Oracle Application Server

Configuring mod_oc4j

The following sections describe all relevant directives in `httpd.conf` and `mod_oc4j.conf`. Sample configurations are also provided.

mod_oc4j Configuration File and Directives

The `mod_oc4j` directives are maintained in `mod_oc4j.conf`. The `mod_oc4j.conf` file is included by default into the `httpd.conf` file, using the following directive:

```
include "ORACLE_HOME/Apache/Apache/conf/mod_oc4j.conf"
```

The following directives are used to configure `mod_oc4j`:

- Oc4jCacheSize
- Oc4jConnTimeout

- Oc4jCookieExtension
- Oc4jExtractSSL
- Oc4jEnvVar
- Oc4jMount
- Oc4jMountCopy
- Oc4jUseOHSErrors

See Also: "Using SSL Configuration Directives" on page 11-4

LoadModule

Loads the `mod_oc4j` module.

Category	Value
Syntax	`LoadModule oc4j_module` *`mod_oc4j shared library file`*
Required	Yes
Default	■ UNIX: None ■ Windows: `LoadModule oc4j_module modules\ApacheModuleOc4j.dll`
Example	■ UNIX: `LoadModule oc4j_module mod_oc4j.so` ■ Windows: `LoadModule oc4j_module modules\ApacheModuleOc4j.dll`

Oc4jCacheSize

Specifies the size of the OC4J connection cache.

Category	Value
Syntax	`Oc4jCacheSize <`*`size of connection cache`*`>`
Required	No
Default	■ UNIX: 1 ■ Windows: 32
Example	`Oc4jCacheSize 64`
Usage	Specifies the number of concurrent OC4J connections that can be cached by each Oracle HTTP Server process. Setting this directive to "0" will disable persistent connections between `mod_oc4j` and the OC4J instances.

Oc4jConnTimeout

Defines maximum idle time (in seconds) for unused connections.

Category	Value
Syntax	`Oc4jConnTimeout <`*`timeout value for AJP13 connections`*`>`
Required	No
Default	None
Example	`Oc4jConnTimeout 10`

Category	Value
Usage	Useful for cases where there is a firewall between `mod_oc4j` and OC4J that times out connections. The value should be set to a value smaller than the timeout value used by the firewall.

Oc4jCookieExtension

Directs `mod_oc4j` to use `JSESSIONID_<cookie_name_extension>` as OC4J's session identifier in the cookie.

Category	Value
Syntax	`Oc4jCookieExtension <cookie_name_extension>`
Required	No
Default	None
Example	`Oc4jCookieExtension MYEXT`
Usage	Directs `mod_oc4j` to use `JSESSIONID_<cookie_name_extension>` as OC4J's session identifier in the cookie, instead of `JSESSIONID`. In the preceding example, `JSESSIONID_MYEXT` is used as the OC4J's session identifier.

Oc4jExtractSSL

Governs passing SSL environment variables.

Category	Value
Syntax	`Oc4jExtractSSL On|Off`
Required	No
Default	`Off`
Example	`Oc4jExtractSSL On`
Usage	Directs `mod_oc4j` to decide whether or not to pass three SSL environment variables, `SSL_CLIENT_CERT`, `SSL_CIPHER`, and `SSL_SESSION_ID` to OC4J. There is a performance cost associated with copying the SSL environment variables to OC4J, so set it to "On" only if the environment variables must be available to OC4J.

Note: If configured, `mod_oc4j` passes some security environment parameters to OC4J set by `mod_ossl` and `mod_osso`, at request time.

Oc4jEnvVar

Directs `mod_oc4j` to pass some environment variables from Oracle HTTP Server to OC4J.

Category	Value
Syntax	`Oc4jEnvVar environment variable name [environment variable default value]`
Required	No
Default	None

Category	Value
Example	`Oc4jEnvVar MY_ENV1` `Oc4jEnvVar MY_ENV2 myenv_value`
Usage	For each `OC4jEnvVar` entry, you must also configure the Oracle HTTP Server directive, `PassEnv`, with the environment variable. Otherwise, `mod_oc4j` cannot acquire and pass the value. Multiple entries are allowed. You could specify the default value for the environment variable as the second parameter, or leave it empty. If the environment variable's value is found in the Oracle HTTP Server environment, its value will be passed to OC4J. Otherwise, if the default value is set, the default value will be passed. If this environment variable's value is not found in the Oracle HTTP Server environment and the default value is not set, nothing is passed to OC4J. There is a performance degradation associated with `mod_oc4j` passing some configured environment variables over to OC4J with each request.

Note: If configured, `mod_oc4j` passes some security environment parameters to OC4J set by `mod_ossl` and `mod_osso`, at request time.

Oc4jMount

Directs `mod_oc4j` to route requests containing a particular path to a destination. A destination can be a single OC4J process, or a set of OC4J instances.

Category	Value

Syntax	`Oc4jMount` *`path [destination]`*
	where path is the context root. The path parameter must be the same as the application context root specified in the OC4J configuration file `xxx-web-site.xml`. The application context root is shown in bold text in the example <web-site> element.
	<default-web-app application="default" name="defaultWebApp" root="/j2ee"/>
	and where destination is one of these types:
	■ `ajp13_dest`
	■ `cluster_dest` (this is the default destination type)
	■ `instance_dest`
	If destination is not specified, the default OC4J instance name of `home` will be used. For example,
	`Oc4jMount /myApp/*`
	provides the same result as:
	`Oc4jMount /myApp/* cluster://local_ias_cluster_name:home`

Required	No
Default	None

Category	Value

Examples	`Oc4jMount /app01/* ajp13://my-sun:8888`
	`Oc4jMount /app02/*`
	`Oc4jMount /app03/* home`
	`Oc4jMount /app04/* ias_cluster_1:home`
	`Oc4jMount /app05/* cluster://ias_cluster_1:home,ias_cluster_2:home`
	`Oc4jMount /app06/* instance://ias_instance_1:home`
	`Oc4jMount /app07/* instance://ias_instance_1:home_1,ias_instance_2:home_2`
	`Oc4jMount /app08/* instance://my-sun:ias_instance_1:home`

Usage	Examples are provided for each routing destination: **`ajp13_dest`** `Oc4jMount` path ajp13://*my-sun:8888* A request with the pattern specified in path is routed to an OC4J process listening on *my-sun*, port 8888 with the AJP 1.3 protocol. (*my-sun* and port 8888 are the AJP 1.3 protocol host and port specified in the OC4J configuration file xxx`-web-site.xml`. **`cluster_dest`** `Oc4jMount` <*path*> cluster: *`//ias_cluster_name:OC4J_instance_name, ias_ cluster_name:OC4J_instance_name...`* A request with the pattern specified in path is load balanced to one or more of the OC4J instances specified (instances are separated by commas). The Oracle Application Server Cluster Name is optional. If it is provided, the destination OC4J instance should be inside the named cluster. If none is provided, the destination OC4J instance should be inside the local Oracle Application Server cluster. **`instance_dest`** `Oc4jMount` <*path*> instance: *`//host:ias_instance_name:OC4J_instance_name, host:ias_instance_name:OC4J_instance_name...`* A request with the pattern specified in <*path*> is load balanced to one or more of the OC4J instances specified (instances are separated by commas). The host name is optional. If it is provided, the destination OC4J instance should be inside the Oracle Application Server instance residing on that host. If none is provided, the destination OC4J instance could be on any host.

Oc4jMountCopy

Copies mount points from the base server.

Category	Value
Syntax	`Oc4jMountCopy` *`On`*`|`*`Off`*
Required	No
Default	On
Example	`Oc4jMountCopy Off`

Category	Value
Usage	Directs `mod_oc4j` to decide whether to copy `Oc4jMount` points from the base server to the virtual host on which this directive is specified. If its value is On, all of the `Oc4jMount` points configured in the base server will be copied to the virtual host. If its value is `Off`, only the `Oc4jMount` points configured within the virtual host scope will be used.

Oc4jUseOHSErrors

Allows users to configure an error range using Oracle HTTP Server's error pages when errors in the range are returned from OC4J.

Category	Value
Syntax	`Oc4jUseOHSErrors` *`On|Off/min-max`*
Required	No
Default	off
Example	`Oc4jUseOHSErrors 400-410`
Usage	`Oc4jUseOHSErrors Off`: This is the default value if `Oc4jUseOHSErrors` is not specified. OC4J error pages are passed back to the client for all error values. `Oc4jUseOHSErrors on`: This returns the Oracle HTTP Server error pages for HTTP errors 400-500 inclusive. `Oc4jUseOHSErrors min-max`: This specifies the min and max for HTTP errors. For example, if you set `Oc4jUseOHSErrors 400-410`, then Oracle HTTP Server error pages for HTTP error 400-410 inclusive are returned from OC4J.

mod_oc4j Sample Configurations

This section provides some sample configurations for `mod_oc4j`.

Example 8–1 Sample mod_oc4j configuration

This configuration mounts all requests starting with the URI /servlet/ to the default instance of OC4J processes.

Make this entry in the `httpd.conf` file:

```
Oc4jMount /servlet/*
```

Example 8–2 Sample mod_oc4j configuration

This configuration performs the same work as the configuration in Example 8–1, using a `<Location>` container directive instead of the `Oc4jMount` directive.

Make this entry in the `httpd.conf` file:

```
<Location /servlet>
    SetHandler oc4j-handler
</Location>
```

Note: This will only route requests to default the OC4J instance

Example 8–3 Sample mod_oc4j configuration

This configuration mounts all requests starting with the URI `/servlet/` or `/j2ee/` and all JSP pages to the default OC4J instance of OC4J processes.

Make these entries in the `mod_oc4j.conf` file:

```
Oc4JMount /servlet/*
Oc4JMount /*.jsp
Oc4JMount /j2ee/*
```

Example 8–4 Sample mod_oc4j configuration

This configuration mounts:

- All requests starting with the URI `/applicationA/` and all JSP pages to `oc4j_instance_A`, in which all OC4J processes are managed by OPMN.
- All requests starting with the URI `/applicationB/` to `oc4j_instance_B`, in which all OC4J processes are managed by OPMN.

  Make these entries in the `mod_oc4j.conf` file:

  ```
  Oc4JMount /applicationA/* oc4j_instance_A
  Oc4JMount /applicationB/* oc4j_instance_B
  Oc4JMount /j2ee/*
  Oc4JMount /*.jsp oc4j_instance_A
  ```

Load Balancing Using mod_oc4j

`mod_oc4j` load balancing, including metric based load balancing, is discussed in detail in Appendix D, "Load Balancing Using mod_oc4j".

Enabling SSL between mod_oc4j and OC4J

Optionally, you can have direct SSL support for communication between `mod_oc4j` and OC4J. To do this, you have to enable SSL on the `mod_oc4j` side as well as the OC4J side.

- Enabling SSL for mod_oc4j
- Enabling SSL for OC4J

Enabling SSL for mod_oc4j

Add the following directives in `mod_oc4j.conf` to enable SSL for `mod_oc4j`:

Oc4jEnableSSL

Indicates whether `mod_oc4j` needs to use SSL when communicating with OC4J processes. It should not be configured to "On" if Oc4jiASPTActive is configured to "On".

Category	Value
Parameter Name	`Oc4jEnableSSL`
Parameter Type	string
Valid Values	`On/Off`
Default Value	`Off`

Oc4jSSLWalletFile

When Oc4jEnableSSL is set to "On", this directive specifies the location of an Oracle Wallet file that contains SSL certificates that are used for SSL communication with OC4J processes.

Category	Value
Parameter Name	`Oc4jSSLWalletFile`
Parameter Type	string
Valid Values	Path to a wallet directory location that contains the SSL certificate to be used when establishing SSL connections to OC4J processes.
Default Value	N/A

Oc4jSSLWalletPassword

When Oc4jEnableSSL is set to "On", this value is the obfuscated password used for authentication when opening the wallet file. This value is obtained using the `iasobf` utility.

Category	Value
Parameter Name	`Oc4jSSLWalletPassword`
Parameter Type	string
Valid Values	Obfuscated password used for authentication when opening the wallet file specified by Oc4jSSLWalletFile.
Default Value	N/A

See Also:

- *Oracle Application Server Administrator's Guide* for information on Oracle Wallet Manager.
- "Using the iasobf Utility" on page 11-15

Note: Wallet passwords have been deprecated. A warning message is generated in the Oracle HTTP Server log if this directive is used. For secure wallets, Oracle recommends that you get a SSO wallet instead.

Enabling SSL for OC4J

To enable SSL communication between `mod_oc4j` and OC4J, you have to enable SSL on the OC4J side too.

See Also: *Oracle Application Server Containers for J2EE Security Guide* for enabling SSL on the OC4J side.

Integrating Generic Apache with Oracle Application Server

You can integrate generic Apache with Oracle Application Server, 10*g* Release 2 (10.1.2). This enables you route requests from generic Apache to OC4J in the same manner as routing requests using Oracle HTTP Server and `mod_oc4j`. The generic Apache version supported is 1.3.xx, and not 2.0.

See Also: "Integrating Generic Apache with Oracle Application Server" on page C-5

mod_onsint

Provides integration support with Oracle Notification Service (ONS) and Oracle Process Manager and Notification Server (OPMN). It is an Oracle module.

Benefits of mod_onsint

`mod_onsint` provides the following functionality:

- Provides a subscription mechanism for ONS notifications within Oracle HTTP Server. This is particularly important on UNIX where Oracle HTTP Server employs a multi-process architecture. In such an architecture, it is not feasible to have an ONS subscriber in each process since there are up to 8192 processes that comprise a single Oracle HTTP Server instance. Instead, `mod_onsint` provides a single process that receives notification for all modules within an Oracle HTTP Server instance.
- Publishes `PROC_READY` ONS notifications so that other components such as OPMN and OC4J are notified that the listener is up and ready. It also provides information such as DMS metrics and information about how the listener can be contacted. These notifications are sent periodically by `mod_onsint` as long as the Oracle HTTP Server instance is running.
- Provides functionality that allows Oracle HTTP Server to terminate as a single unit if the parent process fails. The parent process is responsible for starting and stopping all of the child processes for an Oracle HTTP Server instance. The failure of the parent process without first shutting down the child processes leaves Oracle HTTP Server in an inconsistent state that can only be fixed by manually killing all of the orphaned child processes. Until this is done, a new Oracle HTTP Server instance cannot be started since the orphaned child processes still occupy the ports Oracle HTTP Server wants to use. `mod_onsint` provides a monitor of the parent process. If it detects that the parent process has died, it kills all of the remaining child processes. When combined with OPMN, this provides restartability for Oracle HTTP Server in the case of a parent process failure. `mod_onsint` ensures that all of the Oracle HTTP Server child processes die, leaving the ports open for a new Oracle HTTP Server instance. OPMN ensures that a new instance is started once the failure of the original instance is detected.

Implementation Differences on UNIX and Windows

Due to the difference in architecture of Oracle HTTP Server on UNIX and Windows, the implementation of `mod_onsint` varies slightly on these platforms.

On UNIX, `mod_onsint` spawns a process at module initialization time. This process is responsible for watching the parent process as well as sending and receiving ONS messages. Callback functions from other modules interested in ONS notifications are made in this process. For this information to be shared with other Oracle HTTP Server child processes, the use of an interprocess communication method such as a memory mapped file must be used. If a failure of a parent process is detected on UNIX, a signal is sent to all the other child processes, causing them to shut down.

On Windows, Oracle HTTP Server consists of only two processes, the parent and a multi-threaded child that handles all of the HTTP requests. In this model, `mod_onsint` runs as a thread within the child process. This thread watches the parent

process as well as sending and receiving ONS messages. Callback functions from other modules interested in ONS notifications are made in the child process. If a failure of the parent process is detected, the mod_onsint terminates the child process, effectively shutting down Oracle HTTP Server.

There is an optional directive called OpmnHostPort that can be configured for mod_onsint. This directive enables you to specify a hostname and port that OPMN should use for pinging the Oracle HTTP Server instance that mod_onsint is running in. If OpmnHostPort is not specified, mod_onsint chooses an HTTP port automatically. In certain circumstances, you may want to choose a specific HTTP port and hostname that OPMN should use to ping the listener with.

OpmnHostPort takes a single argument which is a host:port string that specifies the values to pass to OPMN. For example, the following line would specify that OPMN should use the localhost interface and port 7778 to ping this listener:

```
OpmnHostPort localhost: 7778
```

This directive must be in the global section of the httpd.conf file. It cannot be embedded into any virtual host of location container. After installation, an OpmnHostPort directive is located in dms.conf. It points OPMN to the Oracle HTTP Server "diagnostic port", which is a special localhost only virtual host. It does not log internal diagnostic requests such as OPMN pings and DMS metric requests from Application Server Control Console.

mod_oradav

This Oracle module (an OCI application written in C) is an extended implementation of mod_dav, and is integrated with the Oracle HTTP Server. mod_oradav can read and write to local files or to an Oracle database. The Oracle database must have an OraDAV driver (a stored procedure package) that mod_oradav calls to map WebDAV activity to database activity. Essentially, mod_oradav enables WebDAV clients to connect to an Oracle database, read and write content, and query and lock documents in various schemas.

You can configure mod_oradav to an Oracle database using standard Oracle HTTP Server directives. mod_oradav can immediately leverage other module code (such as mime_magic) in order to perform content management tasks. Most OraDAV processing activity involves streaming content to and from a content provider; and mod_oradav uses OCI streaming logic directly within the Oracle HTTP Server.

To configure mod_oradav, you enter parameters within a <Location> container directive in httpd.conf. The <Location> container directive specifies the DAV-enabled URL. The DAV keyword is followed by a single value: On, which tells mod_dav to use the local file system for content.

The following example specifies that the directory myfiles under the Web server documents directory (htdocs by default) is to be DAV-enabled, along with all directories under myfiles in the hierarchy. (Note that there must not be any symlinks defined on myfiles or any of its subdirectories.)

```
<Location /myfiles>
   DAV On
</Location>
```

See Also:

- Chapter 9, "Configuring and Using mod_oradav"
- *Oracle Application Server Portal Configuration Guide*

For information about using `mod_oradav` to access database schemas for access by third-party tools (such as Adobe GoLive and Macromedia Dreamweaver) and Oracle *inter*Media, refer to the OraDAV information available on OTN at

`http://www.oracle.com/technology/index.html`

mod_ossl

Enables strong cryptography for Oracle HTTP Server. This Oracle module is plug-in to Oracle HTTP Server that enables the server to use SSL. It is very similar to the OpenSSL module, `mod_ssl`. However, in contrast to the OpenSSL module, `mod_ossl` is based on the Oracle implementation of SSL, which supports SSL, version 3, and is based on Certicom and RSA Security technology.

See Also:

- *Oracle Application Server Security Guide*
- "Using mod_ossl to Authenticate Users" on page 10-7
- Chapter 11, "Enabling SSL for Oracle HTTP Server"

mod_osso

Enables **single sign-on** for Oracle HTTP Server. An Oracle module, `mod_osso` examines incoming requests and determines whether the resource requested is protected, and if so, retrieves the Oracle HTTP Server cookie for you.

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

mod_perl

Embeds the Perl interpreter into the Oracle HTTP Server. This eliminates start-up overhead and enables you to write modules in Perl. Oracle Application Server uses Perl version 5.6.1.

See Also: `mod_perl` Guide

Database Usage Notes

This section provides information for `mod_perl` users working with databases. It explains how to test a local database connection and set character forms.

Using Perl to Access the Database

The following section contains information about using Perl to access the database. Perl scripts access databases using the DBI/DBD driver for Oracle. The DBI/DBD driver is part of Oracle Application Server. It calls Oracle Callable Interface (OCI) to access the databases.

DBI must be enabled in `httpd.conf` for DBI to function. To do this, perform the following steps:

1. Edit `httpd.conf` using a text editor.
2. Search for `"PerlModule Apache::DBI"`.
3. Uncomment the line `"PerlModule Apache::DBI"`.
4. Restart Oracle HTTP Server using Application Server Control Console, or with the following command:

   ```
   ORACLE_HOME/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server
   ```

See Also: "Starting, Restarting, and Stopping Oracle HTTP Server" on page 7-3

Files must be copied to *ORACLE_HOME*`/Apache/Apache/cgi-bin`

Example 8–5 Using Perl to Access the Database

```
#!<ORACLE_HOME>/perl/bin/perl -w
  use DBI;
  my $dataSource = "host=<hostname.domain>;sid=<orclsid>;port=1521";
  my $userName = "scott";
  my $password = "tiger";
  my $dbhandle = DBI->connect("dbi:Oracle:$dataSource", $userName, $password)
    or die "Can't connect to the Oracle Database: $DBI::errstr\n";
  print "Content-type: text/plain\n\n";
  print "Database connection successful.\n";
  ### Now disconnect from the database
  $dbhandle->disconnect
    or warn "Database disconnect failed; $DBI::errstr\n";
  exit;
```

You can access the DBI scripts from the following locations:

```
http://<hostname.domain>:<port>/cgi-bin/<scriptname>
http://<hostname.domain>:<port>/perl/<scriptname>
```

If the script specifies "`use Apache::DBI`" instead of "`use DBI`", then it will only be able to run from `http://`<hostname.domain>`:`<port>`/perl/`<scriptname>.

Testing Database Connection

The following is a sample Perl script for testing the database connection of a local seed database. To use the script to test another database connection, you must replace `scott/tiger` with the user name and password for the target database.

Example 8–6 Sample Perl Script For Testing Connection for Local Seed Database

```
##### Perl script start ######
use DBI;
print "Content-type: text/plain\n\n";
$dbh = DBI->connect("dbi:Oracle:", "scott/tiger", "") || die $DBI::errstr;
 $stmt = $dbh->prepare("select * from emp order by empno")|| die $DBI::errstr;
$rc = $stmt->execute() || die $DBI::errstr;
while (($empno, $name) = $stmt->fetchrow()) { print "$empno $name\n"; }
warn $DBI::errstr if $DBI::err;
die "fetch error: " . $DBI::errstr if $DBI::err;
$stmt->finish() || die "can't close cursor";
```

```
$dbh->disconnect() || die "cant't log off Oracle";
##### Perl script End ######
```

Using SQL NCHAR Datatypes

SQL NCHAR datatypes have been refined since Oracle9*i*, and are now called reliable Unicode datatypes. SQL NCHAR datatypes such as NCHAR, NVARCHAR2 and NCLOB allow you to store any Unicode characters regardless of the database character set. The character set for those datatypes is specified by the national character set, which is either AL16UTF-16 or UTF8.

> **See Also:** Oracle9*i* documentation for more about SQL NCHAR datatypes.

This release of DBD::Oracle supports SQL NCHAR datatypes and provides driver extension functions to specify the character form for data binding. The following script shows an example to access SQL NCHAR data:

Example 8–7 Sample Script to Access SQLNCHAR Data

```
# declare to use the constants for character forms
use DBD::Oracle qw(:ora_forms);
# connect to the database and get the database handle
$dbh = DBI->connect( ... );
# prepare the statement and get the statement handle
$sth = $dbh->prepare( 'SELECT * FROM TABLE_N WHERE NCOL1 = :nchar1' );
# bind the parameter of a NCHAR type
$sth->bind_param( ':nchar1', $param_1 );
# set the character form to NCHAR
$sth->func( { ':nchar1' => ORA_NCHAR } , 'set_form' );
$sth->execute;
```

As shown in Example 8–7, the set_form function is provided as a private function that you can invoke with the standard DBI func() method. It takes an anonymous hash that specifies which placeholder should be associated with which character form. The valid values of character form are either ORA_IMPLICIT or ORA_NCHAR. Setting the character form to ORA_IMPLICIT causes the application's bound data to be converted to the database character set, and ORA_NCHAR to the national character set. The default form is ORA_IMPLICIT.

Another function is provided to specify the default character set form as follows:

```
# specify the default form to be NCHAR
$dbh->func( ORA_NCHAR, 'set_default_form' );
```

After this call is made, the form of all parameters is ORA_NCHAR, unless otherwise specified with set_form calls. Note that unlike the set_form function, this is a function on the database handle, so every statement from the database handle with its default form specified has the form of your choice by default.

set_form This function sets the character form for parameter(s). Valid forms are either ORA_IMPLICIT (default) or ORA_NCHAR. The constants are available as: ora_forms in DBD::Oracle.

Example 8–8 Sample for set_form

```
# a declaration example for the constants ORA_IMPLICIT and ORA_NCHAR
use DBD::Oracle qw(:ora_forms);
```

```
# set the character form for the placeholder :nchar1 to NCHAR
$sth->func( { ':nchar1' => ORA_NCHAR } , 'set_form' );
# set the character form using the positional index
$sth->func( { 2 => ORA_NCHAR } , 'set_form' );
# set the character form for multiple placeholders at once
$sth->func( { 1 => ORA_NCHAR, 2 => ORA_NCHAR } , 'set_form' );
```

set_default_form This function sets the default character form for a database handle.

Example 8–9 Default Character Form for a Database Handle

```
$dbh->func( ORA_NCHAR , 'set_default_form' );
```

mod_php

PHP (recursive acronym for "PHP: Hypertext Preprocessor") is an open source, widely-used, general-purpose, client-side scripting language, that is embedded in standard HTML. It is used to generate dynamic HTML pages. On Oracle HTTP Server, PHP support is provided through `mod_php` and has Oracle database support enabled. It uses PHP version 4.3.9.

Note: `phpinfo()` prints out very sensitive information about the current state of PHP and Oracle HTTP Server intervals. Users new to PHP, or those who are unaware of `phpinfo()` should not inadvertantly leave a PHP script called `phpinfo()` publically accessible.

`phpinfo()` is used heavily for debugging. There is a good chance that such a debug script could be left in the open by mistake once debugging is finished.

See Also:

- http://php.net/
- "Using PHP with Oracle HTTP Server (OHS)" document on http://www.oracle.com/technology/tech/opensource/index.html if you want to build from the source, or need more information.

mod_plsql

Connects Oracle HTTP Server to an Oracle database, enabling you to create Web applications using Oracle stored procedures. It is an Oracle module.

In order to access a Web-enabled PL/SQL application, configure a PL/SQL Database Access Descriptor (DAD) for `mod_plsql`. A DAD is a set of values that specifies how `mod_plsql` connects to a database server to fulfill an HTTP request. Besides the connect details, a DAD contains important configuration parameters for various operations in the database and for `mod_plsql` in general. Any Web-enabled PL/SQL application which makes use of the PL/SQL Web ToolKit needs to create a DAD to invoke the application.

- Any PL/SQL Application written using the PL/SQL Web ToolKit
- Oracle Application Server Portal

Creating a DAD

If mod_plsql is part of Oracle Application Server, it is recommended that you use Application Server Control Console to create a DAD.

> **See Also:** "Creating a Database Access Descriptor (DAD) for mod_plsql" on page 7-16

If not, then perform the following steps to create a DAD:

1. Edit the DAD configuration file *ORACLE_HOME*/Apache/modplsql/conf/dads.conf.
2. Add a DAD where the DAD has the following format:
   a. The Oracle HTTP Server <Location> directive which defines a virtual path used to access the PL/SQL Web Application. This directive begins enclosing a group of directives that apply to the named Location.

      For example, the directive <Location /myapp> defines a virtual path called "/myapp" that will be used to invoke a PL/SQL Web Application through a URL like http://host:port/myapp/.

      Note: Older versions of mod_plsql were always mounted on a virtual path with a prefix of '/pls'. This restriction is removed in newer versions but might still be a restriction imposed by some of the older PL/SQL applications.

      ---

   b. The Oracle HTTP Server "SetHandler" directive which directs Oracle HTTP Server to enable mod_plsql to handle the request for the virtual path defined by the named Location

      ```
      SetHandler pls_handler
      ```

   c. Additional Oracle HTTP Server directives that are allowed in the context of a <Location> directive. Typically, the following directives are used:

      ```
      Order deny,allow
      Allow from all
      AllowOverride None
      ```

   d. One or more mod_plsql specific directives. For example:

      ```
      PlsqlDatabaseUsername        scott
      PlsqlDatabasePassword        tiger
      PlsqlDatabaseConnectString   orcl
      PlsqlAuthenticationMode      Basic
      ```

   e. An Oracle HTTP Server </Location> directive which closes the group of directives for the named Location, and defines a single DAD.
3. Save the edits.
4. Obfuscate the DAD password by running the "dadTool.pl" script located in *ORACLE_HOME*/Apache/modplsql/conf.

> **See Also:** "PlsqlDatabasePassword" on page 8-36 for instructions on performing the obfuscation.

5. If `mod_plsql` is part of Oracle Application Server, then issue the following command:

   ```
   $ORACLE_HOME/dcm/bin/dcmctl updateConfig -ct ohs
   ```

6. Restart Oracle HTTP Server using Application Server Control Console, or with the following command:

   ```
   ORACLE_HOME/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server
   ```

See Also: "Starting, Restarting, and Stopping Oracle HTTP Server" on page 7-3

You can create additional DADs by defining other uniquely named `Locations` in `dads.conf`.

Configuration Files

`mod_plsql` configuration parameters reside in the following three configuration files:

- plsql.conf
- dads.conf
- cache.conf

plsql.conf

This file contains the `LoadModule` directive to load `mod_plsql` into Oracle HTTP Server, any global settings for `mod_plsql`, and include directives for `dads.conf` and `cache.conf`. This file is included by the Oracle HTTP Server configuration file *ORACLE_HOME*`/Apache/Apache/conf/oracle_apache.conf` on UNIX or *ORACLE_HOME*`\Apache\Apache\conf\oracle_apache.conf` on Windows, which itself gets included in the primary Oracle HTTP Server configuration file `httpd.conf`.

See Also: "oracle_apache.conf" on page E-4

dads.conf

This file contains the configuration parameters for the **PL/SQL database access descriptor** (DAD). A DAD is a set of values that specifies how `mod_plsql` connects to a database server to fulfill a HTTP request.

cache.conf

This file contains the configuration settings for the file system caching functionality implemented in `mod_plsql`. This configuration file is relevant only if PL/SQL applications use the `OWA_CACHE` package to cache dynamically generated content in the file system.

See Also: *Oracle Application Server mod_plsql User's Guide* for details on caching functionality in `mod_plsql`.

Configuration Parameters

Table 8–3 contains a list of `mod_plsql` configuration parameters. They are discussed in detail in later sections.

While specifying a value for a configuration parameter, follow Oracle HTTP Server conventions for specifying values. For instance, if a value has white spaces in it, enclose the value with double quotes. For example: `PlsqlNLSLanguage "TRADITIONAL CHINESE_TAIWAN.UTF8"`

Multi-line directives enable you to specify same directive multiple times in a DAD.

Table 8–3 mod_plsql Configuration Files and Parameters

Configuration File	Parameters
plsql.conf	PlsqlDMSEnable PlsqlLogEnable PlsqlLogDirectory PlsqlIdleSessionCleanupInterval
dads.conf	PlsqlAfterProcedure PlsqlAlwaysDescribeProcedure PlsqlAuthenticationMode PlsqlBeforeProcedure PlsqlBindBucketLengths PlsqlBindBucketWidths PlsqlCGIEnvironmentList PlsqlCompatibilityMode PlsqlConnectionTimeout PlsqlConnectionValidation PlsqlDatabaseConnectString PlsqlDatabasePassword PlsqlDatabaseUserName PlsqlDefaultPage PlsqlDocumentPath PlsqlDocumentProcedure PlsqlDocumentTablename PlsqlErrorStyle PlsqlExclusionList PlsqlFetchBufferSize PlsqlInfoLogging PlsqlMaxRequestsPerSession PlsqlNLSLanguage PlsqlPathAlias PlsqlPathAliasProcedure PlsqlRequestValidationFunction PlsqlSessionCookieName PlsqlSessionStateManagement PlsqlTransferMode PlsqlUploadAsLongRaw

Table 8–3 (Cont.) mod_plsql Configuration Files and Parameters

Configuration File	Parameters
cache.conf	PlsqlCacheCleanupTime PlsqlCacheDirectory PlsqlCacheEnable PlsqlCacheMaxAge PlsqlCacheMaxSize PlsqlCacheTotalSize

plsql.conf

This file contains the `LoadModule` directive to load `mod_plsql` into the Oracle HTTP Server, global settings for `mod_plsql`, and include directives for `dads.conf` and `cache.conf`.

Note: Refer to `plsql.README` located in *`ORACLE_HOME`*`/Apache/modplsql/conf` for detailed description of `plsql.conf`.

The following section discusses the parameters that can be specified in `plsql.conf`:

- PlsqlDMSEnable
- PlsqlLogEnable
- PlsqlLogDirectory
- PlsqlIdleSessionCleanupInterval

PlsqlDMSEnable

Enables Dynamic Monitoring Service (DMS) for `mod_plsql`.

Category	Value
Syntax	`PlsqlDMSEnable` *`On/Off`*
Default	On
Example	`PlsqlDMSEnable On`

PlsqlLogEnable

Enables debug level logging for `mod_plsql`.

Debug level logging is meant to be used for debugging purposes only. When logging is enabled, log files are generated at:

- UNIX: *`ORACLE_HOME`*`/Apache/modplsql/logs`
- Windows: *`ORACLE_HOME`*`\Apache\modplsql\logs`

as configured by PlsqlLogDirectory. This parameter should be set to "Off" unless recommended by Oracle support to debug problems with `mod_plsql`.

To view more details about the internal processing of `mod_plsql`, set this directive to "On". This causes `mod_plsql` to start logging for every request that is processed. The log files are generated as specified by the PlsqlLogDirectory directive.

Category	Value
Syntax	`PlsqlLogEnable` *`On/Off`*
Default	Off
Example	PlsqlLogEnable `Off`

PlsqlLogDirectory

Specifies the directory where debug level logs are written out.

Set the directory name of the location where log files should be generated when logging is enabled. To avoid possible confusion about the location of this directory, an absolute path is recommended.

On UNIX, this directory must have write permissions by the owner of the child httpd processes.

Category	Value
Syntax	`PlsqlLogDirectory` *`directory`*
Default	None
Example	`PlsqlLogDirectory` *`ORACLE_HOME`*`/Apache/modplsql/logs`

PlsqlIdleSessionCleanupInterval

Specifies the time (in minutes) in which the idle database sessions should be closed and cleaned by `mod_plsql`.

This directive is used in conjunction with connection pooling of database connections and sessions in `mod_plsql`. When a session is not used for the specified amount of time, it is closed, and freed. This is done so that unused sessions can be cleaned, and the memory is freed on the database side.

Setting this time to a low number helps in faster cleanup of unused database sessions. Be aware that if this number is too low, then this may adversely affect the performance benefits of connection pooling in `mod_plsql`.

If the number of open database sessions is not a concern, you can increase the value of this parameter for best performance. In such a case, if the site is accessed frequently enough that the idle session cleanup interval is never reached for a session, then the DAD configuration parameter PlsqlMaxRequestsPerSession can be modified so that it is guaranteed that a pooled database session gets recycled on a regular basis.

For most installations, the default parameter value should suffice.

Category	Value
Syntax	`PlsqlIdleSessionCleanupInterval` *number*
Default	15 (minutes)
Example	`PlsqlIdleSessionCleanupInterval` 15

dads.conf

This file contains the configuration parameters for the PL/SQL Database Access Descriptor (DAD).

DAD Parameters

This section describes all the DAD level parameters that can be specified in the `dads.conf` file. Besides these directives, you can also specify additional Oracle HTTP Server directives that can be specified in the context of a `<Location>` directive, such as:

```
Order deny,allow
AllowOverride None
```

The following parameters are discussed in detail in the subsequent sections:

- PlsqlAfterProcedure
- PlsqlAlwaysDescribeProcedure
- PlsqlAuthenticationMode
- PlsqlBeforeProcedure
- PlsqlBindBucketLengths
- PlsqlBindBucketWidths
- PlsqlCGIEnvironmentList
- PlsqlCompatibilityMode
- PlsqlConnectionTimeout
- PlsqlConnectionValidation
- PlsqlDatabaseConnectString
- PlsqlDatabasePassword
- PlsqlDatabaseUserName
- PlsqlDefaultPage
- PlsqlDocumentPath
- PlsqlDocumentProcedure
- PlsqlDocumentTablename
- PlsqlErrorStyle
- PlsqlExclusionList
- PlsqlFetchBufferSize
- PlsqlInfoLogging
- PlsqlMaxRequestsPerSession
- PlsqlNLSLanguage
- PlsqlPathAlias
- PlsqlPathAliasProcedure
- PlsqlRequestValidationFunction
- PlsqlSessionCookieName
- PlsqlSessionStateManagement
- PlsqlTransferMode
- PlsqlUploadAsLongRaw

PlsqlAfterProcedure

Specifies the procedure to be invoked after calling the requested procedure. This enables you to put a hook point after the requested procedure is called. This is useful in doing SQL*Traces/SQL Profiles while debugging a problem with the requested procedure. This is also useful when you want to ensure that a specific call be made after running every procedure.

Category	Value
Syntax	`PlsqlAfterProcedure` *`string`*
Default	None
Example	`PlsqlAfterProcedure portal.mypkg.myafterproc`

- For all purposes, except for debugging, this parameter should be omitted. You could use this parameter to stop SQL Trace/SQL Profiling.
- In older versions of the product, this parameter was called `after_proc`.

PlsqlAlwaysDescribeProcedure

Specifies whether `mod_plsql` should describe a procedure before trying to execute it. If this is set to "On", then `mod_plsql` will always describe a procedure before invoking it. Otherwise, `mod_plsql` will only describe a procedure when its internal heuristics have interpreted a parameter type incorrectly.

Category	Value
Syntax	`PlsqlAlwaysDescribeProcedure` *`On/Off`*
Default	Off
Example	`PlsqlAlwaysDescribeProcedure Off`

- For all purposes, except for debugging, you should leave this parameter set to "Off".
- In older versions of the product, this parameter was called `always_desc`.

PlsqlAuthenticationMode

Specifies the authentication mode to use for allow access through this DAD.

Category	Value
Syntax	`PlsqlAuthenticationMode` *`Basic/SingleSignOn/GlobalOwa/CustomOwa/PerPackageOwa`*
Default	Basic
Example	`PlsqlAuthenticationMode Basic`

- Most customer applications use Basic Authentication. Custom Authentication modes (`GlobalOwa`, `CustomOwa`, `PerPackageOwa`) are used by very few PL/SQL applications. The `SingleSignOn` mode is supported only for Oracle Application Server releases, and is used by Oracle Application Server Portal and Oracle Application Server Single Sign-On.
- If the DAD is not using the `Basic` authentication, then you must include a valid username/password in the DAD configuration. For the `Basic` mode, if you wish

to perform dynamic authentication, the DAD username/password parameters must be omitted.

- In older versions of the product, this configuration parameter was derived from a combination of `enablesso` and `custom_auth`.
  - `enablesso = Yes` translates to `PlsqlAuthenticationMode SingleSignOn`
  - `custom_auth = Global` translates to `PlsqlAuthenticationMode GlobalOwa`
  - `custom_auth = Custom` translates to `PlsqlAuthenticationMode CustomOwa`
  - `custom_auth = PerPackage` translates to `PlsqlAuthenticationMode PerPackageOwa`

  All other combinations translate to `Basic`.

  See Also: "Securing Application Database Access through `mod_plsql`" chapter in the *Oracle Application Server mod_plsql User's Guide* for more information regarding different authentication modes.

PlsqlBeforeProcedure

Specifies the procedure to be invoked before calling the requested procedure. This enables you to put a hook point before the requested procedure is called. This is useful in doing SQL*Traces/SQL Profiles while debugging a problem with the requested procedure. This is also useful when you want to ensure that a specific call be made before running every procedure.

Category	Value
Syntax	`PlsqlBeforeProcedure` *`string`*
Default	None
Example	`PlsqlBeforeProcedure portal.mypkg.mybeforeproc`

- For all purposes, except for debugging purposes, this parameter should be omitted. You could use this parameter to start SQL Trace/SQL Profiling.
- In older versions of the product, this parameter was called `before_proc`.

PlsqlBindBucketLengths

Specifies the rounding size to use while binding the number of elements in a collection bind. While executing PL/SQL statements, the Oracle database maintains a cache of PL/SQL statements in the shared SQL area, and attempts to reuse the cached statement if the same statement is executed again. Oracle's matching criteria requires that the statement texts be identical, and that the bind variable data types match. Unfortunately, the type match for strings is sensitive to the exact byte size specified, and for collection bindings is also sensitive to the number of elements in the collection. Since `mod_plsql` binds statements dynamically, the odds of hitting the shared cache are low, and it may fill up with near-duplicates and lead to contention for the latch on the shared area. This parameter reduces that effect by bucketing bind lengths to the nearest level.

All numbers specified should be in ascending order. After the last specified size, subsequent bucket sizes will be assumed to be twice the last one.

Category	Value
Syntax	`PlsqlBindBucketLengths` *`number multiline`*
Default	4,20,100,400

Example	`PlsqlBindBucketLengths 4`
	`PlsqlBindBucketLengths 25`
	`PlsqlBindBucketLengths 125`

- This parameter is relevant only if you are using procedures with array parameters, and passing varying number of parameters to the procedure.
- The default should be sufficient for most PL/SQL applications.
- To see if this parameter needs to be changed, check the number of versions of a SQL statement in the SQL area.
- Consider using flexible parameter passing to reduce the problem.
- In older versions of the product, this parameter was called `bind_bucket_lengths`.

PlsqlBindBucketWidths

Specifies the rounding size to use while binding the number of elements in a collection bind. While executing PL/SQL statements, the Oracle database maintains a cache of PL/SQL statements in the shared SQL area, and attempts to reuse the cached statement if the same statement is executed again. Oracle's matching criteria requires that the statement texts be identical, and that the bind variable data types match. Unfortunately, the type match for strings is sensitive to the exact byte size specified, and for collection bindings is also sensitive to the number of elements in the collection. Since `mod_plsql` binds statements dynamically, the odds of hitting the shared cache are low, and it may fill up with near-duplicates and lead to contention for the latch on the shared area. This parameter reduces that effect by bucketing bind widths to the nearest level.

All numbers specified should be in ascending order. After the last specified size, subsequent bucket sizes will be assumed to be twice the last one.

The last bucket width must be equal to or less than 4000. This is due to the restriction imposed by OCI where array bind widths cannot be greater than 4000.

Category	Value
Syntax	`PlsqlBindBucketWidths` *`number multiline`*
Default	`32,128,1450,2048,4000`

Example	`PlsqlBindBucketWidths 40`
	`PlsqlBindBucketWidths 400`
	`PlsqlBindBucketWidths 2000`

- This parameter is relevant only of you are using procedures with array parameters, and passing varying number of parameters to the procedure.
- The default should be sufficient for most PL/SQL applications.
- To see if this parameter needs to be changed, check the number of versions of a SQL statement in the SQL area.

- Consider using flexible parameter passing to reduce the problem.
- In older versions of the product, this parameter was called `bind_bucket_widths`.

PlsqlCGIEnvironmentList

Specifies overrides and/or additions of CGI environment variables to the default set of environment variables passed down to a PL/SQL procedure. This is a multi-line directive of name-value pairs to be added, overridden or removed. You can only specify one environment variable for each directive.

You can add CGI environment variables from the Oracle HTTP Server environment by specifying the variable name. To remove a CGI environment variable, set it equal to nothing. To add your own name-value pair, use the syntax `myname=myvalue`.

Category	Value
Syntax	`PlsqlCGIEnvironmentList string` *`multiline`*
Default	None

Example	■ To add a new environment variable from the Oracle HTTP Server environment:
	`PlsqlCGIEnvironmentList DOCUMENT_ROOT`
	■ To remove an environment variable:
	`PlsqlCGIEnvironmentList MYENVAR2=`
	■ To override from the Oracle HTTP Server environment:
	`PlsqlCGIEnvironmentList REQUEST_PROTOCOL=HTTPS`
	■ To add your own environment variable:
	`PlsqlCGIEnvironmentList MY_VARNAME=MY_VALUE`

- Environment variables added here are available in the PL/SQL application through the function `owa_util.get_cgi_env`.
- In older versions of the product, this parameter was called `cgi_env_list`.

PlsqlCompatibilityMode

Specifies the compatibility mode for running `mod_plsql`. This parameter is supported only for Oracle Application Server releases, and is used when you are using `mod_plsql` with an older version of Oracle Application Server Portal. In such situations, if you are running `mod_plsql` against a pre-9.0.2 version of Oracle Application Server Portal, this should be set to 1.

Category	Value
Syntax	`PlsqlCompatibilityMode` *`BitFlag`*
Default	0
Example	`PlsqlCompatibilityMode 1`

This parameter enables an old bug in `mod_plsql` in which `mod_plsql` incorrectly converted the plus symbol (+) to space characters for document downloads. Enabling the first bit in this flag will make it impossible to download documents that have a plus symbol `(+)` in the document name.

PlsqlConnectionTimeout

Specifies the timeout in milliseconds for testing a connection pooled in `mod_plsql`.

When PlsqlConnectionValidation is set to "Automatic" or "AlwaysValidate", `mod_plsql` will attempt to test pooled database connections. This parameter specifies the maximum time `mod_plsql` should wait for the test request to complete before it assumes that the connection is not usable.

Category	Value
Syntax	`PlsqlConnectionTimeout` *`5000`*
Default	10000
Example	`PlsqlConnectionTimeout` *`5000`*

PlsqlConnectionValidation

Specifies the mechanism `mod_plsql` should use to detect terminated connections in its connection pool.

For performance reasons, `mod_plsql` pools database connections. If a database instance goes down, and `mod_plsql` was maintaining a pool of connections to the instance, then each pooled database connection results in an error when it is next used to service a request. This can be a concern in high availability configurations like RAC where even if one node goes down, other nodes servicing the database might have been able to service the request successfully. `mod_plsql` provides for a mechanism whereby it can self-correct after it detects a failure that could be caused by a database node going down. This mechanism to self-correct is controlled by the parameter `PlsqlConnectionValidation`.

The following are the valid values for `PlsqlConnectionValidation`:

- **Automatic**: `mod_plsql` tests all pooled database connections which were created prior to the detection of a failure that could mean an instance failure.
- **ThrowAwayOnFailure**: `mod_plsql` throws away all pooled database connections which were created prior to the detection of a failure that could mean an instance failure.
- **AlwaysValidate**: `mod_plsql` always tests all pooled database connections which were created prior to issuing a request. Since this option has an associated performance overhead for each request, this should be used with caution.
- **NeverValidate**: `mod_plsql` never pings any pooled database connection. This option always for older behavior in `mod_plsql`.

Category	Value
Syntax	`PlsqlConnectionValidation` *`Automatic/ThrowAwayOnFailure/AlwaysValidate/NeverValidate`*
Default	`Automatic`
Example	`PlsqlConnectionValidation ThrowAwayOnFailure`

When `mod_plsql` encounters one of the following errors, it assumes that the database might have been down.

- `00443, 00000, "background process did not start"`
- `00444, 00000, "background process failed while starting"`

- 00445, 00000, "background process did not start after x seconds"
- 00447, 00000, "fatal error in background processes"
- 00448, 00000, "normal completion of background process"
- 00449, 00000, "background process unexpectedly terminated with error"
- 00470, 00000, "LGWR process terminated with error"
- 00471, 00000, "DBWR process terminated with error"
- 00472, 00000, "PMON process terminated with error"
- 00473, 00000, "ARCH process terminated with error"
- 00474, 00000, "SMON process terminated with error"
- 00475, 00000, "TRWR process terminated with error"
- 00476, 00000, "RECO process terminated with error"
- 00480, 00000, "LCK* process terminated with error"
- 00481, 00000, "LMON process terminated with error"
- 00482, 00000, "LMD* process terminated with error"
- 00484, 00000, "LMS* process terminated with error"
- 00485, 00000, "DIAG process terminated with error"
- 01014, 00000, "ORACLE shutdown in progress"
- 01033, 00000, "ORACLE initialization or shutdown in progress"
- 01034, 00000, "ORACLE not available"
- 01041, 00000, "internal error. hostdef extension doesn't exist"
- 01077, 00000, "background process initialization failure"
- 01089, 00000, "immediate shutdown in progress- no operations permitted"
- 01090, 00000, "shutdown in progress- connection is not permitted"
- 01091, 00000, "failure during startup force"
- 01092, 00000, "ORACLE instance terminated. Disconnection forced"
- 03106, 00000, "fatal two-task communication protocol error"
- 03113, 00000, "end-of-file on communication channel"
- 03114, 00000, "not connected to ORACLE"
- 12570, 00000, "TNS: packet writer failure"
- 12571, 00000, "TNS: packet writer failure"

PlsqlDatabaseConnectString

Specifies the connection to an Oracle database.

Category	Value
Syntax	PlsqlDatabaseConnectString *stringServiceNameFormat/SIDFormat/TNSFormat/NetServiceNameFormat*, where string can be one of the following based on the second argument: ■ ServiceNameFormat: HOST:PORT:SERVICE_NAME format where HOST is the hostname running the database, PORT is the port number the TNS listener is listening on, SERVICE_NAME is the database service name. ■ SIDFormat: HOST:PORT:SID format where HOST is the hostname running the database, PORT is the port number the TNS listener is listening on, SID is the database SID. ■ TNSFormat: A valid TNS alias which resolves using Net8 utilities like tnsping and SQL*Plus. ■ NetServiceNameFormat: A valid net service name which resolves to a connect descriptor. A connect descriptor is a specially formatted description of the destination for a network connection. A connect descriptor contains destination service and network route information. If the format argument is not specified, then mod_plsql assumes that "string" is either in the HOST:PORT:SID format, or resolvable by Net8. The differentiation between the two is made by the presence of the colon in the specified string. It is recommended that newer DADs do not use the SIDFormat syntax. This exists only for backward compatibility reasons. Use the new two argument format for newly created DADs.
Default	None
Example	■ PlsqlDatabaseConnectString *myhost.com*:1521:*myhost*.iasdb.inst ServiceNameFormat ■ PlsqlDatabaseConnectString *myhost.com*:1521:iasdb SIDFormat ■ PlsqlDatabaseConnectString *myhost_tns* TNSFormat ■ PlsqlDatabaseConnectString cn=oracle,cn=iasdb NetServiceNameFormat ■ PlsqlDatabaseConnectString (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=*myhost.com*)(Port= 1521))(CONNECT_DATA=(SID=iasdb))) TNSFormat ■ PlsqlDatabaseConnectString *myhost_tns* ■ PlsqlDatabaseConnectString *myhost.com*:1521:iasdb

- If the database is running in the same Oracle home, or the environment variable "TWO_TASK" is set, this parameter need not be specified.
- If the database is running in a separate Oracle home, then this parameter is mandatory.
- If you have problems connecting to the database:
  - Check the username and password information in the DAD.
  - Make sure that you run "tnsping <string>" and execute commands such as:

    sqlplus *DADUsername*/*DADPassword*@<*string*>

  - Ensure that TNS_ADMIN is configured properly.
  - Verify that the HOST:PORT:SERVICE_NAME format makes the connection go through.
  - Ensure that the TNS listener and database are up and running.

  - Ensure that you can ping the host from this machine.
- From a mod_plsql perspective, TNSFormat and NetServiceNameFormat are synonymous and denote connect descriptors that are resolved by Net. The TNSFormat is provided as a convenience so that end-users use this to signify that the name resolution happens through the local tnsnames.ora. For situations where the resolution is through an LDAP lookup as configured in sqlnet.ora, it is recommended that the format specifier of NetServiceNameFormat be used.

  If your database supports high availability, for example, RAC database, it is highly recommended that you use the NetServiceNameFormat such that the resolution for the net service name is through LDAP. This enables you to add or remove RAC nodes accessible through mod_plsql by just changing Oracle Internet Directory with the new/deleted node information. In such situations, hard-coding database listener HOST:PORT information in dads.conf or in the local tnsnames.ora is not recommended.
- In older versions of the product, this configuration parameter was called connect_string.

PlsqlDatabasePassword

Specifies the password to use to log in to the database.

Category	Value
Syntax	PlsqlDatabasePassword *string*
Default	None
Example	PlsqlDatabasePassword tiger

After making manual configuration changes to DAD passwords, it is recommended that the DAD passwords are obfuscated by running the "dadTool.pl" script located in *ORACLE_HOME*/Apache/modplsql/conf.

The following are the steps to obfuscate DAD passwords:

1. If necessary, switch user to the Oracle software owner user, typically oracle using the following command:

   ```
   $su - oracle
   ```

2. Set the *ORACLE_HOME* environment variable to specify the path to the Oracle home directory for the current release and set the PATH environment variable to include the directory containing the Perl executable and the location of the dadTool.pl script.

   On Bourne, Bash, or Korn Shell:

   ```
   ORACLE_HOME=new_ORACLE_HOME_path;export ORACLE_HOME
   PATH=ORACLE_HOME/Apache/modplsql/conf:ORACLE_HOME/perl/bin:PATH;export PATH
   ```

   On C or tcsh Shell:

   ```
   setenv ORACLE_HOME new_ORACLE_HOME_PATH
   setenv PATH ORACLE_HOME/Apache/modplsql/conf:ORACLE_HOME/perl/bin:PATH
   ```

   On Windows:

   ```
   set PATH=ORACLE_HOME\Apache\modplsql\conf;ORACLE_
   HOME\perl\5.6.1\bin\MSWin32-x86;%PATH%
   ```

Note: The preceding command for Windows should be issued in one line.

3. Set the appropriate shared library path environment variable for your platform.
   - On UNIX platforms, include the *ORACLE_HOME*/lib directory in your shared library path. Table 8–4 shows the appropriate environment variable for each platform.

Table 8–4 Platform Type and Corresponding Shared Library Path Environment Variable

Platform	Environment Variable
AIX	LIBPATH
HP-UX	SHLIB_PATH
Linux, Solaris, and Tru64 UNIX	LD_LIBRARY_PATH

   For example, to set the SHLIB_PATH environment in the Bourne shell on HP-UX systems, enter the following command:

```
$SHLIB_PATH=$ORACLE_HOME/lib:$SHLIB_PATH;export SHLIB_PATH
```

   - On Windows, include $*ORACLE_HOME*/bin in your PATH, for example:

```
set PATH=%ORACLE_HOME%\bin;%PATH%
```

4. Change directory to the mod_plsql configuration directory for the current release of Oracle HTTP Server:

```
cd $ORACLE_HOME/Apache/modplsql/conf
```

5. Invoke the following Perl script to obfuscate DAD password:

```
perl dadTool.pl -o
```

Notes:

- This is a mandatory parameter, except for a DAD that sets PlsqlAuthenticationMode to Basic and uses dynamic authentication.
- For DADs using SingleSignOn authentication, this parameter is the name of the schema owner.
- In older versions of the product, this configuration parameter was called password.

PlsqlDatabaseUserName

Specifies the username to use to logon to the database.

Category	Value
Syntax	PlsqlDatabaseUsername *string*
Default	None
Example	PlsqlDatabaseUsername scott

- This is a mandatory parameter, except for a DAD that sets PlsqlAuthenticationMode to Basic and uses dynamic authentication.

- For DADs using `SingleSignOn` authentication, this parameter is the name of the schema owner.
- In older versions of the product, this configuration parameter was called `username`.

PlsqlDefaultPage

Specifies the default procedure to call if none is specified in the URL.

Category	Value
Syntax	`PlsqlDefaultPage` *`string`*
Default	None
Example	`PlsqlDefaultPage myschema.mypackage.home`

- You can also use Oracle HTTP Server Rewrite rules to achieve the same effect as you get by setting this configuration parameter.
- In older versions of the product, this parameter was called `default_page`.

PlsqlDocumentPath

Specifies a virtual path in the URL that initiates document download form the document table. For example, if this parameter is set to `docs`, then the following URLs will start the document downloading process for URLs of the format:

```
/pls/dad/docs
/pls/plsqlapp/docs
```

Category	Value
Syntax	`PlsqlDocumentPath` *`string`*
Default	`docs`
Example	`PlsqlDocumentPath docs`

- Omit this parameter for applications that do not perform document uploads or downloads.

  See Also: *Oracle Application Server mod_plsql User's Guide*

- In older versions of the product, this parameter was called `document_path`.

PlsqlDocumentProcedure

Specifies the procedure to call when a document download is initiated. This procedure is called to process the download.

Category	Value
Syntax	`PlsqlDocumentProcedure` *`string`*
Default	None
Example	`PlsqlDocumentProcedure portal.wwdoc_process.process_download`

- Omit this parameter for applications that do not perform document uploads or downloads.

  See Also: *Oracle Application Server mod_plsql User's Guide*

- In older versions of the product, this parameter was called `document_proc`.

PlsqlDocumentTablename

Specifies the table in the database to which all documents are uploaded.

Category	Value
Syntax	`PlsqlDocumentTablename` *`string`*
Default	None
Example	`PlsqlDocumentTablename myschema.document_table`

- Omit this parameter for applications that do not perform document uploads or downloads.

  See Also: *Oracle Application Server mod_plsql User's Guide*

- In older versions of the product, this parameter was called `document_table`.

PlsqlErrorStyle

Specifies the Error Reporting Mode for `mod_plsql` errors. This parameter accepts the following values:

- **ApacheStyle**: This is the default mode. In this mode, `mod_plsql` indicates to Oracle HTTP Server the HTTP error that was encountered. Oracle HTTP Server then generates the error page. This can be used with the Oracle HTTP Server `ErrorDocument` directive to produce customized error messages.
- **ModplsqlStyle**: `mod_plsql` generates the error pages, usually a short message indicating the PL/SQL error that was encountered and PL/SQL exception stack, if any. For example:

  ```
  scott.foo PROCEDURE NOT FOUND
  ```

- **DebugStyle**: This mode provides more details than `ModplsqlStyle`. `mod_plsql` provides more details about the URL, parameters and also produces server configuration information. This mode is for debugging purposes only. Do not use this in a production system, since displaying internal server variables could be a security risk.

Category	Value
Syntax	`PlsqlErrorStyle ApacheStyle/ModplsqlStyle/DebugStyle`
Default	`ApacheStyle`
Example	`PlsqlErrorStyle ModplsqlStyle`

In older versions of the product, this parameter was called `error_style`.

PlsqlExclusionList

Specifies a pattern for procedures, packages, or schema names which are forbidden to be directly executed from a browser. This is a multi-line directive in which each pattern is on one line. The pattern is case-insensitive and can accept a wildcard such as '*'. The default patterns disallowed from direct URL access are: `sys.*`, `dbms_*`, `utl_*`, `owa_*`, `owa.*`, `htp.*`, `htf.*`, `wpg_docload.*`.

Setting this directive to "`#NONE#`" will disable all protection. This is not recommended for a live site and should not be done (This is sometimes used for debugging purposes).

If this parameter is overridden, the defaults still apply, which means that you do not have to explicitly add the default list to the list of excluded patterns.

Category	Value
Syntax	`PlsqlExclusionList [string/"#NONE#" multiline]`

Default	`sys.*`
	`dbms_*`
	`utl_*`
	`owa_*`
	`owa.*`
	`htp.*`
	`htf.*`
	`wpg_docload.*`

Example	`PlsqlExclusionList myschema.private1.*`
	`PlsqlExclusionList myschema.private.*`
	will disallow access to URLs which contain one of:
	`sys.*, dbms_*, utl_*, owa_*, owa.*, htp.*, htf.*, wpg_docload.*, myschema.private.*, myschema.private1.*`
	`PlsqlExclusionList "#NONE#"` will disable all protection. Again, this is not recommended for live sites as this could be a security concern.

- Besides the patterns specified with this parameter, `mod_plsql` also disallows any procedure name which contains special characters like tabs, newlines, carriage-returns, single-quotes, the reverse slash, the form feed, the open parenthesis, close parenthesis, and space. This cannot be changed.
- In older versions of the product, this parameter was called `exclusion_list`.

See Also: *Oracle Application Server mod_plsql User's Guide*

PlsqlFetchBufferSize

Specifies the number of rows of content to fetch from the database for each trip, using either `owa_util.get_page` or `owa_util.get_page_raw`.

By default, `mod_plsql` attempts to fetch 200 response lines of output where each line is of 255 bytes. In situations where the response bytes are single-bytes, the response buffer is populated to the maximum and can pack 255*200=51000 bytes for each round trip. However, for responses containing multi-byte data, the byte packing for each row could be less than ideal resulting in lesser bytes getting transferred for each round trip. If your application generates large pages frequently and the response does not fit in

one round trip, then consider setting this parameter higher. However, the memory usage for `mod_plsql` will increase.

Category	Value
Syntax	PlsqlFetchBufferSize *number*
Default	200
Example	`PlsqlFetchBufferSize` 256

- This parameter is changed only for performance reasons. The minimum value for this parameter is 28, but it is seldom reduced.
- Change this parameter only under the following circumstances:
  - The average response page is large and you want to reduce the number of round-trips `mod_plsql` makes to the database to fetch the response.
  - The character set in use is multi-byte, and you want to compensate for the problem of `get_page` or `get_page_raw` fetching fewer bytes for each row (calculations in the PL/SQL Web ToolKit are character-based and in the case of multi-byte characters, OWA packages assume a worst-case character byte size and do not attempt to pack each row to its maximum).
- In older versions of the product, this parameter was called `response_array_size`.
- In older versions of the product, the default for this parameter was 128.

PlsqlInfoLogging

Specifies what mode `mod_plsql` should use to do extra performance logging.

The mode is:

InfoDebug: This logs more information to the Apache's `error_log`. This is used in conjunction with Apache's "info" logging level. If the Apache's logging level is not at least set to this high, this setting will be ignored.

Category	Value
Syntax	`PlsqlInfoLogging` *InfoDebug*
Default	Empty
Example	`PlsqlInfoLogging InfoDebug`

This logging setting is useful for debugging problems in your PL/SQL application.

PlsqlMaxRequestsPerSession

Specifies the maximum number of requests a pooled database connection should service before it is closed and re-opened.

Category	Value
Syntax	`PlsqlMaxRequestsPerSession` *number*
Default	1000
Example	`PlsqlMaxRequestsPerSession 1000`

- This parameter helps relieve memory and resource problems that may occur due to prolonged session reuse by a PL/SQL application.
- This parameter should not need to be changed; the default is sufficient in most cases.
- Setting this parameter to a low number can degrade performance. A case for a lower value might be an infrequently used DAD whose performance is not a concern, and for which limiting the number of requests provides some benefit.
- In older versions of the product, the equivalent to this parameter is `reuse`. Instead of taking a value of "Yes" or "No", the new parameter enables you to have finer control over the connection pool reuse in `mod_plsql`.

PlsqlNLSLanguage

Specifies the `NLS_LANG` variable for this DAD. This parameter overrides the `NLS_LANG` environment variable. When this parameter is set, the PL/SQL Gateway uses the specified `NLS_LANG` to connect to the database. Once connected, an alter session command is issued to switch to the specified language and territory. If the middle tier character set matches that of the database, then no alter session call is issued by `mod_plsql`.

Category	Value
Syntax	`PlsqlNLSLanguage` *`string`*
Default	None
Example	`PlsqlNLSLanguage America_America.UTF8`

- Most applications have PlsqlTransferMode set to `CHAR` which means that the character set in PlsqlNLSLanguage needs to match the character set of the database. In one special case, where the database and `mod_plsql` are both using fixed-size character sets, and the character set width matches, the character set can be different. The response character set is always the `mod_plsql` character set.
- If `PlsqlTransferMode` is set to `RAW`, then this parameter can be ignored.
- In older versions of the product, this parameter was called `nls_lang`.

PlsqlPathAlias

Specifies a virtual path alias to map to a procedure call. This is application specific.

Category	Value
Syntax	`PlsqlPathAlias` *`string`*
Default	None
Example	`PlsqlPathAlias url`

- For applications that do not use path aliasing, this parameter may be omitted.

  See Also: *Oracle Application Server mod_plsql User's Guide*

- In older versions of the product, this parameter was called `pathalias`.

PlsqlPathAliasProcedure

Specifies the procedure to call when the virtual path in the URL matches the path alias as configured by PlsqlPathAlias.

Category	Value
Syntax	`PlsqlPathAliasProcedure` *string*
Default	None
Example	`PlsqlPathAliasProcedure portal.wwpth_api_alias.process_download`

- For applications that do not use path aliasing, this parameter may be omitted.

  See Also: *Oracle Application Server mod_plsql User's Guide*

- In older versions of the product, this parameter was called `pathaliasproc`.

PlsqlRequestValidationFunction

Specifies an application-defined PL/SQL function which gives you the opportunity to allow/disallow further processing of the requested procedure. This is useful in implementing tight security for your PL/SQL application by blocking out package/procedure calls which should not be allowed to execute from this DAD.

The function defined by this parameter must have the following prototype:

boolean `function_name` (`procedure_name` IN varchar 2)

Upon invocation, the argument '`procedure_name`' will contain the name of the procedure that the request is trying to execute.

For example, if all the PL/SQL application procedures callable from a browser are inside the package "`mypkg`", then a simple implementation of this function can be as follows:

```
boolean my_validation_check (procedure_name varchar 2
is
begin
  if (upper (procedure_name) like upper ('myschema.mypkg%')) then
    return TRUE
  else
    return FALSE
  end if;
end;
```

Category	Value
Syntax	`PlsqlRequestValidationFunction [string]`
Default	none
Example	`PlsqlRequestValidationFunction myschema.mypkg.my_validation_check`

- By default, `mod_plsql` already disallows direct URL access to certain schemas/packages. For more information, refer to PlsqlExclusionList.

- It is highly recommended that you provide an implementation for this function such that it only allows requests that belong to your application, and are callable from a browser.
- Since this function will be called for every request, be sure to make this function as performant as possible. Suggested recommendations are:
  - Name your PL/SQL packages in a fashion such that the implementation of this function can be similar to the example mentioned earlier.
  - If your implementation performs a table lookup to determine what packages/procedures should be allowed, performance can be improved if you pin the cursor in the shared pool.

PlsqlSessionCookieName

Specifies the cookie name when PlsqlAuthenticationMode is set to `SingleSignOn`. This parameter is supported only for Oracle Application Server releases, and is used by the Oracle Application Server Portal and Oracle Application Server Single Sign-On.

Category	Value
Syntax	`PlsqlSessionCookieName` *`cookie_name`*
Default	Same as DAD name
Example	`PlsqlSessionCookieName mycookie`

- For DADs not using `SingleSignOn` authentication, this parameter can be omitted. In most other cases, the session cookie name should be omitted (and this parameter automatically defaults to the DAD name).
- A session cookie name must be specified only for Oracle Application Server Portal instances that need to participate in a distributed Oracle Application Server Portal environment. For those Oracle Application Server Portal nodes you want to seamlessly participate as a federated cluster, ensure that the session cookie name for all of the participating nodes is the same.
- Independent Oracle Application Server Portal nodes need to use distinct session cookie names.
- In older versions of the product, this configuration parameter was called `sncookiename`.

PlsqlSessionStateManagement

Specifies how package and session state should be cleaned up at the end of each `mod_plsql` request.

- Setting this parameter to `StatelessWithResetPackageState` causes `mod_plsql` to call `dbms_session.reset_package_state` at the end of each `mod_plsql` request.
- Setting this parameter to `StatelessWithPreservePackageState` causes `mod_plsql` to call `htp.init` at the end of each `mod_plsql` request. This cleans up the state of session variables in the PL/SQL Web ToolKit. The PL/SQL application is responsible for cleaning up its own session state. Failure to do so causes erratic behavior, in which a request starts recognizing or manipulating state modified in previous requests.
- Setting this parameter to `StatelessWithFastResetPackageState` causes `mod_plsql` to call `dbms_session.modify_package_state(dbms_`

session.reinitialize) at the end of each mod_plsql request. This API is a lot faster than the mode of StatelessWithResetPackageState, and avoids some latch contention issues, but exists only in database versions 8.1.7.2 and higher. This mode uses up slightly more memory than the default mode.

Category	Value
Syntax	PlsqlSessionStateManagement StatelessWithResetPackageState/StatelessWithFastResetPackageState/StatelessWithPreservePackageState
Default	StatelessWithResetPackageState
Example	PlsqlSessionStateManagement StatelessWithResetPackageState

- In older versions of the product, this configuration parameter was called stateful.
- An older value of stateful=no or stateful=STATELESS_RESET corresponds to PlsqlSessionStateManagement StatelessWithResetPackageState
- An older value of stateful=STATELESS_FAST_RESET corresponds to PlsqlSessionStateManagement StatelessWithFastResetPackageState
- An older value of stateful=STATELESS_PRESERVE corresponds to PlsqlSessionStateManagement StatelessWithPreservePackageState

mod_plsql does not support stateful mode of operation. To equip PL/SQL applications with stateful behavior, save state in cookies and/or in the database.

PlsqlTransferMode

Specifies the transfer mode for data from the database back to mod_plsql. Most applications use the default value of CHAR.

Category	Value
Syntax	PlsqlTransferMode *CHAR/RAW*
Default	CHAR
Example	PlsqlTransferMode CHAR

- This parameter only needs to be changed to enable sending back responses in different character sets from the same DAD. In such a case, the CHAR mode is useless, since it always converts the response data from the database character set to the mod_plsql character set.
- In older versions of the product, RAW transfer mode was not supported.

PlsqlUploadAsLongRaw

Specifies the extensions to be uploaded as LONGRAW data type, as opposed to using the default BLOB data type. The default can be overridden by specifying multi-line directives of file extensions for field. A value of '*' in this field causes all documents to be uploaded as LONGRAW.

Category	Value
Syntax	`PlsqlUploadAsLongRaw` *`string multiline`*
Default	None
Example	`PlsqlUploadAsLongRaw jpg, PlsqlUploadAsLongRaw gif`

- For applications that do not do document uploads or downloads, this parameter may be omitted.

  See Also: *Oracle Application Server mod_plsql User's Guide* for more information about upload and download processes and the structure of the restrictions on the document table format.

- In older versions of the product, this parameter was called `upload_as_log_raw`.

cache.conf

`cache.conf` file contains the cache settings for `mod_plsql`. This file contains parameters which specify the characteristics of the `mod_plsql` cache system.

Note: This file is relevant only if the PL/SQL Application uses the `OWA_CACHE` packages to cache content in the file system. Extremely few customer applications make use of the `OWA_CACHE` packages.

The following parameters are specified in `cache.conf` file:

- PlsqlCacheCleanupTime
- PlsqlCacheDirectory
- PlsqlCacheEnable
- PlsqlCacheMaxAge
- PlsqlCacheMaxSize
- PlsqlCacheTotalSize

PlsqlCacheCleanupTime

Specifies the time to start the cleanup of the cache storage.

This setting defines the exact day and time in which cleanup should occur. The frequency can be set as daily, weekly, and monthly.

- To define daily frequency, the keyword "`Everyday`" is used. The cleanup starts everyday at the time defined. For example, `Everyday 2:00`. This causes the cleanup to happen everyday at 2 AM (local time) in the morning.
- To define weekly frequency, the days of the week such as "`Sunday`", "`Monday`", "`Tuesday`", and so on are used. For example, `Wednesday 15:30`. This causes the cleanup to happen every Wednesday at 3:30 PM (local time) in the afternoon.
- To define monthly frequency, the keyword "`Everymonth`" is used. The cleanup starts at the Saturday of the month at the time defined. For example, `Everymonth 23:00`. This causes the cleanup to happen the first Saturday of every month at 11:00 PM (local time) at night.

Category	Value
Syntax	`PlsqlCacheCleanupTime <Sunday-Saturday, Everyday, Everymonth> <hh:mm>`
Default	Saturday 23:00
Example	`PlsqlCacheCleanupTime Saturday 23:00`

PlsqlCacheDirectory

Specifies the directory where cache files are written out by `mod_plsql`. This directory must exist or else Oracle HTTP Server will not start.

On UNIX, this directory must have write permissions by the owner of the child httpd processes.

Category	Value
Syntax	`PlsqlCacheDirectory <directory>`
Default	none
Example	`PlsqlCacheDirectory ORACLE_HOME/Apache/modplsql/cache`

In older versions, this parameter was called "`cache_dir`" and resides in the "[PLSQL Cache]" section of *`ORACLE_HOME`*`/Apache/modplsql/cfg/cache.cfg`.

PlsqlCacheEnable

Enables `mod_plsql` caching.

Category	Value
Syntax	`PlsqlCacheEnable On/Off`
Default	Off
Example	`PlsqlCacheEnable On`

- If you are sure that your application does not make use of the `OWA_CACHE` packages, in the PL/SQL Web Toolkit, then you can choose to disable caching. In such situations, there will be a very minor performance benefit.
- In older versions, this parameter is called "enabled" and resided in the "[PLSQL Cache]" section of *`ORACLE_HOME`*`/Apache/modplsql/cfg/cache.cfg`.

PlsqlCacheMaxAge

Specifies the maximum time, in days, a cache file can be allowed to reside in a file system cache, after which the cached file will be removed for cache maintenance.

This setting is to ensure that the cache system does not contain old content. This setting removes old cache files and makes space for new ones.

Category	Value
Syntax	`PlsqlCacheMaxAge <number>`
Default	30 (30 days)
Example	`PlsqlCacheMaxAge 30`

PlsqlCacheMaxSize

Specifies the maximum possible size of a cache file.

This setting is to prevent the case in which one file can fill up the entire cache. In general, it is recommended that this be set to about 1-3 percent of the total cache size.

Category	Value
Syntax	`PlsqlCacheMaxSize <number>`
Default	1048576 (1 MB)
Example	`PlsqlCacheMaxSize 1048576`

In older versions, this parameter was called "`max_size`" and resided in the "[PLSQL Cache]" section of *`ORACLE_HOME`*`/Apache/modplsql/cfg/cache/cfg`.

PlsqlCacheTotalSize

Specifies the total size of the cache directory.

This setting limits the amount of space the cache is allowed to use. Both PLSQL cache and Session Cookie cache share this cache space. Note that this setting is not a hard limit. It might exceed the limit temporarily during normal processing. This is normal behavior.

The cleanup algorithm uses this setting to determine how much to reduce the cache files. Therefore, the real space limit is the physical storage's available size.

This parameter takes bytes as values;

- 1 megabytes = 1048576 bytes
- 10 megabytes = 10485760 bytes

Category	Value
Syntax	`PlsqlCacheTotalSize <number>`
Default	20971520 (20 MB)
Example	`PlsqlCacheTotalSize 20971520`

In older versions, this parameter was called "`total_size`" and resided in the "[PLSQL Cache]" section of *`ORACLE_HOME`*`/Apache/modplsql/cfg/cache/cfg`.

mod_proxy

Provides proxy capability for `FTP`, `CONNECT` (for SSL), HTTP/0.9, HTTP/1.0, and HTTP/1.1.

See Also:

- Module `mod_proxy` in the Apache Server documentation.
- "Using mod_proxy Directives" on page 11-16

mod_rewrite

Oracle HTTP Server provides `mod_rewrite` as a tool for URL manipulation. A rewriting engine based on a regular-expression parser is used by `mod_rewrite` to

rewrite requested URLs. The granularity of URL manipulations can be affected by the formats of server variables, environment variables, HTTP headers, and time stamps.

This module operates on the full URLs (including the path-info part) both in per-server context (`httpd.conf`) and per-directory context (`.htaccess`) and can generate query-string parts on result.

The following topics are discussed in subsequent sections:

- mod_rewrite Rules Processing
- mod_rewrite Directives
- Rewrite Rules Hints
- Redirection Examples

mod_rewrite Rules Processing

Apache processes HTTP in phases. A hook for each of these phases is provided by the Apache API. `mod_rewrite` uses two of these hooks - the URL-to-filename translation hook which is used after the HTTP request has been read but before any authorization starts, and the Fixup hook which is triggered after the authorization phases and after the per-directory configuration files (`.htaccess`) have been read, but before the content handler is activated.

`mod_rewrite` reads the configured rulesets from its configuration structure. Server level rulesets are best configured at startup, while directory level rulesets are configured during the directory access of the kernel.

`mod_rewrite` loops through the ruleset rule by rule (`RewriteRule` directive) and when a particular rule matches, it loops through corresponding conditions (`RewriteCond` directives). First the URL is matched against the `Pattern` of each rule. When it fails, `mod_rewrite` looks for corresponding rule conditions. If none are present, it just substitutes the URL with a new value which is constructed from the string `Substitution` and goes on with its rule-looping. But if conditions exist, it starts an inner loop for processing them in the order that they are listed.

For conditions, a string `TestString` is created by expanding variables, back-references map lookups, and then `CondPattern` is matched against the expanded `TestString`. If the pattern does not match, the complete set of conditions and the corresponding rule fails. If the pattern matches, then the next condition is processed until no more conditions are available. If all conditions match, processing is continued with substituting the URL using `Substitution`.

When request seeks a URL with more than one slash (/), for example, `http://yourserver//oldpath/rqstdrsrc`, the "`//oldpath`" may bypass `RewriteCond` and `RewriteRule` directives if they are not correctly written.

For example, consider the following rule:

```
RewriteRule ^/oldpath(.*) /newpath$1 [R]
```

Requesting `http://yourserver/oldpath/files` will redirect and return the page `http://yourserver/newpath/files` as expected.

However, requesting `http://yourserver//oldpath/files` will bypass this particular rule, potentially serving a page that you were not expecting it to. You can work around the problem by making sure that rules will capture more than one slash (/). To fix the example, you should use this replacement:

```
RewriteRule ^/+somepath(.*) /otherpath$1 [R]
```

mod_rewrite Directives

This section discusses the following `mod_rewrite` directives:

- RewriteEngine
- RewriteOptions
- RewriteLog
- RewriteLogLevel
- RewriteBase

RewriteEngine

Enables or disables the runtime rewriting engine. If it is set to "Off", this module does no runtime processing at all. Use this directive to disable the module instead of commenting out all the `RewriteRule` directives.

Rewrite configurations are not inherited by default. This means that you need to have `ReWriteEngine On` directive for each virtual host in which you want to use it.

RewriteOptions

By specifying `RewriteOptions` 'inherit', you can force the configuration of the parent by the children. In virtual-server context this means that the maps, conditions and rules of the main server are inherited. In directory context this means that conditions and rules of the `.htaccess` configuration of the parent directory are inherited.

RewriteLog

Sets the name of the file to which the server logs any rewriting action that it performs. If the name does not begin with a slash (/), then it is assumed to be relative to the `Server Root`. To disable logging, either remove or comment out the `RewriteLog` directive or use `RewriteLogLevel 0`. Avoid setting the filename to `/dev/null` to prevent logging. This can slow down the server with no advantage.

RewriteLogLevel

Sets the verbosity level of the rewriting log file. The default level 0 means no logging, while 9 or more means that practically all actions are logged.

RewriteBase

Explicitly sets the base URL for pre-directory rewrites. Rewrite rule can be used in per-directory configuration (`.htaccess`) files. When a substitution occurs for a new URL, the base URL should be added into the server processing. To be able to do this, the module needs to know what the corresponding URL-prefix or URL-base is. By default, this prefix is the corresponding file path itself. However, at most Web sites, URLs are not directly related to physical filename paths. In such cases, you have to use the `RewriteBase` directives to specify the correct URL-prefix.

If the URLs of your Web server are not directly related to physical file paths, you have to use `RewriteBase` in every `.htaccess` files where you want to use `RewriteRule` directives.

Example 8–10 RewriteBase Directive

Assume the following per-directory configuration file:

```
## /abc/def/.htaccess - - per-dir config file for directory /abc/def
 # /abc/def is the physical path of /xyz,
```

```
RewriteEngine On
RewriteBase /xyz
RewriteRule ^oldstuff\.html$ newstuff.html
```

In Example 8–10, a request to `/xyz/oldstuff.html` gets correctly rewritten to the physical file `/abc/def/newstff.html`.

Rewrite Rules Hints

Table 8–5 provide hints for using rewrite rules.

Table 8–5 Rewrite Rules Hints

Value	Definition
.	Any single character
[char]	Any character listed within a square bracket
b*	Any character b any number of times
.*	Any character any number of times

For example, if you want to redirect requests from `/demo1`, `/demo2`, and `/demo3` to `/alldemos`, write the rewrite rule as one of the following:

```
RewriteRule /demo. /alldemos [R]
```

or,

```
RewriteRule /demo [123] /alldemos [R]
```

If you intend that `/DemoA`, `/DemoB`, and `/DemoC` to be redirected to `/alldemos`, add NC (no case) to the rewrite rules, such as:

```
RewriteRule /demo [123] /alldemos [R, NC]
```

This rewrite rule will not work to redirect from `/demonstration1` to `/demos`, because "." works form one character only. To enable redirection of all URLs beginning with "demo", irrespective of subsequent characters, use the rewrite rule as follows:

```
RewriteRule ^/demo* /alldemos [R, NC]
```

In the preceding example, ^ means the beginning, * means any character after demo.

If there was a request for `/demo1/not_just_index.html`, all the preceding rewrite rules would have redirected the request the request to `/alldemos/index.html`, that may not be what you want. It is quite possible that you may want to redirect to the corresponding files in `/alldemos`, as listed in Table 8–6.

Table 8–6 Request Redirection

Request for	Redirected to
`/demo1/happy.html`	`/alldemos/happy.html`
`/demo1/go.jpg`	`/alldemos/go.jpg`
`/demos1/lucky.jpg`	`/alldemos/lucky.jpg`

Then you have to use substitution in your rewrite rule as follows:

```
RewriteRule ^/demos1(.*)$ //alldemos/$1 [R NC]
```

The explanation for this rule is:

Take the value of the expression, such as `happy.html`, `go.jpg`, and `lucky.jpg`, that appears after `demo1` as variables ($1) and substitute it after `/alldemos/`.

Redirection Examples

For redirecting requests from the `DocumentRoot` to a directory called `newroot`, set the following `mod_rewrite` directives:

```
RewriteEngine On
RewriteRule ^/(.*)$ /newroot/$1 [R]
```

For directing requested for files from one directory (`olddir`) to another (`newdir`), set the following directives:

```
RewriteEngine On
RewriteRule ^/olddir(.*)$ /newdir/$1 [R]
```

In each of these cases, you should ensure that the requested resources are indeed available in the redirected location. The `mod_rewrite` module does not ensure the existence of the requested resource in the new location.

For disabling all requests using the `HTTP TRACE` method, set the following `mod_rewrite` directives:

```
RewriteEngine On
RewriteCond %{REQUEST_METHOD} ^TRACE
RewriteRule .* - [F]
```

> **See Also:** Module `mod_rewrite` in the Apache Server documentation.

mod_security

Increases Web application security by protecting Web application from known and unknown attacks.

> **See Also:** `http://modsecurity.org`

mod_setenvif

Enables you to set environment variables based on characteristics of a request.

> **See Also:** Module `mod_setenvif` in the Apache Server documentation.

mod_speling

Attempts to correct misspelled or miscapitalized URLs.

> **See Also:** Module `mod_speling` in the Apache Server documentation.

mod_status

Displays an HTML page of server activity and performance.

See Also: Module `mod_status` in the Apache Server documentation.

mod_unique_id

Creates a unique ID for each request. This module is available on UNIX only.

See Also: Module `mod_unique_id` in the Apache Server documentation.

mod_userdir

Maps requests to user-specific directories.

See Also: Module `mod_userdir` in the Apache Server documentation.

mod_usertrack

Tracks user activity by creating a log.

See Also: Module `mod_usertrack` in the Apache Server documentation.

mod_vhost_alias

Enables dynamically configured mass virtual hosting.

See Also: Module `mod_vhost_alias` in the Apache Server documentation.

mod_wchandshake

Provides automatic discovery of Oracle HTTP Server by OracleAS Web Cache. If OracleAS Web Cache is not used, this module can be disabled. It is an Oracle module.

9

Configuring and Using mod_oradav

This chapter describes distributed authoring and versioning concepts, and explains how to configure and use the `mod_oradav` module. The `mod_oradav` module enables you to use OraDAV to access content in an Oracle database from a Web browser or a WebDAV client.

Topics discussed are:

OraDAV Concepts

The term **OraDAV** refers to the capabilities available through the `mod_oradav` module. The `mod_oradav` module is an extended implementation of `mod_dav`, which is an implementation of the WebDAV specification. This section explains the following concepts:

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.

WebDAV enables collaboration among authors building Web sites. WebDAV also serves as universal read and write access protocol to arbitrary hierarchies of content (not necessarily Web sites). With WebDAV, you can save content to a URL provided by

an Internet Service Provider (ISP), and then be able to access and optionally change that content from various devices.

WebDAV was initiated as an Internet Engineering Task Force (IETF) standard. The first phase of WebDAV is specified in RFC 2518, which provides the basic primitives for managing hierarchies of information, locking, reading, writing, and querying properties of a WebDAV document. Subsequent work on WebDAV is ongoing and is focusing on completing issues relating to content management over the Web. This includes WebDAV authentication and authorization (access controls), versioning, bindings, ordered collections, and querying (DAV Advanced Searching and Locating).

Microsoft Web folders is a WebDAV client on Windows 2000, and later versions (using Internet Explorer 5.0 and higher). Office 2000 and Office XP applications and the IIS server support WebDAV, meaning that you can start a Microsoft Office application and specify a URL, edit the content, and save it back to the URL from which it was retrieved. WebDAV also has Java clients (such as DAV Explorer), open source tools (such as Cadaver and Sitecopy), and Apple GUI tools (such as Goliath).

Note: When a WebDAV client first connects to Oracle HTTP Server, you must use the full `ServerName` string (as specified in the `httpd.conf` file) in the URL for the connection. Do not use an abbreviated form of the server name.

For example, if the `ServerName` value is `server1.acme.com`, connect to Oracle HTTP Server using the string `http://server1.acme.com:7778`, not an abbreviated form such as `http://server1:7778`.

If you use an abbreviated form, the connection might succeed, but `COPY` and `MOVE` operations will fail to execute and generate `BAD_GATEWAY` errors.

---

mod_dav

`mod_dav` is the Apache Software Foundation native implementation of the WebDAV specification.

mod_oradav

`mod_oradav` is the Oracle module (an OCI application written in C) that is an extended implementation of `mod_dav`, and is integrated with Oracle HTTP Server. `mod_oradav` performs read/write activity to local files and to Oracle databases. Oracle databases must have an OraDAV driver (a stored procedure package) that `mod_oradav` calls to map WebDAV activity to database activity. Essentially, `mod_oradav` enables WebDAV clients to connect to an Oracle database, read and write content, and query and lock documents in various schemas.

You can configure `mod_oradav` to use an Oracle database using standard Oracle HTTP Server directives. `mod_oradav` can immediately leverage other module code (such as mime_magic) in order to perform content management tasks. Most WebDAV processing activity involves streaming content to and from a content provider; and `mod_oradav` uses OCI streaming logic directly within Oracle HTTP Server.

OraDAV

OraDAV refers to the whole set of capabilities that are available through `mod_oradav` to Oracle Application Server users. Some OraDAV-specific terms include:

- **Apache OraDAV**: Code in the Apache HTTP server that supports file-based DAV access and makes calls to Oracle.
- **OraDAV driver API**: Set of stored procedure calls that are used by the OraDAV driver to manage content in an Oracle database, providing support for the following WebDAV functions over the Internet: reading and writing documents, locking and unlocking documents, managing (creating, populating, deleting) hierarchies of information, retrieving properties associated with documents, and associating properties with specific documents.
- **OraDAV driver**: Stored procedure implementation of the OraDAV driver API that executes in Oracle and manages a repository.

OraDAV Architecture

OraDAV fits into an architecture in which `mod_oradav`, within Oracle HTTP Server, provides access to content in one or more schemas in one or more Oracle databases.

A simple form of the architecture is illustrated in Figure 9–1.

Figure 9–1 OraDAV Architecture

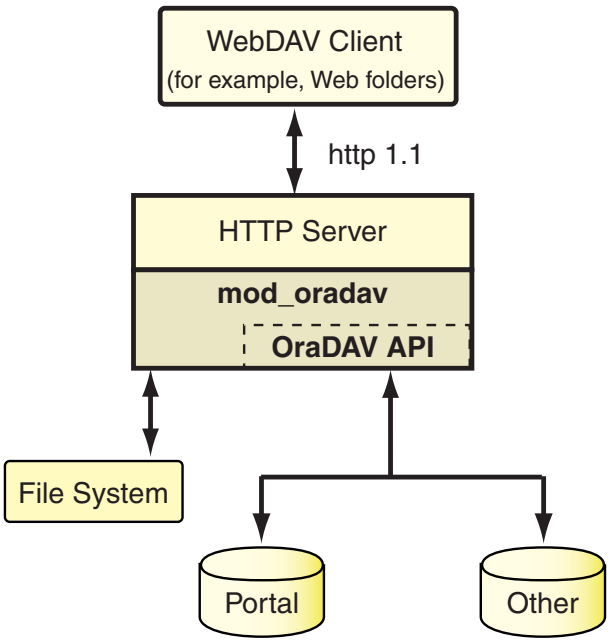


Figure 9–1 shows a WebDAV client, such as Microsoft Web folders, passing HTTP requests to Oracle HTTP Server. If the request is for content stored in the file system (not in an Oracle database), `mod_oradav` handles the access. If the request is for content stored in an Oracle database, the OraDAV API handles the access.

The OraDAV API provides capabilities that are equivalent to using `mod_oradav` running with a file system. The following HTTP methods are supported by the OraDAV API:

- `COPY`
- `DELETE`
- `MOVE`
- `MKCOL`

- GET
- HEAD
- LOCK
- PROPFIND
- PROPPATCH
- PUT
- UNLOCK

The OraDAV API supports shared and exclusive locking, retrieving basic DAV properties, and defining and retrieving server-defined live properties or client-defined dead properties. Set-based operations such as COPY, MOVE, DELETE can be done completely by a single call to an OraDAV driver.

OraDAV Users

The primary direct users of OraDAV are Oracle HTTP Server administrators and database administrators (DBAs) of Oracle databases. End users interact only indirectly with OraDAV through Web browsers or WebDAV client tools.

OraDAV administration involves tasks for a Web master and for a DBA:

- The Web master needs to know how to start and stop Oracle HTTP Server, and how to configure Oracle HTTP Server to direct URL traffic to an OraDAV driver.
- The DBA needs to know how to set up client connectivity to an Oracle database from the system running Oracle HTTP Server, to install and administer the OraDAV driver, and perhaps to tune the content managed by the driver based on physical storage characteristics.

OraDAV Usage Model

OraDAV usage can involve any combination of the following activities:

- **Browsing:** Read-only activity which uses WebDAV to access content in an Oracle database. Its usage model is that of a typical read-only Web site.
- **Restructuring:** Deleting, moving, and copying content. Restructuring is usually done infrequently by a restricted set of individuals who have write access to the WebDAV content. Restructuring has the same limitations and complications that one encounters when restructuring a file directory. In some cases, this directory hierarchy is owned and managed by one user. If the directory is shared, the client doing restructuring is given sole access to the hierarchy through WebDAV exclusive locks.
- **Editing:** Modifying one or a small subset of resources in a hierarchy. Properly designed WebDAV clients take out shared or exclusive locks on such resources to coordinate these activities.
- **Property Management:** Associating properties and attributes (for example, author) with documents for ease of lookup and for categorization. WebDAV clients assign properties to documents using the PROPPATCH directive and retrieve properties using the PROPFIND directive.

OraDAV Configuration Parameters

You configure OraDAV mainly through parameters in the `httpd.conf` file, which is used by an Oracle HTTP Server instance when it is initializing. Some configuration parameters are required for all OraDAV drivers, and others are driver-specific.

When Oracle Application Server is installed, all required OraDAV parameters are set with values that are designed to enable Oracle database content to be accessed through a Web browser or WebDAV client. If the default values do not meet your needs, you can later modify the values for required parameters and specify values for optional parameters. The parameters used in `httpd.conf` to support OraDAV configuration start with `DAV` and `DAVParam`. These parameters are specified within a `<Location>` container directive, and they provide:

- A way of configuring how Oracle HTTP Server connects to the database.
- Coarse controls on OraDAV behavior.

The `DAV` parameter indicates that a URL location is DAV-enabled. The `DAV` keyword is followed by one of the following values:

- `On`, which indicates that `mod_oradav` is to use the local file system for content
- `Oracle`, which indicates that `mod_oradav` is to use OraDAV for all content

`DAVParam` parameters are used to specify name-value pairs. The required pairs are those that enable Oracle HTTP Server to connect to an Oracle database. These include the names `OraService`, `OraUser`, and `OraPassword` or `OraAltPassword`.

Example 9–1 shows a configuration for accessing files on the local system. It specifies that the directory `myfiles` under the Web server documents directory (`htdocs` by default) is to be DAV-enabled, along with all directories under `myfiles` in the hierarchy. Note that there must not be any symlinks defined on `myfiles` or any directory under it in the hierarchy.

Example 9–1 Configuration Parameters: File System Access

```
<Location /myfiles>
  DAV On
</Location>
```

Example 9–2 shows a configuration for accessing content through Oracle Application Server Portal. After OracleAS Portal has been installed in Oracle Application Server, the Oracle HTTP Server configuration file should be populated with a `<Location>` container directive which points to the OracleAS Portal schema. In this example, the location `/portal` will be OraDAV-enabled and will (once populated with the correct values) connect to the OracleAS Portal schema so that users can use WebDAV clients to access OracleAS Portal data.

Example 9–2 Configuration Parameters: Portal Access

```
<Location /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>
```

Each OraDAV driver can use the `DAVParam` mechanism to create its own driver-specific settings. All `DAVParam` name-value pairs are passed to the OraDAV

driver. In addition to the OraDAV parameters, you should consider whether to specify certain DAV parameters, such as DAVDepthInfinity.

See Also: "DAV Directives" on page 9-15

Table 9–1 lists each OraDAV parameter, whether it is required or optional, and its default value. ORAGetSource applies only to file system access; the other parameters apply only to OracleAS Portal driver and other (non-file system) access.

Table 9–1 OraDAV Parameters

Name	Required/Optional	Default Value
ORAAllowIndexDetails	Optional	FALSE
ORAAltPassword	Required. Either ORAPassword or ORAAltPassword must be specified, but not both.	(none)
ORACacheDirectory	Optional	(none)
ORACacheMaxResourceSize	Optional	(none)
ORACachePrunePercent	Optional	25
ORACacheTotalSize	Optional. ORACacheTotalSize is required if ORACacheDirectory is used; otherwise, do not specify the parameter.	(none)
ORAConnect	Required. ORAService, ORAConnect, or ORAConnectSN must be specified, but no more than one.	(none)
ORAConnectSN	Required. ORAService, ORAConnect, or ORAConnectSN must be specified, but no more than one.	(none)
ORAContainerName	Required	(none)
ORAException	Optional	NORAISE
ORAGetSource	Optional	(none)
ORALockExpirationPad	Optional	0 (seconds)
ORAPackageName	Optional	ORDSYS.DAV_API_DRIVER
ORAPassword	Required. Either ORAPassword or ORAAltPassword must be specified, but not both.	(none)
ORARootPrefix	Optional	(none)
ORAService	Required. ORAService, ORAConnect, or ORAConnectSN must be specified, but no more than one.	(none)
ORATraceEvents	Optional	(none)
ORATraceLevel	Optional	0

Table 9–1 (Cont.) OraDAV Parameters

Name	Required/Optional	Default Value
ORAUser	Required	(none)

Note: All OraDAV parameters are passed from Oracle HTTP Server to the routines in the `ORAPackageName` package as part of the `context` parameter. Keys are uppercase in Oracle HTTP Server, such as `ORAUSER`, but the values, such as `scott`, are not.

ORAAllowIndexDetails

In an Oracle HTTP Server environment that is not OraDAV-enabled, `mod_dav` itself does not respond to HTTP `GET` requests. Instead, normal Oracle HTTP Server mechanisms are used to respond to `GET` requests. However, when all your content is in an Oracle database, normal Oracle HTTP Server mechanisms cannot be used to respond to `GET` requests, and thus OraDAV must respond to `GET` requests.

The `ORAAllowIndexDetails` parameter controls how OraDAV responds when a `GET` request is performed on a DAV collection and no `index.html` file is found in that collection (directory). In a typical Oracle HTTP Server environment, a separate module takes control, automatically generating and returning to the client HTML that represents an "index" of the resources (files) in that collection.

An OraDAV-enabled Oracle HTTP Server performs similar actions when responding to a `GET` request on a collection. A Description column (containing links to more detailed information about each resource) is included in the generated index when `ORAAllowIndexDetails` is set to `TRUE`.

The default is `FALSE`, in which case no Description column appears in the generated index, and if `?details` is used in a URL, it is ignored and the URL contents are returned.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional
Values	`TRUE/FALSE`
Default	`FALSE`

ORAAltPassword

Specifies the password associated with the user specified by the `ORAUser` parameter, but the password is a base-64 encoded character string. The `ORAAltPassword` parameter provides an alternative if you do not want the password to appear in unencoded plain text in that parameter.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Required, unless ORAPassword is specified
Values	(character string)
Default	(none)

If the ORAPassword parameter is not specified, the `ORAAltPassword` parameter is used for the password.

ORACacheDirectory

Specifies the directory to use for disk caching operations. If you do not use this parameter, disk caching is not performed for OraDAV operations.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional
Values	(character string)
Default	(none)

The specified directory must exist and be readable by Oracle HTTP Server, but cannot be visible to normal `GET` requests. (If the directory is visible to normal `GET` requests, security measures could be bypassed by users accessing the cache directory.)

The directory should not be an NFS-mounted directory, because most UNIX locking mechanisms caution against this. The directory should be located on a file system that supports a "last accessed" time. On Windows systems, this means using NTFS (not FAT) formatted partitions.

Do not use the cache directory for anything other than caching. Any files in the cache directory are subject to deletion.

If you use the `ORACacheDirectory` parameter, you must also use the ORACacheTotalSize parameter.

See Also: "Using Disk Caching with OraDAV" on page 9-19

ORACacheMaxResourceSize

Specifies a maximum cacheable resource size for disk caching operations.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional
Values	(integer, with optional unit character string)
Default	(none)

Example 9–3 shows how to set ORACacheMaxResourceSize.

Example 9–3 ORACacheMaxResourceSize Parameter

```
DAVParam ORACacheMaxResourceSize 1024KB
```

The setting in Example 9–3 prevents OraDAV from caching any resource larger than one megabyte. The goal is to give Web masters the ability to prevent large media files from dominating the cache. However, be aware that the performance benefit from caching a large file is greater than from caching a small file.

You can specify KB (for kilobytes) or MB (for megabytes) after an integer. If you do not specify a unit after the integer, the default unit is bytes.

See Also: "Using Disk Caching with OraDAV" on page 9-19

ORACachePrunePercent

Specifies the percentage of disk cache usage to be freed up when the cache is full. When the disk cache is full, the oldest files in the cache are deleted (pruned) until the cache disk usage is reduced by the `ORACachePrunePercent` value.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional
Values	integer (1 to 100)
Default	25

See Also: "Using Disk Caching with OraDAV" on page 9-19

ORACacheTotalSize

Specifies the size of the cache to use for disk caching operations.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional, unless ORACacheDirectory is specified
Values	(integer, with optional unit character string, GB or MB) Maximum value is 4GB.
Default	(none)

You can specify MB (for megabytes) or GB (for gigabytes) after an integer, as shown in Example 9–4. If you do not specify a unit after the integer, the default unit is bytes.

Example 9–4 ORACacheTotalSize Parameter

```
DAVParam ORACacheTotalSize 1GB
```

If you use the ORACacheDirectory parameter, you must also use the `ORACacheTotalSize` parameter.

The `ORACacheTotalSize` value should be large enough to hold either a significant fraction of your Web site, or all of your most frequently accessed files plus 25 percent more space. If the value is too small, overall performance degrades because of the extra work of writing BLOB data to the file system and quickly deleting files to make room for newer cache requests.

The actual space utilized by the disk cache might sometimes exceed the `ORACacheTotalSize` value, possibly by as much as the ORACacheMaxResourceSize value. You should also be aware of file system block size issues that could cause the cache to use more disk space than the `ORACacheTotalSize` value.

See Also: "Using Disk Caching with OraDAV" on page 9-19

ORAConnect

Specifies the Oracle database to which to connect. The value must be in the following format:

```
database-host:database-port:database-sid
```

Example 9–5 shows how to use the `ORAConnect` parameter.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Required, unless ORAService or ORAConnectSN is specified
Values	(character string)
Default	(none)

Example 9–5 ORAConnect Parameter

```
DAVParam ORAConnect my-pc.acme.com:1521:mysid
```

The `ORAConnect` parameter lets you connect to a database that is not included in the `tnsnames.ora` file.

You must specify one, and no more than one, of the following parameters: `ORAConnect`, ORAService, or ORAConnectSN.

ORAConnectSN

Specifies the Oracle database to which to connect. The value must be in the following format:

database-host:database-port:database-service-name

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Required, unless ORAService or ORAConnect is specified
Values	(character string)
Default	(none)

The `ORAConnectSN` parameter lets you connect to a database that is not included in the `tnsnames.ora` file, as shown in Example 9–6.

Example 9–6 ORAConnectSN Parameter

```
DAVParam ORAConnectSN my-pc.acme.com:1521:myservice
```

You must specify one, and no more than one, of the following parameters: ORAService, ORAConnect, or `ORAConnectSN`.

ORAContainerName

Within the schema specified by the ORAUser parameter, there must exist a container. The `ORAContainerName` parameter specifies the name of the container to use for the location.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Required
Values	(any valid character string, up to 20 characters)
Default	(none)

ORAException

Writes PS/SQL stack dumps into the Oracle HTTP Server log file, error_log, in the event of an exception in the PL/SQL package.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional
Values	`NORAISE` or `RAISE`
Default	`NORAISE`

Caution: Use this parameter for debugging problems with PL/SQL packages. However, note that it can use a large amount of disk space and can slow the performance of your system.

ORAGetSource

Applies only to file system access. It specifies one or more file extensions to identify types of files that are not to be executed, but rather opened for editing. Include periods (.) with the file extension and use a comma to separate file extensions. For example:

```
".htm, .html, .jsp1, .jsp2"
```

Category	Value
Applies to	File system access
Required/Optional	Optional
Values	(character string in double quotation marks)
Default	(none)

The `ORAGetSource` parameter lets you open for editing files that are usually executed as a result of a `GET` operation.

Note: Because `.jsp` and `.sqljsp` files are by default opened for editing, you do not need to specify them with the `ORAGetSource` parameter.

ORALockExpirationPad

Intended to be used in high-latency network environments, to adjust for the "refresh lock" behavior in Microsoft Office. Microsoft Office attempts to refresh locks on DAV

resources just before the lock is set to expire. However, if there is network congestion between the Microsoft Office client and the DAV server, the refresh request might arrive too late, that is, after the lock has expired.

OraDAV periodically looks for locks on resources that have expired and deletes those locks. The `ORALockExpirationPad` parameter can be used to provide some additional ("pad") time between when a lock expires and when that lock is deleted. For example, if `ORALockExpirationPad` is set to 120, OraDAV does not actually delete locks until at least two minutes after the expiration time.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional
Values	(number of seconds)
Default	0

ORAPackageName

Identifies the OraDAV driver implementation that is to be called when issuing OraDAV commands. The default is the OraDAV driver, which is the `ORDSYS.DAV_API_DRIVER` package.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Required
Values	(character string)
Default	`ORDSYS.DAV_API_DRIVER`

ORAPassword

Specifies the password associated with the user specified by the `ORAUser` parameter.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Required, unless ORAAltPassword is specified
Values	(character string)
Default	(none)

If you do not want to specify the password as an unencoded text string with the `ORAPassword` parameter, you can specify the password as a base-64 encoded string with the ORAAltPassword parameter.

ORARootPrefix

Specifies the directory within the database repository to use as the root. If this parameter is specified, WebDAV clients see this directory as the `root` and are not be able to see the repository directories that lead up to it.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional
Values	(character string)
Default	(none)

In Example 9–7, assume that the database repository contains the directory `/first/second/third/fourth`, and that `ORARootPrefix` is defined as follows (do not include a trailing slash (/) in the value):

Example 9–7 ORARootPrefix Parameter

```
DAVParam ORARootPrefix /first/second
```

In this case, WebDAV clients will see the `/third` directory and be able to navigate to the `/third/fourth` directory, but will not be able to see or navigate to the `/first` or `/first/second` directories.

ORAService

Specifies the Oracle database to which to connect. The specified value must match a SID value in the `tnsnames.ora` file as shown in Example 9–8.

Example 9–8 ORAService Parameter

```
DAVParam ORAService mydbsid.mydomain.com
```

To connect to a database that is not included in the `tnsnames.ora` file, use the ORAConnect parameter. You must specify one, and no more than one, of the following parameters: `ORAService`, ORAConnect, or ORAConnectSN.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Required, unless ORAConnect or ORAConnectSN is specified
Values	(character string matching an entry in the `tnsnames.ora` file)
Default	(none)

ORATraceEvents

Specifies types of events to be recorded in the Apache error log for debugging.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional

Category	Value
Values	One of the following character strings: ■ `getsource`: Traces GET activity against the file system. ■ `hreftoutf8`: Traces the HREF conversion from the native character set to UTF-8. ■ `request`: Traces DAV requests, responses, and status values handled by mod_oradav.
Default	(none)

Caution: Although this parameter is useful for debugging purposes, it can use a large amount of disk space and can slow the performance of your system.

ORATraceLevel

Specifies the level of debugging (trace statements) that will be entered in the Apache error log. The lowest level is 0 (the default), which performs no tracing; the highest level is 4, which performs maximum tracing.

Category	Value
Applies to	Portal driver and other (non-file system) access
Required/Optional	Optional
Values	integer (0 to 4)
Default	0

The higher the number you set, the more information is written to the error log file.

Caution: Although setting this parameter to a high number is useful for debugging purposes, it can use a large amount of disk space and can slow the performance of your system.

ORAUser

Specifies the database user (schema) to use when connecting to the service specified by the `ORAService` parameter.

This user must have the following privileges:

- `CONNECT`
- `RESOURCE`
- `CREATE TABLESPACE`
- `DROP TABLESPACE`
- `CREATE ANY TRIGGER`

Category	Value
Applies to	Portal driver and other (non-file system) access

Category	Value
Required/Optional	Required
Values	(character string)
Default	(none)

DAV Directives

This section describes the following DAV directives that you can set in the `httpd.conf` file:

- DAVDepthInfinity
- DAVLockDB
- DAVMinTimeout
- DAVOraNLS
- DAVOraReadOnly
- DAVOraWebCacheReadOnly
- Limit
- LimitExcept
- LimitXMLRequestBody

Some of the material in this section is taken or adapted from material written by Greg Stein (`gstein@lyra.org`) and available at the following URL:

`http://www.webdav.org/mod_dav/install.html`

DAVDepthInfinity

A `PROPFIND` request with a `Depth: Infinity` header can impose a large burden on the server. These kinds of requests could "walk" the entire repository, returning information about each resource found. `mod_dav` builds the response in memory, so these types of requests can consume a lot of memory. (The memory is released at the end of the request, but the peak memory usage can be high.)

To prevent these types of requests, the `DAVDepthInfinity` directive is provided. It is a simple on/off directive, which can be used on a per-server, per-directory or location basis. If the value is set to `Off`, these types of requests are not allowed. A value of `On` (that is, allowing depth infinity requests) makes it easier for denial of service attacks to occur. However, some clients, such as `sitecopy`, require a `DAVDepthInfinity` value of `On`.

Note: The WebDAV Working Group has stated that it is acceptable for DAV servers to refuse these kinds of requests. Properly written client software should not issue such requests.

DAVLockDB

Creates a DAV lock database. To create the DAV lock database, add a `DAVLockDB` directive at the top level of the configuration file (that is, outside a `<Directory>` or `<Location>` container directive). The `DAVLockDB` directive should specify the name

of a file that mod_dav creates. The directory in which the file is to be created must exist, and Oracle HTTP Server process must have write permission to it.

> **Note:** The directory should not be on an NFS-mounted partition. mod_dav uses flock/fcntl to manage access to the database. Some operating systems cannot use these operations on an NFS-mounted partition.

In Example 9–9, the DAV lock database is stored in the *ORACLE_HOME*/Apache/var directory, which must be writable by Oracle HTTP Server process. When mod_dav creates the file, it is named DAVLock. (Actually, mod_dav creates one or more files using this file name plus an extension).

Example 9–9 DAVLockDB Directive

```
DAVLockDB ORACLE_HOME/Apache/var/DAVLock
```

The DAVLockDB directive can appear outside any container or within a <VirtualHost> specification. It only needs to appear once, and a file extension should not be supplied.

DAVMinTimeout

Specifies the minimum lifetime of a lock, in seconds. If a client requests a lock timeout less than the DAVMinTimeout value, then the DAVMinTimeout value is used and returned instead. For example, Microsoft's Web Folders defaults to a lock timeout of 2 minutes (120 seconds); however, you might decide to specify 10 minutes (600 seconds) instead, to reduce network traffic and the chance that the client might lose a lock due to network latency.

The DAVMinTimeout directive is optional, and may be used on a per-server or per-directory or location basis. The DAVMinTimeout directive takes a single positive integer. Because this value represents a minimum allowed, setting it to zero (0) disables this feature. The default value for DAVMinTimeout is zero.

DAVOraNLS

Provides globalization support for access to the local file systems. This directive specifies whether or not the file names in the file system need to go through conversion using the NLS_LANG setting. A value of Off, the default, means that no conversion is needed. A value of On means that the character set for the file system provides for conversion of all possible characters in client requests.

> **See Also:** "Globalization Support Considerations with OraDAV" on page 9-21

DAVOraReadOnly

Specifies whether or not WebDAV should be used in a read-only mode by WebDAV clients. A value of Off, the default, means that WebDAV clients function normally. A value of On prevents WebDAV clients from performing write operations while using WebDAV; however, it does allow read-only activity by Web browsers and WebDAV clients.

> **See Also:** "DAVOraWebCacheReadOnly" on page 9-17

DAVOraWebCacheReadOnly

Specifies whether or not OracleAS Web Cache should be used in a read-only mode by WebDAV clients. A value of `Off`, the default, means that OracleAS Web Cache functions normally. A value of `On` prevents WebDAV clients from performing write operations while using OracleAS Web Cache; however, it does allow read-only activity by Web browsers and WebDAV clients.

See Also:

- "Using Oracle Application Server Web Cache for Browsing Activities" on page 9-20
- "DAVOraReadOnly" directive on page 9-16

Limit

The `DAV` and `DAVLockDB` directives are the only two configuration changes necessary to operate a DAV server. However, it is usually best to secure the site to be writable only by specific authorized users. This requires the use of the `<Limit>` directive.

The configuration in Example 9–10 allows only authorized users to manipulate the site. However, it does allow them a bit more freedom than you may like. In particular, they may be able to place an `.htaccess` file into the target directory, altering your server configuration. The server may have already been configured to not read `.htaccess` files, but it is best to make sure. Also, you may want to disallow other options within the DAV-enabled directory, such as CGI, symbolic links or server-side includes.

Example 9–10 Securing a Site by Using the <Limit> Directive

```
<Location /mypages>
  DAV On
  <Limit PUT POST DELETE PROPFIND PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
    Require user greg
  </Limit>
</Location>
```

Example 9–11 shows a modified configuration with the additional restrictions placed on it through the addition of `AllowOverride None` and `Options None`:

Example 9–11 Securing Site by Using Additional Restrictions

```
<Location /mypages>
  DAV On
  AllowOverride None
  Options None
  <Limit PUT POST DELETE PROPFIND PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
    Require user greg
  </Limit>
</Location>
<Location /mypages>
  DAV On
  AllowOverride None
  Options None
  <Limit PUT POST DELETE PROPFIND PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
    Require user greg
  </Limit>
</Location>
```

LimitExcept

Rather than using the <Limit> directive and specifying an exhaustive list of HTTP methods to secure, it is also possible to use the <LimitExcept> directive, as shown in Example 9–12. This directive applies the access restrictions to all methods except for the methods listed.

Example 9–12 Securing a Site Using the <LimitExcept> Directive

```
<Location /mypages>
  DAV On
  AllowOverride None
  Options None
  <LimitExcept GET HEAD OPTIONS>
    require user webadmin
  </LimitExcept>
</Location>
```

Choosing to use one or the other is a matter of preference. The <Limit> directive is precise and explicit, but the <LimitExcept> directive automatically restricts methods that are added in the future.

LimitXMLRequestBody

mod_dav parses XML request bodies into memory. One technique used in denial of service attacks is to send a large request body to a mod_dav server. Oracle HTTP Server defines a directive named LimitRequestBody, which limits all methods' request bodies. Unfortunately, this is not an effective mechanism for a mod_dav server because large PUT operations should be allowed.

To limit just the methods that have an XML request body, mod_dav provides the LimitXMLRequestBody directive. The default for this value is a compile-time constant, which is set to one million (1,000,000) bytes in the standard distribution. Setting the value to zero (0) disables the size limit.

LimitXMLRequestBody may be set on a per-server or a per-directory or location basis, and takes a single non-negative integer argument.

WebDAV Security Considerations

Because WebDAV enables read-write capabilities, users on the Internet can write to your Web site or to an Oracle database. A major concern is preventing users from placing an inappropriate file (a "Trojan horse") that can execute on the Web server system. If the WebDAV configuration and authorization is not set up properly, an inappropriate file from the file system can be executed. This problem does not apply to content from an Oracle database, because such content cannot execute in the middle tier.

The HTTP protocol issues GET requests both to static and executable files, without differentiation. Oracle HTTP Server executes files based on their location or extension. For example, a shell script (which typically has no file extension) is executed if it is in the cgi-bin directory, but is retrieved as a static text file if it is in the htdocs directory. On the other hand, a JavaServer Pages (JSP) file, which has a .jsp extension, will normally be executed regardless of its location. However, by default, mod_oradav prevents a WebDAV-enabled directory from executing a .jsp or .sqljsp file. For a file with one of these extensions, mod_oradav reads the content directly, bypassing any Oracle HTTP Server logic that attempts to execute the file. Files with these extensions are retrieved as having the text/plain MIME type and can be

edited. You can add to the list of file types that are never to be executed and always retrieved as `text/plain` by using the ORAGetSource parameter.

One way to limit execution of files is to use the Apache `ForceType` directive in a `<Location>` container directive. This forces all content under a location to be retrieved as `text/plain`. However, this simple and sweeping approach may not be what you want in many cases, wherein you want the standard behavior associated with the actual MIME type, for example, for GIF files, to be used.

To decide how to handle these security issues with content on file systems, you should determine what kinds of WebDAV users will have access to the content. WebDAV users typically fall into two categories: Web authors who want to collaborate and manage a Web site, and end users who want to use WebDAV as a public storage area. Because end users should never be able to upload and execute a file, you may want to specify many file extensions with the ORAGetSource parameter or to use the `ForceType` directive for end users.

Be sure to apply the standard Basic or Digest authentication and authorization mechanisms supported by Oracle HTTP Server. Generally, you do this for the default location, such as `dav_public`, in the supplied moddav.conf file. This restricts who can use your system for remote storage, preventing unauthorized users from filling up your disks. You should always apply Oracle HTTP Server authentication and authorization to authors of the Web site.

You should also provide both an execution context and an editing context, so that Web authors, after being properly authenticated and authorized, can edit a JSP file or other executable file and then see how it executes. To do this, create an alias for the directory associated with the execution context, and then DAV-enable the aliased location. For example, assume that you want to be able to execute a script if the URL specifies the `cgi-bin` directory (for example, `http://www.acme.com/cgi-bin/printenv`), but to edit the script if the URL specifies an alias named `edit-cgi-bin` (for example, `http://www.acme.com/edit-cgi-bin/printenv`). In Example 9–13, the configuration file entries achieve this goal, setting up `edit-cgi-bin` as an editing context for content in the `cgi-bin` directory:

Example 9–13 Editing Context

```
Alias /edit-cgi-bin /usr/local/apache/cgi-bin
<Location /edit-cgi-bin>
  DAV On
  ForceType text/plain
</Location>
```

OraDAV Performance Considerations

This section provides information that can help you optimize the performance of various kinds of operations. It contains the following topics:

- Using Disk Caching with OraDAV
- Bypassing Oracle Application Server Web Cache for WebDAV Activities
- Using Oracle Application Server Web Cache for Browsing Activities

Using Disk Caching with OraDAV

Oracle Application Server can use local file system disk caching with data that is retrieved from an Oracle database. Disk caching is designed to improve the performance of HTTP `GET` operations on frequently accessed database data. When

data is requested from the database, it is retrieved and is stored in a disk cache on the local file system. If a subsequent request is for the same data and if the data is still in the disk cache, Oracle Application Server checks to see if the data has changed in the database (by examining the `etag` value). If the data has not changed, it is retrieved from the cache, which is more efficient than retrieving a substantial amount of data from the database.

The performance benefit from disk caching is greatest with medium to large-size files (roughly 50 KB and larger). However, with smaller files, the performance benefit is less, and with very small files the performance can be worse with disk caching than without disk caching. For example, if the file `myfile.dat` is requested and if the file size is only 24 bytes, the time required for copying the file from the database to the local system is very small compared to the time required for accessing the database to check if the file has changed. If disk caching is not used, there is no check of the database to see if the file has changed, and the file is copied from the database in all cases.

You can set the following OraDAV parameters to control disk caching for OraDAV operations:

- ORACacheDirectory
- ORACacheTotalSize
- ORACacheMaxResourceSize
- ORACachePrunePercent

If you specify `ORACacheDirectory`, disk caching for OraDAV operations is enabled. In this case, you must also specify a value for `ORACacheTotalSize`, and you can specify values for `ORACacheMaxResourceSize` and `ORACachePrunePercent` parameters. If you do not specify `ORACacheDirectory`, disk caching for OraDAV operations is not enabled, and other disk cache-related parameters are not relevant.

See Also: "OraDAV Configuration Parameters" on page 9-5

Bypassing Oracle Application Server Web Cache for WebDAV Activities

Oracle Application Server Web Cache enhances performance for most Web activity, which involves client read-only operations of data on the Web server system. However, OracleAS Web Cache does not cache OraDAV operations, which are designed for read/write capability. Thus, for better performance, WebDAV clients can connect directly to Oracle HTTP Server.

To bypass OracleAS Web Cache for WebDAV clients, you can use port 7778, which is the standard port for Oracle HTTP Server. If you do this, WebDAV clients connect directly to the Web server, resulting in better performance than if OracleAS Web Cache was used.

Using Oracle Application Server Web Cache for Browsing Activities

If your WebDAV clients always bypass OracleAS Web Cache, you may want to tune OracleAS Web Cache for read-only clients such as Web browsers. To do so, add the `DAVOraWebCacheReadOnly On` setting for an OraDAV-enabled location in the `httpd.conf` file, as shown in Example 9–14.

Example 9–14 Using OracleAS Web Cache for Browsing Activities

```
<Location /dav_public>
  DAV On
```

```
  DAVOraWebCacheReadOnly On
</Location>
```

This setting prevents WebDAV clients from performing write operations while using OracleAS Web Cache; however, it does allow read-only activity by Web browsers and WebDAV clients.

See Also: "DAVOraWebCacheReadOnly" on page 9-17

mod_oradav Usage Notes

This section contains usage notes relating to `mod_oradav`. Some of the information is taken or adapted from material written by Greg Stein (`gstein@lyra.org`) and available at the following URL:

`http://www.webdav.org/mod_dav/install.html`

Mapping Containers Under the Root Location

Note the following when mapping containers under the root location:

- Do not map the root itself. That is, do not specify `<Location />`.
- Do not map a container as a subelement in the hierarchy to another container. For example, do not specify the following two containers: `<Location /project1>` and `<Location /project1/project2>`. However, it is acceptable to specify `<Location /project1>` and `<Location /project2>`.
- Do not create any symbolic links to the container or any location under the container in the hierarchy.

Globalization Support Considerations with OraDAV

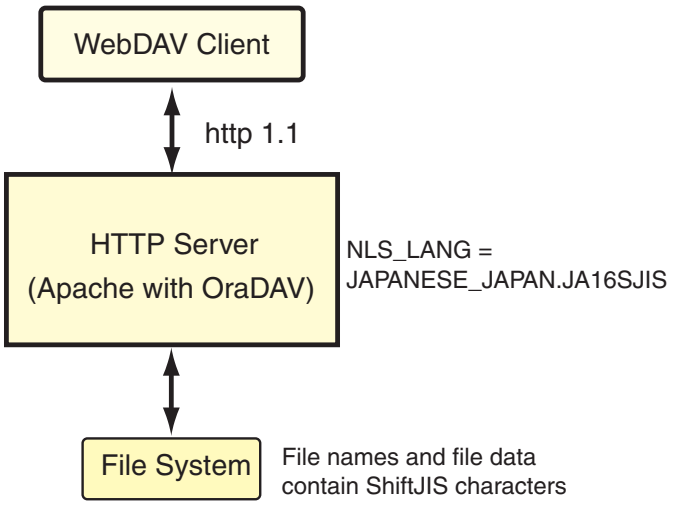
For access to database data, the character set used for client requests, such as in URLs and file names, must be compatible with the character set used for the database. Specifically, if the character set for the database is not the same as for the client requests, the character set for the database must provide for conversion of all possible characters in client requests (and thus must be a superset of the character set for client requests). That is, the character set for the database must not cause replacement characters during the conversion.

When you start Oracle HTTP Server, the `NLS_LANG` environment variable must reflect the character set for client requests. For example, if file names and URLs contain Kanji characters, you can specify `NLS_LANG=JAPANESE_JAPAN.JA16SJIS` (for ShiftJIS characters). In this case, the database character set must be one that accommodates `SJIS` characters, for example, `UTF8`.

For access to the local file system, as opposed to database access, the character set for the file system must be the same as, or compatible with, the character set for URLs embedded in client requests. The character set for the file system must provide for conversion of all possible characters in client requests. The `NLS_LANG` parameter value must represent the character set of both the client and the OraDAV server. You must also specify a value of `On` for the parameter `DAVOraNLS`.

For example, assume that you are using Web folders on a system where the files have ShiftJIS characters and that the file system under `dav_public` is represented by the operating system in the `JAPANESE_JAPAN.JA16SJIS` character sets shown in Figure 9–2.

Figure 9–2 OraDAV Access to File System with ShiftJIS Characters



In this case, you must do the following:

1. Set the `NLS_LANG` value to `JAPANESE_JAPAN.JA16SJIS`.
2. Include the following in the `httpd.conf` file:

```
<Location /dav_public>
  DAV On
  DAVOraNLS On
</Location>
```

Note: If you use Microsoft Internet Explorer with OraDAV and a multibyte character set, you must disable (uncheck) the Internet option (Internet Options, Advanced tab) `Always send URLs as UTF-8`. (By default, this option is enabled.) The requirement to disable this option applies to both database access and file system access.

PROPFIND Security

In the example configurations in the preceding sections on the `<Limit>` and `<LimitExcept>` directives, the `PROPFIND` method was limited, even though it is read-only. This is because the `PROPFIND` method can be used to list all the files in the DAV-enabled directory. For security reasons, it is probably best to protect the list of files from general read access.

An alternative is to limit the `PROPFIND` to a group of people, a set of domains, or a set of hosts, while the methods that modify content are limited to just a few authors. This scenario allows, for example, your company's employees to browse the files on the server, yet only a few people can change them. Anonymous (non-authenticated) visitors cannot browse or modify.

Finally, you can simply omit `PROPFIND` from the limits if your Web server is intended as a general, read-only repository of files. This allows anybody to arbitrarily browse the directories and then to fetch the files.

10

Managing Security

This chapter contains an overview of Oracle HTTP Server security features, and provides configuration information for setting up a secure Web site.

Topics discussed are:

- About Oracle HTTP Server Security
- Classes of Users and Their Privileges
- Resources Protected
- Authentication and Authorization Enforcement
- Understanding Port Tunneling
- Leveraging Oracle Identity Management Infrastructure

> **See Also:** For additional information about security, refer to the following documents:
>
> - The *Oracle Application Server Security Guide* provides an overview of Oracle Application Server security and its core functionality.
> - The *Oracle Identity Management Concepts and Deployment Planning Guide* provides guidance for administrators of the Oracle security infrastructure.

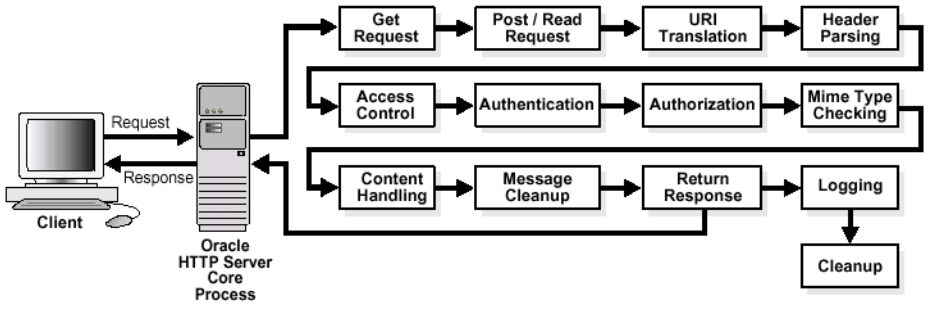
About Oracle HTTP Server Security

Security can be organized into the three categories of authentication, authorization, and confidentiality. Oracle HTTP Server provides support for all three of these categories. It is based on the Apache Web server, and its security infrastructure is primarily provided by the Apache modules, mod_auth and mod_access, and the Oracle modules, mod_ossl and mod_osso. `mod_auth` provides authentication based on user name and password pairs, `mod_access` controls access to the server based on the characteristics of a request, such as hostname or IP address, `mod_ossl` provides confidentiality and authentication with X.509 client certificates over SSL, and `mod_osso` enables **single sign-on** authentication for Web applications.

Based on the Apache model, Oracle HTTP Server provides access control, authentication, and authorization methods that can be configured with access control directives in the `httpd.conf` file. When URL requests arrive at Oracle HTTP Server, they are processed in a sequence of steps determined by server defaults and configuration parameters. The steps for handling URL requests are implemented through a module or plug-in architecture that is common to many Web listeners.

Figure 10–1 shows how URL requests are handled by the server. Each step in this process is handled by a server module depending on how the server is configured. For example, if basic authentication is used, then the steps labeled "Authentication" and "Authorization" in Figure 10–1 represent the processing of the `mod_auth` module.

Figure 10–1 Steps for Handling URL Requests in Oracle HTTP Server



Classes of Users and Their Privileges

Oracle HTTP Server authorizes and authenticates users before allowing them to access, or modify resources on the server. The following are three classes of users that access the server using Oracle HTTP Server, and their privileges:

- Users that access the server without providing any authentication. They have access to unprotected resources only.
- Users that have been authenticated and potentially authorized by modules within Oracle HTTP Server. This includes users authenticated by `mod_auth` and `mod_ossl`. Such users have access to URLs defined in `http.conf` file.

  See Also: "Authentication and Authorization Enforcement" on page 10-3

- Users that have been authenticated through `mod_osso` and Single Sign-On server. These users have access to resources allowed by Single Sign-On.

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

Resources Protected

Oracle HTTP Server is configured to protect resources such as:

- Static content such as static HTML pages, graphics interchange format, `.gif`, files, and other static files that Oracle HTTP Server provides directly.
- CGI/FastCGI scripts, simple scripts or programs that Oracle HTTP Server invokes directly.
- Content generated by modules within Oracle HTTP Server. Modules such as mod_perl, mod_dms generate responses that are returned to the client.
- Oracle Application Server components that exist behind Oracle HTTP Server, including servlets and JSPs running with OC4J that are accessed through mod_

oc4j. Oracle HTTP Server forms the first line of authentication and authorization for these components, although further authentication may occur at the component level.

Authentication and Authorization Enforcement

Oracle HTTP Server provides user authentication and authorization at two stages:

- Host-based Access Control (stage one): This is based on the details of the incoming HTTP request and its headers, such as IP addresses or host names.
- User Authentication and Authorization (stage two): This is based on different criteria depending on the HTTP server configuration. The server can be configured to authenticate users with user name and password pairs that are checked against a list of known users and passwords. You can also configure the server to use single sign-on authentication for Web applications or X.509 client certificates over SSL.

Host-based Access Control

Early in the request processing cycle, access control is applied, which can inhibit further processing based on the host name, IP address, or other characteristics such as browser type. You use the `deny`, `allow`, and `order` directives to set this type of access control. These restrictions are configured with Oracle HTTP Server configuration directives and can be based on particular files, directories, or URL formats using the <Files>, <Directory>, and <Location> container directives as shown in the Example 10–1:

Example 10–1 Host-based Access Control

```
<Directory /internalonly/>
  order deny, allow
  deny from all
  allow from 192.168.1.* us.oracle.com
</Directory>
```

In Example 10–1, the `order` directive determines the order in which Oracle HTTP Server reads the conditions of the `deny` and `allow` directives. The `deny` directive ensures that all requests are denied access. Then, using the `allow` directive, requests originating from any IP address in the 192.168.1.* range, or with the domain name `us.oracle.com` are allowed access to files in the directory `/internalonly/`. It is common practice to specify both `allow` and `deny` in host-based authentication to make the access policy explicit.

If you want to match objects at the file system level, then you must use `<Directory>` or `<Files>`. If you want to match objects at the URL level, then you must use `<Location>`.

Note: Allowing or restricting access based on a host name for Internet access is not considered a very good method of providing security, because host names are easy to spoof. While the same is true of IP addresses, sabotage is more difficult. However, setting access control with intranet IP address ranges is reasonable because the same risks do not apply. This assumes that your firewalls have been properly configured.

Access Control for Virtual Hosts

To set up access control for virtual hosts, place the `AccessConfig` directive inside a virtual host container in the server configuration file, `httpd.conf`. When used in a virtual host container, the `AccessConfig` directive specifies an access control policy contained in a file. Example 10–2 shows an excerpt from an `httpd.conf` file which provides the syntax for using `AccessConfig` this way:

Example 10–2 Using AccessConfig to Set Up Access Control

```
...
<VirtualHost ip_address_of_host.some_domain.com>
  ... virtual host directives ...
  AccessConfig conf/access.conf
</VirtualHost>
```

Using mod_access and mod_setenvif for Host-based Access Control

Using host-based access control schemes, you can control access to restricted areas based on where HTTP requests originate. Oracle HTTP Server uses mod_access and mod_setenvif to perform host-based access control. `mod_access` provides access control based on client hostname, IP address, or other characteristics of the client request, and `mod_setenvif` provides the ability to set environment variables based upon attributes of the request. When you enter configuration directives into the `httpd.conf` file that use these modules, the server fulfills or denies requests based on the address or name of the host, or based on the HTTP request header contents.

You can use host-based access control to protect static HTML pages, applications, or components.

Oracle HTTP Server supports four host-based access control schemes:

- Controlling Access by IP Address
- Controlling Access by Domain Name
- Controlling Access by Network or Netmask
- Controlling Access with Environment Variables

All of these allow you to specify the machines from which access to protected areas is granted or denied. Your decision to choose one or more of the host-based access control schemes is determined by which scheme most efficiently protects your restricted content and applications, or which scheme is easiest to maintain.

Controlling Access by IP Address Controlling access with IP addresses is a preferred method of host-based access control. It does not require DNS lookups that consume time, system resources, and make your server vulnerable to DNS spoofing attacks.

Example 10–3 Controlling Access by IP Address

```
<Directory /secure_only/>
  order deny,allow
  deny from all
  allow from 207.175.42.*
</Directory>
```

In Example 10–3, requests originating from all IP addresses except 207.175.42.* range are denied access to the `/secure_only/` directory.

Controlling Access by Domain Name Domain name-based access control can be used with IP address-based access control to solve the problem of IP addresses changing without warning. When you combine these methods, if an IP address changes, then the secure areas of your site are still protected because the domain names you want to keep out will still be denied access.

To combine domain name-based with IP address-based access control, use the syntax shown in Example 10–4:

Example 10–4 controlling Access by Domain Name

```
<Directory /co_backgr/>
  order allow,deny
  allow from all
  # 141.217.24.* is the IP for malicious.cracker.com
  deny from malicious.cracker.com 141.217.24.*
</Directory>
```

In Example 10–4, all requests for directory `/co_backgr/` are accepted except those that originate from the domain name `malicious.cracker.com` or the IP address 141.217.24.* range. Although this is not a fool proof precaution against domain name or IP address spoofing, it protects your site from `malicious.cracker.com` even if they change their IP address.

Controlling Access by Network or Netmask You can control access based on subsets of networks, specified by IP address. The syntax is shown in Example 10–5:

Example 10–5 Controlling Access by Network or Netmask

```
<Directory /payroll/>
  order deny,allow
  deny from all
  allow from 10.1.0.0/255.255.0.0
</Directory>
```

In Example 10–5, access is allowed from a network/netmask pair. A netmask shows how an IP address is to be divided into network, subnet, and host identifiers. Netmasks enable you to refer to only the host ID portion of an IP address.

The netmask in Example 10–5, 255.255.0.0, is the default netmask setting for a Class B address. The binary ones (decimal 255) mask the network ID and the binary zeroes (decimal 0) retain the host ID of a given IP address.

Controlling Access with Environment Variables You can use arbitrary environment variables for access control, instead of using IP addresses or domain names. Use `BrowserMatch` and `SetEnvIf` directives for this type of access control.

Note: Typically, `BrowserMatch` and `SetEnvIf` are not used to implement security policies. Instead they are used to provide different handling of requests based on browser types and versions.

Use `BrowserMatch` when you want to base access on the type of browser used to send a request. For instance, if you want to allow access only to requests that come from a Netscape browser, then use the syntax shown in Example 10–6:

Example 10–6 Controlling Access with Environment Variables

```
BrowserMatch ^Mozilla netscape_browser
<Directory /mozilla-area/>
  order deny,allow
  deny from all
  allow from env=netscape_browser
</Directory>
```

Use `SetEnvIf` when you want to base access on header information contained in the HTTP request. For instance, if you want to deny access from any browsers using HTTP version 1.0 or earlier, then use the syntax shown in Example 10–7:

Example 10–7 Controlling Access with SetEnv

```
SetEnvIf Request_Protocol ^HTTP/1.1 http_11_ok
<Directory /http1.1only/>
  order deny,allow
  deny from all
  allow from env=http_11_ok
</Directory>
```

See Also: "Scope of Directives" on page 2-4

User Authentication and Authorization

Basic authentication prompts for a user name and password before serving an HTTP request. When a browser requests a page from a protected area, Oracle HTTP Server responds with an unauthorized message (status code 401) containing a `WWW-Authenticate:` header and the name of the realm configured by the configuration directive, `AuthName`. When the browser receives this response, it prompts for a user name and password. After the user enters a user name and password combination, the browser sends this information back to the server in an Authorization header. In the authorization header message, the user name and password are encoded as a base 64 encoded string.

User authorization involves checking the authenticated user against an access control list that is associated with a specific server resource such as a file or directory. To configure user authorization, place the `require` directive in the `httpd.conf` file, usually within a virtual host container. User authorization is commonly used in combination with user authentication. After the server has authenticated a user's user name and password, then the server compares the user to an access control list associated with the requested server resource. If Oracle HTTP Server finds the user or the user's group on the list, then the resource is made available to that user.

Using mod_auth to Authenticate Users

User authentication is based on user names and passwords that are checked against a list of known users and passwords. These user name and password pairs may be stored in a variety of forms, such as a text file, database, or directory service. Then configuration directives are used in `httpd.conf` to configure this type of user authentication on the server. `mod_auth` uses the `AuthUserFile` directive to set up basic authentication. It supports only files.

Any authentication scheme that you devise requires that you use a combination of the configuration directives listed in Table 10–1.

Table 10–1 Directives Descriptions

Directive Name	Description
`AuthName`	Defines the name of the realm in which the user names and passwords are valid. Use quotation marks if the name includes spaces.
`AuthType`	Specifies the authentication type. Most authentication modules use basic authentication, which transmits user names and passwords in clear text. This is not recommended.
`AuthUserFile`	Specifies the path to a file that contains user names and passwords.
`AuthGroupFile`	Specifies the path to a file that contains group names and their members.

Using mod_osso to Authenticate Users

`mod_osso` enables single-sign on for Oracle HTTP Server. `mod_osso` examines incoming requests and determines whether the resource requested is protected, and if so, retrieves the Oracle HTTP Server cookie for the user.

Through `mod_osso`, Oracle HTTP Server becomes a single sign-on (SSO) partner application enabled to use SSO to authenticate users and obtain their identity using Oracle Application Server Single Sign-On, and to make user identities available to Web applications as an Apache header variable.

Using `mod_osso`, Web applications can register URLs that require SSO authentication. When Oracle HTTP Server receives URL requests, `mod_osso` detects which requests require SSO authentication and redirects them to the SSO server. Once SSO server authenticates the users, it passes the user's authenticated identity back to `mod_osso` in a secure token, or cookie. `mod_osso` retrieves the user's identity from the cookie and propagates the user's identity information to applications running in Oracle HTTP Server instance. `mod_osso` can propagate the user's identity information to applications running in CGI, and those running in OC4J, and it can also authenticate users for access to static files.

See Also:

- *Oracle Application Server Single Sign-On Administrator's Guide*
- "Leveraging Oracle Identity Management Infrastructure" on page 10-15

Using mod_ossl to Authenticate Users

`mod_ossl` is a plug-in to Oracle HTTP Server that enables the server to use SSL. `mod_ossl` replaces `mod_ssl` in the Oracle HTTP Server distribution. Oracle no longer supports `mod_ssl`.

See Also: "Enabling SSL for Oracle HTTP Server" on page 11-1

Understanding Port Tunneling

Port tunneling allows all communication between Oracle HTTP Server and OC4J to happen on a single, or a small number of ports. Previously, the firewall configuration had to include port information for many ports to handle communication between Oracle HTTP Server and multiple OC4J instances. Using port tunneling, a daemon called `iaspt` routes requests to the appropriate OC4J instances. Only one, or a small number of ports have to be opened through the firewall regardless of the number of OC4J instances involved, thereby offering a higher degree of security for the communication between Oracle HTTP Server and OC4J.

To enable this, a **de-militarized zone** environment is provided where a firewall exists typically between the client and the Oracle HTTP Server, and another that exists between Oracle HTTP Server and OC4J. In this configuration, Oracle HTTP Server exists in the DMZ bracketed by the two firewalls. OC4J, and other business logic components, exist behind both firewalls in the intranet. To ensure the highest degree of security, all communication transmitted between machines is encrypted using SSL. Port tunneling provides the framework to support this level of security in a flexible, manageable manner, which enhances performance.

The suggested port range is 7501-7599, the default being 7501, but you can select a port of your choice.

Figure 10–2 Port Tunneling

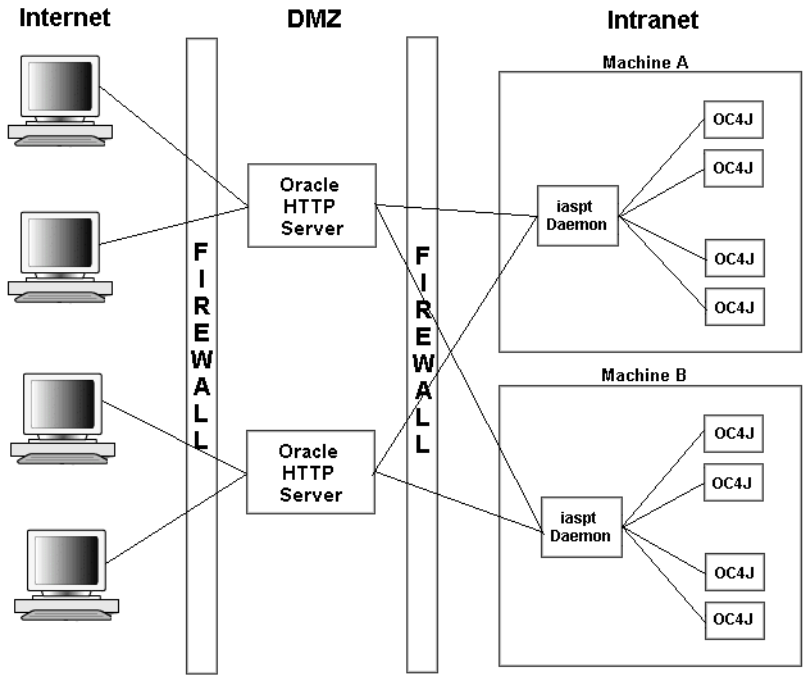


Figure 10–2 shows an Oracle Application Server configuration using port tunneling. The `iaspt` daemon, a standalone component, acts as a communication concentrator for connections between Oracle HTTP Server and the Java Virtual Machine (JVM), which contains OC4J. Oracle HTTP Server does not connect directly to OC4J. Instead, it connects to the `iaspt` daemon which then dispatches communication on to OC4J. By doing this concentration of connections, only one port is opened per port tunneling on the internal firewall, instead of one port per OC4J instance.

The communication between Oracle HTTP Server and the `iaspt` daemon is encrypted using SSL. Authentication is enabled when these connections are established using SSL Client Certificates. These connections are persistent, and are maintained for a reasonable time depending on connection resources. The AJP 1.3 protocol, modified to include routing information that indicates which servlet engine a request is to be routed to, is used.

Port tunneling supports connections between Oracle HTTP Server and OC4J. To support OC4J, Oracle HTTP Server module `mod_oc4j` is modified to use SSL

encrypted communication and to route requests through the port tunneling processes. Port tunneling supports static configurations.

There must be at least one `iaspt` daemon per machine. More than one `iaspt` daemon can be run for higher **availability**. Oracle HTTP Server supports round robin partitioning of requests across `iaspt` daemons, and supports application partitioning. Oracle HTTP Server also supports automatic **failover** of requests which cannot be sent to a given `iaspt` daemon.

Configuring Port Tunneling

Perform the following three tasks to configure port tunneling:

- Task 1: Configure opmn.xml
- Task 2: Configure iaspt.conf
- Task 3: Configure mod_oc4j.conf

Task 1: Configure opmn.xml

Perform the following steps to start one or more `iaspt` daemons:

1. By default, there is an `opmn.xml` entry for `iaspt` that is disabled. Enable `iaspt` by editing `opmn.xml` and changing `status="disable"` to `status="enable"`.
2. Optionally, you may also change the TCP/IP ports used by the `iaspt` daemon by changing the port "range" and the number of the `iaspt` daemon processes by changing "numprocs".

   The following is a complete example configuration for the `iaspt` daemon. It contains all possible configuration elements/attributes that can be used with this component.

   ```
   <module path="/ORACLE_HOME/opmn/lib/libopmniaspt">
     <module-id id="IASPT" />
   </module>
   <ias-component id="IASPT" status="enabled" id-matching="false">
     <process-type id="IASPT" module-id="IASPT">
       <port id="ajp" range="6701-6703"/>
       <process-set id="IASPT" restart-on-death="true" numprocs="3"/>
     </process-type>
   </ias-component>
   ```

3. Run the following command to direct the opmn daemon to reload its configuration file:

   ```
   opmnctl reload
   ```

Task 2: Configure iaspt.conf

Perform the following steps to configure `iaspt.conf` to specify an SSL wallet for the `iaspt` daemon(s) to use:

1. Communication between `mod_oc4j` and `iaspt` is always encrypted, therefore an SSL wallet file must be configured for the `iaspt` daemon(s). By default, this wallet is the same as the Oracle HTTP Server wallet. You may change the default by editing the following values in `iaspt.conf`:

   ```
   wallet-file=<path to wallet file>
   wallet-password=<password>
   ```

See Also:

- "wallet-file" on page 10-13
- "wallet-password" on page 10-14

2. Start the iaspt daemon(s) using the following command:

```
opmnctl startall
```

Task 3: Configure mod_oc4j.conf

Perform the following steps to configure `mod_oc4j.conf` to route requests using `iaspt`:

1. Enable port tunneling by adding the following line in `mod_oc4j.conf`:

```
Oc4jiASPTActive on
```

See Also: "Oc4jiASPTActive" on page 10-12

2. Specify an SSL wallet and wallet password for `mod_oc4j.conf` by adding the following two lines in `mod_oc4j.conf`:

```
Oc4jiASPTWalletFile <path to wallet file>
Oc4jiASPTWalletPassword <password of wallet>
```

This wallet may be the same as used by Oracle HTTP Server and/or `iaspt`.

See Also:

- "Oc4jiASPTWalletFile" on page 10-12
- "Oc4jiASPTWalletPassword" on page 10-13
- *Oracle Application Server Administrator's Guide* for information on Oracle Wallet Manager.

3. Specify the host and port addresses of the `iaspt` daemons. For example, add the following line to `mod_oc4j.conf`:

```
Oc4jiASPTProcess myhost.us.oracle.com:6701
```

You may add as many `Oc4jiASPTProcess` lines as you have `iaspt` daemons. The host and port addresses must match those of your configured `iaspt` daemons. For example, to route requests to the three `iaspt` daemons configured in the example in step 2 of "Task 1: Configure opmn.xml" on page 10-9, add the following three lines:

```
Oc4jiASPTProcess myhost.us.oracle.com:6701
Oc4jiASPTProcess myhost.us.oracle.com:6702
Oc4jiASPTProcess myhost.us.oracle.com:6703
```

See Also: "Oc4jiASPTProcess" on page 10-12

4. Restart Oracle HTTP Server for the changes to take effect, using the following command:

- UNIX: *ORACLE_HOME*`/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`

- Windows: *ORACLE_HOME*\opmn\bin> opmnctl [verbose] restartproc ias-component=HTTP_Server

Configuring SSL for Port Tunneling

This section contains information about configuring SSL between iaspt and OC4J

By default, the iaspt daemons and the OC4J processes communicate with unencrypted data. Perform the following steps to configure SSL communication between these processes:

1. In iaspt.conf, change the value "destination-ssl" from "false" to "true".
2. To configure the OC4J process to use SSL, refer to the *Oracle Application Server Containers for J2EE Security Guide*.

Port Tunneling Configuration Reference

This section contains information about the following configuration files and their parameters:

- opmn.xml
- mod_oc4j.conf
- iaspt.conf

opmn.xml

Describes the process that OPMN manages within an Oracle Application Server installation.

See Also: "opmn.xml" on page E-3

As part of port tunneling, an **entry** that describes the iaspt daemon process to be started should exist in OPMN. This entry describes the following:

- number of iaspt daemon processes to start.
- ports that these processes can use.

See Also: iaspt.conf on page 10-13

An out of the box Oracle Application Server installation contains an iaspt component in opmn.xml, but it is disabled by default.

mod_oc4j.conf

Configures mod_oc4j with Oracle HTTP Server.

See Also: mod_oc4j.conf on page E-3

For port tunneling, you need to add directives that specify the following:

- whether port tunneling should be used
- static location for an iaspt daemon process
- location of SSL certificates to be used in establishing connections with the iaspt daemon processes.

See Also: iaspt.conf on page 10-13

By default, `mod_oc4j` communicates directly to OC4J. For port tunneling process, `mod_oc4j` should communicate to OC4J through the `iaspt` daemon.

The following directives used to connect `mod_oc4j` to the `iaspt` daemon:

- Oc4jiASPTActive
- Oc4jiASPTProcess

Oc4jiASPTActive Indicates whether `mod_oc4j` needs to consider port tunneling when routing requests. This should not be configured to "On" if Oc4jEnableSSL is configured to "On". To enable port tunneling process, set this directive to "On".

Category	Value
Parameter Name	`Oc4jiASPTActive`
Parameter Type	string
Valid Values	`On/Off`
Default Value	`Off`

Oc4jiASPTProcess Describes the listening host and port of a port tunneling process. There can be multiple instances of this directive within a `mod_oc4j.conf` file for multiple port tunneling processes.

The syntax for this directive is `host:port`. The host value should be the hostname of a machine where an `iaspt` daemon is running. The port value should match the port configured in `opmn.xml` for that `iaspt`. Both regular hostname and IP address are allowed for host.

Category	Value
Parameter Name	`Oc4jiASPTProcess`
Parameter Type	string
Valid Values	`host:port` values of the available `iaspt` daemons.
Default Value	N/A
Syntax	`host:port` For example: *`myhost.us.oracle.com`*`:6667`

`mod_oc4j` should use SSL when communicating with the `iaspt` daemon. The following are the directives used to enable SSL:

- Oc4jiASPTWalletFile
- Oc4jiASPTWalletPassword

Oc4jiASPTWalletFile Specifies the location of an Oracle Wallet file that contains SSL certificates that are used for SSL communication with the `iaspt` daemon.

Category	Value
Parameter Name	`Oc4jiASPTWalletFile`
Parameter Type	string

Category	Value
Valid Values	Path to a wallet file that contains the SSL certificate to be used when establishing SSL connections to the iaspt daemon.
Default Value	N/A
Syntax	Valid filename For example: `/foo/bar/myfilename`

Oc4jiASPTWalletPassword Specifies the value of the obfuscated password used for authentication when opening the wallet file. This value is obtained using the utility provided with Oracle Wallet Manager.

Category	Value
Parameter Name	`Oc4jiASPTWalletPassword`
Parameter Type:	string
Valid Values	Password used for authentication when opening the wallet file specified by Oc4jiASPTWalletFile.
Default Value	N/A

> **See Also:** *Oracle Application Server Administrator's Guide* for information on Oracle Wallet Manager.

iaspt.conf

Configures port tunneling.

> **See Also:** "iaspt.conf" on page E-2

It specifies the following information:

- Wallet file and password that should be used
- Log file location and log level
- Port that `iaspt` daemon should listen on (optionally). This port can either be specified in `iaspt.conf`, or can be passed in from `opmn.xml` by specifying a range of ports. By doing so, more than one port tunneling process can use the same `iaspt.conf` file.

The `iaspt.conf` file is a set of name value pairs. The following are the names of the parameters accepted:

- wallet-file
- wallet-password
- log-file
- log-level
- iaspt-port

wallet-file Specifies the location of an Oracle Wallet file that contains SSL certificates that are used for SSL communication with peers.

Category	Value
Parameter Name	`wallet-file`
Parameter Type	string
Valid Values	Path to a wallet file that contains the SSL certificate to be used when establishing SSL connections to other processes.
Default Value	N/A
Syntax	Valid filename For example: `/foo/bar/myfilename`

wallet-password Specifies the value of the password used for authentication when opening the wallet file. This value is obtained using the utility provided with Oracle Wallet Manager.

Category	Value
Parameter Name	`wallet-password`
Parameter Type	string
Valid Values	Password used for authentication when opening the wallet file specified by wallet-file
Default Value	N/A

> **See Also:** *Oracle Application Server Administrator's Guide* for information on Oracle Wallet Manager.

log-file Specifies the path to a log file where `iaspt` daemon logging messages are written to.

Category	Value
Parameter Name	`log-file`
Parameter Type	string
Valid Values	Path to a log file where `iaspt` daemon logging messages are written to.
Default Value	N/A
Syntax	Valid filename For example: `/foo/bar/myfilename`

log-level Specifies the logging level where 9 is the highest and 0 implies no logging.

Category	Value
Parameter Name	`log-level`
Parameter Type	integer
Valid Values	Integer from 0 to 9
Default Value	3

iaspt-port Specifies the port value that the `iaspt` daemon should accept connections on. This is optional.

Category	Value
Parameter Name	`iaspt-port`
Parameter Type	integer
Valid Values	Valid TCP/IP port value
Syntax	Integer For example: 9898
Default Value	N/A

Leveraging Oracle Identity Management Infrastructure

This section discusses how Oracle HTTP Server uses the Oracle Identity Management Infrastructure.

Overview

Oracle Identity Management is an integrated infrastructure that the Oracle Application Server relies on for distributed security. It consists of Oracle Internet Directory, Oracle Directory Integration and Provisioning, Delegated Administrative Service, Oracle Application Server Single Sign-On, and Oracle Certificate Authority.

> **See Also:** *Oracle Identity Management Concepts and Deployment Planning Guide*

Using Oracle Application Server Single Sign-On and mod_osso

Oracle Application Server supports single sign-on (SSO) to Web-based applications through Oracle Application Server Single Sign-On. Oracle Application Server Single Sign-On enables you to log in to Oracle Application Server and gain access to those applications for which you have authorization, without requiring to re-enter a user name and password for each application. It is fully integrated with Oracle Internet Directory, which stores user information. It supports LDAP-based user and password management through Oracle Internet Directory.

`mod_osso`, an Oracle HTTP Server module, enables the transparent use of Oracle Application Server Single Sign-On across all of Oracle Application Server. Through `mod_osso`, Oracle HTTP Server becomes a SSO partner application enabled to use SSO to authenticate users and obtain their identity, and to make user identities available to Web applications as an Apache header variable.

> **See Also:** "Using mod_osso to Authenticate Users" on page 10-7

11

Enabling SSL for Oracle HTTP Server

This chapter contains information about enabling and configuring SSL for Oracle HTTP Server.

Topics discussed are:

Overview

Secure Sockets Layer (SSL) is an encrypted communication protocol that is designed to securely send messages across the Internet. It resides between Oracle HTTP Server on the application layer and the TCP/IP layer, transparently handling encryption and decryption when a secure connection is made by a client.

One common use of SSL is to secure Web HTTP communication between a browser and a Web server. This case does not preclude the use of non-secured HTTP. The secure version is simply HTTP over SSL (named HTTPS). The differences are that HTTPS uses the URL scheme `https://` rather than `http://`, and its default communication port is 4443 on UNIX or 443 on Windows.

`mod_ossl` is a plug-in to Oracle HTTP Server that enables the server to use SSL.

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

Configuring SSL

By default, SSL is disabled when you install Oracle Application Server. Perform the following tasks to enable and configure SSL:

Task 1: Creating a Real Wallet

To configure Oracle HTTP Server for SSL, you need a **wallet** that contains the certificate for the server. Wallets store your credentials, such as certificate requests, certificates, and private keys.

The default wallet that is automatically installed with Oracle HTTP Server is for testing purposes only. A real wallet has to be created for your production server. The default wallet is located in *ORACLE_HOME*/Apache/Apache/conf/ssl.wlt/default. You can either place the new wallet in that location, or change the SSLWallet directive in *ORACLE_HOME*/Apache/Apache/conf/ssl.conf to point to the location of your real wallet.

> **See Also:** *Oracle Application Server Administrator's Guide* for instructions on creating a wallet. It is important that you do the following:
>
> 1. Generate a certificate request: For the Common Name, specify the name or alias of the site you are configuring.
> 2. Set the auto-login feature for your wallet: Make sure that you enable this auto-login feature. The default wallet has this feature disabled.

Task 2: Enabling SSL

Perform the following steps to enable SSL:

1. Open opmn.xml in a text editor.
2. In the <ias-component id="HTTP_Server"> entry, change the start mode from "ssl-disabled" to "ssl-enabled". After modification is made, the entry should look like the following:

   ```
   <data id="start-mode" value="ssl-enabled"/>
   ```

3. Save and close opmn.xml.
4. Update the distributed cluster management database with the change:

   ```
   ORACLE_HOME/dcm/bin/dcmctl updateconfig -ct opmn
   ```

5. Reload OPMN using the following command:

   ```
   opmnctl reload
   ```

6. Stop Oracle HTTP Server using Application Server Control Console, or with the following command:

   - UNIX: *ORACLE_HOME*/opmn/bin> opmnctl [verbose] stopproc ias-component=HTTP_Server
   - Windows: *ORACLE_HOME*\opmn\bin> opmnctl [verbose] stopproc ias-component=HTTP_Server

   > **See Also:** "Starting, Restarting, and Stopping Oracle HTTP Server" on page 7-3

7. Start Oracle HTTP Server using Application Server Control Console, or with the following command:

   - UNIX: *ORACLE_HOME*/opmn/bin> opmnctl [verbose] startproc ias-component=HTTP_Server
   - Windows: *ORACLE_HOME*\opmn\bin> opmnctl [verbose] startproc ias-component=HTTP_Server

   > **See Also:** "Starting, Restarting, and Stopping Oracle HTTP Server" on page 7-3

Note: Be sure that you stop and start Oracle HTTP Server as per the instructions. Restarting Oracle HTTP Server does not yield the same result as stopping and starting it.

8. You can verify if SSL was enabled successfully by navigating to the SSL port, for example:

```
HTTPS://hostname:4443
```

Task 3: (Optional) Customizing Your Configuration

Optionally, you can further customize your configuration using `mod_ossl` directives.

See Also: "Using mod_ossl Directives" on page 11-4

Note: The templates files installed during installation contain all the necessary SSL configuration directives and a default setup for SSL.

To enable client authentication, do the following:

1. Specify SSLVerifyClient on the server side.
2. Use proper client certificate on your client side for the HTTPS connection. Refer to your client documentation for information on getting and using a client certificate. Be sure that your client certificate is trusted by the server wallet.

See Also: *Oracle Application Server Administrator's Guide* for instructions on how to import a trusted certificate into your wallet.

Additional SSL Features

This section contains SSL features that are supported for this release.

- Global Server ID Support
- PKCS #11 Support

Global Server ID Support

This feature adds support SSL protocol features called variously "step-up", "server gated crypto" or "global server ID". "Step-up" is a feature that allows old, weak encryption browsers, to "step-up" so that public keys greater than 512 bits and bulk encryption keys greater than 64 bits can be used in the SSL protocol. This means that server **X.509** certificates that contain public keys in excess of 512 bits and which contain "step-up" digital rights can now be used by Oracle Application Server. Such certificates are often called "128 bit" certificates, even though the certificate itself typically contains a 1024 bit certificate. The Verisign Secure Site Pro is an example of such a certificate which can now be used by Oracle Application Server.

Global Server ID functionality is provided by default, there is no configuration necessary.

PKCS #11 Support

Public-Key Cryptography Standards #11, or PKCS #11 for short, is a public key cryptography specification that outlines how systems use hardware security modules, which are basically "boxes" where cryptographic functions (encryption/decryption) are performed and where encryption keys are stored.

Oracle HTTP Server supports the option of having dedicated SSL hardware through nCipher. nCipher is a certified third party accelerator that improves the performance of the PKI cryptography that SSL uses.

See Also:

- *Oracle Application Server Administrator's Guide*
- `http://www.ncipher.com`

Using SSL Configuration Directives

`mod_ossl` provides standard support for HTTPS protocol connections to Oracle Application Server. It enables secure connections between Oracle HTTP Server and a browser client by using an Oracle-provided encryption mechanism over SSL. It may also be used for authentication over the Internet through the use of digital certificate technology. It supports SSL v. 3.0, and provides:

- Encrypted communication between client and server, using **RSA** or **DES** encryption standards.
- Integrity checking of client-server communication using **MD5** or **SHA** checksum algorithms.
- Certificate management with Oracle **wallet**s.

The following `mod_ssl` directives are not supported by `mod_ossl`.

- `SSLRandomSeed`
- `SSLCertificateFile`
- `SSLCertificateKeyFile`
- `SSLCertificateChainFile`
- `SSLCACertificateFile`
- `SSLCACertificatePath`
- `SSLVerifyDepth`

Note: The server will not start if these directives are used.

Using mod_ossl Directives

To configure SSL for your Oracle HTTP Server, enter the `mod_ossl` directives you want to use in the `httpd.conf` file.

The following directives are described in subsequent sections:

- SSLAccelerator
- SSLCARevocationFile
- SSLCARevocationPath

- SSLCipherSuite
- SSLEngine
- SSLLog
- SSLLogLevel
- SSLMutex
- SSLOptions
- SSLPassPhraseDialog
- SSLProtocol
- SSLRequire
- SSLRequireSSL
- SSLSessionCache
- SSLSessionCacheTimeout
- SSLVerifyClient
- SSLWallet
- SSLWalletPassword

SSLAccelerator

Specifies if SSL accelerator is used. Currently only nFast card is supported.

Category	Value
Valid Values	`yes/no`
Syntax	`SSLAccelerator yes|no`
Default	`SSLAccelerator no`
Context	server configuration

Note: The `SSLAccelerator` directive has been deprecated. For information on enabling SSL acceleration support using a wallet, refer to the *Oracle Advanced Security Administrator's Guide* on `http://www.oracle.com/technology/documentation`.

SSLCARevocationFile

Specifies the file where you can assemble the Certificate Revocation Lists (CRLs) from **CA**s (Certificate Authorities) that you accept certificates from. These are used for client authentication. Such a file is the concatenation of various **PEM**-encoded CRL files in order of preference. This directive can be used alternatively or additionally to SSLCARevocationPath.

Category	Value
Syntax	`SSLCARevocationFile` *`file_name`*
Example	`SSLCARevocationFile /`*`ORACLE_HOME`*`/Apache/Apache/conf/ssl.crl/ca_bundle.crl`
Default	None

Category	Value
Context	server configuration, virtual host

SSLCARevocationPath

Specifies the directory where **PEM**-encoded Certificate Revocation Lists (CRLs) are stored. These CRLs come from the **CA**s (Certificate Authorities) that you accept certificates from. If a client attempts to authenticate itself with a certificate that is on one of these CRLs, then the certificate is revoked and the client cannot authenticate itself with your server.

Category	Value
Syntax	`SSLCARevocationPath` *`path/to/CRL_directory/`*
Example	`SSLCARevocationPath` */ORACLE_HOME*`/Apache/Apache/conf/ssl.crl/`
Default	None
Context	server configuration, virtual host

SSLCipherSuite

Specifies the SSL **cipher suite** that the client can use during the SSL handshake. This directive uses a colon-separated cipher specification string to identify the cipher suite. Table 11–2 shows the tags you can use in the string to describe the cipher suite you want.

Tags are joined together with prefixes to form a cipher specification string.

Category	Value
Valid Values	`none`: Adds the cipher to the list `+` : Adds the cipher to the list and place them in the correct location in the list `-` : Remove the cipher from the list (can be added later) `!` : Remove the cipher from the list permanently
Example	`SSLCipherSuite` *`ALL:!LOW:!DH`* In this example, all ciphers are specified except low strength ciphers and those using the **Diffie-Hellman key negotiation algorithm**.
Syntax	`SSLCipherSuite` *`cipher-spec`*
Default	`ALL:!ADH:!EXPORT56:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP`
Context	server configuration, virtual host, directory

Table 11–1 SSLCipher Suite Tags

Function	Tag	Meaning
Key exchange	`kRSA`	`RSA` key exchange
Key exchange	`kDHr`	Diffie-Hellman key exchange with RSA key
Authentication	`aNULL`	No authentication
Authentication	`aRSA`	`RSA` authentication
Authentication	`aDH`	Diffie-Hellman authentication

Table 11–1 (Cont.) SSLCipher Suite Tags

Function	Tag	Meaning
Encryption	eNULL	No encryption
Encryption	DES	DES encoding
Encryption	3DES	Triple DES encoding
Encryption	RC4	RC4 encoding
Data Integrity	MD5	MD5 hash function
Data Integrity	SHA	SHA hash function
Aliases	SSLv3	All SSL version 3.0 ciphers
Aliases	EXP	All export ciphers
Aliases	EXP40	All 40-bit export ciphers only
Aliases	EXP56	All 56-bit export ciphers only
Aliases	LOW	All low strength ciphers (export and single DES)
Aliases	MEDIUM	All ciphers with 128-bit encryption
Aliases	HIGH	All ciphers using triple DES
Aliases	RSA	All ciphers using RSA key exchange
Aliases	DH	All ciphers using Diffie-Hellman key exchange

Note: There are restrictions if export versions of browsers are used. Oracle module, mod_ossl, supports RC4-40 encryption only when the server uses 512 bit key size wallets.

Table 11–2 Cipher Suites Supported in Oracle Advanced Security 10i

Cipher Suite	Authentication	Encryption	Data Integrity
SSL_RSA_WITH_3DES_EDE_CBC_SHA	RSA	3DES (168)	SHA
SSL_RSA_WITH_RC4_128_SHA	RSA	RC4 (128)	SHA
SSL_RSA_WITH_RC4_128_MD5	RSA	RC4 (128)	MD5
SSL_RSA_WITH_DES_CBC_SHA	RSA	DES (56)	SHA
SSL_DH_anon_WITH_3DES_EDE_CBC_SHA	DH anon	3DES (168)	SHA
SSL_DH_anon_WITH_RC4_128_MD5	DH anon	RC4 (128)	MD5
SSL_DH_anon_WITH_DES_CBC_SHA	DH anon	DES (56)	SHA
SSL_RSA_EXPORT_WITH_RC4_40_MD5	RSA	RC4 (40)	MD5
SSL_RSA_EXPORT_WITH_DES40_CBC_SHA	RSA	DES40 (40)	SHA
SSL_RSA_WITH_AES_128_CBC_SHA	RSA	AES (128)	SHA
SSL_RSA_WITH_AES_256_CBC_SHA	RSA	AES (256)	SHA
SSL_DHE_DSS_EXPORT_WITH_DES40_CBS_SHA	DH DSS	DES (40)	SHA
SSL_DHE_DSS_WITH_DES_CBC_SHA	DH DSS	DES (50)	SHA
SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DH DSS	3DES (168)	SHA

Table 11–2 (Cont.) Cipher Suites Supported in Oracle Advanced Security 10i

Cipher Suite	Authentication	Encryption	Data Integrity
SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA	DH RSA	DES (40)	SHA
SSL_DHE_RSA_WITH_DES_CBC_SHA	DH RSA	DES (56)	SHA
SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA	DH RSA	3DES (168)	SHA
SSL_DHE_DSS_EXPORT1024_WITH_DES_CBC_SHA	DH DSS	DES (40)	SHA
SSL_DHE_DSS_WITH_RC4_128_SHA	DH DSS	RC4 (128)	SHA
SSL_DHE_DSS_EXPORT1024_WITH_RC4_56_SHA	DH DSS	RC4 (56)	SHA

SSLEngine

Toggles the usage of the SSL Protocol Engine. This is usually used inside a `<VirtualHost>` section to enable SSL for a particular virtual host. By default, the SSL Protocol Engine is disabled for both the main server and all configured virtual hosts. Example 11–1 is an example for using SSLEngine directive. The default SSL is 4443 on UNIX and 443 on Windows.

Example 11–1 Using SSLEngine Directive

```
<VirtualHost_dafault_:4443>
  SSLEngine on
  ...
</VirtualHost>
```

Category	Value
Syntax	`SSLEngine on|off`
Default	`SSLEngine off`
Context	server configuration, virtual host

SSLLog

Specifies where the SSL engine log file will be written. (Error messages will also be duplicated to the standard Oracle HTTP Server log file specified by the ErrorLog directive.)

Place this file at a location where only root can write, so that it cannot be used for symlink attacks. If the filename does not begin with a slash (/), it is assumed to be relative to the ServerRoot. If the filename begins with a bar (|), then the string following the bar is expected to be a path to an executable program to which a reliable pipe can be established.

This directive should occur only once per virtual server configuration.

Category	Value
Syntax	`SSLVerifyClient` *`path/to/filename`*
Default	None
Context	server configuration, virtual host

SSLLogLevel

Specifies the verbosity degree of the SSL engine log file.

Category	Value
Valid Values	The levels are (in ascending order, where each level is included in the levels preceding it): ■ `none`: No dedicated SSL logging is done. Messages of type 'error' are duplicated to the standard HTTP server log file specified by the `ErrorLog` directive. ■ `error`: Only messages of the type 'error' (conditions that stop processing) are logged. ■ `warn`: Messages that notify of non-fatal problems (conditions that do not stop processing) are logged. ■ `info`: Messages that summarize major processing actions are logged. ■ `trace`: Messages that summarize minor processing actions are logged. ■ `debug`: Messages that summarize development and low-level I/O operations are logged.
Syntax	`SSLLogLevel` *`level`*
Default	None
Context	server configuration, virtual host

SSLMutex

Type of semaphore (lock) for SSL engine's mutual exclusion of operations that have to be synchronized between Oracle HTTP Server processes.

Category	Value
Valid Values	■ `none`: Uses no mutex at all. Not recommended, because the mutex synchronizes the write access to the SSL session cache. If you do not configure a mutex, the session cache can become garbled. ■ `file:`path/to/mutex: Uses a file for locking. The process ID (PID) of the Oracle HTTP Server parent process is appended to the filename to ensure uniqueness. If the filename does not begin with a slash (/), it is assumed to be relative to `ServerRoot`. This setting is not available on Windows. ■ `sem`: Uses an operating system semaphore to synchronize writes. On UNIX, it would be a Sys V IPC semaphore; on Windows, it is a Windows Mutex. This is the best choice, if the operating system supports it.
Example	`SSLMutex file:`*`/usr/local/apache/logs/ssl_mutex`*
Syntax	`SSLMutex` *`type`*
Default	`SSLMutex` *`none`*
Context	server configuration

SSLOptions

Controls various runtime options on a per-directory basis. In general, if multiple options apply to a directory, the most comprehensive option is applied (options are not merged). However, if all of the options in an `SSLOptions` directive are preceded by a plus ('+') or minus ('-') symbol, then the options are merged. Options preceded by a plus are added to the options currently in force, and options preceded by a minus are removed from the options currently in force.

Category	Value

Valid Values	■ `StdEnvVars`: Creates the standard set of CGI/SSI environment variables that are related to SSL. This is disabled by default because the extraction operation uses a lot of CPU time and usually has no application when serving static content. Typically, you only enable this for CGI/SSI requests.
	■ `ExportCertData`: Enables the following additional CGI/SSI variables:
	`SSL_SERVER_CERT`
	`SSL_CLIENT_CERT`
	`SSL_CLIENT_CERT_CHAIN_n` (where n= 0, 1, 2...)
	These variables contain the Privacy Enhanced Mail (**PEM**)-encoded X.509 certificates for the server and the client for the current HTTPS connection, and can be used by CGI scripts for deeper certificate checking. All other certificates of the client certificate chain are provided. This option is "Off" by default because there is a performance cost associated with using it.
	`SSL_CLIENT_CERT_CHAIN_n` variables are in the following order: `SSL_CLIENT_CERT_CHAIN_0` is the intermediate CA who signs `SSL_CLIENT_CERT`. `SSL_CLIENT_CERT_CHAIN_1` is the intermediate CA who signs `SSL_CLIENT_CERT_CHAIN_0`, and so forth, with `SSL_CLIENT_ROOT_CERT` as the root CA.
	■ `FakeBasicAuth`: Translates the subject **distinguished name** of the client **X.509** certificate into an HTTP basic authorization user name. This means that the standard HTTP server authentication methods can be used for access control. Note that no password is obtained from the user; the string 'password' is substituted.

Valid Values (for `SSLOptions` continued)	■ `StrictRequire`: Denies access when, according to SSLRequireSSL or directives, access should be forbidden. Without `StrictRequire`, it is possible for a `'Satisfy any'` directive setting to override the `SSLRequire` or `SSLRequireSSL` directive, allowing access if the client passes the host restriction or supplies a valid user name and password. Thus, the combination of `SSLRequireSSL` or `SSLRequire` with `SSLOptions +StrictRequire` gives `mod_ossl` the ability to override a `'Satisfy any'` directive in all cases. ■ `CompatEnvVars`: Exports obsolete environment variables for backward compatibility to Apache SSL 1.x, `mod_ssl` 2.0.x, Sioux 1.0, and Stronghold 2.x. Use this to provide compatibility to existing CGI scripts. ■ `OptRenegotiate`: This enables optimized SSL connection renegotiation handling when SSL directives are used in a per-directory context.
Syntax	`SSLOptions` *`[+-] option`*
Default	None
Context	server configuration, virtual host, directory

SSLPassPhraseDialog

Type of pass phrase dialog for wallet access. `mod_ossl` asks the administrator for a pass phrase in order to access the wallet.

Category	Value
Valid Values	■ `builtin`: when the server is started, `mod_ossl` prompts for a password for each wallet. This cannot be used when Oracle HTTP Server is managed by OPMN. No user interaction is allowed when Oracle HTTP Server is started by OPMN. ■ `exec`: path/to/program - when the server is started, `mod_ossl` calls an external program configured for each wallet. This program is invoked with two arguments: servername:portnumber and `RSA` or `DSA`.
Syntax	`SSLPassPhraseDialog` *type*
Example	`SSLPassPhraseDialog exec:`*`/usr/local/apache/sbin/pfilter`*
Default	`SSLPassPhraseDialog` *`builtin`*
Context	server configuration

SSLProtocol

Specifies SSL protocol(s) for `mod_ossl` to use when establishing the server environment. Clients can only connect with one of the specified protocols.

Category	Value
Valid Values	SSLv2, SSLv3, TLSv1, ALL
Example	To specify only SSL version 3.0, set this directive to the following: `SSLProtocol` *`+SSLv3`*
Syntax	`SSLProtocol` *`[+-] protocol`*
Default	`SSLProtocol ALL`
Context	server configuration, virtual host

SSLRequire

Denies access unless an arbitrarily complex boolean expression is true.

Category	Value
Syntax	`SSLRequire` *`expression`*
Default	None
Context	directory

The *expression* must match the following syntax (given as a BNF grammar notation):

```
expr ::= "true" | "false"
"!" expr
expr "&&" expr
expr "||" expr
"(" expr ")"

comp ::=word "==" word | word "eq" word
word "!=" word |word "ne" word
word "<" word |word "lt" word
word "<=" word |word "le" word
word ">" word |word "gt" word
word ">=" word |word "ge" word
word "=~" regex
```

```
word "!~" regex
wordlist ::= word
wordlist "," word

word ::= digit
cstring
variable
function

digit ::= [0-9]+

cstring ::= "..."

variable ::= "%{varname}"
```

Table 11–3 and Table 11–4 list standard and SSL variables. These are valid values for varname.

```
function ::= funcname "(" funcargs ")"
```

For `funcname`, the following function is available:

```
file(filename)
```

The file function takes one string argument, the filename, and expands to the contents of the file. This is useful for evaluating the file's contents against a regular expression.

Table 11–3 lists the standard variables for SSLRequire varname.

Table 11–3 Standard Variables for SSLRequire Varname

Standard Variables	Standard Variables	Standard Variables
HTTP_USER_AGENT	PATH_INFO	AUTH_TYPE
HTTP_REFERER	QUERY_STRING	SERVER_SOFTWARE
HTTP_COOKIE	REMOTE_HOST	API_VERSION
HTTP_FORWARDED	REMOTE_IDENT	TIME_YEAR
HTTP_HOST	IS_SUBREQ	TIME_MON
HTTP_PROXY_CONNECTION	DOCUMENT_ROOT	TIME_DAY
HTTP_ACCEPT	SERVER_ADMIN	TIME_HOUR
HTTP:headername	SERVER_NAME	TIME_MIN
THE_REQUEST	SERVER_PORT	TIME_SEC
REQUEST_METHOD	SERVER_PROTOCOL	TIME_WDAY
REQUEST_SCHEME	REMOTE_ADDR	TIME
REQUEST_URI	REMOTE_USER	ENV:variablename
REQUEST_FILENAME		

Table 11–4 lists the SSL variables for SSLRequire varname.

Table 11–4 SSL Variables for SSLRequire Varname

SSL Variables	SSL Variables	SSL Variables
HTTPS	SSL_PROTOCOL	SSL_CIPHER_ALGKEYSIZE
SSL_CIPHER	SSL_CIPHER_EXPORT	SSL_VERSION_INTERFACE
SSL_CIPHER_USEKEYSIZE	SSL_VERSION_LIBRARY	SSL_SESSION_ID

Table 11–4 (Cont.) SSL Variables for SSLRequire Varname

SSL Variables	SSL Variables	SSL Variables
SSL_CLIENT_V_END	SSL_CLIENT_M_SERIAL	SSL_CLIENT_V_START
SSL_CLIENT_S_DN_ST	SSL_CLIENT_S_DN	SSL_CLIENT_S_DN_C
SSL_CLIENT_S_DN_CN	SSL_CLIENT_S_DN_O	SSL_CLIENT_S_DN_OU
SSL_CLIENT_S_DN_G	SSL_CLIENT_S_DN_T	SSL_CLIENT_S_DN_I
SSL_CLIENT_S_DN_UID	SSL_CLIENT_S_DN_S	SSL_CLIENT_S_DN_D
SSL_CLIENT_I_DN_C	SSL_CLIENT_S_DN_Email	SSL_CLIENT_I_DN
SSL_CLIENT_I_DN_O	SSL_CLIENT_I_DN_ST	SSL_CLIENT_I_DN_L
SSL_CLIENT_I_DN_T	SSL_CLIENT_I_DN_OU	SSL_CLIENT_I_DN_CN
SSL_CLIENT_I_DN_S	SSL_CLIENT_I_DN_I	SSL_CLIENT_I_DN_G
SSL_CLIENT_I_DN_Email	SSL_CLIENT_I_DN_D	SSL_CLIENT_I_DN_UID
SSL_CLIENT_CERT	SSL_CLIENT_CERT_CHAIN_n	SSL_CLIENT_ROOT_CERT
SSL_CLIENT_VERIFY	SSL_CLIENT_M_VERSION	SSL_SERVER_M_VERSION
SSL_SERVER_V_START	SSL_SERVER_V_END	SSL_SERVER_M_SERIAL
SSL_SERVER_S_DN_C	SSL_SERVERT_S_DN_ST	SSL_SERVER_S_DN
SSL_SERVER_S_DN_OU	SSL_SERVER_S_DN_CN	SSL_SERVER_S_DN_O
SSL_SERVER_S_DN_I	SSL_SERVER_S_DN_G	SSL_SERVER_S_DN_T
SSL_SERVER_S_DN_D	SSL_SERVER_S_DN_UID	SSL_SERVER_S_DN_S
SSL_SERVER_I_DN	SSL_SERVER_I_DN_C	SSL_SERVER_S_DN_Email
SSL_SERVER_I_DN_L	SSL_SERVER_I_DN_O	SSL_SERVER_I_DN_ST
SSL_SERVER_I_DN_CN	SSSL_SERVER_I_DN_T	SSL_SERVER_I_DN_OU
SSL_SERVER_I_DN_G	SSL_SERVER_I_DN_I	

SSLRequireSSL

Denies access to clients not using SSL. This is a useful directive for absolute protection of a SSL-enabled virtual host or directories in which configuration errors could create security vulnerabilities.

Category	Value
Syntax	`SSLRequireSSL`
Default	None
Context	directory

SSLSessionCache

Specifies the global/interprocess session cache storage type. The cache provides an optional way to speed up parallel request processing.

Category	Value
Valid Values	▪ `none`: disables the global/interprocess session cache. Produces no impact on functionality, but makes a major difference in performance. ▪ `shmht:/path/to/datafile[bytes]`: Uses a high-performance hash table (`bytes` specifies approximate size) inside a shared memory segment in RAM, which is established by the `/path/to/datafile`. This hash table synchronizes the local SSL memory caches of the server processes. ▪ `shmcb:/path/to/datafile[bytes]`: Uses a high-performance Shared Memory Cyclic Buffer (SHMCB) session cache to synchronize the local SSL memory caches of the server processes. The performance of `shmcb` is more uniform in all environments when compared to `shmht`.
Syntax	`SSLSessionCache` *`type`*
Examples	`SSLSessionCache shmht: /`*`ORACLE_HOME`*`/Apache/Apache/logs/ssl_scache(512000)` `SSLSessionCache shmcb: /`*`ORACLE_HOME`*`/Apache/Apache/logs/ssl_scache(512000)`
Default	`SSLSessionCache` *`none`*

SSLSessionCacheTimeout

Specifies the number of seconds before a SSL session in the session cache expires.

Category	Value
Syntax	`SSLSessionCacheTimeout` *`seconds`*
Default	300
Context	server configuration

SSLVerifyClient

Specifies whether or not a client must present a certificate when connecting.

Category	Value
Valid Values	▪ `none`: No client certificate is required ▪ `optional`: Client may present a valid certificate ▪ `require`: Client must present a valid certificate
Syntax	`SSLVerifyClient` *`level`*
Default	None
Context	server configuration, virtual host

Note: The level `optional_no_ca` included with `mod_ssl` (in which the client can present a valid certificate, but it need not be verifiable) is not supported in `mod_ossl`.

SSLWallet

Specifies the location of the wallet with its **WRL**.

Category	Value
Syntax	`SSLWallet wrl` The format of `wrl` is: *`file:path to wallet`*
Example	`SSLWallet file:/etc/ORACLE/WALLETS/server` Other values of wrl may be used as permitted by the Oracle SSL product.
Default	None
Context	server configuration, virtual host

SSLWalletPassword

Specifies the Wallet password needed to access the wallet specified within the same context. You can choose either a cleartext wallet password or an obfuscated password. The obfuscated password is created with the command line tool `iasobf`. If you must use a regular wallet, Oracle recommends that you use the obfuscated password instead of a cleartext password.

See Also: "Using the iasobf Utility" on page 11-15

Category	Value
Syntax	`SSLWalletPassword password` If no password is required do not set this directive. Note: If a wallet created with the Auto Login feature of Oracle Wallet Manager is used, then do not set this directive because these wallets do not require passwords.
Default	None
Context	server configuration, virtual host

Note: `SSLWalletPassword` has been deprecated. A warning message is generated in the Oracle HTTP Server log if this directive is used. For secure wallets, Oracle recommends that you get a SSO wallet, with auto-login enabled, instead. Refer to the"Task 1: Creating a Real Wallet" on page 11-1.

Using the iasobf Utility

The `iasobf` utility enables you to generate an obfuscated wallet password from a **cleartext** password.

If you are using an Oracle Wallet that has been created with Auto Login enabled (an SSO wallet), then you do not need to use this utility. However, if you must use a regular wallet with a password, then Oracle recommends that you use the password obfuscation tool `iasobf`, which is located in *`ORACLE_HOME`*`/Apache/Apache/bin`, to generate an obfuscated wallet password from a cleartext password.

To generate an obfuscated wallet password, the command syntax is:

```
iasobf -p password
```

The obfuscated password is printed to the terminal. `iasobf` requires operating system user of httpd process. Accordingly, use the `root` argument for UNIX or `system`

argument for Windows. For example, on UNIX, the command will be `iasobf -password root`.

> **Note:** The corresponding tool for Windows environments is called `osslpassword`, which can be used in the same way as `iasobf`.

Using mod_proxy Directives

The following directives are for mod_proxy support only:

- SSLProxyCache
- SSLProxyCipherSuite
- SSLProxyProtocol
- SSLProxyWallet
- SSLProxyWalletPassword

SSLProxyCache

Specifies whether the proxy cache will be used. The proxy will use the same session as the SSL server uses.

Category	Value
Syntax	`SSLProxyCache` *`on/off`*
Default	`SSLProxyCache off`
Context	server configuration, virtual host

SSLProxyCipherSuite

Specifies the proxy server's cipher suite.

Category	Value
Syntax	`SSLCipherSuite` *`cipher-spec`*
Default	`None`
Context	server configuration, virtual host

SSLProxyProtocol

Controls the proxy server's SSL protocol flavors.

Category	Value
Syntax	`SSLProxyProtocol [+-]` *`protocol`*
Default	`None`
Context	server configuration, virtual host

SSLProxyWallet

Specifies the location of the wallet containing the certificates to use when opening proxy connections.

Category	Value
Syntax	`SSLProxyWallet` *`wrl`*
Default	`None`
Context	server configuration, virtual host

SSLProxyWalletPassword

Specifies the proxy wallet password.

Category	Value
Syntax	`SSLProxyWalletPassword` *`password`*
Default	`None`
Context	server configuration, virtual host

Note: `SSLProxyWalletPassword` has been deprecated. A warning message is generated in the Oracle HTTP Server log if this directive is used. For secure wallets, Oracle recommends that you get a SSO wallet instead.

12

Using PHP with Oracle Application Server

The chapter provides information about PHP Hypertext Preprocessor (PHP) for use with Oracle Application Server.

This chapter contains the following sections:

12.1 PHP Overview

PHP is a scripting language capable of being embedded in HTML. This capability makes PHP well-suited for Web development. PHP's object model facillitates project development using standard object-oriented methodologies.

PHP is part of the Oracle Application Server installation package located in the following directory:

`\OH\10.1.2\OracleAS\portal\Apache\Apache\php`

Oracle provides information about PHP at the Open Source Developer Center:

`http://www.oracle.com/technology/tech/opensource/index.html`

You also can find information about PHP at:

`http://www.php.net/`

12.2 Using the PHP Extension for Oracle JDeveloper

The following link provides information on installing and configuring the PHP Extension for Oracle JDeveloper:

`http://www.oracle.com/technology/products/jdev/htdocs/partners/addins/exchange/php/index.html`

For a tutorial on using JDeveloper with PHP, see "*Oracle Application Server Standard Edition One Quick Tour*".

12.3 Connecting to an Oracle Database with PHP

The following link provides information on connecting to an Oracle Database with PHP:

```
http://www.oracle.com/technology/products/jdev/htdocs/partners/addins/exchange/php
/readme_php1.2.html
```

12.4 Oracle Application Server mod_php Extensions

Oracle Application Server provides a server side module for PHP named mod_php. Configuration options for mod_php include extensions. Oracle Application Server supports the extensions listed in Table 12–1, " Oracle Application Server mod_php Extensions".

Table 12–1 Oracle Application Server mod_php Extensions

mod_php Extension	Installation Support	Description	Runtime Dependencies
BC Math	Default enabled on Windows only	For arbitrary precision mathematics, PHP offers the Binary Calculator which supports numbers of any size and precision, represented as strings.	No external libraries are needed for this extension.
Calendar	Default enabled on Windows only	The calendar extension presents a series of functions to simplify converting between different calendar formats.	No external libraries are needed for this extension.
C Type	Available on Unix and Windows	The functions provided by this extension check whether a character or string falls into a certain character class according to the current locale.	No external libraries are needed for this extension.
COM	Available on Windows only	COM is one of the main ways to glue applications and components together on the Windows platform. Using COM you can launch Microsoft Word, fill in a document template, save the result as a Word document, and send it to a visitor of your web site.	No external libraries are needed for this extension. However, COM objects, like MS Word, must be installed by the user.
FTP	Default enabled on Windows only	This extension provides access to an FTP server and provides a wide range of control to the executing script.	No external libraries are needed for this extension.
OCI8	Enabled on UNIX and Windows	The OCI 8 functions allow you to connect to Oracle 9, Oracle 8 and Oracle 7 databases.	Oracle client libraries are required to use this extension. Oracle client libraries are provided with an Oracle Application Server install. On Windows the dependency is on php_ oci8.dll which resides in %ORACLE_HOME% /Apache/Apache/php/ext ensions This functionality can be enabled or disabled on Windows from the php.ini file.

Table 12–1 (Cont.) Oracle Application Server mod_php Extensions

mod_php Extension	Installation Support	Description	Runtime Dependencies
Object Overload	Enabled on UNIX and Windows	The purpose of this extension is to allow overloading of object property access and method calls. This extension is experimental.	No external libraries are needed for this extension.
ODBC	Default enabled on Windows only	This function is enabled by default on Windows. The Unified ODBC functions allows the user to access most databases like Adabas D, IBM DB2 etc.	During install time, the libraries for the database, which requires support, must be provided. This is not possible to provide hence not supported.
PCRE	Enabled on UNIX and Windows	Perl-compatible regular expressions or PCRE functions are enabled with this directive.	This extension uses the default bundled library and does not require any additional libraries.
POSIX	Default enabled on UNIX only	This module provides access to POSIX functions and is available only for UNIX systems.	No external libraries are needed to build this extension.
SESSION	Enabled on UNIX and Windows	Session support in PHP consists of a way to preserve certain data across subsequent accesses. The session related information is either stored in a cookie or propagated though a URL.	No external libraries are needed to build this extension.
MYSQL	Enabled on UNIX and Windows	This extension is enabled by default in PHP and provides access to MySQL databases.	On Solaris it uses the bundled client libraries and on Windows it requires the php_mysql.dll along with the libmysql.dll.
TOKENIZER	Enabled on UNIX and Windows	The tokenizer functions provide an interface to the PHP tokenizer embedded in the Zend Engine. Using these functions you may write your own PHP source analyzing or modification tools without having to deal with the language specification at the lexical level.	No external libraries are needed to build this extension.
XML	Enabled on UNIX and Windows	This PHP extension implements support for the XML parser Expat. Since PHP is built using Apache, it uses the bundled expat library from Apache.	No external libraries are needed to build this extension.
ZLib	Default enabled on Windows only	This module enables the customer to transparently read compressed files like gzip.	No additional extensions are required to use this functionality.

Note: For more information on mod_php extensions, see http://us4.php.net/manual/en/funcref.php

A

Using Oracle Application Server Proxy Plug-in

This appendix explains how the Oracle Application Server Proxy Plug-in (OracleAS Proxy Plug-in) enables you to use components in conjunction with a third-party HTTP listener. OracleAS Proxy Plug-in works with the Sun ONE Web Server Enterprise Edition on UNIX and Windows systems, or the Microsoft Internet Information Server (IIS) on Windows systems, to send requests to Oracle Application Server.

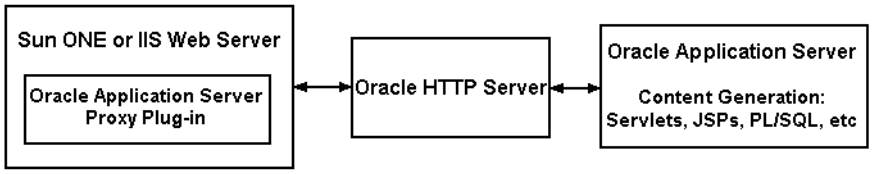
> **See Also:**
>
> `http://www.oracle.com/technology/products/ias/ohs/htdocs/plugincerts.html` for complete certification information.

Topics discussed are:

Overview

OracleAS Proxy Plug-in is a reverse HTTP proxy. The **plug-in** forwards incoming HTTP requests to an Oracle Application Server instance as shown in Figure A–1.

Figure A–1 OracleAS Proxy Plug-in



This proxy logic is provided as a plug-in, a shared library that is loaded by the third-party HTTP listeners. The plug-in uses APIs provided with the third-party listeners to directly handle HTTP requests, in much the same way that modules are plugged into Oracle HTTP Server.

Oracle HTTP Server can mimic the address and port that the third-party listener is using. That is, when sending a request to Oracle HTTP Server, the proxy can be configured to send a different Host: HTTP header than the actual hostname and port that the request is being sent to, so that downstream applications are shielded from the introduction of the reverse proxy.

Downloading OracleAS Proxy Plug-in

OracleAS Proxy Plug-in is available on the Oracle Application Server 10*g* Companion CD, which is included in your Oracle Application Server CD Pack.

Installing OracleAS Proxy Plug-in

After downloading OracleAS Proxy Plug-in, place the appropriate configuration file and shared library in directories that the third-party listener can access.

On the Oracle Application Server 10*g* Companion CD, the files are located at `/plugins/solaris/` for UNIX and `/plugins/win32/` for Windows.

Table A–1 contains information about the shared libraries for OracleAS Proxy Plug-in.

Table A–1 OracleAS Proxy Plug-in Shared Libraries

Platform	File Name	Location and Description	Instructions
UNIX	`oracle_proxy.so`	`oracle_proxy.so` is the OracleAS Proxy Plug-in file for Sun ONE Web listener. It is located in the `/plugins/solaris/sunone` directory	To install the plug-in into the listener, place `oracle_proxy.so` in a directory to which the listener has read and execute privileges.
Windows	`oracle_proxy.dll`	`oracle_proxy.dll` is the OracleAS Proxy Plug-in file for the IIS Web listener. It is located in the `/plugins/win32/iis` directory.	To install the plug-in into the listener, copy `oracle_proxy.dll` to a directory the listener can access.
	`oracle_proxy_sunone.dll`	`oracle_proxy_sunone.dll` is the OracleAS Proxy Plug-in file for Sun ONE Web listener. It is located in the `/plugins/win32/sunone` directory	To install the plug-in into the listener, copy `oracle_proxy_sunone.dll` to a directory the listener can access.

Using Application Server Control Console

When you install Oracle Application Server, you can administer Oracle HTTP Server using Application Server Control Console. However, if you choose to use Sun ONE or

IIS instead of Oracle HTTP Server, then it is recommended that you disable Oracle HTTP Server on Application Server Control Console so that it no longer appears there.

Oracle does not support monitoring or administering of non-Oracle HTTP Server listener with Application Server Control Console.

See Also:

- Chapter 7, "Application Server Control Console Management"
- *Oracle Enterprise Manager Concepts*

Configuring OracleAS Proxy Plug-in

There is one configuration file for OracleAS Proxy Plug-in. It controls the proxy functionality. The presence of the configuration file in the Web server's file system makes the functionality active.

You also need to modify configuration files specific to the third-party listener to enable the plug-in on to these listeners.

Proxy Server Definition File

The **proxy server** definition file must reside in a directory that is readable by the third-party listener. For simplicity, you could create a directory called `proxy` in a convenient location on your system, and place the proxy server definition file, the proxy shared library file, and proxy log files in it.

Described in detail in Proxy Configuration File Parameters section, the proxy server definition file contains:

- Name value pairs that describe the servers that will be used to proxy requests to Oracle Application Server.
- Options for communicating with the servers.
- A set of rules that map URLs to the servers.

You can create this file with the text editor of your choice. The oproxy.serverlist parameter must list at least one server name, or the proxy will not function.

Example A–1 provides a sample proxy server definition file.

Example A–1 Sample Proxy Server Definition File

```
# This file defines proxy server behavior.
#
# Server names that the proxy plug-in will recognize.
oproxy.serverlist=ias1

# Hostname to use when communicating with a specific server.
oproxy.ias1.hostname=oasdocs.us.oracle.com

# Port to use when communicating with a specific server.
oproxy.ias1.port=7777

# Description of URL(s) that will be redirected to this server.
oproxy.ias1.urlrule=/*
```

Proxy Configuration File Parameters

The following proxy configuration file parameters are described in the subsequent sections:

- oproxy.serverlist
- oproxy.servername.hostname
- oproxy.servername.port
- oproxy.servername.alias
- oproxy.servername.resolveall
- oproxy.servername.urlrule

oproxy.serverlist

Lists all of the server names that the plug-in recognizes.

Category	Value
Parameter Type	string list
Allowable Values	Comma separated list of server names, one for each Oracle HTTP Server to which requests will be sent. All servers in the serverlist must also be defined in the file.
Default Value	None. At least one server name must be provided for the proxy to be functional.
Example	`oproxy.serverlist=ias1,ias2`

oproxy.*servername*.hostname

Defines the hostname to use when communicating with a specific server.

Category	Value
Parameter Type	string
Allowable Values	Valid hostname
Default Value	None
Example	`oproxy.ias1.hostname=www1.us.oracle.com`

oproxy.*servername*.port

Defines the port to use when communicating with a specific server.

Category	Value
Allowable Values	Valid port value
Default Value	80
Example	`oproxy.ias1.port=7777`

oproxy.*servername*.alias

Supports the mimicing feature of the proxy by defining the hostname and port that clients use to access the third-party HTTP listener. If defined, this value will be passed as the Host: HTTP header. If not defined, the hostname and port of the machine actually being communicated with will be sent.

Category	Value
Parameter Type	string
Allowable Values	host:port
Default Value	`oproxy.`*`servername.hostname:oproxy.servername.port`*
Example	`oproxy.ias1.alias=www.oracle.com:80`

oproxy.*servername*.resolveall

Directs the proxy plug-in to resolve the hostname to the backend server on every request. This enables DNS based failover or routing between the proxy plug-in and backend servers. The use of this parameter incurs the cost of going to the DNS server for every incoming request, and hence should only be used if the mapping from hostname to IP address will change dynamically.

Category	Value
Allowable Values	true or false
Default Value	false
Example	`oproxy.ias1.resolveall=true`

oproxy.*servername*.urlrule

Describes a URL or set of URLs that are redirected to this server. A given server can have any number of `urlrule` properties assigned to it.

Category	Value
Parameter Type	string
Example	`oproxy.ias1.urlrule=/foo/*`

Three types of rules can be used: exact match, context match, or suffix match.

- **Exact matches**: One URL is mapped to a server.

  For example:

  `oproxy.ias1.urlrule=/foo/bar/foo.html` would map only the URL `/foo/bar/foo.html` to be proxied to the server with the name `ias1` (the details for the server `ias1` are configured in the server configuration file).

- **Context matches**: A set of URLs with a common prefix or context are all mapped to a server. For example, `oproxy.ias1.urlrule=/foo/*` would map URLs beginning with `/foo` to the server with the name `ias1`.

  For context matches, you can use the stripcontext option with the `urlrule` parameter to send only the portion of the url *following* the wildcard to the server. The default for the stripcontext option is false, so you do not need to include it unless you are setting it to true. It is shown for completeness of the example.

  Example: In following configuration:

  ```
  oproxy.ias1.urlrule=/ias1/*
  oproxy.ias1.stripcontext=false
  ```

  and the URL request:

  `http://`*`hostname`*`/ias1/header1.gif`

retrieves

ORACLE_HOME/Apache/Apache/htdocs/ias1/header1.gif

In the following configuration:

```
oproxy.ias1.urlrule=/ias1/*
oproxy.ias1.stripcontext=true
```

and the URL request:

http://*hostname*/ias1/header1.gif

retrieves

ORACLE_HOME/Apache/Apache/htdocs/header1.gif

- **Suffix matches**: All files with a common file extension are mapped to a server.

  For example, `oproxy.ias1.urlrule=/*.jsp` would map all of the URLs that end in `.jsp` to the server `ias1`. This can be combined with the context rule to have something like `/foo/bar/*.jsp` so that only URLs that start with `/foo/bar` and end in `.jsp` would be proxied.

Note: For the `oproxy.`*`servername`*`.urlrule`, when multiple rules apply to the same URL, the following precedence applies:

1. Exact matches
2. Longest context match plus suffix match
3. Longest context match

Some examples of the precedence are:

```
/foo/bar/index.html would take precedence over
/foo/bar/*
/foo/bar/*.jsp would take precedence over /foo/bar/*
/foo/bar/* would take precedence over /foo/*
```

Defining OracleAS Proxy Plug-in Behavior

In the proxy server definition file, you define which servers and URLs to proxy to the plug-in.

1. In the first line of the file, specify the list of all the servers that can be used by the plugins. For example:

   ```
   oproxy.serverlist=ias1,ias2
   ```

2. Set the relevant properties (hostname, port, and server alias) for each server. For example:

   ```
   oproxy.ias1.hostname=myhost.us.oracle.com
   oproxy.ias1.port=7777
   oproxy.ias1.alias=www.oracle.com
   ```

   The hostname must be provided. If you do not specify the port, 80 is assigned. If an alias value is not given, the combination of the hostname and port given are used. The alias enables the back end server to receive requests that have an HTTP

Host: header that looks exactly like the one the client delivers to the third-party listener.

3. Set the `urlrule` parameter to specify redirection between servers. For example, the rule:

```
oproxy.ias1.urlrule=/*
```

maps all incoming requests to be proxied to the Web server on the server `ias1`. These rules can be of three forms, exact URL, context match, or extension-based. An exact match maps exactly one URL to a server, for example:

```
oproxy.ias1.urlrule=/my/path/index.html
```

maps only accesses to `/my/path/index.html` for proxying. An example of a context rule is:

```
oproxy.ias1.urlrule=/app1/*
```

which maps any URL beginning with `/app1`. An extension-based rule, such as:

```
oproxy.ias1.urlrule=/*.jsp
```

maps any URL ending with `.jsp`.

All requests sent to a mapped URL are proxied through HTTP/1.1 to the specified server.

Configuring Sun ONE Listener to Use OracleAS Proxy Plug-in

This section provides proxy plug-in configuration instructions for Sun ONE Enterprise Server listener on UNIX and Windows systems.

Notes: If you are configuring the Sun ONE listener on Windows, use forward slashes (/) in all paths.

The default configuration files for Sun ONE route all incoming requests for the URI `/servlet` to the Sun ONE servlet handler. The OracleAS Proxy Plug-in does not override the Sun ONE server's configuration settings. You must ensure that the URL mappings to the OracleAS Proxy Plug-in are distinct from the URL mappings to the Sun ONE servlet engine.

---

1. Open the `magnus.conf` file in version 6, or `obj.conf` in version 4 in the Sun ONE listener `/config` directory.
2. Add the load-modules line:

On UNIX:

```
Init fn="load-modules" shlib="/path/oracle_proxy.so" funcs=op_init,op_
objecttype,op_service
```

On Windows:

```
Init fn="load-modules" shlib="/path/oracle_proxy_sunone.dll" funcs=op_init,op_
objecttype,op_service
```

where /path/ is the path to the shared library for the plug-in. This line tells the listener where the proxy shared library is, and which functions are exposed by this library.

3. Add the configuration parameters line:

```
Init fn="op_init" server_defs="/path/servers"
logfile="/path/oproxy.log" log_level=error
```

where /path/ is the path to the proxy server definition and log files. The proxy server definition file contains all of the configuration information for the servers that the proxy plug-in can communicate with. A log file and log level to log messages from the plug-in can also be specified (optional).

> **See Also:** "Proxy Server Definition File" on page A-3 for a complete description and example.

4. Add the following line to the <Object name=default> section of the obj.conf file, before all other lines beginning with the word ObjectType:

```
ObjectType fn=op_objecttype
```

5. Add the following line before all other lines that begin with the word "Service":

```
Service type="oracle/proxy" fn="op_service"
```

6. Start the listener using the GUI or the shell script.

Configuring IIS Listener to Use OracleAS Proxy Plug-in

This section provides proxy plug-in configuration instructions for the IIS listener on Windows systems. The process involves creating Windows registry entries and using the IIS management console to add directories and filters. You must restart the listener after configuring the plug-in.

To configure the plug-in, perform the following steps:

1. From the Start menu, select **Run**.
2. In the run dialog box, type regedit and click **OK**.

   The Registry Editor window opens.
3. In the Registry Editor window, expand the HKEY_LOCAL_MACHINE folder (click the + preceding its name).
4. Expand the SOFTWARE folder (click the + preceding its name).
5. Click the ORACLE folder.
6. From the Edit menu, select New, then Key.

   A new folder is added under the ORACLE folder with the name New Key #1.
7. Type IIS Proxy Adapter for the key name.
8. From the Edit menu, select New, then String Value.

   A new value is added in the right window pane with the name New Value #1.
9. Type server_defs for the value name.
10. From the Edit menu, select Modify. The Edit String dialog box appears.

11. In the Value data field, type the full path of your proxy server definition file. Click **OK**.
12. Specify `log_file` and `log_level` using the procedure specified in steps 8-11. This is optional.
    a. Add a string value with the name `log_file` and the desired location of the log file (for example, `d:\proxy\proxy.log`)
    b. Add a string value with the name `log_level` and a value for the desired log level. Valid values are debug, inform, error and emerg.
13. Using the IIS management console, add a new virtual directory to your IIS Web site with the same physical path as that of `oracle_proxy.dll`. Name the directory `oproxy` and give it execute access.
14. Using the IIS management console, add `oracle_proxy.dll` as a filter in your IIS Web site. The name of the filter should be `oproxy` and its executable must point to the directory containing `oracle_proxy.dll` (for example, `d:\proxy\oracle_proxy.dll`).
15. Restart IIS (stop and then start the IIS Server), ensuring that the `oproxy` filter is marked with a green upward arrow.

Note: To restart IIS, you must stop all of the IIS services through the control panel, or restart the computer. This is the only way to ensure that the `.dll` is reloaded. Restarting IIS through the management console is not sufficient.

OracleAS Proxy Plug-in Usage Notes

This section highlights development and usage practices to consider when developing an application that runs behind the OracleAS Proxy Plug-in. Some of these also have relevance when enabling an application to run behind Oracle Application Server Web Cache.

- **Check for configurations based on the Oracle HTTP Server being the entry point into the network.**

  This is usually only relevant if an application has a module that plugs directly into the Oracle HTTP Server. Specifically, look for dependencies on obtaining information about the client based on the connection made to the Oracle HTTP Server, such as using the SSL certificate for authentication. Currently, SSL is not supported, so even if the client uses SSL to connect to the third-party listener, an unencrypted HTTP message will be sent from the third-party listener to the Oracle HTTP Server. This means that client certificates will not be available to components that reside behind the plug-in. The environment variable `REMOTE_ADDR` has been specifically preserved when OracleAS Proxy Plug-in and Oracle Application Server Web Cache are used, but other client information may, in practice, represent the machine on which the proxy resides rather than the actual client host. These behaviors must be discovered and eliminated in cases where the Oracle HTTP Server is not the external listener for Oracle Application Server.

- **Avoid returning non-relative links in HTML, that is, avoid embedding host names into HTML unless the link is external to the Web site.**

  This includes static HTML pages, dynamic pages generated by servlets, JSPs, PL/SQL, and so on. Examine all code that obtains the server name of Oracle HTTP Server to ensure that it is not embedding the server name into pages that are sent

back to the client. To test for this behavior, use a "spider" application that traverses all links in a Web site. Open source tools with this functionality are available.

- **Avoid returning host and port information in applications (such as applets or javascript) downloaded to the client.**

  If you have an application that uses browser-based code, ensure that the code does not contain the hostname and port of Oracle HTTP Server that actually delivers the content. Instead, it must have the actual client-accessible address used by the third-party listener.

- **Ensure that all URLs within an application can be easily mapped to a set of rules that the proxy can use.**

  In order to successfully proxy all requests for an application, the OracleAS Proxy Plug-in must have a complete description of the URL space for that application. Each Oracle Application Server application must describe the set of rules necessary to configure the plug-in for that application. This set of rules must include all URLs that the application could generate. If an application generates a URL that is not described by the proxy `urlrule` parameters, the request will be served by the third-party HTTP listener, and a "document not found" error may occur (or, worse, a document other then the intended document may be delivered to the client).

  Developers of applications that use common top level directories (such as a reliance on mapping /images) should be prepared to:

  - Change these common links to something that will not conflict with applications that might already be deployed on the third-party listener.
  - Instruct the user to copy the necessary content to the third-party listener's directory structure. For performance reasons, it is a good idea to have the third-party listener handle static `.gif` and `.jpg` files anyway, but it requires extra effort.

Troubleshooting

This section describes common problems and possible solutions.

Listener Fails to Start

- Ensure that you have the newest version of the OracleAS Proxy Plug-in.
- Verify that your listener configuration is set up correctly. (The IIS listener may need to be restarted in order to make the filter work properly.) A proxy server definition file must exist.

  See Also: "Proxy Server Definition File" on page A-3 for a description of this file and parameter requirements.

- Check for problems in the proxy server definition file. Each server in the serverlist line must be defined later in the file, and you must have at least one server defined. If a server name is listed but not defined, the listener may not start (although the reverse is not true). Ensure that there are no typographical errors or missing quotes in the proxy server definition file.
- **For Sun ONE 6.0 on UNIX and Windows:** Ensure that `Init` lines are added to the `magnus.conf` file and `ObjectType` and `Service` lines are added to the `obj.conf` file.

Listener Returns Incorrect URLs

- Verify that changes to the proxy server definition file have been saved and the listener has been restarted.
- Ensure that there are no typographical errors in the proxy server definition file.
- Ensure that the `urlrule` parameter is set up correctly, and consider whether the stripcontext option should be set to true.
- Verify that the `serverlist` line in the proxy server definition file specifies the back-end server you are trying to reach.
- Verify that the back-end server is running, and that the file you are attempting to retrieve exists and is accessible on the back-end server.
- Verify that the host, port and `urlrule` parameters in the proxy server definition file target the correct area on the back-end server.
- Ensure that client requests are being sent to the correct port on the third-party listener machine.
- Check the listener log files, the proxy log (may need to be turned on in "debug" mode, and may require restarting the listener), and the back-end server logs to verify that requests are getting through.

Changes Made to Proxy Server Definition File are Not Reflected

- Ensure that you have saved the proxy server definition file and restarted the listener.
- **For IIS:** To pick up the changes, you must stop and start the WWW Publishing Service from the Control Panel. This takes a few minutes.

IIS Listener Displays Incomplete Pages or Garbled Characters

Do not display an IIS pages with a Sun ONE browser.

Parsing Error Occurs with Sun ONE 6.0

If you try to change the ports or turn on security (for SSL), the server may return the error message "Unable to parse `magnus.conf`".

Remove any comments and added lines preceding and following the `Init` lines in the `magnus.conf` file.

"File Not Found" Error Occurs

If you are using a context-based `urlrule` parameter to retrieve a file that is known to exist, and the listener returns "`Not Found`", you probably need to set "`stripcontext=true`".

See Also: "oproxy.servername.urlrule" on page A-5

Partial URL Requests Return Unexpected Results

The IIS and Sun ONE servers auto-complete URLs differently. Requests of "`http://serviceman`", "`http://serviceman/`", and "`http://serviceman/index.html`" do not necessarily return the same results on different platforms. The `oproxy.`*`servername`*`.urlrule` parameter can be used to work around this problem.

See Also: "oproxy.servername.urlrule" on page A-5

Sun ONE Server Returns "Server Error" with "/servlet" Request

The default Sun ONE configuration maps any URL requests to "/servlet" to its own servlet handler. You must edit the proxy server definition file, or change the Sun ONE configuration to correct this.

Server Returns Page with Broken Image Links

If you use an exact `urlrule` parameter, for example, `"urlrule=/*.html"`, in the proxy server definition file (or a similar scenario), the server retrieves the specified page, but all other links are forbidden to the user, including inline images in the page. (If you use an exact `urlrule` with `stripcontext=true`, a "Server Error" is returned.)

Unexpected Pages are Displayed

Clear the memory cache in your client browser. Earlier versions of Sun ONE and IE cache pages even when told to retrieve the page every time, when no memory is allocated for caching (you may need to restart the browser to get this behavior to work). If you see a page you're not expecting, try refreshing or reloading the page.

REMOTE_ADDR Contains Unexpected IP Address

The `REMOTE_ADDR` field usually contains the IP address of the client machine. In some URL request cases, if there is a proxy server in the environment, the field may contain the IP address of the proxy server.

Redirects Go To Network Entry Point

If the back-end server returns a redirect to the entry point of the network, do one of the following, the first option being the preferred one:

- Set the following directives in the `httpd.conf` file:

  ```
  UseCanonicalName On
  ServerName name of listener host
  Port port of listener host
  ```

- Set the following directives in the `httpd.conf` file:

  ```
  UseCanonicalName port
  Port port of listener host
  ```

  Edit the proxy plug-in server configuration file:

  ```
  oproxy.serverName.alias=name of listener host:port of listener host
  ```

SSL Requests Yield Unexpected Results

The proxy plug-in supports SSL connections made between the client and the proxy host, but does not support SSL connections between the proxy and the back-end server. To implement the latter, set up the listener to receive SSL connections and start the back-end server in non-SSL mode. No changes to the proxy configuration are needed.

B

Using Oracle Application Server SSO Plug-in

This appendix explains how to use Oracle Application Server SSO Plug-in (OracleAS SSO Plug-in) to a protect third-party HTTP listener and its applications. The OracleAS SSO Plug-in works with the Sun ONE Web Server Enterprise Edition on UNIX and Windows systems, and the Microsoft Internet Information Server (IIS) on Windows systems.

> **See Also:**
> `http://www.oracle.com/technology/products/ias/ohs/htdocs/plugincerts.html` for complete certification information.

Topics discussed are:

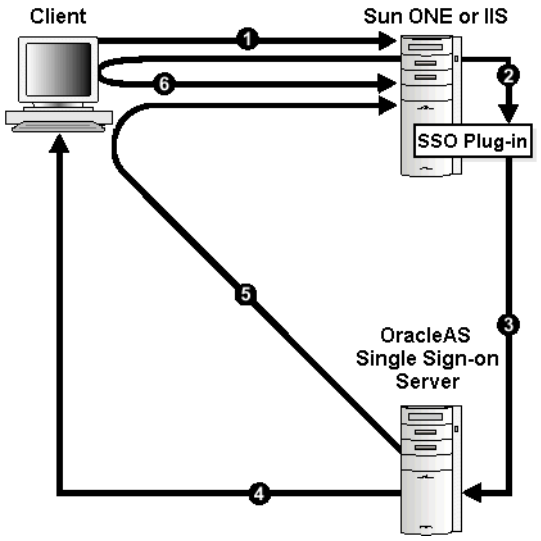
Overview

OracleAS SSO Plug-in is Oracle's single sign-on (SSO) solution for third-party listeners such as Sun ONE and IIS. The **plug-in** is designed to protect native third-party listener applications using the SSO infrastructure. With the help of the OracleAS SSO Plug-in, you can be authenticated to different third-party listener applications using only one SSO password. You can integrate these SSO-protected third-party listener applications with SSO enabled Oracle HTTP Server applications or legacy Oracle SSO enabled application together as long as they are all protected on the same SSO server.

OracleAS SSO Plug-in is a simple version of mod_osso, and only implements some of its basic functionality. Features such as dynamic authentication, global logout, idle timeout and global timeout, and basic authentication for legacy application are not implemented in the current OracleAS SSO Plug-in release.

Figure B–1 illustrates the process involved when you request a URL protected by the OracleAS SSO Plug-in.

Figure B–1 OracleAS SSO Plug-in



1. The user requests a URL through a Web browser.
2. The Web server looks for an OracleAS SSO Plug-in cookie for the user. If the cookie exists, the Web server extracts the user's information and uses it to log the user in to the requested application.
3. If the cookie does not exist, then the OracleAS SSO Plug-in redirects the user to the single sign-on server.
4. The single sign-on server looks for its own cookie in the browser. If it finds none, it tries to authenticate the user with a user name and password. If authentication is successful, the single sign-on server creates a cookie in the browser as a reminder that the user has been authenticated. If a cookie exists, the single sign-on server authenticates using the cookie.
5. The single sign-on server returns the user's encrypted information to the OracleAS SSO Plug-in.
6. OracleAS SSO Plug-in creates its own cookie for the user in the browser and redirects the user to the requested URL.

   During the same session, if the user again seeks access to the same or to a different application, the user is not prompted for a user name and password; the application uses an HTTP header to obtain this information from the OracleAS SSO Plug-in session cookie.

Downloading OracleAS SSO Plug-in

OracleAS SSO Plug-in is available on the Oracle Application Server 10*g* Companion CD, which is included in your Oracle Application Server CD Pack.

Installing OracleAS SSO Plug-in

Install OracleAS SSO Plug-in on a machine that has an Oracle Application Server installation. This installation is required only for the network and security dependent libraries and single sign-on registration tool; it is not required to be running. After the Oracle Application Server installation on UNIX systems, add *ORACLE_HOME*`/lib` to the `LD_LIBRARY_PATH` in the listener's start script. For example, the "start" script in

Sun ONE. On Windows systems, the installation automatically sets the environment variable `PATH`. For example, *`ORACLE_HOME`*`\bin`.

Download, or copy, the plug-in, and place the configuration file and shared libraries in directories that the third-party listener can access. For security reasons, ensure that all the configuration files and plug-in modules are given minimum privileges.

On the Oracle Application Server 10*g* Companion CD, the files are located at `/plugins/solaris/` for UNIX and `/plugins/win32/` for Windows.

Table B–1 contains information about the shared libraries for OracleAS SSO Plug-in.

Table B–1 OracleAS SSO Plug-in Shared Libraries

Platform	File Name	Location and Description	Instructions
UNIX	`oracle_proxy.so`	`oracle_proxy.so` is the OracleAS SSO Plug-in file for Sun ONE Web listener. It is located in the `/plugins/solaris/sunone` directory	To install the plug-in into the listener, place `oracle_proxy.so` in a directory to which the listener has read and execute privileges.
Windows	`oracle_osso.dll`	`oracle_osso.dll` is the OracleAS SSO Plug-in file for the IIS Web listener. It is located in the `/plugins/win32/iis` directory.	To install the plug-in into the listener, copy `oracle_osso.dll` to a directory the listener can access.
	`oracle_proxy_sunone.dll`	`oracle_proxy_sunone.dll` is the OracleAS SSO Plug-in file for Sun ONE Web listener. It is located in the `/plugins/win32/sunone` directory	To install the plug-in into the listener, copy `oracle_proxy_sunone.dll` to a directory the listener has read and execute privileges.

Registering with Single Sign-On

The **single sign-on** registration process enables the single sign-on server and the listener to share information such as server location, protocol version, and common encryption key, before they communicate. After the registration process, this information is stored on the single sign-on server side as a single sign-on partner application entry. On the listener side, a single sign-on file called `sso_conf` is created. `sso_conf` is obfuscated for security purposes. Copy it to an appropriate location so that the listener can access it.

Oracle provides a Java-based single sign-on registration tool to automatically complete this process.

Using the Single Sign-On Registration Tool

Register the third-party listener with a single sign-on server using the following command:

```
ORACLE_HOME/jdk/bin/java -jar ORACLE_HOME/sso/lib/ossoreg.jar [arguments]
```

where *`ORACLE_HOME`* is the home directory of your Oracle Application Server installation. Be sure that `LD_LIBRARY_PATH` is set to include `$ORACLE_HOME/lib32`.

Note: You can only run this tool on the same machine where your listener resides. Also, the resulting single sign-on configuration file has to be generated directly on the same machine. Do not copy the single sign-on configuration file across different machines.

A different version `ossoreg.jar` could have very different command arguments. If needed, run the command using "-help" first to get the complete usage information.

Common Single Sign-On Registrar Command Arguments

Table B–2 lists some important common arguments for the single sign-on registrar.

Table B–2 SSO Registrar Command Arguments

Argument	Description
`-oracle_home_path`	Absolute path to the Oracle home of the Oracle Application Server installation
`-site_name`	Name of the site, typically expressed as the contiguous string *`host:port`*.
`-ssoDBConnect`	Single sign-on database JDBC connect string.
`-pass`	`ORASSO_PA` password.
`-mod_osso_url`	`http://<`*`listener_hostname.domain`*`>:`*`port`*
`-admin_id`	User name of the third-party administrator. This argument is optional.
`-admin_info`	Information associated with the administrator's user name, such as e-mail address. This argument is optional.
`-config_ mod_osso`	Set to TRUE. This parameter indicates that the application being registered is `mod_osso`. This argument is necessary to generate the `sso_conf` file.
`-u`	Specifies the name of the account used to start the third-party listener. For example, use the value of `User` specified in the `magnus.conf` for Sun ONE and `SYSTEM` for IIS. The default is the user who runs the single sign-on registrar tool.
`-sso_server_version`	Must be set to `v1.2`.
`-virtualhost`	Be sure to include this argument.
`-config_file`	Specifies a path of the final obfuscated single sign-on configuration file. It has to be set under `$ORACLE_HOME`. Default is set to: ■ UNIX: *`ORACLE_HOME`*`/Apache/Apache/conf/osso.conf` ■ Windows: *`ORACLE_HOME`*`/Apache\Apache\conf\osso.conf`

Example B–1 Using Common Single Sign-On Registrar Command Arguments

On UNIX:

```
ORACLE_HOME/jdk/bin/java -jar ORACLE_HOME/sso/lib/ossoreg.jar \
-ssoDBConnect <host.domain>:1521:iasdb -pass your_password \
-oracle_home_path ORACLE_HOME -site_name <host.domain>:7778 \
-config_mod_osso TRUE -mod_osso_url http://<host.domain>:7778 \
-u nobody -admin_id admin_name -admin_info admin@company.com \
-sso_partner_version v1.2 \
-virtualhost \
-config_file ORACLE_HOME/Apache/Apache/conf/osso/sso_conf
```

On Windows:

```
ORACLE_HOME/jdk/bin/java -jar ORACLE_HOME/sso/lib/ossoreg.jar \
-ssoDBConnect <host.domain>:1521:iasdb -pass your_password \
-oracle_home_path ORACLE_HOME -site_name <host.domain>:8080 \
-config_mod_osso TRUE -mod_osso_url http://<host.domain>:8080 \
-u SYSTEM -admin_id admin_name -admin_info admin@company.com \
-sso_partner_version v1.2 \
-virtualhost \
-config_file ORACLE_HOME/Apache/Apache/conf/osso/sso_conf
```

Configuring OracleAS SSO Plug-in

Create a plug-in configuration file such as `osso_plugin.conf`. This is the file where you define all the plug-in functionality. It can also be referred as the `osso` property file. The syntax is exactly the same for all third-party listeners. This file must reside in a directory that is readable by the third-party listener. This file also contains the following:

- Plug-in directives such as `LoginServerFile` and `IpCheck`
- A set of rules that match resources to be protected.

OracleAS SSO Plug-in Configuration Directives

Table B–3 lists the configuration directives for the OracleAS SSO Plug-in.

Table B–3 SSO Plug-in Configuration Directives

Directive Name	Function
`LoginServerFile`	Specifies the location of the Single Sign-On Server configuration file such as `sso.conf` that is attained from the SSO registration process. This directive gets its name from Login Server, which is now called Single Sign-On Server, for historical reasons. This is a global parameter and should not be used on a per-resource basis. That is, you must provide one and only one Single Sign-On Server configuration file. ■ **Parameter Type**: string ■ **Allowable Values**: the full path of your Single Sign-On Server configuration file, such as `sso_conf`. ■ **Default Value**: None. You must provide the exact location of the Single Sign-On server configuration file to allow SSO plug-in to be functional. ■ **Example**: `LoginServerFile=/path/config/sso_conf`
`IpCheck`	Specifies whether the SSO plug-in should check the IP address of each request when it examines the cookie. Valid values are true and false. Setting `IpCheck` to true prevents cookies being stolen. ■ **Parameter Type**: Boolean ■ **Allowable Values**: true/false. ■ **Default Value**: false. ■ **Example**: `IpCheck=true` Note: Set `IpCheck` to false if you have a proxy server or firewall between your Sun ONE server and your clients' browser.
`HardTimeout`	Deprecated.

Resource Protection

Use the following format to protect resources:

```
<OSSO url-matching-rule>
  SSO_configuration_directives
</OSSO>
```

Use the following rules to define the url-matching-rule:

Rule Name	Description
Exact Match	This option identifies an exact file as a protected resource, for example: `/examples/Hello.html`

Rule Name	Description
Context Match	This option identifies a directory as a protected resource, for example: `/examples/*`
Extension Match	This option identifies files with a certain extension in a particular directory as a protected resource, for example: `/examples/*.jsp`

When multiple rules apply to the same URL, the following precedence applies:

1. Exact matches
2. Longest context match plus suffix match
3. Longest context match

Some examples of the precedence are:

```
/foo/bar/index.html would take precedence over /foo/bar/*
/foo/bar/*.jsp would take precedence over /foo/bar/*
/foo/bar/* would take precedence over /foo/*
```

Example B–2 Simple Single Sign-on Configuration File, osso_plugin.conf

```
LoginServerFile=/path/sso_conf
<OSSO /private/hello.html>
  IpCheck = false
</OSSO>
<OSSO /private1/*>
</OSSO>
<OSSO /private2/*.jsp>
  IpCheck = true
</OSSO>
```

Configuring Sun ONE Listener for Single Sign-on

This section provides OracleAS SSO Plug-in configuration instructions for the Sun ONE listener on UNIX and Windows systems.

> **Note:** If you are configuring the Sun ONE listener on Windows, use forward slashes (/) in all paths.

1. Open the `magnus.conf` file, version 6, or `obj.conf`, version 4, in the Sun ONE listener `/config` directory.
2. Add the load-modules line:

   On UNIX:

   ```
   Init fn="load-modules" shlib="/path/oracle_proxy.so" funcs="osso_init,oracle_
   single_sign_on,osso_redirect_service,osso_success_service"
   ```

   On Windows:

   ```
   Init fn="load-modules" shlib="/path/oracle_proxy_sunone.dll" funcs="osso_
   init,oracle_single_sign_on,osso_redirect_service,osso_success_service"
   ```

where /path/ is the path to the shared library for the plug-in. This line tells the listener where the proxy shared library is, and which functions are exposed by this library.

3. Add the configuration parameters line:

```
Init fn="osso_init" osso_properties="/path/osso_plugin.conf" log_
file="/path/plugin.log" log_level=error
```

where /path/osso_plugin.conf is the exact location of the plug-in configuration file you just created. Also this line can specify a log file and log level to log messages from the plug-in (optional).

4. Add the following line to the <Object name=default> section of the obj.conf file, before all other lines:

```
AuthTrans fn="oracle_single_sign_on"
```

5. Add the following line to the <Object name=default> section before all other lines that begin with the word Service:

```
Service type="oracle/sso_redirect" fn="osso_redirect_service"
```

6. Add the following lines where /path/ is the path of your document root. For example: /home/Sun ONE/docs/ or $docroot.

```
<Object ppath="/path/osso_login_success">
  Service fn="osso_success_service"
</Object>
```

7. Change the LD_LIBRARY_PATH variable in your start script to include the location of *ORACLE_HOME*/lib32, where *ORACLE_HOME* is the Oracle Application Server installation home directory.
8. Restart the listener.

Usage Notes for Sun ONE Enterprise Server Version 6.0

For version 6.0, the same shared library can be used as with version 4.1. The configuration is virtually the same, but the configuration files for Sun ONE have changed slightly in version 6.0. In this version, the two lines beginning with Init that need to be added must be added at the end of the magnus.conf file rather than to the obj.conf file. The other two lines that should be added to obj.conf remain the same.

Configuring IIS Listener for Single Sign-On

This section provides instructions on configuring the IIS Listener to use OracleAS SSO Plug-in. The plug-in consists of a single .dll, oracle_osso.dll. To install the plug-in, copy the .dll to the host on which IIS resides and perform the following steps:

1. Edit your registry to create a new registry key named HKEY_LOCAL_MACHINE\SOFTWARE\Oracle\IIS OSSO Adapter.
2. Specify the exact location of your plug-in configuration file you just created. For example: d:\osso\osso_plugin.conf, by adding this string value with the name cfg_file and a value pointing to the location of your configuration file.
3. Specify a log_file and log_level. This is optional.

a. Add a string value with the name `log_file` and the desired location of the log file. For example: `d:\ossoplugin.log`.

b. Add a string value with the name `log_level` and a value for the desired log level. Valid values are `debug`, `inform`, `error` and `emergency`.

4. Using the IIS management console, add a new virtual directory to your IIS Web site with the same physical path as that of `oracle_osso.dll`. Name the directory `osso` and give it execute access.

5. Using the IIS management console, add `oracle_osso.dll` as a filter in your IIS Web site. The name of the filter should be osso and its executable must point to the directory containing `oracle_osso.dll`.For example, `d:\osso\oracle_osso.dll`.

6. Stop and then start the IIS Server, ensuring that the filter is marked with a green up-pointing arrow.

Note:

- To restart IIS, you must stop all of the IIS services through the control panel, or restart the computer. This is the only way to ensure that the `.dll` is reloaded. Restarting IIS through the management console is not sufficient.
- If you want multiple Oracle installations on the same home, the `ORACLE_HOME\bin` `PATH` entry for the installation that you wish to use in conjunction with the OracleAS SSO Plug-in must appear first in your `PATH`.

Troubleshooting

This section describes common problems and possible reasons.

Oracle Dependency Libraries Not Found

OracleAS SSO Plug-in could not find the libraries it needs. Possible reason would be that you do not have *`ORACLE_HOME`*`/lib` included in your `LD_LIBRARY_PATH` on UNIX. On Windows, you do not have *`ORACLE_HOME`*`/bin` included in your `PATH`.

Single Sign-On Server Configuration File De-obfuscation Fails

Single sign-on server configuration file, for example, `sso_conf`, is obfuscated using a certain account and this account has to be the one being used to start your listener. For example, use the value of `User` specified in `magnus.conf` for Sun ONE and usually `SYSTEM` for IIS.

IIS Oracle Application Server OracleAS SSO Plug-in Does Not Work with HTML Authentication

OracleAS SSO Plug-in is designed to not work with other authentication modules. It is either a native listener authentication module, or third-party module.

C

Using Oracle Application Server Containers for J2EE Plug-in

This appendix explains how the Oracle Application Server Containers for J2EE Plug-in (OC4J Plug-in) enables you to use third party HTTP listeners to access servlets running in OC4J J2EE within Oracle Application Server. OC4J Plug-in works with Sun ONE Web Server for UNIX, and Microsoft Internet Information Server (IIS) for Windows.

> **See Also:**
>
> `http://www.oracle.com/technology/products/ias/ohs/htdocs/plugincerts.html` for complete certification information.

It also contains information about using `mod_oc4j` in a non-Oracle Apache.

Topics discussed are:

Overview

OC4J Plug-in is a shared library that can be loaded into IIS, or Sun ONE HTTP listener. It provides functionality to route requests directly from a third party listener to OC4J in the same manner `mod_oc4j` routes requests from Oracle HTTP Server to OC4J. Thus, requests for OC4J can be directly routed from IIS or Sun ONE to one or more OC4J JVMs using the AJP or AJP over SSL protocol.

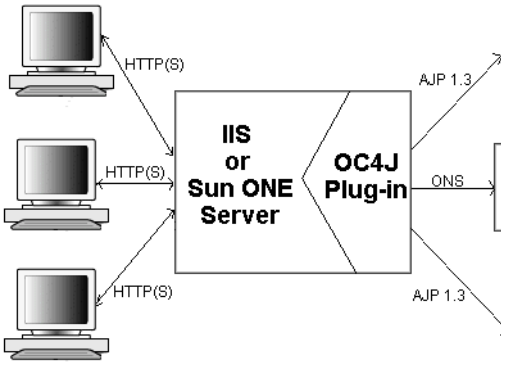
Note: OracleAS Proxy Plug-in and OracleAS SSO Plug-in are certified to work with the entire Oracle Application Server stack, such as Oracle Application Server Portal and Oracle Application Server Wireless. However, OC4J Plug-in is only certified to run customer applications. For example, you cannot route requests from Sun ONE using OC4J Plug-in to an OC4J container where Oracle Application Server Wireless is running.

See Also:

- Appendix A, "Using Oracle Application Server Proxy Plug-in"
- Appendix B, "Using Oracle Application Server SSO Plug-in"
- "mod_oc4j" on page 8-9

Figure C–1 illustrates how the OC4J Plug-in routes requests directly from a third party listener to OC4J.

Figure C–1 OC4J Plug-in for IIS and Sun ONE



Downloading OC4J Plug-in

OC4J Plug-in is available on the Oracle Application Server 10*g* Companion CD, which is included in your Oracle Application Server CD Pack.

Installing OC4J Plug-in

On the Oracle Application Server 10*g* Companion CD, the files are located at `/plugins/solaris/` for UNIX and `/plugins/win32/` for Windows.

Table C–1 contains information about the shared libraries for OC4J Plug-in.

Table C–1 OC4J Plug-in Shared Libraries

Platform	File Name	Location and Description	Instructions
UNIX	`opii.so`	`opii.so` is the OC4J plug-in for Sun ONE Web listener. It is located in the `/plugins/solaris/sunone` directory	To install the plug-in into the listener, place `opii.so` in a directory to which the listener has read and execute privileges.
Windows	`opii.dll`	`opii.dll` is the OC4J plug-in file for the IIS Web listener. It is located in the `/plugins/win32/iis` directory.	To install the plug-in into the listener, copy `opii.dll` to a directory the listener can access.
	`opii_sunone.dll`	`opii_sunone.dll` is the OC4J plug-in for SUN ONE Web listener. It is located in the `/plugins/win32/sunone` directory	To install the plug-in into the listener, copy `opii_sunone.dll` to a directory the listener can access.

Note: In order to use the OracleAS SSO Plug-in or to turn on SSL with the OC4J Plug-in when using the plug-ins within IIS 6.0 on Microsoft Windows 2003, the permissions on the .dll files in the `%ORACLE_HOME%\bin` location must be changed. The .dll files must have execution permissions for the user `NETWORK SERVICE` added.

Do not complete this operation during normal installation.

Because IIS 6.0 is run as the user `NETWORK SYSTEM`, the plug-ins will not function correctly unless the necessary permissions are granted.

Configuring OC4J Plug-in on Sun ONE

Place the `opii.so` file in a directory such as `/sunone/opii.so`, where it is readable by the Sun ONE listener. The following examples assume that Sun ONE is installed in a directory called `/sunone`, and that the instance being configured exists under `/sunone/`*`https-mymachine`*.

1. Add the following lines at the end of `/sunone/`*`https-mymachine`*`/magnus.conf`:

```
Init fn="load-modules" shlib="/sunone/opii.so" funcs=opii_init,opii_
objecttype,opii_service,opii_child_init

Init fn="opii_init" log_file="/sunone/https-mymachine/logs/opii.log"
  log_level=error server_defs="/sunone/https-machine/config/opii.conf"
Init fn="opii_child_init" LateInit=yes
```

   where, `log_file` points to a file where OC4J Plug-in messages will be logged and `server_defs` points to an OC4J configuration file.

   See Also: "OC4J Plug-in Configuration File" on page C-5

2. Make the following modifications to `/sunone/`*`https-mymachine`*`/obj.conf`:

   a. Add the following line before any `ObjectType` line:

```
ObjectType fn=opii_objecttype
```

   b. Add the following line before all other lines that begin with the word `"Service"`:

```
Service type="oracle/opii" fn="opii_service" UseOutStreamSize=8192
```

   Note: Not all versions of Sun ONE support `UseOutputStreamSize`. Refer to Sun ONE documentation for details.

3. If you want to enable the OC4J status page, which is equivalent to the URL `/oc4j-service` in `mod_oc4j` for Oracle HTTP Server, then make the following two changes to `obj.conf`:

   a. Add the following line above the other `NameTrans` entries in`obj.conf`:

```
NameTrans fn=assign-name from="/oc4j-service" name="opii-status"
```

   b. Add the following section at the end of `obj.conf`:

```
<Object name="opii-status">
  Service fn="opii_status_service"
 </Object>
```

Configuring OC4J Plug-in for IIS

Perform the following steps to configure OC4J Plug-in for IIS:

1. Edit your registry to create a new registry key named `HKEY_LOCAL_MACHINE\SOFTWARE\Oracle\OPII`
2. Specify the exact location of your configuration file with the name `server_defs`, and a value pointing to the location of your configuration file, for example `d:\opii\opii.conf`.

   See Also: "OC4J Plug-in Configuration File" on page C-5

3. (Optional) Specify a `log_file` and `log_level`:
   a. Add a string value with the name `log_file`, and the desired location of the log file, for example, `d:\opii\plugin.log`.
   b. Add a string value with the name `log_level`, and a value for the desired log level. Valid values are `debug`, `inform`, `error`, and `emerg`.
   c. If you want to enable the OC4J Plug-in status page, a page equivalent to `mod_oc4j /oc4j-service` URL, add a string value with the name `status_uri` and a value like `/oc4j-service`.
4. Using the IIS management console, add a new virtual directory to your IIS Web site with the same physical path as that of `opii.dll`. Name the directory `opii` and give it execute access.
5. Using the IIS management console, configure the plugin as an allowed Web Service Extension.
6. Using the IIS management console, add `opii.dll` as a filter in your IIS Web site. The name of the filter should be `opii` and its executable must point to the directory contain `opii.dll`, for example, `d:\opii\opii.dll`.
7. If you want to use dynamic routing, ensure that *`ORACLE_HOME`* is set in the system environment or plugin config file using the `Oc4jOracleHome` directive. Using the IIS management console, set the Identity of the Application Pool containing the plugin to "Local System" so that the plugin can communicate with OPMN.
8. Restart IIS (stop and then start the IIS server), ensuring that the filter is marked with a green arrow pointing up.

Note: To restart IIS, you must stop all of the IIS services through the control panel, or restart the computer. This is the only way to ensure that the `.dll` is reloaded. Restarting IIS through the management console is not sufficient.

Configuring Anonymous Access for IIS

Perform the following steps if you want OC4J to perform the authentication:

1. In the IIS Management GUI, right click the default Web site and select Properties.

2. Select the Directory Security tab and click the Edit button under Anonymous Access and Authentication Control. Be sure that the Anonymous access is checked and that basic authentication and integrated Windows Authentication are both unchecked.

> **Note:** If you want IIS to perform authentication and simply pass the user name to OC4J, then do not disable authentication in IIS.

OC4J Plug-in Configuration File

When you set up the OC4J Plug-in in the third party listener, the configuration file points at a `server_defs` file, or the OC4J Plug-in configuration file. This file defines the OC4J instances that the OC4J Plug-in communicates with. It has the same syntax as the `mod_oc4j` file for Oracle HTTP Server. For example, a configuration file that contains only the following line

```
Oc4jMount /j2ee/* ajp13://localhost:3000
```

routes any requests to URLs that begin with `/j2ee/` to the OC4J instance that has an AJP listener on the localhost interface on port 3000.

All of the `Oc4j*` directives defined for `mod_oc4j` also work for OC4J Plug-in. In additional to these directives, the OC4J Plug-in-specific directives `Oc4jOracleHome` can be used in place of setting the `ORACLE_HOME` directive in the environment for the third party listener. An `ORACLE_HOME` value is required if you want to use the dynamic functionality of the OC4J Plug-in.

> **See Also:** "mod_oc4j Configuration File and Directives" on page 8-9

The dynamic routing functionality of the OC4J Plug-in provides the same `Oc4jMount` syntax as `mod_oc4j` for routing to OC4J instances that are managed by OPMN. Accordingly, you can mount OC4J instances or clusters instead of just pointing at the host and port of a single JVM. In order to accomplish this, the OC4J Plug-in must be able to communicate with an ONS daemon on the same machine. If Oracle Application Server is installed on the same machine as the OC4J Plug-in, then this can be accomplished simply by setting either `ORACLE_HOME` in the environment, or setting the `Oc4jOracleHome` directive to point to the location of the `ORACLE_HOME`, and ensuring that the third party listener is running as the same user as Oracle Application Server, or `root` on UNIX.

> **See Also:** "Running Oracle HTTP Server as Root" on page 4-4

If Oracle Application Server is not installed on the same machine, then the standalone ONS daemon must be installed. OC4J Plug-in supports all the `mod_oc4j` functionality, including AJP over SSL and use of port tunneling.

> **See Also:** "Understanding Port Tunneling" on page 10-7

Integrating Generic Apache with Oracle Application Server

In Oracle Application Server, 10*g* Release 2 (10.1.2), you can integrate generic Apache with Oracle Application Server. In doing so, you can route requests from generic Apache to OC4J in the same manner as routing requests using Oracle HTTP Server and `mod_oc4j`. Generic Apache is Apache version 1.3.xx, and not Apache 2.x.

See Also:

- *Oracle HTTP Server Standalone Administrator's Guide Based On Apache 2.0*
- *Oracle HTTP Server Standalone Administrator's Guide Based On Apache 1.3*

Using Oracle Notification Service (ONS), the communication method utilized between `mod_oc4j` and OC4J, you can load `mod_oc4j` into a generic Apache instance and use it in a static routing configuration without requiring any other Oracle infrastructure. In combination with `mod_onsint` and ONS, `mod_oc4j` can utilize the same dynamic configuration and failover options that are available when using Oracle HTTP Server.

See Also: "mod_onsint" on page 8-17

This section discusses the following topics:

Integration Requirements

`mod_oc4j` and `mod_onsint` require the following open source software packages to be included in generic Apache builds that will be integrated with Oracle Application Server:

- Enhanced API (EAPI) provided as part of `mod_ssl`. EAPI provides a context mechanism that is used extensively by `mod_onsint` and `mod_oc4j` to share information and function callbacks between modules without introducing link time dependencies. EAPI is provided by configuring Apache with `mod_ssl`.

  Note: `mod_ssl` does not need to be loaded at runtime as it is the integration of EAPI provided by `mod_ssl` that is important.

  See Also: `http://www.modssl.org`

- (UNIX only) mm shared memory library. The mm library is used on UNIX platforms to share routing information between all of the child processes that make up an Apache instance on UNIX. This library is not necessary on Windows platforms since the Apache architecture on Windows uses a single multi-threaded process instead of many single-threaded processes.

  See Also: `http://www.ossp.org/pkg/lib/mm/`

  Note: `mod_oc4j` is supported in Apache versions 1.3.x only. It is not supported in Apache 2.0.x versions.

Generic Apache Files

Four libraries need to be accessible by the generic Apache instance that is going to be integrated with Oracle Application Server. If `mod_oc4j` is used in a static configuration and `mod_onsint` is not used, then you need only two libraries.

On UNIX, the four files and their locations within an *ORACLE_HOME* are:

```
$ORACLE_HOME/Apache/Apache/libexec/mod_oc4j.so
$ORACLE_HOME/Apache/Apache/libexec/mod_onsint.so
$ORACLE_HOME/opmn/lib/libons.so
$ORACLE_HOME/lib/libdms2.so
```

On Windows, the four files and their locations within an *ORACLE_HOME* are:

```
%ORACLE_HOME%\Apache\Apache\modules\ApacheModuleOc4j.dll
%ORACLE_HOME%\Apache\Apache\modules\ApacheModuleOnsint.dll
%ORACLE_HOME%\opmn\bin\onsclient.dll
%ORACLE_HOME%\bin\yod.dll
```

> **Note:** These files are also available on the Oracle Application Server 10*g* Companion CD. On the CD, they are located in the `/plugins/solaris/` directory for UNIX and `/plugin/win32/` directory for Windows.

It is easiest if the binaries for the two modules are copied into the same location as the other modules in the generic Apache installation (`libexec` on UNIX, and `modules` on Windows), although this is not a requirement. Full paths can be used if you want to place the binaries elsewhere. The `dms` and `ons` libraries do not need to be in any specific location, but they must be in the your `LD_LIBRARY_PATH` on UNIX, and the `PATH` on Windows. On UNIX, this is most easily accomplished by editing the `apachectl` script used to start the generic Apache instance to set the `LD_LIBRARY_PATH` appropriately. On Windows, this is most easily accomplished by placing the appropriate directory into the System environment variable `PATH`. However, if more than one generic Apache instance is running on the same machine, then some other mechanism might be needed.

Setting Up a Static Configuration with mod_oc4j

A simple configuration can be constructed using generic Apache and only `mod_oc4j`.

In this configuration, the host and port of all OC4J instances must be statically configured. There is no automatic registration of new JVMs, nor are failed JVMs ever removed from the routing table used by `mod_oc4j`. The advantage of this configuration is its simplicity, including the fact that it does not require the availability of other Oracle Application Server infrastructure components, such as ONS. The following is an example of such a configuration. This configuration loads `mod_oc4j` and provides routing of all requests starting with `/j2ee/` to two different JVMs, both located on the same machine, one at port 3001, and the other at port 3002:

```
LoadModule onsint_module libexec/mod_onsint.so
LoadModule oc4j_module libexec/mod_oc4j.so
Oc4jMount /j2ee/* ajp13://localhost:3001,localhost:3002
```

On Windows, change the line used to load `mod_oc4j` to the following:

```
LoadModule onsint_module modules/ApacheModuleOnsint.dll
LoadModule oc4j_module modules/ApacheModuleOc4j.dll
```

On UNIX, this assumes that the mod_oc4j.so file will be copied into the libexec directory within the Apache installation. On Windows, it means that the ApacheModuleOc4j.dll file will be copied to the modules directory within the Apache installation.

In order to utilize mod_oc4j, *ORACLE_HOME* must be set to point to an Oracle Application Server. On UNIX, this can be accomplished by adding the setting of *ORACLE_HOME* to the apachectl script used to start the generic Apache instance. On Windows, this is most easily accomplished by setting *ORACLE_HOME* as a System environment variable.

Setting Up a Dynamic Configuration with mod_oc4j and mod_onsint

In order to provide full mod_oc4j functionality including dynamic detection of new JVMs and Oracle Application Server installations, mod_oc4j must be combined with mod_onsint.

In order to utilize mod_onsint and mod_oc4j, *ORACLE_HOME* must be set to point to an Oracle Application Server instance where OPMN is running. On UNIX, this can be accomplished by adding the setting of *ORACLE_HOME* to the apachectl script used to start the generic Apache instance. On Windows, this is most easily accomplished by setting *ORACLE_HOME* as a System environment variable.

The following configuration shows how to load the modules and mount /j2ee to the OC4J instance, myinstance, running within the Oracle Application Server cluster, mycluster.

```
LoadModule onsint_module libexec/mod_onsint.so
LoadModule oc4j_module libexec/mod_oc4j.so
Oc4jMount /j2ee/* cluster://mycluster:myinstance
```

Any allowable Oc4jMount syntax available from within Oracle HTTP Server is available when used with generic Apache. This configuration supports all of the same routing and availability features that mod_oc4j offers when running within Oracle HTTP Server, including dynamic discovery of new OC4J processes and instances as they are added and failover of both stateless and session based requests.

Integrating with Oracle Process Manager and Notification Server

Oracle Process Manager and Notification Server (OPMN) can be configured to provide process management, such as starting, stopping, and restart capability for a generic Apache installation. To do this, the Apache instance must have mod_onsint configured. It should have the standard Apache directory layout, that is, the directory structure created by doing a standard Apache 1.3 installation.

> **Note:** When generic Apache is started using OPMN, you have to set OpmnHostPort directive in httpd.conf or Apache will fail to start. For more information on OpmnHostPort, refer to "mod_onsint" on page 8-17

To configure OPMN to manage this Apache instance, the following changes must be made to opmn.xml:

In the module section add the GENERIC_APACHE module-id to the configuration for libopmnohs, such as:

```
<module path="$ORACLE_HOME/opmn/lib/libopmnohs">
```

```
  <module-id id="OHS"/>
  <module-id id="GENERIC_APACHE"/>
</module>
```

In the `HTTP_Server` section, you must set the module to `GENERIC_APACHE` and set an `apache-home`, such as:

```
</ias-component>
  <ias-component id="HTTP_Server">
    <process-type id="HTTP_Server" module-id="GENERIC_APACHE">
      <module-data>
        <category id="start-parameters">
          <data id="apache-home" value="/private/my/path/to/APACHE"/>
        </category>
      </module-data>
    <process-set id="HTTP_Server" numprocs="1"/>
  </process-type>
</ias-component>
```

You can configure either an Oracle HTTP Server instance or a generic Apache instance into any `opmn.xml`. Configuring both in the same `opmn.xml` is currently not supported.

See Also: "opmn.xml" on page E-3

D

Load Balancing Using mod_oc4j

This chapter contains information about `mod_oc4j` load balancing, including metric-based load balancing. Topics include:

- Load Balancing Policies
- Load Balancing Parameters
- Metric-based Load Balancing

Load Balancing Policies

This section contains information about load balancing policies that `mod_oc4j` supports:

- Random
- Round Robin
- Random with Local Affinity
- Round Robin with Local Affinity
- Random using Routing Weight
- Round Robin using Routing Weight
- Metric Based
- Metric Based with Local Affinity

Random

`mod_oc4j` randomly selects an OC4J instance from a list of OC4J instances that are candidates to service a request.

Round Robin

`mod_oc4j` randomly selects an OC4J instance from an ordered list of OC4J instances that are candidates to service a request. Other OC4J instances are selected from the ordered list in turn, until the initially selected server is selected again. This sequence is repeated. If a particular OC4J instance is stopped or is unavailable, then that instance is skipped (no attempt is made to select it) until it can be brought back in service.

Random with Local Affinity

mod_oc4j randomly selects local OC4J processes to service requests. When no local OC4J processes are available, mod_oc4j randomly selects remote OC4J processes and gives them equal opportunity to be selected.

Round Robin with Local Affinity

mod_oc4j routes all requests to local OC4J processes in a round robin manner. When no local processes are available, mod_oc4j routes requests equally to each OC4J process on different hosts.

Random using Routing Weight

mod_oc4j distributes requests according to the routing weight configured for each host. One OC4J process is selected randomly from the OC4J processes on that host.

Round Robin using Routing Weight

mod_oc4j distributes the total request load to OC4J processes on each host based on the routing weight configured to each host. mod_oc4j selects an OC4J process in round robin manner from the OC4J processes on that host.

Metric Based

mod_oc4j routes requests based on run time metrics from OC4J processes that indicate how much load can be placed on the OC4J process.

Metric Based with Local Affinity

mod_oc4j routes all requests to local OC4J processes based on the run time performance metrics of OC4J processes. When there are no local OC4J processes available, mod_oc4j routes requests to each OC4J process on different hosts as per their performance metrics only.

Load Balancing Parameters

This section discusses the following load balancing parameters:

- Oc4jSelectMethod
- Oc4jRoutingWeight

Oc4jSelectMethod

Selects an OC4J instance for load balancing.

Category	Value
Syntax	Oc4jSelectMethod roundrobin | roundrobin:local | roundrobin:weighted | random | random:local | random:weighted | metric | metric:local
Required	No
Default	If Oc4jSelectMethod is not specified, it defaults to "Oc4jSelectMethod roundrobin".

Category	Value
Example	`Oc4jSelecctMethod random:local` `Oc4jSelecctMethod metric`
Usage	■ `Oc4jSelectMethod random`: Selects an OC4J process according to "Random" load balancing policy. ■ `Oc4jSelectMethod roundrobin:weighted`: Selects an OC4J process according to "Round Robin using Routing Weight" load balancing policy. ■ `Oc4jSelecctMethod metric:local:` Selects an OC4J process according to "Metric Based with Local Affinity" load balancing policy.

This directive is only applicable to the base server for Oracle Application Server 10*g* Release 2 (10.1.2) and an error will be printed at startup if specified within a `VirtualHost` container.

Oc4jRoutingWeight

Associates a request routing weight for each machine during load balancing. Weighted routing is a load balancing strategy that distributes requests according to a predefined value assigned to each machine based on the predicted ability to handle load.

Category	Value
Syntax	`Oc4jRoutingWeight` <*node_name*> <*routing_weight*>
Required	No
Default	It defaults to OC4J processes on all the nodes with routing weight as 1. If `Oc4jRoutingWeight` is specified, but some hosts are not specified, it defaults to OC4J processes on any non-specified node with routing weight as 1.

Category	Value

Example	■ There are two hosts in an Oracle Application Server cluster: Host_A and Host_B. Each has Oracle HTTP Server and OC4J processes running on them.
	`Oc4jSelectMethod random:local`
	`Oc4jRoutingWeight Host_A 3`
	`Oc4jRoutingWeight Host_B 2`
	`Oc4jRoutingWeight` directives are ignored. `mod_oc4j` on Host_A randomly routes all requests to OC4J processes on Host_A, `mod_oc4j` on Host_B randomly routes all requests to OC4J processes on Host_B.
	■ There are four hosts in an Oracle Application Server cluster: Host_A, Host_B, Host_C, and Host_D. Each has Oracle HTTP Server and OC4J processes running on them.
	`Oc4jSelectMethod roundrobin:weighted`
	`Oc4jRoutingWeight Host_A 3`
	`Oc4jRoutingWeight Host_B 2`
	`mod_oc4j` on all the machines route three times the number of requests to OC4J processes running on Host_A, two times the number of requests on Host_B, one time the number of requests on Host_C, and one time the number of requests on Host_D in a round robin manner.
	■ There are four hosts in an Oracle Application Server cluster: Host_A, Host_B, Host_C, and Host_D. Each has Oracle HTTP Server and OC4J processes running on them.
	`Oc4jSelectMethod roundrobin:weighted`
	`mod_oc4j` on all the machines route requests equally to OC4J processes on Host_A, Host_B, Host_C, and Host_D in a round robin manner.

Usage	`Oc4jRoutingWeight` is taken into account only when Oc4jSelectMethod specifies weighted. "`Oc4jRoutingWeight` <*node_name*> <*routing_weight*>" associates a request routing weight to each node. `node_name` can be in host name or IP address format. For hosts with multiple interfaces, if different interfaces are specified, it is assumed that they are different hosts.

This directive is only applicable to the base server for Oracle Application Server 10*g* Release 2 (10.1.2) and an error will be printed at startup if specified within a `VirtualHost` container.

Metric-based Load Balancing

Metric-based load balancing is a way to distribute request load among OC4Js based on a "health" metric that each OC4J reports. The metric range is between 0 and 100, where 0 is very busy, or unhealthy, and 100 is not busy, or healthy. When metric-based load balancing is enabled, requests are distributed among OC4Js based on a ratio of a metric received for an individual OC4J, divided by the total of the metrics received from all the OC4Js.

For example, OC4J process p1 reports a metric of 20, process p2 reports a metric of 40, and process p3 reports a metric of 90. The requests would be distributed as follows:

- p1 is routed 20 out of every 150 requests (13%)
- p2 is routed 40 out of every 150 requests (27%)
- p3 is routed 90 out of every 150 requests (60%)

You must configure Oracle HTTP Server and OC4J to enable metric-based load balancing. The following sections contain the required configuration information:

- Configuring Oracle HTTP Server
- Configuring OC4J

Configuring Oracle HTTP Server

On the Oracle HTTP Server side, specify "`Oc4jSelectMethod metric`" or "`Oc4jSelectMethod metric:local`" in mod_oc4j.conf.

See Also:

- "Oc4jSelectMethod" on page D-2
- "Metric Based" on page D-2
- "Metric Based with Local Affinity" on page D-2

Configuring OC4J

On the OC4J side, you must configure the metric collector in *ORACLE_HOME*`/j2ee/home/config/server.xml` in UNIX or *ORACLE_HOME*`/j2ee\home\config\server.xml` on Windows. Configuring the `<metric-collector>` element tells OC4J to start sending a metric to `mod_oc4j` so that `mod_oc4j` can make routing decisions to load balance incoming requests to a list of available OC4J instances.

The metric sent from OC4J to `mod_oc4j` is used only when metric-based load balancing is specified for `mod_oc4j` and when OC4J runs in an Oracle Application Server environment.

If you specify metric-based load balancing in `mod_oc4j` and do not specify the `<metric-collector>` element in `server.xml`, then `mod_oc4j` expects OC4J to send metrics, but OC4J does not send metrics. In this case, `mod_oc4j` reports the following warning message:

```
No run time metrics for oc4j(opmnid=%s) in notification Oc4jSelectMethod is
configured to use run time metrics, please make sure OC4J side is configured
accordingly. Default to 50.
```

In this case, `mod_oc4j` uses the value "50" for each of the OC4J processes and continues.

Likewise, if you specify the `<metric-collector>` element in `server.xml`, but do not specify metric-based load balancing in `mod_oc4j`, then OC4J sends metrics but `mod_oc4j` is not configured to receive metrics. In this case, `mod_oc4j` ignores the metrics and uses whatever the configured method is for load balancing. You specify the load balancing method with Oc4jSelectMethod. If no `Oc4jSelectMethod` is specified, then `mod_oc4j` uses the default, which is roundrobin.

All OC4Js that are used from this Oracle HTTP Server instance must be configured identically. Otherwise, the number returned from the OC4Js will not be comparable and can produce some very poor load balancing results.

When `mod_oc4j` receives a notification containing the metrics information from OC4J, it immediately changes the request routing behavior. The default interval between notifications from OC4J is 30 seconds. This value can be configured using the system property `opmnPingInterval`, which is passed on the command line when OC4J is started by OPMN. To change the interval between notifications, specify the following in `opmn.xml` under the `OC4J <process-set>` configuration element:

```
<module-data>
  <category id="start-parameters">
    <data id="java-options" value="-DopmnPingInterval=<new ping interval value>"/>
  </category>
</module-data>
```

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

Specifying Metrics for OC4J

The following two methods can be used to specify metrics for OC4J:

- Configuring Metric-based Load Balancing to Use the DMSMetricCollector
- Building Your Own Metric Collector

Configuring Metric-based Load Balancing to Use the DMSMetricCollector

In this out of the box method, the `<metric-collector>` element takes a single attribute: `classname`. This attribute defines an interface for gathering and calculating a server-wide metric. Use `oracle.oc4j.server.DMSMetricCollector` for the `classname` attribute when using a DMS-based metric collector. A `DMSMetricCollector` instance takes several parameters.

The DMS metric is specified using the 'dms-noun' parameter, which is shown in the configuration example below. This is the metric on which the `DMSMetricCollector` bases its calculation. The recommended DMS metric for metric-based load balancing is `/oc4j/default/WEBS/processRequest.time`. This metric represents the processing time of the servlets in the default Web application.

The value sent to OC4J is a weighted average of the value computed based on the current DMS metric value, and the last value computed the last time a value was sent. The default weight is 0.7 for the current value, and 0.3 for the previous value. To modify the weights, one may set the "`history-proportion`" as shown the following example. This results in a weight of 0.8 for the current value and 0.2 for the previous value.

```
<metric-collector classname="oracle.oc4j.server.DMSMetricCollector">
   <init-param>
     <param-name>
       dms-noun
     </param-name>
     <param-value>
       /oc4j/default/WEBs/processRequest.time
     </param-value>
   </init-param>
   <init-param>
     <param-name>
       history-proportion
     </param-name>
     <param-value>
       0.2
     </param-value>
   </init-param>
   <init-param>
     <param-name>
       debug
     </param-name>
     <param-value>
```

```
      false
    </param-value>
  </init-param>
</metric-collector>
```

> **See Also:** *Oracle Application Server Performance Guide* for a list of DMS metrics.

How DMS Metrics are Converted for Metric-based Load Balancing When `getMetrics()` is called, the value of the DMS metric specified by the `dms-noun` parameter is obtained. The `delta` with the previous measurement is computed.

Since the scale is from 0 to 100, and the measurement is potentially unbounded, the following formula is applied:

```
metric = 100 / (1 + (log (1 + delta)))
```

As outlined above, this metric should match the current situation, but also reflect on previous metric history. You can assign a weight of 1/3 to the previous history and 2/3 for the average just collected in order to form the new metric.

This new metric becomes the next metric's history, and factors in less and less as time passes. As shown in the configuration example, you can specify the proportion of the previous metric that enters in new metrics, by setting the `history-proportion` parameter to a floating point value between 0.0 and 1.0. Higher values mean that historical values matter more, making the metric less volatile. A lower value makes the metric reflect the most recent metric. The metric returned is as follows:

```
smoothedmetric = ((1 - history-proportion) * metric) + (history-proportion *
previousmetric)
```

The `smoothedmetric` is what is sent to `mod_oc4j` for load balancing purposes.

Building Your Own Metric Collector

You can implement the interface oracle.oc4j.api.MetricCollector to supply your own metric collector to `mod_oc4j`. The metric has to be between 0 and 100.

All metric collectors must implement the interface `oracle.oc4j.api.MetricCollector`. The concrete metric collector must have a constructor with no parameters so it can be instantiated.

The schema elements for the feature are in `server.xml`, and look like:

```
<metric-collector classname="my.package.name.MyClassName">
  <init-param>
    <param-name>
      mysetting
    </param-name>
    <param-value>
      12345
    </param-value>
  </init-param>
</metric-collector>
```

As per the preceding example, 0 or more parameters can be set on the metric collector, and are passed to it when `setParameters()` method is called. `setEnabled(true)` is called once after `setParameters()`. It signals the metric collector that it can begin gathering data as needed.

> **Note:** If the custom metric collector starts threads, the threads need to be daemon threads. Otherwise they may prevent the server from shutting down in an orderly fashion.

After `oracle.oc4j.api.MetricCollector` has been implemented, package your metric in a jar file and add it to your library path, using the following, in `server.xml`:

```
<library path="<path to>/mymetric.jar"/>
```

oracle.oc4j.api.MetricCollector Here is the `oracle.oc4j.api.MetricCollector` interface:

```
package oracle.oc4j.api;

 import java.util.Map;

 /**
  * Defines an interface for gathering and obtaining a server-wide metric.
  * The metric is used in iAS mode, by mod_oc4j, to load balance between
  * virtual oc4j instances.
  * The metric value is relative, and should be between 0 and 100, both
  * inclusive.
  * When configured for metric load balancing,
  * Mod_oc4j will route preferably to an oc4j with the greater value.
  * <p>Concrete instances of this class must have a public empty constructor in
  * order to be loaded and instantiated.
  */
 public interface MetricCollector {
   /**
    * Support for debugging: This is a property name to set to true in order
    * to display the metric that is sent from the server
    */
   String OC4J_METRIC_DEBUG_PROPERTY = "oc4j.metric.debug";

   /**
    * Debugging flag that depends on @{link #OC4J_METRIC_DEBUG_PROPERTY}
    */
   boolean DEBUG = Boolean.getBoolean( OC4J_METRIC_DEBUG_PROPERTY );

   /**
    * Initial metric to return when no measurement has been made (property key)
    */
   String OC4J_INITIAL_METRIC_PROPERTY = "oc4j.metric.initial";
   /**
    * Initial metric to return when no measurement has been made.
    * Default is 50
    */
   int INITIAL_METRIC = Integer.getInteger( OC4J_INITIAL_METRIC_PROPERTY, 50
).intValue();

   /**
    * Enabled flag for the collector.
    * @return true if the collector is collecting data
    */
   boolean isEnabled();

   /**
```

```
   */
  void setEnabled( boolean enabled);

  /**
   * The parameters the metric collector is configured with.
   * This method will be called even when the set of parameters is null.
   * @param params the key/value pairs the metric collector is configured with,
   * or <code>null</code> if none
   */
  void setParameters( Map params );

  /**
   * @return a metric between 0 and 100, inclusive. 100 is better, 0 is worse
   */
  int getMetric();

}
```

E

Configuration Files

This appendix lists commonly used Oracle HTTP Server configuration files.

Files discussed are:

- dms.conf
- httpd.conf
- iaspt.conf
- mime.types
- mod_oc4j.conf
- mod_osso.conf
- opmn.xml
- oracle_apache.conf
- php.ini
- ssl.conf

dms.conf

Enables you to monitor performance of site components with Oracle's Dynamic Monitoring Service (DMS).

It is located at:

- UNIX: *ORACLE_HOME*/Apache/Apache/conf
- Windows: *ORACLE_HOME*\Apache\Apache\conf

See Also: *Oracle Application Server Performance Guide*

httpd.conf

This is a server configuration file which typically contains directives that affect how the server runs, such as user and group IDs it should use, and location of other files. Because the server configuration file is the main file that the server starts with, Oracle HTTP Server does not include any directive that says where to locate it. The location is passed on command line when the server starts.

It is located at:

- UNIX: *ORACLE_HOME*/Apache/Apache/conf
- Windows: *ORACLE_HOME*\Apache\Apache\conf

You should use only this file, and not `srm.conf` or `access.conf` because it is much easier to manage a single configuration file.

Note: If you have an Oracle Application Server installation in `/home/your_directory/orahome` and it is linked to `/private/your_directory/orahome`, the files in the installation are accessible from either `/home/your_directory/orahome` or `/private/your_directory/orahome`.

After installation, the `httpd.conf` file contains an entry for all the files that are included in it that use the original Oracle home path. For example:

```
include /home/your_directory/orahome/Apache/Apache/conf/dms.conf
```

Do not replace the original Oracle home path with the linked Oracle home path.

---

httpd.conf File Structure

`httpd.conf` is arranged in the following sections:

- Global Environment
- Main Server Configuration
- Virtual Hosts Parameters

Global Environment

This is section one of the `httpd.conf` file. It contains configuration directives dealing with Oracle HTTP Server.

See Also:

- "Specifying File Locations" on page 3-3
- "Configuring the Number of Processes and Connections" on page 4-2
- "Specifying Listener Ports and Addresses" on page 5-1

Main Server Configuration

This is section two of the `httpd.conf` file. It contains the directives of the default server.

See Also: "Setting Server and Administrator Functions" on page 3-1.

Virtual Hosts Parameters

This is section three of the `httpd.conf` file. It contains parameters specific to virtual hosts, which override some of the main server configuration defaults.

iaspt.conf

Configures the port tunneling process. Port tunneling allows all communication between Oracle HTTP Server and OC4J to happen on a single, or a small number of ports.

It is located at:

- UNIX: *ORACLE_HOME*/iaspt/conf
- Windows: *ORACLE_HOME*\iaspt\conf

See Also: "Understanding Port Tunneling" on page 10-7

mime.types

Controls the Multi Internet media types that are sent to the client for the given file extensions. Sending the correct media type to the client is important so that the client knows how to handle the content of the file. You can add extra types in the mime type file or add an AddType directive in the configuration file.

It is located at:

- UNIX: *ORACLE_HOME*/Apache/Apache/conf
- Windows: *ORACLE_HOME*\Apache\Apache\conf

See Also: "mod_mime" on page 8-8

mod_oc4j.conf

Configures and loads the mod_oc4j module, and is enabled by default. It routes requests from Oracle HTTP Server to OC4J, and therefore contains routing information.

It is located at:

- UNIX: *ORACLE_HOME*/Apache/Apache/conf
- Windows: *ORACLE_HOME*\Apache\Apache\conf

See Also: "mod_oc4j" on page 8-9

mod_osso.conf

Configures mod_osso, which enables single sign-on for Oracle HTTP Server.

It is located at:

- UNIX: *ORACLE_HOME*/Apache/Apache/conf
- Windows: *ORACLE_HOME*\Apache\Apache\conf

See Also: "mod_osso" on page 8-19

opmn.xml

Describes the processes that Oracle Process Manager and Notification Server (OPMN) manages within an Oracle Application Server installation.

The opmn.xml file is the main configuration file for OPMN. It contains information for the ONS, the PM, and Oracle Application Server component-specific configuration. opmn.xml shows you which Oracle Application Server components OPMN is managing on your system. It contains Oracle Application Server component entries arranged in the following hierarchical structure:

```
<ias-component>
  <process-type>
```

```
<process-set>
```

- **<ias-component>**: This entry represents the Oracle Application Server component. It enables management of the component for processes such as starting and stopping.
- **<process-type>**: This subcomponent of the `<ias-component>` entry declares the type of process to run by association with a specific PM module.
- **<process-set>**: This sub-subcomponent of the `<ias-component>` entry enables you to declare different sets of optional runtime arguments and environments for the Oracle Application Server component.

`opmn.xml` is located at:

- UNIX: *ORACLE_HOME*`/opmn/conf`
- Windows: *ORACLE_HOME*`\opmn\conf`

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

oracle_apache.conf

Stores configuration files of supported modules. It contains directives to include the following configuration files:

- aqxml.conf
- moddav.conf
- ojsp.conf
- plsql.conf
- uix.conf

> **Note:** For the Oracle Application Server Infrastructure install type, another configuration file is included by `oracle_apache.conf` called `oracle_ocm.conf`. It contains configuration for Oracle Application Server Certificate Authority.

aqxml.conf

Enables and configures Advanced Queuing.

It is located at:

- UNIX: *ORACLE_HOME*`/Apache/Apache/conf`
- Windows: *ORACLE_HOME*`\Apache\Apache\conf`

moddav.conf

Configures and loads the `mod_oradav` module to enable distributed authoring and versioning of Web documents.

It is located at:

- UNIX: *ORACLE_HOME*`/Apache/oradav/conf`
- Windows: *ORACLE_HOME*`\Apache\oradav\conf`

See Also:

- Chapter 9, "Configuring and Using mod_oradav"
- "mod_oradav" on page 8-18

ojsp.conf

Configures Java Server Pages.

It is located at:

- UNIX: *ORACLE_HOME*/Apache/jsp/conf
- Windows: *ORACLE_HOME*\Apache\jsp\conf

plsql.conf

Configures and loads the PL/SQL module.

It is located at:

- UNIX: *ORACLE_HOME*/Apache/modplsql/conf
- Windows: *ORACLE_HOME*\Apache\modplsql\conf

See Also: "mod_plsql" on page 8-22

uix.conf

Configures Oracle uix.

It is located at:

- UNIX: *ORACLE_HOME*/uix
- Windows: *ORACLE_HOME*\uix

oiddas.conf

Configures the OiD DAS module.

It is located at:

- UNIX: *ORACLE_HOME*/ldap/das
- Windows: *ORACLE_HOME*\ldap\das

php.ini

Configures mod_php. This file should not be renamed as PHP looks for this specific file name.

It is located at:

- UNIX: *ORACLE_HOME*/Apache/Apache/conf
- Windows: *ORACLE_HOME*\Apache\Apache\conf

See Also: "mod_php" on page 8-22

ssl.conf

`ssl.conf` includes the SSL definitions and virtual host container. Out of the box, SSL is disabled by default.

It is located at:

- UNIX: *ORACLE_HOME*`/Apache/Apache/conf`
- Windows: *ORACLE_HOME*`\Apache\Apache\conf`

See Also: Chapter 11, "Enabling SSL for Oracle HTTP Server"

F

Frequently Asked Questions

This appendix provides answers to frequently asked questions about Oracle HTTP Server.

> **See Also:** "Frequently Asked Questions" in the Apache Server documentation.

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Creating Application-specific Error Pages

Oracle HTTP Server has a default content handler for dealing with errors. You can use the `ErrorDocument` directive to override the defaults.

> **See Also:** "ErrorDocument directive" in the Apache Server documentation.

Offering HTTPS to ISP (Virtual Host) Customers

For HTTP, Oracle HTTP Server supports two types of virtual hosts: name-based and IP-based. HTTPS supports only IP-based virtual hosts.

If you are using IP-based virtual hosts for HTTP, then the customer has a virtual server listening on port 80 of a per-customer IP address. To provide HTTPS for these customers, simply add an additional virtual host per user listening on port 4443 of that same per-customer IP address and use SSL directives, such as Using mod_ossl Directives to specify the per-customer SSL characteristics. Note that each customer can have their own wallet and server certificate.

If you are using name-based virtual hosts for HTTP, each customer has a virtual server listening on port 80 of a shared IP address. To provide HTTPS for those customers, you can add a single shared IP virtual host listening on port 4443 of the shared IP address. All customers will share the SSL configuration, including the wallet and ISP's server certificate.

> **See Also:** "Running Oracle HTTP Server as Root" on page 4-4

Using Oracle HTTP Server as Cache

You can use Oracle HTTP Server as a cache by setting the `ProxyRequests` to "On" and `CacheRoot` directives.

> **See Also:** "`ProxyRequests` and `CacheRoot` directives in the Apache Server documentation.

Using Different Language and Character Set Versions of Document

You can use *multiviews*, a general name given to the Apache server's ability to provide language and character-specific document variants in response to a request.

> **See Also:** "Multiviews" in the Apache Server documentation.

Using OracleAS Web Cache as Front-end

You can use directives such as `ExpiresActive`, `ExpiresByType`, `ExpiresDefault`, to set the length of time that any cache existing between the client and the Web server will cache the returned Web pages.

> **See Also:** "ExpiresActive, ExpiresByType, ExpiresDefault directives" in the Apache Server documentation.

Sending Proxy Sensitive Requests to HTTP Server Behind a Firewall

You should use the Proxy directives, and not the Cache directives, to send proxy sensitive requests across firewalls.

mod_oc4j Information

`mod_oc4j` is a module that integrates with Web servers, typically Oracle HTTP Server, and routes request to the backend OC4J processes. OPMN module keeps `mod_oc4j` aware of the status of different OC4J processes, so `mod_oc4j` routes only to the processes that are up and running. `mod_oc4j` also understands the concepts of Oracle Application Server Clusters and OC4J islands, and routes accordingly to provide as much transparent failover as possible.

> **See Also:** "mod_oc4j" on page 8-9

mod_oc4j Compatibility with Other Web Servers

`mod_oc4j` supports other Web servers including IIS, Sun ONE, and non-Oracle HTTP Server Apache Servers.

mod_oc4j Communication to OC4J using SSL

The AJP communication between `mod_oc4j` and OC4J processes can now be over AJP/SSL. Previously, this was in the clear. Also, the SSL negotiation does not happen each time `mod_oc4j` and OC4J communicate, resulting in less performance impact.

> **See Also:** "Enabling SSL between mod_oc4j and OC4J" on page 8-15

Oracle HTTP Server Version Number

Oracle HTTP Server is based on Apache version 1.3.31.

Applying Apache Security patches to Oracle HTTP Server

You cannot apply the Apache security patches to Oracle HTTP Server for the following reasons:

- Oracle tests and appropriately modifies security patches before releasing them to Oracle HTTP Server users.
- In many cases those alerts may not be applicable, for example, `openSSL` alerts, since Oracle has removed those components from the stack in use.
- Oracle releases these patches soon enough that the time-delay impact of getting the patch from Oracle versus open source organization should be minimal and the benefit with respect to supportability, tremendous.

Compressing Output from Oracle HTTP Server

In general, Oracle recommends the use of OracleAS Web Cache for this purpose. There are other freeware modules, such as `mod_gzip` that may be plugged in for this purpose, but their use is not supported. When using these, there may be an error message with respect to EAPI, but in general that can be ignored.

Supporting PHP

`mod_php` is fully supported in Release 2 (10.1.2).

See Also: "mod_php" on page 8-22

Creating Namespace that Works Across Firewalls, Clusters, Web Cache

The general idea is that all servers in a distributed Web site should agree on a single URL namespace. Every server serves some part of that namespace, and is able to redirect or proxy requests for URLs that it does not serve to a server that is "closer" to that URL. For example, your namespaces could be the following:

```
/app1/login.html
/app1/catalog.html
/app1/dologin.jsp
/app2/orderForm.html
/apps/placeOrder.jsp
```

We could initially map this namespace to two Web servers by putting `app1` on server1 and `app2` on server2. Server1's configuration might look like the following:

```
Redirect permanent /app2 http://server2/app2
Alias /app1 /myApps/application1
<Directory /myApps/application1>
  ...
</Directory>
```

Server2's configuration is complementary. If you decide to partition the namespace by content type (HTML on server, JSP on server2), change server configuration and move files around, but do not have to make changes to the application itself. The resulting configuration of server1 might look like the following:

```
RedirectMatch permanent (.*) \.jsp$ http://server2/$1.jsp
AliasMatch ^/app(.*) \.html$ /myPages/application$1.html
<DirectoryMatch "^/myPages/application\d">
  ...
</DirectoryMatch>
```

Note that the amount of actual redirection can be minimized by configuring a hardware load balancer like F5 system's BigIP to send requests to server1 or server2 based on the URL.

Protecting Web Site From Hackers

There are many attacks, and new attacks are invented everyday. The following are some general guidelines for securing your site. You can never be completely secure, but you can avoid being an easy target.

- Use a commercial firewall, such as Checkpoint FW-1 or Cisco PIX between your ISP and your Web server. Recognize, however, that not all hackers are outside your organization.
- Use switched ethernet to limit the amount of traffic a compromised server can sniff. Use additional firewalls between Web server machines and highly sensitive internal servers running database and enterprise applications.
- Remove unnecessary network services such as RPC, Finger, telnet from your server machine.
- Carefully validate all input from Web forms. Be especially wary of long input strings and input that contains non-printable characters, HTML tags, or javascript tags.
- Encrypt or randomize the contents of cookies that contain sensitive information. For example, it should be difficult to guess a valid sessionID to prevent a hacker from hijacking a valid session.
- Check often for security patches for all your system and application software, and install them as soon as possible. Be sure these patches come from bona fide sources; download from trusted sites and verify the cryptographic checksum.
- Use an intrusion detection package to monitor for defaced Web pages, viruses, and presence of "rootkits" that indicate hackers have broken in. If possible, mount system executables and Web content on read-only file systems.
- Have a "forensic analysis" package on hand to capture evidence of a break in as soon as detected. This aids in prosecution of the hackers.

G

Troubleshooting Oracle HTTP Server

This appendix describes common problems that you might encounter when using Oracle HTTP Server, and explains how to solve them.

It contains the following topics:

Problems and Solutions

This section describes common problems and solutions. It contains the following topics:

Intermittent HTTP-500 errors

Certain Microsoft Internet Explorer security patches have resulted in intermittent HTTP-500 errors, such as `MOD_OC4J_0145`, `MOD_OC4J_0119`, `MOD_OC4J_0013` errors, when the `KeepAlive` directive is set on "On" in Oracle HTTP Server.

Problem

Intermittent HTTP-500 errors caused by bug in Microsoft Internet Explorer.

Solution

There are two possible solutions for this problem:

- Patch all the client Internet Explorer browsers.
- If the preceding option is not practical, set `KeepAlive` to "Off" in Oracle HTTP Server.

Consult Metalink Note 269980.1 on `http://metalink.oracle.com` for details regarding this issue. The easiest way to access the note is to click the Advanced Search button at the top of the Oracle*Metalink* site, and search for Doc ID "269980.1".

See Also: "KeepAlive" on page 5-4

Firewall Between Oracle HTTP Server and OC4J Blocks Connections

Oracle HTTP Server is unable to forward requests to OC4J when certain firewalls are used between them.

Problem

Oracle HTTP Server processes maintain persistent connections with OC4J processes. If the firewall times out a connection before Oracle HTTP Server does, then requests to the OC4J processes can result in errors, or can take a very long time, depending on how the firewall and the operating system are configured.

Solution

Set the Oracle HTTP Server directive `OC4JConnTimeout` to a value less than that of the firewall timeout (this is firewall specific).

See Also: "Oc4jConnTimeout" on page 8-10

Client IP Address Not Passed Through OracleAS Web Cache

When client IP is not passed through OracleAS Web Cache, it can create issues such as:

- Client IP security does not work for allow/deny.
- `Access_log` has OracleAS Web Cache IP address instead of the client's IP address.
- OC4J `request.getRemoteAddr()` returns OracleAS Web Cache IP address.

Problem

Normally, when OracleAS Web Cache is used, its IP address is the address seen by Oracle HTTP Server. You can use the `UseWebCacheIp` configuration option, which is not set by default, to enable Oracle HTTP Server to obtain the IP address of the client.

Solution

Set `UseWebCacheIp On` in your `httpd.conf` file.

See Also: "Obtaining Client IP Address" on page 5-4

Certificate Information Lost When Using OracleAS Web Cache

Certificate information is not available to OC4J applications when OracleAS Web Cache is used.

Problem

When SSL is terminated at OracleAS Web Cache, information is not passed to Oracle HTTP Server and OC4J applications by default.

Solution

Use Certheaders feature, which enables the passing of SSL information, such as configuration information, from OracleAS Web Cache to Oracle HTTP Server.

See Also: "mod_certheaders" on page 8-3

Oracle HTTP Server Unable to Start Due to Port Conflict

You can get the following error if Oracle HTTP Server is unable to start due to port conflict:

```
[crit] (98) Address already in use: make_sock: could not bind to
port 7778
```

Problem

Oracle HTTP Server is unable to start as its port number is being used by another process.

Solution

Determine what process is already using the port by pointing a browser at the address assigned to Oracle HTTP Server and viewing the results. Depending on the results, either change the IP:port address of Oracle HTTP Server, or that of the conflicting process.

Machine Overloaded by Number of HTTPD Processes

When there are too many httpd processes running on a machine, the response time plummets.

Problem

When too many httpd processes are started, there are insufficient resources for normal processing.

Solution

Lower value of `MaxClients` to a value the hardware box can accommodate.

See Also: "MaxClients" on page 4-3

Permission Denied When Starting Oracle HTTP Server on Port Below 1024

You will get the following errors if you try to start Oracle HTTP Server on port below 1024:

Bind errors on ports below 1024: `PERMISSION DENIED: MAKE_SOCK: COULD NOT BIND TO PORT 443.`

Problem

Oracle HTTP Server will not start on ports below 1024 because root privileges are needed to bind these ports. Also, steps to configure `.apachectl` have not been followed.

Solution

Perform the following steps to enable Oracle HTTP Server to run as `root` on ports below 1024:

1. Log in as `root`.
2. Run the following commands in the middle-tier Oracle home:

   ```
   cd ORACLE_HOME/Apache/Apache/bin
   ```

```
chown root .apachectl
chmod 6750 .apachectl
```

Oracle HTTP Server May Fail To Start If PM Files Are Not Located Correctly

Oracle HTTP Server may encounter the following error, and fail to start:

```
"[error] Can't locate mod_perl.pm in @INC (@INC contains:$ORACLE_HOME/perl/...)
```

or,

```
[error] Can't locate Apache::Registry.pm in @INC (@INC contains: $ORACLE_
HOME/per/...)
```

Problem

`mod_perl` needs to locate PM files kept under the *`ORACLE_HOME`*`/Apache/Apache/mod_perl` directory. Without these PM files, `mod_perl` will not start.

Solution

For UNIX, check that `apachectl` has correctly defined in the variable `PERL5LIB`. It should point to *`ORACLE_HOME`*`/Apache/Apache/mod_perl/lib/site_perl/5.6.1/sun4-solaris`

For Windows, check that the environment sub-section in the HTTP Server section in `opmn.xml` has a correct entry for `PERL5LIB`. It should point to *`ORACLE_HOME`*`\Apache\Apache\mod_perl\lib\site_perl\5.6.1\lib`

SSO Client Authentication Fails with Webcache Reverse Proxy

SSO client authentication fails with Webcache reverse proxy.

Problem

During SSO client login, the client certificate should be authenticated from the browser with the SSO server and connect successfully. However, it fails because the `ssoServer.log` shows it is trying to authenticate the certificate stored in the Webcache wallet and not the one from the browser.

Solution

Perform the following steps:

1. Edit `$ORACLE_HOME/Apache/Apache/conf/httpd.conf` and make sure it has the following:

   ```
   LoadModule certheaders_module libexec/mod_certheaders.so
   AddCertHeader HTTPS
   AddCertHeader SSL_CLIENT_CERT
   ```

2. Edit the `$ORACLE_HOME/sso/conf/sso_apache.conf`, and comment out the following line:

   ```
   #SSLOptions +ExportCertData +StdEnvVars
   ```

3. Run `dcmctl updateconfig -ct ohs`
4. Run `opmnctl restartproc type=ohs`
5. Test that the SSO server can be logged into with client authentication.

Need More Help?

You can find more solutions on Oracle*MetaLink*, http://metalink.oracle.com. If you do not find a solution for your problem, log a service request.

> **See Also:** *Oracle Application Server Release Notes*, available on the Oracle Technology Network: http://www.oracle.com/technology/documentation/index.html

H

Third Party Licenses

This appendix includes the Third Party License for all the third party products included with Oracle Application Server.

Topics discussed are:

Apache HTTP Server

Under the terms of the Apache license, Oracle is required to provide the following notices. However, the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Apache software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the Apache software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or Apache.

The Apache Software License

```
/* ====================================================================
 * The Apache Software License, Version 1.1
 *
 * Copyright (c) 2000-2002 The Apache Software Foundation.  All rights
 * reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
```

```
 *    distribution.
 *
 * 3. The end-user documentation included with the redistribution,
 *    if any, must include the following acknowledgment:
 *       "This product includes software developed by the
 *        Apache Software Foundation (http://www.apache.org/)."
 *    Alternately, this acknowledgment may appear in the software itself,
 *    if and wherever such third-party acknowledgments normally appear.
 *
 * 4. The names "Apache" and "Apache Software Foundation" must
 *    not be used to endorse or promote products derived from this
 *    software without prior written permission. For written
 *    permission, please contact apache@apache.org.
 *
 * 5. Products derived from this software may not be called "Apache",
 *    nor may "Apache" appear in their name, without prior written
 *    permission of the Apache Software Foundation.
 *
 * THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED
 * WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
 * DISCLAIMED.  IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR
 * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
 * USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
 * ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
 * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
 * OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 * ====================================================================
 *
 * This software consists of voluntary contributions made by many
 * individuals on behalf of the Apache Software Foundation.  For more
 * information on the Apache Software Foundation, please see
 * <http://www.apache.org/>.
 *
 * Portions of this software are based upon public domain software
 * originally written at the National Center for Supercomputing
Applications,
 * University of Illinois, Urbana-Champaign.
```

Apache SOAP

Under the terms of the Apache license, Oracle is required to provide the following notices. However, the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Apache software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the Apache software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or Apache.

Apache SOAP License

Apache SOAP license 2.3.1

```
TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION
1. Definitions.
```

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual,

worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

   You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with

the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

DBI Module

Oracle is required to provide the text of the third-party license, but the third-party program will be subject to the Oracle license, and Oracle will NOT provide warranties and technical support for the third-party technology.

This program contains third-party code from DBI. Under the terms of the DBI license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the

Oracle program, including the DBI software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the DBI software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or DBI.

You may distribute under the terms of either the GNU General Public License or the Artistic License, as specified in the Perl README file.

Perl Artistic License

The "Artistic License"

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions

"Package" refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.

"Standard Version" refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.

"Copyright Holder" is whoever is named in the copyright or copyrights for the package.

"You" is you, if you're thinking about copying or distributing this Package.

"Reasonable copying fee" is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

"Freely Available" means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.
2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.
3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:
    a. place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.

b. use the modified Package only within your corporation or organization.

c. rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.

d. make other distribution arrangements with the Copyright Holder.

4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:

a. distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.

b. accompany the distribution with the machine-readable source of the Package with your modifications.

c. give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.

d. make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.

6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whoever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package through the so-called "undump" or "unexec" methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.

7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.

8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.

9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.

10. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED

WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

Perl

Oracle is required to provide the text of the third-party license, but the third-party program will be subject to the Oracle license, and Oracle will NOT provide warranties and technical support for the third-party technology.

This program contains third-party code from Perl. Under the terms of the Perl license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Perl software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the Perl software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or Perl.

Perl Kit Readme

Copyright 1989-2001, Larry Wall

All rights reserved.

This program is free software; you can redistribute it and/or modify it under the terms of either:

1. the GNU General Public License as published by the Free Software Foundation; either version 1, or (at your option) any later version, or
2. the "Artistic License" which comes with this Kit.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See either the GNU General Public License or the Artistic License for more details.

You should have received a copy of the Artistic License with this Kit, in the file named "Artistic". If not, I'll be glad to provide one.

You should also have received a copy of the GNU General Public License along with this program in the file named "Copying". If not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307, USA or visit their Web page on the internet at http://www.gnu.org/copyleft/gpl.html.

For those of you that choose to use the GNU General Public License, my interpretation of the GNU General Public License is that no Perl script falls under the terms of the GPL unless you explicitly put said script under the terms of the GPL yourself. Furthermore, any object code linked with perl does not automatically fall under the terms of the GPL, provided such object code only adds definitions of subroutines and variables, and does not otherwise impair the resulting interpreter from executing any standard Perl script. I consider linking in C subroutines in this manner to be the moral equivalent of defining subroutines in the Perl language itself. You may sell such an object file as proprietary provided that you provide or offer to provide the Perl source, as specified by the GNU General Public License. (This is merely an alternate way of specifying input to the program.) You may also sell a binary produced by the dumping of a running Perl script that belongs to you, provided that you provide or offer to provide the Perl source as specified by the GPL. (The fact that a Perl interpreter and your code are in the same binary file is, in this case, a form of mere aggregation.) This

is my interpretation of the GPL. If you still have concerns or difficulties understanding my intent, feel free to contact me. Of course, the Artistic License spells all this out for your protection, so you may prefer to use that.

mod_perl License

```
/* ====================================================================
 * The Apache Software License, Version 1.1
 *
 * Copyright (c) 1996-2000 The Apache Software Foundation.  All rights
 * reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * 3. The end-user documentation included with the redistribution,
 *    if any, must include the following acknowledgment:
 *       "This product includes software developed by the
 *        Apache Software Foundation (http://www.apache.org/)."
 *    Alternately, this acknowledgment may appear in the software itself,
 *    if and wherever such third-party acknowledgments normally appear.
 *
 * 4. The names "Apache" and "Apache Software Foundation" must
 *    not be used to endorse or promote products derived from this
 *    software without prior written permission. For written
 *    permission, please contact apache@apache.org.
 *
 * 5. Products derived from this software may not be called "Apache",
 *    nor may "Apache" appear in their name, without prior written
 *    permission of the Apache Software Foundation.
 *
 * THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED
 * WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
 * DISCLAIMED.  IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR
 * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
 * USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
 * ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
 * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
 * OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 * ====================================================================
 */
```

Perl Artistic License

The "Artistic License"

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions

"Package" refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.

"Standard Version" refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.

"Copyright Holder" is whoever is named in the copyright or copyrights for the package.

"You" is you, if you're thinking about copying or distributing this Package.

"Reasonable copying fee" is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

"Freely Available" means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.
2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.
3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:
   a. place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.
   b. use the modified Package only within your corporation or organization.
   c. rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.
   d. make other distribution arrangements with the Copyright Holder.
4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:
   a. distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.

   b. accompany the distribution with the machine-readable source of the Package with your modifications.

   c. give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.

   d. make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.

6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whoever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package through the so-called "undump" or "unexec" methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.

7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.

8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.

9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.

10. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

PHP

Oracle is required to provide the text of the third-party license, but the third-party program will be subject to the Oracle license, and Oracle will NOT provide warranties and technical support for the third-party technology.

This program contains third-party code from PHP. Under the terms of the PHP license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the

Oracle program, including the PHP software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the PHP software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or PHP.

The PHP License

```
The PHP License, version 3.0
Copyright(c) 1999-2004 The PHP Group. All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, is permitted provided that the following conditions
are met:

  1. Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

  3. The name "PHP" must not be used to endorse or promote products
     derived from this software without prior written permission. For
     written permission, please contact group@php.net.

  4. Products derived from this software may not be called "PHP", nor
     may "PHP" appear in their name, without prior written permission
     from group@php.net.  You may indicate that your software works in
     conjunction with PHP by saying "Foo for PHP" instead of calling
     it "PHP Foo" or "phpfoo"

  5. The PHP Group may publish revised and/or new versions of the
     license from time to time. Each version will be given a
     distinguishing version number.
     Once covered code has been published under a particular version
     of the license, you may always continue to use it under the terms
     of that version. You may also choose to use such covered code
     under the terms of any subsequent version of the license
     published by the PHP Group. No one other than the PHP Group has
     the right to modify the terms applicable to covered code created
     under this License.

  6. Redistributions of any form whatsoever must retain the following
     acknowledgment:
     "This product includes PHP, freely available from
     <http://www.php.net/>".

THIS SOFTWARE IS PROVIDED BY THE PHP DEVELOPMENT TEAM ``AS IS'' AND
ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE PHP
DEVELOPMENT TEAM OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
OF THE POSSIBILITY OF SUCH DAMAGE.
```

mod_dav

mod_dav has been licensed to Oracle free of charge by Greg Stein under a license similar to the Apache Software Foundation license. The following copyright notice applies to mod_dav and Oracle's use of mod_dav:

```
Copyright © 1998-2001 Greg Stein. All rights reserved.

Redistribution and use in source and binary forms, with or without modification,
are permitted provided that the following conditions are met:

  1. Redistributions of source code must retain the above copyright notice,
     this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in the
     documentation and/or other materials provided with the distribution.

  3. All advertising materials mentioning features or use of this software
     must display the following acknowledgment:

          This product includes software developed by Greg Stein
          <gstein@lyra.org> for use in the mod_dav module for Apache
          (http://www.webdav.org/mod_dav/).

  4. Products derived from this software may not be called "mod_dav" nor may
     "mod_dav" appear in their names without prior written permission of
     Greg Stein. For written permission, please contact gstein@lyra.org.

  5. Redistributions of any form whatsoever must retain the following
     acknowledgment:

          This product includes software developed by Greg Stein
          <gstein@lyra.org> for use in the mod_dav module for Apache
          (http://www.webdav.org/mod_dav/).

THIS SOFTWARE IS PROVIDED BY GREG STEIN ``AS IS'' AND ANY EXPRESSED OR
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO
EVENT SHALL GREG STEIN OR THE SOFTWARE'S CONTRIBUTORS BE LIABLE FOR ANY
DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

  ------------------------------------------------------------------------
Greg Stein
Last modified: Thu Feb 3 17:34:42 PST 2000
```

FastCGI

Oracle is required to provide the text of the third-party license, but the third-party program will be subject to the Oracle license, and Oracle will NOT provide warranties and technical support for the third-party technology.

This program contains third-party code from FastCGI. Under the terms of the FastCGI license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use

the Oracle program, including the FastCGI software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the FastCGI software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or FastCGI.

FastCGI Developer's Kit License

This FastCGI application library source and object code (the "Software") and its documentation (the "Documentation") are copyrighted by Open Market, Inc ("Open Market"). The following terms apply to all files associated with the Software and Documentation unless explicitly disclaimed in individual files.

Open Market permits you to use, copy, modify, distribute, and license this Software and the Documentation solely for the purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions.

No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this Software and Documentation may be copyrighted by their authors and need not follow the licensing terms described here, but the modified Software and Documentation must be used for the sole purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other purpose. If modifications to this Software and Documentation have new licensing terms, the new terms must protect Open Market's proprietary rights in the Software and Documentation to the same extent as these licensing terms and must be clearly indicated on the first page of each file where they apply.

Open Market shall retain all right, title and interest in and to the Software and Documentation, including without limitation all patent, copyright, trade secret and other proprietary rights.

OPEN MARKET MAKES NO EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE SOFTWARE OR THE DOCUMENTATION, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL OPEN MARKET BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY DAMAGES ARISING FROM OR RELATING TO THIS SOFTWARE OR THE DOCUMENTATION, INCLUDING, WITHOUT LIMITATION, ANY INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES OR SIMILAR DAMAGES, INCLUDING LOST PROFITS OR LOST DATA, EVEN IF OPEN MARKET HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS". OPEN MARKET HAS NO LIABILITY IN CONTRACT, TORT, NEGLIGENCE OR OTHERWISE ARISING OUT OF THIS SOFTWARE OR THE DOCUMENTATION.

Module mod_fastcgi License

This FastCGI application library source and object code (the "Software") and its documentation (the "Documentation") are copyrighted by Open Market, Inc ("Open Market"). The following terms apply to all files associated with the Software and Documentation unless explicitly disclaimed in individual files.

Open Market permits you to use, copy, modify, distribute, and license this Software and the Documentation solely for the purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other

purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions.

No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this Software and Documentation may be copyrighted by their authors and need not follow the licensing terms described here, but the modified Software and Documentation must be used for the sole purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other purpose. If modifications to this Software and Documentation have new licensing terms, the new terms must protect Open Market's proprietary rights in the Software and Documentation to the same extent as these licensing terms and must be clearly indicated on the first page of each file where they apply.

Open Market shall retain all right, title and interest in and to the Software and Documentation, including without limitation all patent, copyright, trade secret and other proprietary rights.

OPEN MARKET MAKES NO EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE SOFTWARE OR THE DOCUMENTATION, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL OPEN MARKET BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY DAMAGES ARISING FROM OR RELATING TO THIS SOFTWARE OR THE DOCUMENTATION, INCLUDING, WITHOUT LIMITATION, ANY INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES OR SIMILAR DAMAGES, INCLUDING LOST PROFITS OR LOST DATA, EVEN IF OPEN MARKET HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS". OPEN MARKET HAS NO LIABILITY IN CONTRACT, TORT, NEGLIGENCE OR OTHERWISE ARISING OUT OF THIS SOFTWARE OR THE DOCUMENTATION.

Glossary

Apache

Apache is a public domain HTTP server derived from the National Center for Supercomputing Applications (NCSA).

Application Server Control Console

See **Oracle Enterprise Manager 10g Application Server Control Console**.

authentication

The process of verifying the identity of a user, device, or other entity in a host system, often as a prerequisite to granting access to resources in a system. A recipient of an authenticated message can be certain of the message's origin (its sender). Authentication is presumed to preclude the possibility that another party has impersonated the sender.

availability

The percentage or amount of scheduled time that a computing system provides application service.

CA

See **certificate authority**.

certificate

Also called a **digital certificate**. An ITU x.509 v3 standard data structure that securely binds an identity to a public key.

A certificate is created when an entity's public key is signed by a trusted identity, a **certificate authority** The certificate ensures that the entity's information is correct and that the public key actually belongs to that entity.

A certificate contains the entity's name, identifying information, and public key. It is also likely to contain a serial number, expiration date, and information about the rights, uses, and privileges associated with the certificate. Finally, it contains information about the certificate authority that issued it.

certificate authority

A trusted third party that certifies that other entities—users, databases, administrators, clients, servers—are who they say they are. When it certifies a user, the certificate authority first seeks verification that the user is not on the certificate revocation list (CRL), then verifies the user's identity and grants a certificate, signing it with the certificate authority's private key. The certificate authority has its own certificate and public key which it publishes. Servers and clients use these to verify signatures the

certificate authority has made. A certificate authority might be an external company that offers certificate services, or an internal organization such as a corporate MIS department.

CGI

Common Gateway Interface (CGI) is the industry-standard technique for transferring information between a Web server and any program designed to accept and return data that conforms to the CGI specifications.

ciphertext

Data that has been encrypted. Cipher text is unreadable until it has been converted to plain text (decrypted) with a key. See **decryption**.

cipher suite

A set of authentication, encryption, and data integrity algorithms used for exchanging messages between network nodes. During an SSL handshake, for example, the two nodes negotiate to see which cipher suite they will use when transmitting messages back and forth.

cleartext

See **plaintext**.

cryptography

The art of protecting information by transforming it (encrypting) into an unreadable format. See **encryption**.

DAD

See **database access descriptor**.

database access descriptor

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 username (which also specifies the schema and the privileges), password, connect-string, error log file, standard error message, and national language support (NLS) parameters such as NLS language, NLS date format, NLS date language, and NLS currency.

DCM

See **Distributed Configuration Management**.

decryption

The process of converting the contents of an encrypted message (**ciphertext**) back into its original readable format (**plaintext**).

DES

Data Encryption Standard. A commonly used symmetric key **encryption** method that uses a 56-bit key.

de-militarized zone

A de-militarized zone (DMZ) is a set of machines that are isolated from the internet by a firewall on one side, and from a company's intranet by a firewall on the other side. This set of machines are viewed as semi-secure. They are protected from the open internet, but are not completely trusted like machines that are inside the second

firewall and part of the company's intranet. In a typical application server setup with a DMZ, only the Web listener and the static content for the Web site are placed in the DMZ. All business logic, databases, and other critical data and systems in the intranet are protected.

Diffie-Hellman key negotiation algorithm

Diffie-Hellman key negotiation algorithm is a method that lets two parties communicating over an insecure channel to agree upon a random number known only to them. Though the parties exchange information over the insecure channel during execution of the Diffie-Hellman key negotiation algorithm, it is computationally infeasible for an attacker to deduce the random number they agree upon by analyzing their network communications. Oracle Advanced Security uses the Diffie-Hellman key negotiation algorithm to generate session keys.

digital certificate

See **certificate**.

digital wallet

See **wallet**.

directory information tree

A hierarchical tree-like structure consisting of the DNs of the directory entries. See **distinguished name**.

distinguished name

The unique name of a directory entry. It comprises all of the individual names of the parent entries back to the root in the **directory information tree**.

Distributed Configuration Management

Distributed Configuration Management (DCM) manages configuration by propagating the cluster-wide configuration for the application server instances and its components. When you add application server instances to the cluster, it is the DCM component that automatically replicates the base configuration to all instances in the cluster. When you modify the cluster-wide configuration, DCM propagates the changes to all application server instances in the cluster.

DIT

See **directory information tree**.

DMZ

See **de-militarized zone**.

DN

See **distinguished name**.

encryption

The process of disguising a message thereby rendering it unreadable to any but the intended recipient. Encryption is performed by translating data into secret code. There are two main types of encryption: **public-key encryption** (or asymmetric-key encryption) and symmetric-key encryption.

entry

In the context of a directory service, entries are the building blocks of a directory. An entry is a collection of information about an object in the directory. Each entry is composed of a set of attributes that describe one particular trait of the object. For example, if a directory entry describes a person, that entry can have attributes such as first name, last name, telephone number, or e-mail address.

failover

The ability to reconfigure a computing system to utilize an alternate active component when a similar component fails.

HTTP

See **Hypertext Transfer Protocol**.

Hypertext Transfer Protocol

Hypertext Transfer Protocol (HTTP) is the underlying format used by the Web to format and transmit messages and determine what actions Web servers and browsers should take in response to various commands. HTTP is the protocol used between Oracle Application Server and clients.

LDAP

See **Lightweight Directory Access Protocol**.

Lightweight Directory Access Protocol

A standard, extensible directory access protocol. It is a common language that **LDAP** clients and servers use to communicate. The framework of design conventions supporting industry-standard directory products, such as the Oracle Internet Directory.

MD5

A hashing algorithm intended for use on 32-bit machines to create digital signatures. MD5 is a **one-way hash function**, meaning that it converts a message into a fixed string of digits that form a **message digest**.

message digest

Representation of text as a string of single digits. It is created using a formula called a **one-way hash function**.

modules

Modules extend the basic functionality of the Web server and support integration between Oracle HTTP Server and other Oracle Application Server components.

Oracle Enterprise Manager 10*g* Application Server Control Console

Oracle Enterprise Manager 10*g* Application Server Control Console (Application Server Control Console) provides Web-based management tools designed specifically for Oracle Application Server. Using the Application Server Control Console, you can monitor and configure the components of your application server. You can deploy applications, manage security, and create and manage Oracle Application Server clusters.

one-way hash function

An algorithm that turns a message into a single string of digits. "One way" means that it is almost impossible to derive the original message from the string of digits. The

calculated **message digest** can be compared with the message digest that is decrypted with a **public key** to verify that the message has not been tampered with.

OPMN

See **Oracle Process Manager and Notification Server**.

Oracle Process Manager and Notification Server

Oracle Process Manager and Notification Server (OPMN) manages Oracle HTTP Server and OC4J processes within an application server instance. It channels all events from different components to all components interested in receiving them.

PEM

Privacy-Enhanced Electronic Mail. An **encryption** technique that provides encryption, authentication, message integrity, and key management.

PL/SQL

PL/SQL is Oracle's proprietary extension to the SQL language. PL/SQL adds procedural and other constructs to SQL that make it suitable for writing applications.

plaintext

Also called cleartext. Unencrypted data in ASCII format.

plug-in

A module that adds a specific feature or service to a larger system. For example, Oracle Application Server Proxy Plug-in, Oracle Application Server SSO Plug-in, or Oracle Application Server Containers for J2EE Plug-in.

port

A port is a number that TCP uses to route transmitted data to and from a particular program.

private key

In **public-key cryptography**, this key is the secret key. It is primarily used for decryption but is also used for encryption with digital signatures. See **public/private key pair**.

proxy server

A proxy server typically sits on a network firewall and allows clients behind the firewall to access Web resources. All requests from clients go to the proxy server rather than directly to the destination server. The proxy server forwards the request to the destination server and passes the received information back to the client. The proxy server channels all Web traffic at a site through a single, secure port; this allows an organization to create a secure firewall by preventing Internet access to internal machines, while allowing Web access.

public key

In **public-key cryptography**, this key is made public to all. It is primarily used for encryption but can be used for verifying signatures. See **public/private key pair**.

public-key cryptography

Encryption method that uses two different random numbers (keys). See **public key** and **public-key encryption**.

public-key encryption

The process where the sender of a message encrypts the message with the public key of the recipient. Upon delivery, the message is decrypted by the recipient using its private key.

public/private key pair

A set of two numbers used for **encryption** and **decryption**, where one is called the **private key** and the other is called the **public key**. Public keys are typically made widely available, while private keys are held by their respective owners. Though mathematically related, it is generally viewed as computationally infeasible to derive the private key from the public key. Public and private keys are used only with asymmetric encryption algorithms, also called **public-key encryption** algorithms, or public-key cryptosystems. Data encrypted with either a public key or a private key from a key pair can be decrypted with its associated key from the key-pair. However, data encrypted with a public key cannot be decrypted with the same public key, and data encrypted with a private key cannot be decrypted with the same private key.

RSA

A **public-key encryption** technology developed by RSA Data Security. The RSA algorithm is based on the fact that it is laborious to factor very large numbers. This makes it mathematically unfeasible, because of the computing power and time required to decode an RSA key.

scalability

A measure of how well the software or hardware product is able to adapt to future business needs.

SHA

See **Secure Hash Algorithm**.

Secure Hash Algorithm

Secure Hash Algorithm assures data integrity by generating a 160-bit cryptographic message digest value from given data. If as little as a single bit in the data is modified, the Secure Hash Algorithm checksum for the data changes. Forgery of a given data set in a way that will cause the Secure Hash Algorithm to generate the same result as that for the original data is considered computationally infeasible.

An algorithm that takes a message of less than 264 bits in length and produces a 160-bit message digest. The algorithm is slightly slower than MD5, but the larger message digest makes it more secure against brute-force collision and inversion attacks.

Secure Shell

Secure Shell (SSH) is a well known protocol and has widely available implementation that provide a secure connection tunneling solution, very similar to what port tunneling offers. SSH provides a daemon on both the client and server sides of a connection. Clients connect to the local daemon rather than connecting directly to the server. The local SSH daemon then establishes a secure connection to the daemon on the server side. Communication is then routed from the client, through the client side daemon to the server side daemon and then on to the actual server. This allows a client/server program that uses an insecure protocol to be tunneled through a secure channel. For our purposes, the disadvantage of SSH is that it requires two hops to occur and that the implementations available do not perform and scale well enough. More information on SSH can be obtained from

http:www.ssh.org

Secure Sockets Layer

Secure Sockets Layer (SSL) is a standard for the secure transmission of documents over the Internet using HTTPS (secure HTTP). SSL uses digital signatures to ensure that transmitted data is not tampered with.

single sign-on

Single sign-on enables a you to authenticate once, combined with strong authentication occurring transparently in subsequent connections to other databases or applications. It lets you access multiple accounts and applications with a single password, entered during a single connection.

SSL

See **Secure Sockets Layer**.

SSH

See **Secure Shell**.

wallet

Also called a **digital wallet**. A wallet is a data structure used to store and manage security credentials for an individual entity. It implements the storage and retrieval of credentials for use with various cryptographic services. A **Wallet Resource Locator** (WRL) provides all the necessary information to locate the wallet.

Wallet Resource Locator

A wallet resource locator (WRL) provides all necessary information to locate a wallet. It is a path to an operating system directory that contains a wallet.

WRL

See **Wallet Resource Locator**.

X.509

Public keys can be formed in various data formats. The X.509 v3 format is one such popular format.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X