

## **Oracle® Fusion Middleware**

Administrator's Guide for Oracle HTTP Server

11g Release 1 (11.1.1)

**E10144-08**

May 2013

This guide describes how to manage Oracle HTTP Server, including how to start and stop Oracle HTTP Server, how to manage network components, configure listening ports, and extend basic functionality using modules.

Oracle Fusion Middleware Administrator's Guide for Oracle HTTP Server, 11g Release 1 (11.1.1)

E10144-08

Copyright © 2002, 2013, Oracle and/or its affiliates. All rights reserved.

Primary Author: Srinivas Sudhindra

Contributors: Jeff Trawick, Ken Vincent, Maria Choudhary, Edwin Spear

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

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

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

---

---

# Contents

<b>Preface</b> .....	xi
Audience .....	xi
Documentation Accessibility .....	xi
Related Documents .....	xii
Conventions .....	xii
<b>Part I Understanding Oracle HTTP Server</b>	
<b>1 Introduction to Oracle HTTP Server</b>	
1.1 What is Oracle HTTP Server .....	1-1
1.1.1 Key Features of Oracle HTTP Server .....	1-2
1.2 Understanding Oracle HTTP Server Directory Structure .....	1-5
1.3 Understanding Configuration Files .....	1-5
1.4 Oracle HTTP Server Support .....	1-6
<b>2 Management Tools for Oracle HTTP Server</b>	
2.1 Overview of Oracle HTTP Server Management .....	2-1
2.2 Accessing Fusion Middleware Control .....	2-1
2.3 Accessing the Oracle HTTP Server Home Page .....	2-2
2.3.1 Navigating Within Fusion Middleware Control .....	2-2
2.4 Using the opmnctl Command-line Tool .....	2-3
<b>3 Understanding Oracle HTTP Server Modules</b>	
3.1 List of Included Modules .....	3-1
3.2 mod_certheaders .....	3-2
3.3 mod_dms .....	3-3
3.4 mod_onsint .....	3-3
3.5 mod_oradav .....	3-4
3.6 mod_ossl .....	3-4
3.7 mod_osso .....	3-5
3.8 mod_perl .....	3-5
3.8.1 Using mod_perl with a Database .....	3-6
3.9 mod_reqtimeout .....	3-8
3.10 mod_plsql .....	3-8

3.10.1	Creating a DAD.....	3-8
3.10.2	Configuration Files for mod_plsql .....	3-10
3.10.3	Using Configuration Files and Parameters.....	3-11
3.11	mod_security .....	3-11
3.11.1	Enabling mod_security .....	3-12
3.11.2	Configuring mod_security .....	3-12
3.12	mod_wl_ohs.....	3-12

## Part II Managing Oracle HTTP Server

### 4 Getting Started with Oracle HTTP Server

4.1	Starting, Stopping, and Restarting Oracle HTTP Server.....	4-1
4.1.1	Understanding the PID File.....	4-2
4.1.2	Starting Oracle HTTP Server.....	4-3
4.1.3	Stopping Oracle HTTP Server.....	4-4
4.1.4	Restarting Oracle HTTP Server.....	4-4
4.2	Creating an Oracle HTTP Server Component .....	4-5
4.3	Specifying Server Properties .....	4-6
4.3.1	Using Fusion Middleware Control to Specify Server Properties .....	4-6
4.3.2	Editing the httpd.conf File to Specify Server Properties .....	4-7
4.4	Configuring Oracle HTTP Server .....	4-8
4.4.1	Configuring Secure Sockets Layer .....	4-9
4.4.2	Configuring MIME Settings .....	4-9
4.4.3	Configuring the mod_perl Module.....	4-11
4.4.4	Configuring mod_wl_ohs.....	4-12
4.4.5	Modifying an Oracle HTTP Server Configuration File.....	4-12
4.4.6	Configuring the Oracle HTTP Server Environment to Use the apxs Script.....	4-12
4.4.7	Disabling the Options Method .....	4-13
4.4.8	Updating the Configuration for OHS Instances on a Shared Filesystem.....	4-14
4.5	Deleting an Oracle HTTP Server Component .....	4-14

### 5 Managing and Monitoring Server Processes

5.1	Oracle HTTP Server Processing Model .....	5-1
5.1.1	Request Process Model .....	5-1
5.1.2	Single Unit Process Model.....	5-2
5.2	Monitoring Oracle HTTP Server Performance .....	5-2
5.2.1	Viewing Oracle HTTP Server Performance Metrics.....	5-2
5.2.2	Understanding Oracle HTTP Server Performance Metrics .....	5-3
5.3	Configuring Oracle HTTP Server Performance Directives.....	5-4
5.3.1	Using Fusion Middleware Control to Set the Request Configuration.....	5-5
5.3.2	Using Fusion Middleware Control to Set the Connection Configuration.....	5-6
5.3.3	Using Fusion Middleware Control to Set the Process Configuration.....	5-6
5.4	Understanding Process Security.....	5-7

### 6 Managing Connectivity

6.1	Viewing Port Number Usage.....	6-1
-----	--------------------------------	-----

6.1.1	Using the Fusion Middleware Control to View Port Number Usage.....	6-2
6.2	Managing Ports .....	6-2
6.2.1	Using Fusion Middleware Control to Create Ports .....	6-3
6.2.2	Using Fusion Middleware Control to Edit Ports .....	6-3
6.2.3	Updating OHS Registration with a WebLogic Domain.....	6-4
6.3	Configuring Virtual Hosts.....	6-4
6.3.1	Using Fusion Middleware Control to Create Virtual Hosts.....	6-5
6.3.2	Using Fusion Middleware Control to Configure Virtual Hosts .....	6-6

## 7 Managing Oracle HTTP Server Logs

7.1	Overview of Server Logs .....	7-1
7.1.1	About Error Logs .....	7-2
7.1.2	About Access Logs.....	7-2
7.1.3	Log Rotation .....	7-3
7.2	Configuring Oracle HTTP Server Logs.....	7-4
7.2.1	Using Fusion Middleware Control to Configure Error Logs .....	7-4
7.2.2	Using Fusion Middleware Control to Configure Access Logs .....	7-6
7.3	Log Directives for Oracle HTTP Server .....	7-8
7.3.1	Oracle Diagnostic Logging Directives .....	7-8
7.3.2	Apache HTTP Server Log Directives .....	7-10
7.4	Viewing Oracle HTTP Server Logs .....	7-11

## 8 Managing Application Security

8.1	About Oracle HTTP Server Security .....	8-1
8.2	Classes of Users and Their Privileges .....	8-2
8.3	Resources Protected.....	8-2
8.4	Authentication, Authorization and Access Control .....	8-3
8.4.1	Access Control.....	8-3
8.4.2	User Authentication and Authorization .....	8-3
8.4.3	Support for FMW Audit Framework.....	8-4

## 9 Configuring mod\_oradav

9.1	Introduction to the mod_oradav Module.....	9-1
9.1.1	WebDAV .....	9-2
9.1.2	OraDAV .....	9-2
9.1.3	OraDAV Architecture .....	9-3
9.1.4	OraDAV Usage Model .....	9-4
9.1.5	PROPFIND Security .....	9-4
9.2	Configuring mod_oradav .....	9-5
9.2.1	OraDAV Configuration Directives.....	9-5
9.2.2	Using Fusion Middleware Control to Configure mod_oradav .....	9-6
9.2.3	Editing mod_oradav.conf .....	9-6
9.3	WebDAV Security Considerations .....	9-7
9.4	OraDAV Performance Considerations .....	9-7
9.4.1	Using Disk Caching with OraDAV .....	9-7
9.4.2	Bypassing Oracle Web Cache for WebDAV Activities .....	9-8

9.5	Globalization Support Considerations with OraDAV .....	9-8
9.6	Location of DAV Files .....	9-9

## Part III Appendices and Glossary

### A Using Oracle Plug-Ins for Third-Party Web Servers

A.1	Using Oracle Proxy Plug-In.....	A-1
A.1.1	Overview of Oracle Proxy Plug-In.....	A-2
A.1.2	Installing Oracle Proxy Plug-In .....	A-2
A.1.3	Configuring Oracle Proxy Plug-In .....	A-3
A.1.4	Configuring Sun Java System Web Server to Use Oracle Proxy Plug-In .....	A-7
A.1.5	Configuring Microsoft IIS 6.0 Listener to Use Oracle Proxy Plug-In.....	A-8
A.1.6	Configuring Microsoft IIS 7.0 Listener to Use Oracle Proxy Plug-In.....	A-10
A.1.7	Concurrently Serving Files From IIS While Proxying Requests to OHS.....	A-12
A.1.8	Oracle Proxy Plug-In Usage Information.....	A-12
A.1.9	Troubleshooting Oracle Proxy Plug-In.....	A-13
A.2	Using Oracle SSO Plug-In.....	A-16
A.2.1	Overview of Oracle SSO Plug-In.....	A-16
A.2.2	Installing Oracle SSO Plug-In .....	A-17
A.2.3	Registering with the Oracle Single Sign-On Server .....	A-18
A.2.4	Configuring the Oracle SSO Plug-In.....	A-18
A.2.5	Configuring Microsoft IIS 6.0 Listener to Use Oracle Single Sign-On .....	A-19
A.2.6	Configuring Microsoft IIS 7.0 Listener to Use Oracle Single Sign-On .....	A-22
A.2.7	Troubleshooting Oracle SSO Plug-In.....	A-23

### B Frequently Asked Questions

B.1	How Do I Create Application-Specific Error Pages? .....	B-1
B.2	What Type of Virtual Hosts Are Supported for HTTP and HTTPS? .....	B-2
B.3	Can I Use Oracle HTTP Server As Cache? .....	B-2
B.4	Can I Use Different Language and Character Set Versions of Document? .....	B-2
B.5	Can I Apply Apache Security Patches to Oracle HTTP Server? .....	B-2
B.6	Can I Upgrade the Apache Version of Oracle HTTP Server?.....	B-3
B.7	Can I Compress Output From Oracle HTTP Server? .....	B-3
B.8	How Do I Create a Namespace That Works Through Firewalls and Clusters? .....	B-3
B.9	How do I Protect the Web Site from Hackers? .....	B-4
B.10	Do I Need to Re-register Partner Applications with the SSO Server If I Disable or Enable SSL? B-4	
B.11	Why REDIRECT_ERROR_NOTES is not set for file-not-found errors? .....	B-4
B.12	How can I hide information about the Web Server Vendor and Version .....	B-5

### C Troubleshooting Oracle HTTP Server

C.1	Oracle HTTP Server Unable to Start Due to Port Conflict.....	C-1
C.2	System Overloaded by Number of httpd Processes .....	C-1
C.3	Permission Denied When Starting Oracle HTTP Server On a Port Below 1024 .....	C-2
C.4	Oracle HTTP Server May Fail To Start If PM Files Are Not Located Correctly .....	C-2

C.5	Unsetting PerSetEnv and Removing a Previously Configured PerSetEnv variable in EM Throws an Exception	C-2
C.6	Using Log Files to Locate Errors.....	C-3
C.6.1	Rewrite Log.....	C-3
C.6.2	Script Log.....	C-3
C.6.3	Error Log.....	C-3
C.7	Client IP Address Not Used in a Configuration with Oracle Web Cache.....	C-3

## D Configuring mod\_security

## E OHS Module Directives

E.1	mod_certheaders.....	E-1
E.1.1	AddCertHeader.....	E-1
E.1.2	SimulateHttps.....	E-2
E.2	mod_onsint.....	E-2
E.2.1	OpmnHostPort.....	E-2
E.3	mod_oradav.....	E-2
E.3.1	ORAAllowIndexDetails.....	E-3
E.3.2	ORAAltPassword.....	E-3
E.3.3	ORACacheDirectory.....	E-4
E.3.4	ORACacheMaxResourceSize.....	E-4
E.3.5	ORACachePrunePercent.....	E-4
E.3.6	ORACacheTotalSize.....	E-5
E.3.7	ORAConnect.....	E-5
E.3.8	ORAConnectSN.....	E-5
E.3.9	ORAContainerName.....	E-6
E.3.10	ORAException.....	E-6
E.3.11	ORAGetSource.....	E-6
E.3.12	ORALockExpirationPad.....	E-7
E.3.13	ORAPackageName.....	E-7
E.3.14	ORAPassword.....	E-7
E.3.15	ORARootPrefix.....	E-8
E.3.16	ORAService.....	E-8
E.3.17	ORATraceEvents.....	E-8
E.3.18	ORATraceLevel.....	E-9
E.3.19	ORAUser.....	E-9
E.4	mod_ssl.....	E-9
E.4.1	SSLAccelerator.....	E-10
E.4.2	SSLCARevocationFile.....	E-10
E.4.3	SSLCARevocationPath.....	E-11
E.4.4	SSLCipherSuite.....	E-11
E.4.5	SSLEngine.....	E-13
E.4.6	SSLLog.....	E-13
E.4.7	SSLLogLevel.....	E-13
E.4.8	SSLMutex.....	E-14
E.4.9	SSLOptions.....	E-14
E.4.10	SSLPassPhraseDialog.....	E-15

E.4.11	SSLProtocol.....	E-16
E.4.12	SSLRequire.....	E-16
E.4.13	SSLRequireSSL.....	E-18
E.4.14	SSLSessionCache.....	E-18
E.4.15	SSLSessionCacheTimeout.....	E-19
E.4.16	SSLVerifyClient.....	E-19
E.4.17	SSLWallet.....	E-20
E.4.18	SSLWalletPassword.....	E-20
E.5	mod_osso.....	E-20
E.5.1	OssoConfigFile.....	E-21
E.5.2	OssoIdleTimeout.....	E-21
E.5.3	OssoIgnoreUri.....	E-21
E.5.4	OssoIpCheck.....	E-21
E.5.5	OssoLegacyApp.....	E-22
E.5.6	OssoProtectedOnly.....	E-22
E.5.7	OssoRedirectByForm.....	E-22
E.5.8	OssoSecureCookies.....	E-22
E.5.9	OssoSendCacheHeaders.....	E-23
E.5.10	OssoHTTPOnly.....	E-23
E.6	mod_plsql.....	E-23
E.6.1	plsql.conf.....	E-23
E.6.2	dads.conf.....	E-25
E.6.3	cache.conf.....	E-43
E.7	mod_wl_ohs.....	E-45
E.7.1	ConnectRetrySecs.....	E-45
E.7.2	ConnectTimeoutSecs.....	E-46
E.7.3	Debug.....	E-46
E.7.4	DebugConfigInfo.....	E-47
E.7.5	DefaultFileName.....	E-47
E.7.6	DynamicServerList.....	E-48
E.7.7	ErrorPage.....	E-48
E.7.8	FileCaching.....	E-48
E.7.9	Idempotent.....	E-49
E.7.10	KeepAliveEnabled.....	E-49
E.7.11	KeepAliveSecs.....	E-49
E.7.12	MatchExpression.....	E-50
E.7.13	MaxPostSize.....	E-50
E.7.14	MaxSkipTime.....	E-51
E.7.15	PathPrepend.....	E-51
E.7.16	PathTrim.....	E-51
E.7.17	QueryFromRequest.....	E-52
E.7.18	WebLogicCluster.....	E-52
E.7.19	WebLogicHost.....	E-53
E.7.20	WebLogicPort.....	E-53
E.7.21	WebLogicSSLVersion.....	E-53
E.7.22	WLCookieName.....	E-54
E.7.23	WLDNSRefreshInterval.....	E-54



E.7.24	WLEXcludePathOrMimeType .....	E-54
E.7.25	WLForwardUriUnparsed .....	E-55
E.7.26	WLIOTimeoutSecs.....	E-55
E.7.27	WLLocalIP .....	E-55
E.7.28	WLLogFile .....	E-56
E.7.29	WLProxyPassThrough .....	E-56
E.7.30	WLProxySSL.....	E-56
E.7.31	WLProxySSLPassThrough .....	E-56
E.7.32	WLServerInitiatedFailover .....	E-57
E.7.33	WLSocketTimeoutSecs.....	E-57
E.7.34	WLSRequest.....	E-57
E.7.35	WLTempDir.....	E-58

## Glossary

## Index



---

---

# Preface

This guide describes how to manage Oracle HTTP Server, including how to start and stop Oracle HTTP Server, how to manage network components, configure listening ports, and extend basic functionality using modules.

## Audience

*Oracle Fusion Middleware Administrator's Guide for Oracle HTTP Server* is intended for application server administrators, security managers, and managers of databases used by application servers. This documentation is based on the assumption that readers are already familiar with Apache HTTP Server.

Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier in Stand-Alone Mode" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*."

## Documentation Accessibility

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

### Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=accid=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=accid=trs> if you are hearing impaired.

### 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.

## Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/support/contact.html> or visit <http://www.oracle.com/accessibility/support.html> if you are hearing impaired.

## Related Documents

For more information, see the following documents in the Oracle Fusion Middleware 11g Release 1 (11.1.1) documentation set:

- *Oracle Fusion Middleware Concepts*
- *Oracle Fusion Middleware Administrator's Guide*
- *Oracle Fusion Middleware High Availability Guide*
- Apache documentation included in this library

---

---

**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.

---

---

## Conventions

The following text conventions are used in this document:

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

# Part I

---

## Understanding Oracle HTTP Server

This part presents introductory and conceptual information about Oracle HTTP Server. It contains the following chapters:

- [Chapter 1, "Introduction to Oracle HTTP Server"](#)
- [Chapter 2, "Management Tools for Oracle HTTP Server"](#)
- [Chapter 3, "Understanding Oracle HTTP Server Modules"](#)



---

---

# Introduction to Oracle HTTP Server

Oracle HTTP Server is the Web server component for Oracle Fusion Middleware. It provides a listener for Oracle WebLogic Server and the framework for hosting static pages, dynamic pages, and applications over the Web.

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier in Stand-Alone Mode" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

This chapter includes the following sections:

- [Section 1.1, "What is Oracle HTTP Server"](#)
- [Section 1.2, "Understanding Oracle HTTP Server Directory Structure"](#)
- [Section 1.3, "Understanding Configuration Files"](#)
- [Section 1.4, "Oracle HTTP Server Support"](#)

## 1.1 What is Oracle HTTP Server

Oracle HTTP Server 11g, Release 1 (11.1.1.7.0) is based on Apache HTTP Server 2.2.22 (with critical bug fixes from higher versions) infrastructure, and includes modules developed specifically by Oracle. The features of single sign-on, clustered deployment, and high availability enhance the operation of the Oracle HTTP Server. Oracle HTTP Server has the following components to handle client requests:

- HTTP listener, to handle incoming requests and route them to the appropriate processing utility.
- Modules (mods), to implement and extend the basic functionality of Oracle HTTP Server. Many of the standard Apache HTTP Server modules are included with Oracle HTTP Server. Oracle also includes several modules that are specific to Oracle Fusion Middleware to support integration between Oracle HTTP Server and other Oracle Fusion Middleware components.
- Perl interpreter, a persistent Perl runtime environment embedded in Oracle HTTP Server through mod\_perl.

Oracle HTTP Server enables developers to program their site in a variety of languages and technologies, such as the following:

- Perl (through mod\_perl and CGI)
- C (through CGI and FastCGI)
- C++ (through FastCGI)
- PHP (through mod\_php)
- Oracle PL/SQL

Oracle HTTP Server can also be a proxy server, both forward and reverse. A reverse proxy enables content served by different servers to appear as if coming from one server.

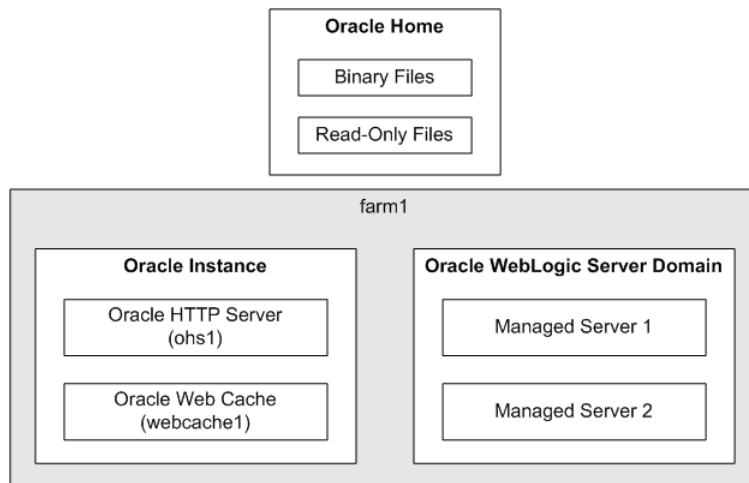
Figure 1–1 shows an example of an Oracle home and a farm (farm1) consisting of an Oracle instance and an Oracle WebLogic Server domain. The Oracle instance in this example includes two components: Oracle HTTP Server (ohs1) and Oracle Web Cache (webcache1). The Oracle WebLogic Server domain includes two managed servers.

---

**Note:** For more information about Fusion Middleware concepts, see the *Oracle Fusion Middleware Concepts*.

---

**Figure 1–1 Oracle Fusion Middleware Farm**



### 1.1.1 Key Features of Oracle HTTP Server

The following sections describe some of the key features of Oracle HTTP Server:

- [Section 1.1.1.1, "Security: Encryption with Secure Sockets Layer"](#)
- [Section 1.1.1.2, "Security: Single Sign-On"](#)
- [Section 1.1.1.3, "Distributed Authoring and Versioning \(DAV\) Support"](#)
- [Section 1.1.1.4, "URL Rewriting and Proxy Server Capabilities"](#)
- [Section 1.1.1.5, "Oracle Process Manager and Notification Server"](#)
- [Section 1.1.1.6, "Oracle Plug-Ins"](#)
- [Section 1.1.1.7, "PL/SQL Server Pages"](#)
- [Section 1.1.1.8, "Server-Side Includes"](#)



- [Section 1.1.1.9, "Perl"](#)
- [Section 1.1.1.10, "PHP"](#)
- [Section 1.1.1.11, "C / C++ \(CGI and FastCGI\)"](#)
- [Section 1.1.1.12, "Load Balancing"](#)

#### 1.1.1.1 Security: Encryption with Secure Sockets Layer

Secure Sockets Layer (SSL) is required to run any Web site securely. Oracle HTTP Server supports SSL encryption based on patented, industry standard, algorithms. SSL works seamlessly with commonly-supported Internet browsers. Security features include the following:

- SSL hardware acceleration support uses dedicated hardware for SSL. Hardware encryption is faster than software encryption.
- Variable security per directory allows individual directories to be protected by different strength encryption.
- Oracle HTTP Server and Oracle WebLogic Server communicate using the HTTP protocol to provide both encryption and authentication. You can also enable HTTP tunneling for the T3 or IIOP protocols to provide non-browser clients access to WebLogic Server services.

**See Also:** *Oracle Fusion Middleware Security Guide*

#### 1.1.1.2 Security: Single Sign-On

Basic authentication for HTTP servers uses a flat file with encrypted passwords. Oracle HTTP Server supports standard authentication as well as single sign-on. The `mod_osso` module is included to support single sign-on across sites and across applications. This security feature provides a better end user experience because users only have to log in once. It also helps the development cycle because most of the security is declarative.

**See Also:**

- *Oracle Fusion Middleware Security Guide*
- [Section 3.7, "mod\\_osso"](#)

#### 1.1.1.3 Distributed Authoring and Versioning (DAV) Support

WebDAV is an HTTP based protocol that allows DAV enabled clients, such as Microsoft Office and Microsoft Windows Explorer, to edit files on a server. Oracle HTTP Server enhances DAV support with the `mod_oradav` module. This module enables WebDAV clients to connect to an Oracle database, read and write content, query, and lock documents in various schemas.

#### 1.1.1.4 URL Rewriting and Proxy Server Capabilities

Active Web sites usually update their Web pages and directory contents often, and possibly their URLs as well. Oracle HTTP Server makes it easy to accommodate the changes by including an engine that supports URL rewriting so end users do not have to change their bookmarks.

Oracle HTTP Server also supports reverse proxy capabilities, making it easier to make content served by different servers to appear from one single server.

### 1.1.1.5 Oracle Process Manager and Notification Server

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

**See Also:**

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

### 1.1.1.6 Oracle Plug-Ins

Oracle provides the following plug-ins to enable third-party web servers to work with Oracle HTTP Server:

- Oracle Proxy Plug-In is a separately-available component that enables Microsoft IIS and Sun Java System Web Server to route requests to Oracle HTTP Server.
- Oracle SSO Plug-In is a separately-available component that enables Microsoft IIS to be integrated with Oracle Fusion Middleware Single Sign-On.

For more information about these plug-ins, see [Appendix A, "Using Oracle Plug-Ins for Third-Party Web Servers."](#)

### 1.1.1.7 PL/SQL Server Pages

PL/SQL Server Pages are similar in concept to the JavaServer Pages. The `mod_plsql` module enables PL/SQL to be used as the scripting language within an HTML page. PL/SQL Server Pages get translated into a stored procedure, which then uses the module to send the output to the browser.

### 1.1.1.8 Server-Side Includes

Server-Side Includes provide an easy way of adding dynamic or uniform static content across all pages on a site. It is typically used for header and footer information. Oracle HTTP Server supports special directives to enable these only for certain types of files, or for specified virtual hosts.

### 1.1.1.9 Perl

Perl is a scripting language often used to provide dynamic content. Perl scripts can either be called as a CGI program, or directly through the `mod_perl` module. Oracle Fusion Middleware uses Perl version 5.10.

**See Also:** [Section 3.8, "mod\\_perl"](#)

### 1.1.1.10 PHP

PHP (Hypertext Preprocessor) is a scripting language capable of being embedded in HTML, which makes it well-suited for Web development.

### 1.1.1.11 C / C++ (CGI and FastCGI)

CGI programs are commonly used to program Web applications. Oracle HTTP Server enhances the programs by providing a mechanism to keep them alive beyond the request lifecycle.

### 1.1.1.12 Load Balancing

Oracle HTTP Server includes the `mod_wl_ohs` module, which routes requests to Oracle WebLogic Server. The `mod_wl_ohs` module provides the same load balancing functionality as the Oracle WebLogic Server plug-in for Apache HTTP Server (`mod_wl`). Note that the modules mentioned in this section are different from the plug-ins described in [Section 1.1.1.6, "Oracle Plug-Ins."](#)

For more information, see "The Dynamic Server List" section of *Using Web Server Plug-Ins with Oracle WebLogic Server*.

## 1.2 Understanding Oracle HTTP Server Directory Structure

Oracle HTTP Server directories are divided between the Oracle home and the Oracle instance. The Oracle home directories are read-only, and contain the Oracle Fusion Middleware binaries. The Oracle instance directories contain the module configuration files and content pages for Oracle HTTP Server. [Table 1–1](#) shows the subdirectories for Oracle HTTP Server in the Oracle home directory.

**Table 1–1 Oracle Home Directories**

Directory	Contents
<code>ohs/bin</code>	Oracle HTTP Server binary files.
<code>ohs/conf</code>	Oracle HTTP Server template configuration files, which get provisioned to an Oracle instance when an Oracle HTTP Server component is configured.  <b>Note:</b> These files should only be edited by advanced Oracle HTTP Server users.
<code>ohs/modules</code>	Oracle HTTP Server modules

[Table 1–2](#) shows the subdirectories for Oracle Fusion Middleware in the Oracle instance directory.

**Table 1–2 Oracle Instance Directories**

Directory Name	Contents
<code>config/OHS/component_name</code>	Oracle HTTP Server configuration files.
<code>config/OHS/component_name/htdocs</code>	Static content and CGI scripts for Oracle HTTP Server.
<code>config/OHS/component_name/moduleconf</code>	Configuration files that are automatically included in Oracle HTTP Server configuration. Be careful not to create any files with a <code>.conf</code> extension in this directory that you do not want to be included in the configuration.
<code>diagnostics/logs/OHS/component_name</code>	Oracle HTTP Server component instance log files.

## 1.3 Understanding Configuration Files

Configuration for Oracle HTTP Server are specified through **directives** in configuration files in the exact same manner as Apache HTTP Server configuration files. For more information about Apache HTTP Server configuration files, see the *Apache HTTP Server 2.2 Users Guide*.

## 1.4 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 including those modules.
- Problems that can be reproduced within an Oracle HTTP Server configuration consisting only of supported Oracle HTTP Server modules.
- Use of the included Perl interpreter with the supported Oracle HTTP Server configuration.

---

---

# Management Tools for Oracle HTTP Server

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier in Stand-Alone Mode" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

Oracle provides the following management tools for Oracle HTTP Server:

- Fusion Middleware Control, which is a browser-based management tool
- `opmnctl`, which is a command-line management tool.

This chapter includes the following sections:

- [Section 2.1, "Overview of Oracle HTTP Server Management"](#)
- [Section 2.2, "Accessing Fusion Middleware Control"](#)
- [Section 2.3, "Accessing the Oracle HTTP Server Home Page"](#)
- [Section 2.4, "Using the `opmnctl` Command-line Tool"](#)

## 2.1 Overview of Oracle HTTP Server Management

The main tool for managing Oracle HTTP Server is Fusion Middleware Control, which is a browser-based tool for administering and monitoring the Oracle Fusion Middleware environment.

**See Also:** *Oracle Fusion Middleware Administrator's Guide*

## 2.2 Accessing Fusion Middleware Control

To display Fusion Middleware Control, you enter the Fusion Middleware Control URL, which includes the name of the WebLogic Administration Server host and the port number assigned to Fusion Middleware Control during the installation. The following shows the format of the URL:

```
http://hostname.domain:port/em
```

If you saved the installation information by clicking **Save** on the last installation screen, the URL for Fusion Middleware Control is included in the file that is written to disk.

1. Display Fusion Middleware Control by entering the URL in your Web browser. For example:

```
http://host1.acme.com:7001/em
```

The Welcome page is displayed:

2. Enter the Fusion Middleware Control administrator user name and password and click **Login**.

The default user name for the administrator user is `weblogic`. This is the account you can use to log in to the Fusion Middleware Control for the first time. The `weblogic` password is the one you supplied during the installation of Fusion Middleware Control.

## 2.3 Accessing the Oracle HTTP Server Home Page

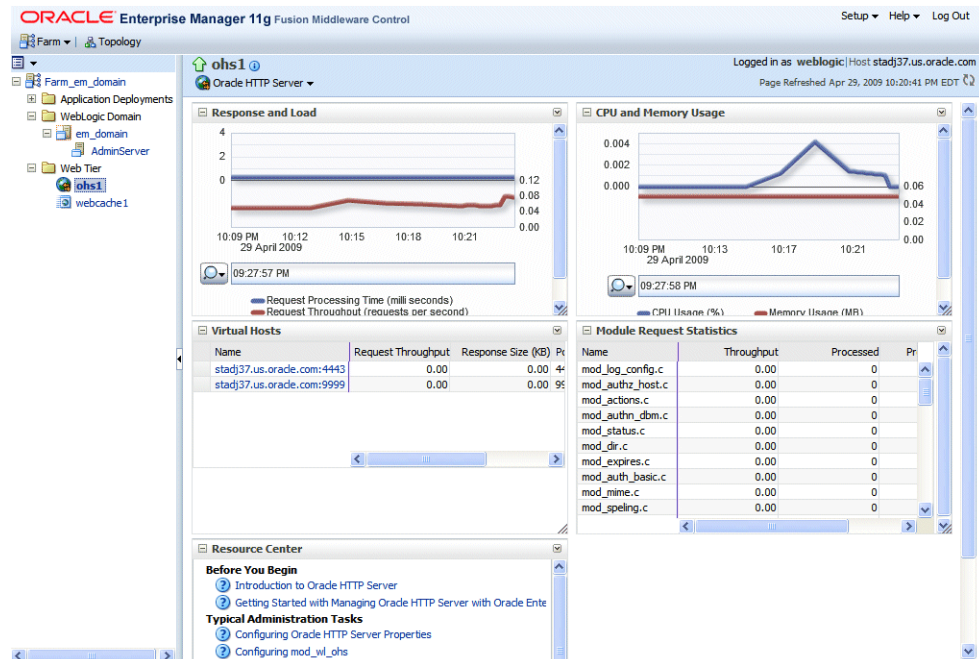
The Oracle HTTP Server Home page in Fusion Middleware Control contains menus and regions that enable you to manage the server. Use the menus for monitoring, managing, routing, and viewing general information.

### 2.3.1 Navigating Within Fusion Middleware Control

When you select a target, such as a WebLogic Managed Server or a component, such as Oracle HTTP Server, the target's home page is displayed in the content pane and that target's menu is displayed at the top of the page, in the context pane. For example, if you select an Oracle HTTP Server instance from the Web Tier folder, the Oracle HTTP Server menu is displayed. You can also view the menu for a target by right-clicking the target in the navigation pane.

[Figure 2–1](#) shows the target navigation pane and the home page of Oracle HTTP Server.

Figure 2–1 Oracle HTTP Server Home Page



The Oracle HTTP Server home page contains the following regions:

- **Response and Load Region:** Provides information such as the number of active requests, how many requests were submitted, and how long it took for Oracle HTTP Server to respond to a request. It also provides information about the number of bytes processed with the requests.
- **CPU and Memory Usage Region:** Shows how much CPU (by percentage) and memory (in megabytes) are being used by an Oracle HTTP Server instance.
- **Virtual Hosts Region:** Shows the virtual hosts for Oracle HTTP Server.
- **Module Request Statistics Region:** Shows the modules for Oracle HTTP Server.
- **Resource Center:** Provides links to books and topics related to Oracle HTTP Server.

---

**See Also:** The *Oracle Fusion Middleware Administrator's Guide* contains detailed descriptions of all the items on the target navigation pane and the home page.

---

## 2.4 Using the opmnctl Command-line Tool

You can use the `opmnctl` command-line interface to start and stop system components, monitor system components, and perform many other tasks related to process management.

The following are some of the `opmnctl` operations that you can perform with Oracle HTTP Server components:

- Create additional components.
- Delete existing components.
- Start and/or stop components.

- Check a component's status.
- Check a component's port usage (see [Chapter 6, "Managing Connectivity"](#)).

**See Also:** *Oracle Process Manager and Notification Server*

The `opmnctl` command is located in the `ORACLE_HOME/opmn/bin` directory in an Oracle home and in the `ORACLE_INSTANCE/bin` directory in an Oracle instance.

---

**Note:** Oracle recommends running `opmnctl` from the same `ORACLE_INSTANCE` in which Oracle HTTP Server is running, unless that instance is unavailable.

If an `ORACLE_INSTANCE` has not been created, then run `opmnctl` from `ORACLE_HOME/opmn/bin`.

---

The following example demonstrates using `opmnctl` in a command shell to start an Oracle HTTP Server component, and to then verify the status of that component.

```
> $ORACLE_INSTANCE/bin/opmnctl startproc ias-component=ohs1
> $ORACLE_INSTANCE/bin/opmnctl status
Processes in Instance: instancel
```

ias-component	process-type	pid	status
webcache1	WebCache-admin	19556	Alive
webcache1	WebCache	19555	Alive
ohs1	OHS	7249	Alive



---

---

## Understanding Oracle HTTP Server Modules

Modules (mods) extend the basic functionality of Oracle HTTP Server, and support integration between Oracle HTTP Server and other Oracle Fusion Middleware components.

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier in Stand-Alone Mode" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

This chapter discusses the modules developed specifically by Oracle for Oracle HTTP Server. It includes the following sections:

- [Section 3.1, "List of Included Modules"](#)
- [Section 3.2, "mod\\_certheaders"](#)
- [Section 3.3, "mod\\_dms"](#)
- [Section 3.4, "mod\\_onsint"](#)
- [Section 3.5, "mod\\_oradav"](#)
- [Section 3.6, "mod\\_oss1"](#)
- [Section 3.7, "mod\\_osso"](#)
- [Section 3.8, "mod\\_perl"](#)
- [Section 3.10, "mod\\_plsql"](#)
- [Section 3.11, "mod\\_security"](#)
- [Section 3.12, "mod\\_wl\\_ohs"](#)

### 3.1 List of Included Modules

This section lists all of the modules bundled with Oracle HTTP Server.

#### Oracle-developed Modules for Oracle HTTP Server

The following modules have developed specifically by Oracle for Oracle HTTP Server:

- [mod\\_certheaders](#)
- [mod\\_dms](#)
- [mod\\_onsint](#)
- [mod\\_oradav](#)
- [mod\\_oss](#)
- [mod\\_osso](#)
- [mod\\_plsql](#)
- [mod\\_security](#)
- [mod\\_wl\\_ohs](#)

### Apache HTTP Server and Third-party Modules in Oracle HTTP Server

Oracle HTTP Server also includes the following Apache HTTP Server and third-party modules out-of-the-box. These modules are *not* developed by Oracle.

**See Also:** For information about Apache HTTP Server modules, including lists and descriptions of their valid directives, see the Apache documentation at:

<http://httpd.apache.org/docs/2.2/mod/>

- |                                       |  |                                      |
|---------------------------------------|--|--------------------------------------|
| ▪ <a href="#">mod_authz_host</a>      | ▪ <a href="#">mod_cgid</a> (Unix only) | ▪ <a href="#">mod_perl</a>           |
| ▪ <a href="#">mod_actions</a>         | ▪ <a href="#">mod_deflate</a>          | ▪ <a href="#">mod_proxy</a>          |
| ▪ <a href="#">mod_alias</a>           | ▪ <a href="#">mod_dir</a>              | ▪ <a href="#">mod_proxy_balancer</a> |
| ▪ <a href="#">mod_asis</a>            | ▪ <a href="#">mod_env</a>              | ▪ <a href="#">mod_proxy_connect</a>  |
| ▪ <a href="#">mod_auth_basic</a>      | ▪ <a href="#">mod_expires</a>          | ▪ <a href="#">mod_proxy_ftp</a>      |
| ▪ <a href="#">mod_auth_dbm</a>        | ▪ <a href="#">mod_fastcgi</a>          | ▪ <a href="#">mod_proxy_http</a>     |
| ▪ <a href="#">mod_authn_anon</a>      | ▪ <a href="#">mod_headers</a>          | ▪ <a href="#">mod_rewrite</a>        |
| ▪ <a href="#">mod_autoindex</a>       | ▪ <a href="#">mod_imagemap</a>         | ▪ <a href="#">mod_reqtimeout</a>     |
| ▪ <a href="#">mod_authn_dbm</a>       | ▪ <a href="#">mod_include</a>          | ▪ <a href="#">mod_setenvif</a>       |
| ▪ <a href="#">mod_authn_file</a>      | ▪ <a href="#">mod_info</a>             | ▪ <a href="#">mod_speling</a>        |
| ▪ <a href="#">mod_authz_groupfile</a> | ▪ <a href="#">mod_log_config</a>       | ▪ <a href="#">mod_status</a>         |
| ▪ <a href="#">mod_authz_host</a>      | ▪ <a href="#">mod_logio</a>            | ▪ <a href="#">mod_unique_id</a>      |
| ▪ <a href="#">mod_authz_user</a>      | ▪ <a href="#">mod_mime</a>             | ▪ <a href="#">mod_userdir</a>        |
| ▪ <a href="#">mod_cern_meta</a>       | ▪ <a href="#">mod_mime_magic</a>       | ▪ <a href="#">mod_usertrack</a>      |
| ▪ <a href="#">mod_cgi</a>             | ▪ <a href="#">mod_negotiation</a>      |                                      |

## 3.2 mod\_certheaders

The `mod_certheaders` module enables reverse proxies that terminate Secure Sockets Layer (SSL) connections in front of Oracle HTTP Server to transfer information regarding the SSL connection, such as SSL client certificate information, to Oracle

HTTP Server and the applications running behind Oracle HTTP Server. This information is transferred from the reverse proxy to Oracle HTTP Server using HTTP headers. The information is then transferred from the headers to the standard CGI environment variable. The mod\_ossl module or the mod\_ssl module populate the variable if the SSL connection is terminated by Oracle HTTP Server.

The mod\_certheaders module also enables certain requests to be treated as HTTPS requests even though they are received through HTTP. This is done using the `SimulateHttps` directive.

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

### 3.3 mod\_dms

---



---

**Note:** mod\_dms is disabled by default in Oracle HTTP Server 11.1.1.

---



---

The mod\_dms module enables you to monitor the performance of site components using Oracle Dynamic Monitoring Service (DMS).

For more information about DMS, see the "Oracle Dynamic Monitoring Service" chapter in the *Oracle Fusion Middleware Performance and Tuning Guide*.

### 3.4 mod\_onsint

The mod\_onsint module provides integration support with Oracle Notification Service (ONS) and Oracle Process Manager and Notification Server (OPMN). It is an Oracle module and provides the following functionality.

- Provides a subscription mechanism for ONS notifications within Oracle HTTP Server. mod\_onsint receives notification for all modules within an Oracle HTTP Server instance.
- Publishes `PROC_READY` ONS notifications so that OPMN knows 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.

mod\_onsint runs as a thread within a Oracle HTTP Server parent process on UNIX and within a child process on Windows. This thread is responsible for sending and receiving ONS messages.

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` has the following syntax that specifies the values to pass to OPMN:

```
OpmnHostPort [<http> | <https>://]<host>:<port>
```

For example, the following line would specify that OPMN should use HTTP, the localhost interface and port 7778 to ping this listener:

```
OpmnHostPort http://localhost:7778
```

For more information about ONS and OPMN, see the "Oracle Process Manager and Notification Overview" chapter in the *Oracle Fusion Middleware Oracle Process Manager and Notification Server Administrator's Guide*.

### **mod\_onsint Directives**

See [Section E.2](#) for a list and descriptions of directives accepted by mod\_onsint.

## **3.5 mod\_oradav**

The mod\_oradav module is an Oracle Call Interface (OCI) application written in C that extends the implementation of mod\_dav. The mod\_oradav directive can read and write to local files or to an Oracle database. The Oracle database must have an OraDAV driver (a stored procedure package) for the mod\_oradav module to map WebDAV activity to database activity. Essentially the mod\_oradav module enables WebDAV clients to connect to an Oracle database, read and write content, and query and lock documents in various schemas.

You can configure the mod\_oradav module using standard Oracle HTTP Server directives. Use the Advanced Server Configuration page of Fusion Middleware Control to configure the mod\_oradav module. The mod\_oradav directive can immediately leverage other module code (such as mime\_magic) to perform content management tasks. Most OraDAV processing activity involves streaming content to and from a content provider. The mod\_oradav directive uses OCI streaming logic directly within Oracle HTTP Server.

### **See Also:**

- [Chapter 9, "Configuring mod\\_oradav"](#)
- *Oracle Portal Administrator's Guide*
- For information about using the mod\_oradav module to access database schemas for access by third-party tools, such as Adobe GoLive, Macromedia Dreamweaver, and Oracle *interMedia*, go to the OraDAV information available on OTN at:

<http://www.oracle.com/technology/index.html>

### **mod\_oradav Directives**

See [Section E.3](#) for a list and descriptions of directives accepted by mod\_oradav.

## **3.6 mod\_ossl**

The mod\_ossl module enables strong cryptography for Oracle HTTP Server. This Oracle module is a plug-in to Oracle HTTP Server that enables the server to use SSL. It is very similar to the OpenSSL module, mod\_ssl. The mod\_ossl module is based on the Oracle implementation of SSL, which supports SSL version 3 and TLS version 1, and is based on Certicom and RSA Security technology.

### **mod\_ossl Directives**

See [Section E.4](#) for a list and descriptions of directives accepted by mod\_ossl.

---

---

**See Also:** For more information, see the "Configuring SSL for the Web Tier" section of the *Oracle Fusion Middleware Administrator's Guide*.

---

---

## 3.7 mod\_osso

---

---

**Note:** mod\_osso is deprecated with Oracle Single Sign-On Server 10g. You should use one of the following options:

- Use the Webgate plug-in; for more information, see one of the following documents:
    - *Oracle® Fusion Middleware Installing WebGates for Oracle Access Manager*
    - Part VII "Using 10g Webgates with Oracle Access Manager 11g" in *Oracle® Fusion Middleware Administrator's Guide for Oracle Access Manager with Oracle Security Token Service*
  - Use the SSO Agent solution provided in Oracle Access Manager 11g, which uses the same mod\_osso module and configuration. For more information, see Chapter 9, "Registering Partners (Agents and Applications) by Using the Console" in *Oracle® Fusion Middleware Administrator's Guide for Oracle Access Manager with Oracle Security Token Service*.
- 
- 

The mod\_osso module enables single sign-on for Oracle HTTP Server by examining incoming requests and determining whether the requested resource is protected. If the resource is protected, then the module retrieves the Oracle HTTP Server cookie.

The module is disabled by default. You can enable mod\_osso module on the Server Configuration Properties page of Fusion Middleware Control. For more information see [Section 4.3.1, "Using Fusion Middleware Control to Specify Server Properties."](#)

### mod\_osso Directives

See [Section E.5](#) for a list and descriptions of directives accepted by mod\_osso.

---

---

**See also:** For information about forced authentication, see the *Oracle Fusion Middleware Application Developer's Guide for Oracle Identity Management*.

For information about single sign-on, see the *Oracle Fusion Middleware Security Guide*.

---

---

## 3.8 mod\_perl

The mod\_perl module embeds the Perl interpreter into Oracle HTTP Server. This eliminates start-up overhead and enables you to write modules in Perl. Oracle Fusion Middleware uses Perl version 5.10.

The module is disabled, by default. To enable the mod\_perl module, follow the instructions in [Section 4.4.3, "Configuring the mod\\_perl Module"](#).

**See Also:** mod\_perl documentation at <http://perl.apache.org/docs/index.html>

### 3.8.1 Using mod\_perl with a Database

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

#### 3.8.1.1 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 Fusion Middleware. It calls Oracle Call Interface (OCI) to access the databases.

Once mod\_perl is enabled, DBI must be enabled in the `mod_perl.conf` file to function. To enable DBI, perform the following steps:

1. Edit the `mod_perl.conf` file:
  - a. In Fusion Middleware Control, navigate to the Oracle HTTP Server Advanced Server Configuration page.
  - b. Select the `mod_perl.conf` file from the menu and click **Go**.
  - c. Add the following line to the `mod_perl.conf` file:

```
PerlModule Apache::DBI
```
2. Click **Apply** to save the file.
3. Restart Oracle HTTP Server as described in [Section 4.1.4, "Restarting Oracle HTTP Server."](#)

Place the Perl scripts that you want to run in the `ORACLE_INSTANCE/config/OHS/component_name/cgi-bin` directory.

#### **Example 3–1 Using a Perl Script to Access a Database**

```
#!ORACLE_HOME/perl/bin/perl -w
use DBI;
my $dataSource = "host=hostname.domain;sid=orclsid;port=1521";
my $userName = "userid";
my $password = "password";
my $dbhhandle = DBI->connect("dbi:Oracle:$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
$dbhhandle->disconnect
    or warn "Database disconnect failed; $DBI::errstr\n";
exit;
```

To run the DBI scripts, the URLs would look like the following:

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

If a script specifies "use Apache::DBI" instead of "use DBI", then it can only run from the URL `http://hostname.domain:port/perl/scriptname`.

### 3.8.1.2 Testing a Database Connection

[Example 3-2](#) shows a sample Perl script for testing a database connection. Replace the instance name, user ID, and password in the `connect` statement with proper values for the target database.

#### **Example 3-2 Sample Perl Script For Testing Connection for Local Seed Database**

```
use DBI;
print "Content-type: text/plain\n\n";
$dbh = DBI->connect("dbi:Oracle:instance_name", userid/password, "") ||
    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";
```

### 3.8.1.3 Using SQL NCHAR Data Types

SQL NCHAR data types (NCHAR, NVARCHAR2 and NCLOB) are reliable Unicode data types. SQL NCHAR data types enable you to store Unicode characters regardless of the database character set. The character set for those data types is specified by the national character set, which is either AL16UTF-16 or UTF8.

[Example 3-3](#) shows an example of accessing SQL NCHAR data.

#### **Example 3-3 Sample Script to Access SQL NCHAR 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
$stmt = $dbh->prepare( 'SELECT * FROM TABLE_N WHERE NCOL1 = :nchar1' );

# bind the parameter of a NCHAR type
$stmt->bind_param( ':nchar1', $param_1 );
# set the character form to NCHAR
$stmt->func( { ':nchar1' => ORA_NCHAR }, 'set_form' );

$stmt->execute;
```

As shown in [Example 3-3](#), the `set_form` function is provided as a private function that you can invoke with the standard DBI `func` method. The `set_form` function takes an anonymous hash that enables you to set the character form for parameters.

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 is `ORA_IMPLICIT`.

The constants are available as `ora_forms` in `DBD::Oracle`.

`set_default_form` sets the default character form for a database handle. The following example shows its syntax:

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

This syntax causes the form of all parameters to be `ORA_NCHAR`, unless otherwise specified with `set_form` calls. Unlike the `set_form` function, the `set_default_form` functions on the database handle, so every statement from the database handle has the form of your choice.

#### **Example 3-4 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' );
```

## 3.9 mod\_reqtimeout

`mod_reqtimeout` sets timeout and minimum data rate for receiving requests. `mod_reqtimeout` module provides protection against some DOS attacks and related issues. For more information, see:

[http://httpd.apache.org/docs/2.2/mod/mod\\_reqtimeout.html](http://httpd.apache.org/docs/2.2/mod/mod_reqtimeout.html)

## 3.10 mod\_plsql

The `mod_plsql` module connects Oracle HTTP Server to an Oracle database, enabling you to create Web applications using Oracle stored procedures.

To access a Web-enabled PL/SQL application, configure a PL/SQL **database access descriptor** (DAD) for the `mod_plsql` module. A DAD is a set of values that specifies how the module connects to a database server to fulfill an HTTP request. Besides the connection details, a DAD contains important configuration parameters for various operations in the database and for the `mod_plsql` module 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.

This section contains the following topics:

- [Section 3.10.1, "Creating a DAD"](#)
- [Section 3.10.2, "Configuration Files for `mod\_plsql`"](#)
- [Section 3.10.3, "Using Configuration Files and Parameters"](#)

### **mod\_plsql Directives**

See [Section E.6](#) for a list and descriptions of directives accepted by `mod_plsql`.

### 3.10.1 Creating a DAD

To create a DAD, perform the following steps:



1. Open the `dads.conf` configuration file.

For the locations of `mod_plsql` configuration files, see [Table 3-1](#).

---

**Note:** You can also open and edit the `dads.conf` file by using Oracle Fusion Middleware Control, on the Oracle HTTP Server Advanced Server Configuration page, as described in [Section 4.4.5, "Modifying an Oracle HTTP Server Configuration File."](#)

---

2. Add the following:

- a. The `<Location>` element, which defines a virtual path used to access the PL/SQL Web Application. This directive groups a set of directives that apply to the named `Location`.

For example, the following directive defines a virtual path called `/myapp` that will be used to invoke a PL/SQL Web application through a URL such as `http://host:port/myapp/`.

```
<Location /myapp>
```

---

**Note:** Earlier releases of the `mod_plsql` module were always mounted on a virtual path with a prefix of `/pls`. This restriction is removed in later releases but might still be a restriction imposed by some of the earlier PL/SQL applications.

---

- b. The `SetHandler` directive, which directs Oracle HTTP Server to enable the `mod_plsql` module 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
```

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

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

- e. The `</Location>` tag to close the `<Location>` element.

3. Save the edits.

4. Obfuscate the DAD password by running the `dadTool.pl` script located in the `ORACLE_HOME/bin` directory.

**See Also:** [Section E.6.2.11, "PlsqlDatabasePassword"](#) for instructions on performing the obfuscation.

5. Restart Oracle HTTP Server as described in [Section 4.1.4, "Restarting Oracle HTTP Server."](#)

You can create additional DADs by defining other uniquely named `<Location>` elements in `dads.conf`.

### Example DADs

The following DAD connects as a specific user and has a default home page:

```
<Location /pls/mydad>
SetHandler pls_handler
Order allow,deny
Allow from All
PlsqlDatabaseUsername scott
PlsqlDatabasePassword tiger
PlsqlDatabaseConnectionString prod_db
PlsqlDefaultPage scott.myapp.home
</Location>
```

The following DAD uses HTTP Basic Authentication and supports document upload/download operations:

```
<Location /pls/mydad2>
SetHandler pls_handler
Order allow,deny
Allow from All
PlsqlDatabaseConnectionString prod_db2
PlsqlDefaultPage scott.myapp.my_home
PlsqlDocumentTablename scott.my_documents
PlsqlDocumentPath docs
PlsqlDocumentProcedure scott.docpkg.process_download
</Location>
```

## 3.10.2 Configuration Files for mod\_plsql

The `mod_plsql` configuration parameters reside in the configuration files that are located in the `ORACLE_INSTANCE` directory, as described in [Table 3-1](#).

**Table 3-1** *mod\_plsql Configuration Files In an Oracle Instance*

Directory Name	Contents
<code>config/OHS/component_name/moduleconf</code>	<code>plssql.conf</code> configuration file.
<code>config/OHS/component_name/mod_plsql</code>	<code>dads.conf</code> and <code>cache.conf</code> configuration files.

The `mod_plsql` configuration directives are used in these configuration files:

- [plssql.conf](#)
- [dads.conf](#)
- [cache.conf](#)

### 3.10.2.1 plssql.conf

The `plssql.conf` file resides in the `ORACLE_INSTANCE/config/OHS/config/OHS/component_name/moduleconf` directory and Oracle HTTP Server automatically loads all `.conf` files under this location. The `plssql.conf` file contains the `LoadModule` directive to load the `mod_plsql` module into Oracle HTTP Server, any global settings for the `mod_plsql` module, and include directives for `dads.conf` and `cache.conf`.

**mod\_plsql Directives in plsql.conf**

See [Section E.6.1](#) for a list and description of the directives used in `plsql.conf`.

**See Also:** The `plsql.README` file, located in `ORACLE_HOME/ohs/mod_plsql`, for a detailed description of `plsql.conf`

**3.10.2.2 dads.conf**

The `dads.conf` file contains the configuration parameters for the PL/SQL database access descriptor. (See [Table 3-1](#) for the file location.) A DAD is a set of values that specifies how the `mod_plsql` module connects to a database server to fulfill a HTTP request.

**mod\_plsql Directives in dads.conf**

See [Section E.6.2](#) for a list and description of the directives used in `dads.conf`.

**3.10.2.3 cache.conf**

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

**mod\_plsql Directives in cache.conf**

See [Section E.6.3](#) for a list and description of the directives used in `cache.conf`.

**3.10.3 Using Configuration Files and Parameters**

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.

**3.11 mod\_security**


---

---

**Notes:** Be aware of the following:

- `mod_security` was removed from earlier versions of Oracle HTTP Server but is reintroduced in version 11.1.1.7. This version follows the recommendations and practices prescribed for open source `mod_security` 2.6.2. Only documentation applicable to open source Apache `mod_security` 2.6.2 is applicable to the Oracle HTTP Server implementation of the module.
  - In Oracle HTTP Server 11.1.1.7 and later, `mod_security` is not loaded or configured by default; however, if you have an installation patched from 11.1.1.6, it might have loaded and configured the module due to the patch implementation.
  - Oracle only supports the Oracle-supplied version of `mod_security`. Newer versions from `modsecurity.org` will not be supported.
- 
-

mod\_security is an open-source module that you can use to detect and prevent intrusion attacks against Oracle HTTP Server; for example, you can specify a mod\_security rule to screen all incoming requests and deny requests that match the conditions specified in the rule. The mod\_security module (version 2.6.2) and its prerequisites are included in the Oracle HTTP Server installation as a shared object named mod\_security2.so in the ORACLE\_HOME/ohs/modules directory.

This version of OHS supports only mod\_security (version 2.6.2) directives, variables, action, phases and functions. *It will not be supported if you replace this module with a later version.* For more information about these directives, see the mod\_security documentation, at:

<http://www.modsecurity.org/documentation/>.

### 3.11.1 Enabling mod\_security

To make the mod\_security module available for use when Oracle HTTP Server is running, ensure that mod\_security.conf begins with the following lines:

```
#Load Module
LoadModule security2_module "${ORACLE_HOME}/ohs/modules/mod_security2.so"
```

as shown in the mod\_security.conf example in [Appendix D, "Configuring mod\\_security"](#). If you are an Oracle HTTP Server 11.1.1.6 user who is now using 11.1.1.7, the module is already configured properly due to previous patch implementations.

### 3.11.2 Configuring mod\_security

Configuring mod\_security involves specifying certain directives in the Oracle HTTP Server configuration file. You can specify the directives directly in the httpd.conf file in an IfModule container. Alternatively, you can specify the mod\_security directives in a separate mod\_security.conf file and include that .conf file in httpd.conf by using the Include directive.

Note that, by default, mod\_security.conf does not exist, so you need to create one for your system. [Appendix D, "Configuring mod\\_security"](#), contains a sample mod\_security.conf file that you can use for this purpose. Copy and paste the sample into a text editor and read the entire file, editing it for your system. Then save it as your own mod\_security.conf and included from your httpd.conf.

## 3.12 mod\_wl\_ohs

The mod\_wl\_ohs module enables requests to be proxied from Oracle HTTP Server 11g to Oracle WebLogic Server.

For information about the prerequisites and procedure for configuring mod\_wl\_ohs, see "Configuring the mod\_wl\_ohs Plug-In for Oracle HTTP Server" in *Using Web Server 1.1 Plug-Ins with Oracle WebLogic server*.

### mod\_wl\_ohs Directives

See [Section E.7](#) for a list and descriptions of directives accepted by mod\_wl\_ohs.

---

---

**Note:** mod\_wl\_ohs is similar to the mod\_wl plug-in, which you can use to proxy requests from Apache HTTP Server to Oracle WebLogic server. However, while the mod\_wl plug-in for Apache HTTP Server should be downloaded and installed separately, the mod\_wl\_ohs plug-in is included in the Oracle HTTP Server installation.

---

---



# Part II

---

## Managing Oracle HTTP Server

This part presents information about management tasks for Oracle HTTP Server. It contains the following chapters:

- [Chapter 4, "Getting Started with Oracle HTTP Server"](#)
- [Chapter 5, "Managing and Monitoring Server Processes"](#)
- [Chapter 6, "Managing Connectivity"](#)
- [Chapter 7, "Managing Oracle HTTP Server Logs"](#)
- [Chapter 8, "Managing Application Security"](#)
- [Chapter 9, "Configuring mod\\_oradav"](#)





---

---

## Getting Started with Oracle HTTP Server

This chapter provides information on getting started with Oracle HTTP Server. It discusses the procedures needed to configure and use Oracle HTTP Server in your environment.

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier in Stand-Alone Mode" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

This chapter includes the following sections:

- [Section 4.1, "Starting, Stopping, and Restarting Oracle HTTP Server"](#)
- [Section 4.2, "Creating an Oracle HTTP Server Component"](#)
- [Section 4.3, "Specifying Server Properties"](#)
- [Section 4.4, "Configuring Oracle HTTP Server"](#)
- [Section 4.5, "Deleting an Oracle HTTP Server Component"](#)

### 4.1 Starting, Stopping, and Restarting Oracle HTTP Server

You can use Fusion Middleware Control or the `opmnctl` command to start, stop, and restart Oracle HTTP Server.

---

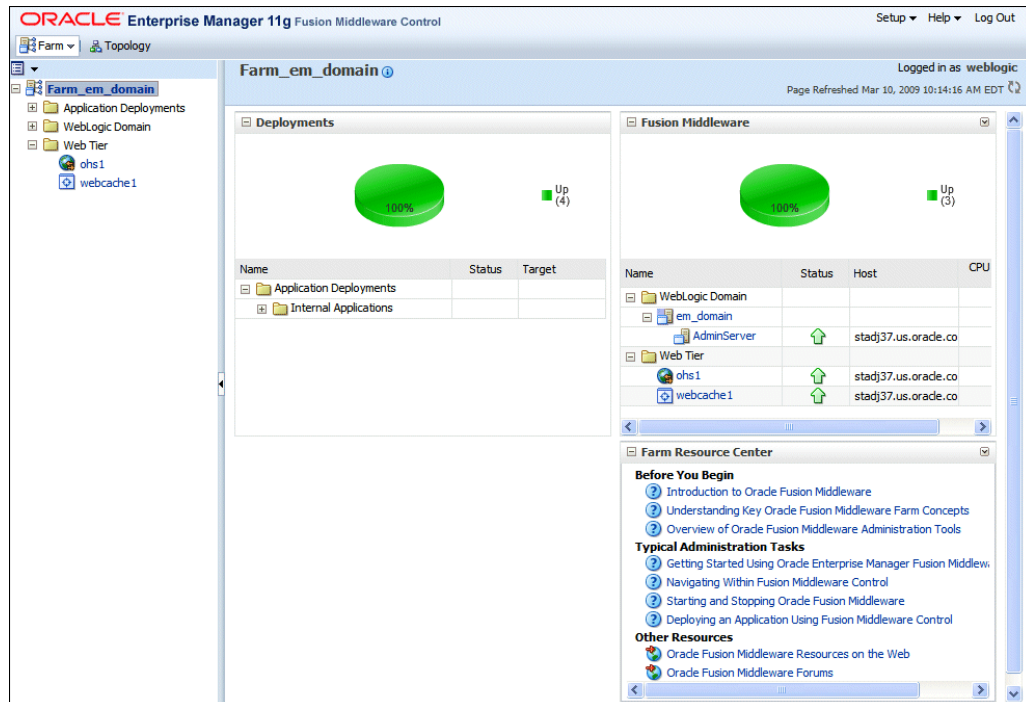
---

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

---

---

The Fusion Middleware Control home page shows the status of all installed components, including Oracle HTTP Server, as illustrated in the following figure:



You can determine the status of Oracle HTTP Server components using `opmnctl`:

```
> $ORACLE_INSTANCE/bin/opmnctl status
```

```
Processes in Instance: instance1
```

ias-component	process-type	pid	status
webcache1	WebCache-admin	19556	Alive
webcache1	WebCache	19555	Alive
ohs1	OHS	7249	Alive

### 4.1.1 Understanding the PID File

When Oracle HTTP Server starts up, it writes the process ID (PID) of the parent `httpd` process to the `httpd.pid` file located, by default, in the following directory:

```
ORACLE_INSTANCE/diagnostics/logs/OHS/component_name
```

The process ID can be used by the administrator when restarting and terminating the daemon. If a process stops abnormally, it is necessary to stop the `httpd` child processes using the `kill` command.

The `PidFile` directive in `httpd.conf` specifies the location of the PID file.

---

**Note:** On UNIX/Linux platforms, if you edit the `PidFile` directive, you also have to edit the `ORACLE_HOME/ohs/bin/apachectl` file to specify the new location of the PID file.

---

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

## 4.1.2 Starting Oracle HTTP Server

This section describes how to start Oracle HTTP Server using Fusion Middleware Control and `opmnctl`.

### 4.1.2.1 Using Fusion Middleware Control to Start Oracle HTTP Server

To start Oracle HTTP Server using Fusion Middleware Control:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Control** from the Oracle HTTP Server menu.
3. Select **Start Up** from the Control menu.

### 4.1.2.2 Using `opmnctl` to Start Oracle HTTP Server

To start all Oracle HTTP Server components in an Oracle instance using `opmnctl`:

```
> $ORACLE_INSTANCE/bin/opmnctl startproc process-type=OHS
```

To start a specific Oracle HTTP Server component using `opmnctl`:

```
> $ORACLE_INSTANCE/bin/opmnctl startproc ias-component=component_name
```

To get detailed information about the start process, include the verbose option with the command, as follows:

```
> $ORACLE_INSTANCE/bin/opmnctl verbose startproc ias-component=component_name
```

The following is an example of the information provided using the verbose option:

```
HTTP/1.1 200 OK
Content-Length: 656
Content-Type: text/html
Response: 1 of 1 processes started.

<?xml version='1.0' encoding='WINDOWS-1252'?>
<response>
<opmn id="stadk58:6701" http-status="200" http-response="1 of 1 processes
started.">
  <ias-instance id="inst1">
    <ias-component id="ohs1">
      <process-type id="OHS">
        <process-set id="OHS">
          <process id="699033550" pid="11366" status="Alive" index="1"
log="/scratch/oracle/product/11110/test090306/instances/inst1/diagnostics/logs/OHS
/ohs1/console-OHS~1.log" operation="request" result="success">
            <msg code="0" text="">
              </msg>
            </process>
          </process-set>
        </process-type>
      </ias-component>
    </ias-instance>
  </opmn>
</response>
```

### 4.1.2.3 Starting Oracle HTTP Server on a Privileged Port

On a UNIX system the TCP/IP port numbers below 1024 are special in that only processes with root privileges are allowed to listen on those ports.

By default, Oracle HTTP Server runs as a non-root user (the user that installed Oracle Fusion Middleware). Therefore, on UNIX systems, if you plan on running Oracle HTTP Server on a privileged port (for example, port 80), you must enable Oracle HTTP Server to run as root, as follows:

1. Stop Oracle HTTP Server using Fusion Middleware Control, or with the following `opmnctl` command:

```
> $ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=component_name
```

2. Change to the root user.

3. Navigate to `ORACLE_HOME/ohs/bin` and run the following commands:

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

4. Exit as the root user.

5. Add or uncomment the User and Group directives in the `httpd.conf` file and set them to the user and group that were used to install and configure Oracle Fusion Middleware. On the Unix operating system, ensure the user and group in the `httpd.conf` file is set to the user and group who has performed the installation.

6. Start Oracle HTTP Server using Fusion Middleware Control, or with the following command:

```
> $ORACLE_INSTANCE/bin/opmnctl startproc ias-component=component_name
```

### 4.1.3 Stopping Oracle HTTP Server

This section describes how to stop Oracle HTTP Server using Fusion Middleware Control and `opmnctl`. Other services may be impacted when Oracle HTTP Server is stopped.

#### 4.1.3.1 Using Fusion Middleware Control to Stop Oracle HTTP Server

To stop Oracle HTTP Server using Fusion Middleware Control:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Control** from the Oracle HTTP Server menu.
3. Select **Shut Down** from the Control menu.

#### 4.1.3.2 Using `opmnctl` to Stop Oracle HTTP Server

To stop all Oracle HTTP Server components in an Oracle instance using `opmnctl`:

```
> $ORACLE_INSTANCE/bin/opmnctl stopproc process-type=OHS
```

To stop a specific Oracle HTTP Server component using `opmnctl`:

```
> $ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=component_name
```

To get detailed information about the stop process, include the verbose option with the command, as follows:

```
> $ORACLE_INSTANCE/bin/opmnctl verbose stopproc ias-component=component_name
```

### 4.1.4 Restarting Oracle HTTP Server

Restarting Oracle HTTP Server causes the Apache parent process to advise its child processes to exit after their current request (or to exit immediately if they are not

serving any requests). Upon restarting, the parent process re-reads its configuration files and reopens its log files. As each child process exits, the parent replaces it with a child process from the new generation of the configuration file, which begins serving new requests immediately.

The following sections describe how to restart Oracle HTTP Server using Fusion Middleware Control and `opmnctl`.

#### 4.1.4.1 Using Fusion Middleware Control to Restart Oracle HTTP Server

To restart Oracle HTTP Server using Fusion Middleware Control:

1. Navigate to the Oracle HTTP Server home page.
2. Select the **Control** from the Oracle HTTP Server menu.
3. Select **Restart** from the Control menu.

#### 4.1.4.2 Using `opmnctl` to Restart Oracle HTTP Server

To restart all Oracle HTTP Server components in an Oracle instance using `opmnctl`:

```
> $ORACLE_INSTANCE/bin/opmnctl restartproc process-type=OHS
```

To restart a specific Oracle HTTP Server component using `opmnctl`:

```
> $ORACLE_INSTANCE/bin/opmnctl restartproc ias-component=component_name
```

To get detailed information about the start process, include the verbose option with the command, as follows:

```
> $ORACLE_INSTANCE/bin/opmnctl verbose restartproc ias-component=component_name
```

## 4.2 Creating an Oracle HTTP Server Component

When you install Oracle Web Tier, you can choose one of the following approaches:

- Install the software and configure an Oracle instance with an Oracle HTTP Server component.
- Install only the software.

If you choose to install only the software, you should subsequently configure an Oracle instance by running the configuration tool (`config.sh` on UNIX and `config.bat` on Windows), which is located in the `ORACLE_HOME/bin` directory. For more information, see the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

In either approach, you can create additional Oracle HTTP Server components in an Oracle instance by using the `opmnctl` command available in the `ORACLE_INSTANCE/bin` directory, as described in this section. Note that you cannot create Oracle HTTP Server components by using Fusion Middleware Control.

To create an Oracle HTTP Server component by using `opmnctl`, run the following command:

```
> $ORACLE_INSTANCE/bin/opmnctl createcomponent -componentType OHS -componentName component_name
```

For example, to create an Oracle HTTP Server component named `ohs2`, use the following command:

```
> $ORACLE_INSTANCE/bin/opmnctl createcomponent -componentType OHS -componentName ohs2
```

When you create the Oracle HTTP Server component, ports are automatically assigned. However, you can use the following parameters to specify the ports of your choice:

- `-listenPort`: HTTP listening port
- `-sslPort`: HTTPS (SSL) listening port
- `-proxyPort`: Proxy MBean port internally used by Oracle HTTP Server to communicate with Fusion Middleware Control

## 4.3 Specifying Server Properties

Server properties for Oracle HTTP Server can be set using Fusion Middleware Control or direct editing of the Oracle HTTP Server configuration files. You cannot specify the server properties using `opmnctl` commands.

- [Using Fusion Middleware Control to Specify Server Properties](#)
- [Editing the `httpd.conf` File to Specify Server Properties](#)

### 4.3.1 Using Fusion Middleware Control to Specify Server Properties

To specify the server properties using the Fusion Middleware Control:

1. Select **Administration** from the Oracle HTTP Server menu.
2. Select **Server Configuration** from the Administration menu. The Server Configuration page appears.

**Information**  
All fields on this page will require a restart to take effect.

**Server Configuration** Apply Revert

Configure basic OHS settings, such as server root directory, document root directory, the user and group under which the server runs, installed modules, and aliases.

Server Root Directory `"${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_NAME}"`

Document Root `"${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_NAME}"`

Administrator's E-mail

Directory Index `index.html index.html.var`

Operating System User

Operating System Group

**Modules**

The following are the installed OHS modules that can be enabled or disabled.

mod\_perl  mod\_fcgi  mod\_osso

**Aliases**

Alias is used to map URLs to filesystem locations. This allows for documents to be stored in the local filesystem other than under the Document Root.

+ Add Row ✕ Remove

URL Path	File Path or Directory Path
<code>/icons/</code>	<code>"\${ORACLE_INSTANCE}/config/\${COMPONENT_TYPE}/\${COMPONENT_NAME}/icons/"</code>

**TIP** For example, the columns can have the following values: URL Path: /image, File Path or Directory Path: ftp/pub/image.

3. Enter the documentation root directory in the Document Root field that forms the main document tree visible from the Web site.
4. Enter the e-mail address in the Administrator's E-mail Address field that the server will include in error messages sent to the client.
5. Enter the directory index in the Directory Index field. This is the main (index) page that will be displayed when a client first accesses the Web site.

6. Enter the user name in the Operating System User field.

This is the user name for the server, when sending and responding to requests. The user should not have privileges that allow it to access files or run programs that are for internal-use only. For example, when a request comes from Oracle Portal, Oracle HTTP Server will respond as the user defined in this field, and should have privileges to access the content in Oracle Portal. However, the user should not have privileges to access company-confidential content.

Oracle recommends that you set up a user specifically for running the server. Oracle also recommends that you do **not** set the user to root.

7. Enter the group name in the Operating System Group field. This is the group for the server, when sending and responding to requests. The user defined for Oracle HTTP Server must be a member of this group.

Oracle recommends that you set up a group specifically for running the server. Oracle also recommends that you do **not** set the group as root.

8. The Modules region is used to enable or disable modules. There are three modules that you can enable or disable: `mod_perl`, `mod_fcgi`, and `mod_osso`.

For instructions on configuring the `mod_perl` module, see "[Configuring the mod\\_perl Module](#)" on page 4-11.

9. Create an alias, if necessary in the Aliases table. An alias maps to a specified directory. For example, to use a specific set of content pages for a group you can create an alias to the directory that has the content pages.

10. Review the settings. If the settings are correct, then click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, then click **Revert** to return to the original settings.

11. Restart Oracle HTTP Server as described in [Section 4.1.4](#).

The server properties are saved, and shown on the Server Configuration page.

### 4.3.2 Editing the `httpd.conf` File to Specify Server Properties

To specify the server properties using the `httpd.conf` file:

1. Open the `httpd.conf` file using either a text editor or the Advanced Server Configuration page in Fusion Middleware Control. (See [Section 4.4.5, "Modifying an Oracle HTTP Server Configuration File."](#))

2. In the `DocumentRoot` section of the file, enter the directory that stores the main content for the Web site. The following is an example of the syntax:

```
DocumentRoot "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_NAME}/htdocs"
```

3. In the `ServerAdmin` section of the file, enter the administrator's e-mail address. This is the e-mail address that will appear on client pages. The following is an example of the syntax:

```
ServerAdmin WebMaster@example.com
```

4. In the `DirectoryIndex` section of the file, enter the directory index. This is the main (index) page that will be displayed when a client first accesses the Web site. The following is an example of the syntax:

```
DirectoryIndex index.html index.html.var
```

5. In the `User` and `Group` section of the file, enter the user name and group. The following is an example of the syntax:

```
User nobody
Group nobody
```

The user name is for the server, when sending and responding to requests. The user should not have privileges that allow it to access files or run programs that are for internal-use only. For example, when a request comes from Oracle Portal, Oracle HTTP Server will respond as the user defined in this field, and should have privileges to access the content in Oracle Portal. However, the user should not have privileges to access company-confidential content.

Oracle recommends that you set up a group specifically for running the server. The user defined for Oracle HTTP Server must be a member of this group. Oracle also recommends that you do **not** set the group as root.

---

---

**Note:** `User` and `Group` are relevant only when running Oracle HTTP Server as root on UNIX.

---

---

6. Create aliases, if needed. An alias maps to a specified directory. For example, to use a specific set of icons, you can create an alias to the directory that has the icons for the Web pages. The following is an example of the syntax:

```
Alias /icons/ "${ORACLE_HOME}/config/${COMPONENT_TYPE}/${COMPONENT_
NAME}/icons/"

<Directory "${ORACLE_HOME}/content/${COMPONENT_TYPE}/${COMPONENT_NAME}/icons/">
    Options MultiViews
    Order allow, deny
    Allow from all
</Directory>
```

7. Save the file.
8. Restart Oracle HTTP Server as described in [Section 4.1.4](#).

## 4.4 Configuring Oracle HTTP Server

This section includes the following sections:

- [Section 4.4.1, "Configuring Secure Sockets Layer"](#)
- [Section 4.4.2, "Configuring MIME Settings"](#)
- [Section 4.4.3, "Configuring the mod\\_perl Module"](#)
- [Section 4.4.4, "Configuring mod\\_wl\\_ohs"](#)
- [Section 4.4.5, "Modifying an Oracle HTTP Server Configuration File"](#)
- [Section 4.4.6, "Configuring the Oracle HTTP Server Environment to Use the apxs Script"](#)
- [Section 4.4.7, "Disabling the Options Method"](#)
- [Section 4.4.8, "Updating the Configuration for OHS Instances on a Shared Filesystem"](#)



## 4.4.1 Configuring Secure Sockets Layer

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 (HTTPS). The differences are that HTTPS uses the URL scheme `https://` rather than `http://`.

By default, an SSL listen port is configured and enabled using a default **wallet** during installation. 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 must be created for your production server. The default wallet is located in the `ORACLE_INSTANCE/config/OHS/component_name/keystores/default` directory. You can either place the new wallet in this location, or change the `SSLWallet` directive in `ORACLE_INSTANCE/config/OHS/component_name/ssl.conf` to point to the location of your real wallet.

For the changes to take effect, you should restart the Oracle HTTP Server components as described in [Section 4.1.4](#).

For information about configuring wallets and SSL using Fusion Middleware Control, see "Enabling SSL for Oracle HTTP Server Virtual Hosts" in the *Oracle Fusion Middleware Administrator's Guide*.

## 4.4.2 Configuring MIME Settings

Multipurpose Internet Mail Extension (MIME) settings are used by Oracle HTTP Server to interpret file types, encodings, and languages. MIME settings for Oracle HTTP Server can only be set using Fusion Middleware Control. You cannot specify the MIME settings using `opmnctl` commands.

The following tasks can be completed on the MIME Configuration page:

- [Configuring MIME Types](#)
- [Configuring MIME Encoding](#)
- [Configuring MIME Languages](#)

### 4.4.2.1 Configuring MIME Types

MIME type maps a given file extension to a specified content type. The MIME type is used for filenames containing an extension.

**4.4.2.1.1 Using Fusion Middleware Control to Configure MIME Types** To configure a MIME type using Fusion Middleware Control, do the following:

1. Select **Administration** from the Oracle HTTP Server menu.
2. Select **MIME Configuration** from the Administration menu. The MIME configuration page appears.
3. Click **Add Row** in MIME Configuration region. A new, blank row is added to the list.
4. Enter the MIME type.

5. Enter the file extension.
6. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
7. Restart Oracle HTTP Server, as described in [Section 4.1.4](#).

The MIME configuration is saved, and shown on the MIME Configuration page.

#### 4.4.2.2 Configuring MIME Encoding

MIME encoding enables Oracle HTTP Server to determine the file type based on the file extension. You can add and remove MIME encodings. The encoding directive maps the file extension to a specified encoding type.

1. Select **Administration** from the Oracle HTTP Server menu.
2. Select **MIME Configuration** from the Administration menu. The MIME configuration page appears.
3. Expand the MIME Encoding region by clicking the plus sign (+) next to MIME Encoding.
4. Click **Add Row** in MIME Encoding region. A new, blank row is added to the list.
5. Enter the MIME encoding, such as x-gzip.
6. Enter the file extension, such as .gx.
7. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
8. Restart Oracle HTTP Server as described in [Section 4.1.4](#).

#### 4.4.2.3 Configuring MIME Languages

The MIME language setting maps file extensions to a particular language. This directive is commonly used for content negotiation, in which Oracle HTTP Server returns the document that most closely matched the preferences set by the client.

1. Select **Administration** from the Oracle HTTP Server menu.
2. Select **MIME Configuration** from the Administration menu. The MIME configuration page appears.
3. Expand the MIME Languages region by clicking the plus sign (+) next to MIME Languages.
4. Click **Add Row** in MIME Languages region. A new, blank row is added to the list.
5. Enter the MIME language, such as en-US.
6. Enter the file extension, such as en-us.
7. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
8. Restart Oracle HTTP Server as described in [Section 4.1.4](#).

### 4.4.3 Configuring the mod\_perl Module

The mod\_perl module embeds the Perl interpreter into Oracle HTTP Server. This eliminates start-up overhead and enables you to write modules in Perl. The module is disabled, by default.

To enable the mod\_perl module using Fusion Middleware Control, do the following:

1. Select **Administration** from the Oracle HTTP Server menu.
2. Select **mod\_perl Configuration** from the Administration menu. The mod\_perl configuration page appears.

---

**Note:** If mod\_perl has not been enabled, then you will be redirected to the Server Configuration page. Select mod\_perl and click **Apply** to enable mod\_perl. After the confirmation page has been displayed, restart Oracle HTTP Server, and then return to the mod\_perl Configuration page.

---

3. Enter the switch information in the Switches field.
4. Enter the environment variables to be passed to the scripts in the Environment field.
5. Enter the required script names in the Require field.
6. Click **Add Row** to create a new row.
7. Configure mod\_perl directives for a Location in the Perl Locations table. The Location assigns a number of rules that the server should follow when the request's URI matches the Location.
  - a. Enter the base URI for the Perl scripts in the Locations field. Just as it is the widely accepted convention to use /cgi-bin for your mod\_cgi scripts, it is also conventional to use /perl as the base URI of the Perl scripts that are running under mod\_perl.
  - b. Enter options in the Options field. The PerlOptions directive provides fine-grained configuration by providing control over which class of Perl interpreter pool to be used. Options are enabled by prepending them with a plus sign (+) and are disabled by prepending them with a minus sign (-).
  - c. If you want to send headers, then click the **Send Header** check box. The PerlSendHeader directive is for mod\_perl 1.0 backwards-compatibility. When enabled, the server sends an HTTP header to the browser on every script invocation. You should disable this option for NPH (non-parsed-headers) scripts.
  - d. Enter the environment in the Environment field. The PerlSetEnv directive allows you to specify system environment variables and pass them into your mod\_perl handlers.
  - e. Enter the response handler in the Response Handler field. The PerlResponseHandler directive tells mod\_perl which callback is going to do the job.
  - f. Enter the authentication handler in the Authentication Handler field. The PerlAuthenHandler directive is used to set the handler to verify a user's identification credentials.

8. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
9. Restart Oracle HTTP Server as described in [Section 4.1.4](#).

The mod\_perl module configuration is saved and shown on the mod\_perl Configuration page.

---

**Note:** If you are manually editing the mod\_perl configuration instead of using Fusion Middleware Control, then all directives must be defined within the `<IfModule mod_perl.c>` block of the `mod_perl.conf` file. Any mod\_perl related directives defined outside of this block might be ignored.

---

#### 4.4.4 Configuring mod\_wl\_ohs

You can configure mod\_wl\_ohs either by using Fusion Middleware Control or by editing the `mod_wl_ohs.conf` configuration file manually.

For information about the prerequisites and procedure for configuring mod\_wl\_ohs to proxy requests from Oracle HTTP Server to Oracle WebLogic Server, see "Configuring the mod\_wl\_ohs Plug-In for Oracle HTTP Server" in *Using Web Server 1.1 Plug-Ins with Oracle WebLogic server*.

#### 4.4.5 Modifying an Oracle HTTP Server Configuration File

To modify an Oracle HTTP Server configuration file using Fusion Middleware Control, do the following:

1. Select **Administration** from the HTTP Server menu.
2. Select **Advanced Configuration** from the Administration menu item. The Advanced Server Configuration page appears.
3. Select the configuration file from the list, such as the `httpd.conf` file.
4. Edit the file, as needed.
5. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
6. Restart Oracle HTTP Server as described in [Section 4.1.4](#).

The file is saved and shown on the Advanced Server Configuration page.

#### 4.4.6 Configuring the Oracle HTTP Server Environment to Use the apxs Script

You can use the `apxs` command in the `$ORACLE_HOME/ohs/bin` directory to build and install Apache extension modules for Oracle HTTP Server. To be able to use the `apxs` command, you should configure the Oracle HTTP Server environment as follows:

Set the following environment variables from command line:

- `ORACLE_HOME=~/oraHome1`
- `ORACLE_INSTANCE=~/oraInstance1`
- `CONFIG_FILE_PATH=$ORACLE_INSTANCE/config/OHS/oraInstance1`

- `LD_LIBRARY_PATH=$ORACLE_HOME/lib:$ORACLE_HOME/ohs/lib:$LD_LIBRARY_PATH`

Where `ORACLE_HOME` and `ORACLE_INSTANCE` are set to the appropriate values for the customer; for example:

```
export ORACLE_HOME=/scratch/jjones/PS6/Middleware/Oracle_WT1
export ORACLE_INSTANCE=$ORACLE_HOME/instances/instance1
export CONFIG_FILE_PATH=$ORACLE_INSTANCE/config/OHS/ohs1
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:$ORACLE_HOME/ohs/lib:$LD_LIBRARY_PATH
```

You can now use the `apxs` from `ORACLE_HOME` to build, install and configure a new module; for example, to build, install and configure a new extension module called `mod_experimental`, from the module's source directory, you would enter the following:

```
$ORACLE_HOME/ohs/bin/apxs -c mod_experimental.c
$ORACLE_HOME/ohs/bin/apxs -i -a mod_experimental.c
```

For more information about the `apxs` command, see the Apache HTTP Server documentation at

<http://httpd.apache.org/docs/2.0/programs/apxs.html>.

---



---

**Note:** `apxs` is available only to Unix/Linux versions of OHS.

---



---

#### 4.4.7 Disabling the Options Method

The Options method enables clients to determine which methods are supported by a web server. If enabled, it appears in the `Allow` line of HTTP response headers.

For example, if you send a request such as:

```
---- Request -----
OPTIONS / HTTP/1.0
Content-Length: 0
Accept: */*
Accept-Language: en-US
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Win32)
Host: host123:80
```

you might get the following response from the web server:

```
---- Response -----
HTTP/1.1 200 OK
Date: Wed, 23 Apr 2008 20:20:49 GMT
Server: Oracle-Application-Server-11g/11.1.1.0.0 Oracle-HTTP-Server
Allow: GET, HEAD, POST, OPTIONS
Content-Length: 0
Connection: close
Content-Type: text/html
```

Some sources consider exposing the Options method a low security risk because malicious clients could use it to determine the methods supported by a web server. However, because web servers support only a limited number of methods, disabling this method will just slow down malicious clients, not stop them. In addition, the Options method may be used by legitimate clients.

If your Oracle Fusion Middleware environment does not have clients that require the Options method, you can disable it by including the following lines in the `httpd.conf` file:

```
<IfModule mod_rewrite.c>
RewriteEngine on
RewriteCond %{REQUEST_METHOD} ^OPTIONS
RewriteRule .* - [F]
</IfModule>
```

#### 4.4.8 Updating the Configuration for OHS Instances on a Shared Filesystem

Functional or performance issues may be encountered when an OHS instance is created on a shared filesystem, including NFS (Network File System). In particular, lock files or Unix sockets used by OHS may not work or may have severe performance degradation; WLS requests routed by `mod_wl_ohs` may have severe performance degradation due to filesystem accesses in the default configuration.

Table 4-1 provides information about the Lock file issues and the suggested changes in the `httpd.conf` file specific to the operating systems.

**Table 4-1 Lock File issues**

Operating System	Description	httpd.conf changes
Linux	Lock files are not required. The Sys V semaphore is the preferred cross-process mutex implementation.	Change <code>AcceptMutex fcntl</code> to <code>AcceptMutex sysvsem</code> (two places). Comment out the <code>LockFile</code> directive (three places).
Solaris	Lock files are not required. The cross-process <code>pthread</code> mutex is the preferred cross-process mutex implementation.	Change <code>AcceptMutex fcntl</code> to <code>AcceptMutex pthread</code> (two places). Comment out the <code>LockFile</code> directive (three places).
Other Unix platforms		Change the <code>LockFile</code> directive to point to a local filesystem (three places).
Unix socket issues	<code>mod_cgid</code> - is not enabled by default. If enabled, use the <code>ScriptSock</code> directive to place <code>mod_cgid</code> 's Unix socket on a local filesystem.  <code>mod_fastcgi</code> - is not enabled by default. If enabled, use the <code>FastCgiIpcDir</code> directive to place <code>mod_fastcgi</code> 's Unix sockets on a local file system.	

#### 4.5 Deleting an Oracle HTTP Server Component

This section describes how to delete an Oracle HTTP Server component using `opmnctl`. You cannot delete an Oracle HTTP Server component using Fusion Middleware Control.

To delete an Oracle HTTP Server component using `opmnctl`:

```
> $ORACLE_INSTANCE/bin/opmnctl deletecomponent -componentName component_name
```

For example, to delete an Oracle HTTP Server component named `ohs2` use the following command:

```
> $ORACLE_INSTANCE/bin/opmnctl deletecomponent -componentName ohs2
```

---

---

# Managing and Monitoring Server Processes

This chapter describes how to manage and monitor Oracle HTTP Server. It discusses the procedures and tools to manage Oracle HTTP Server in your environment.

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier Without Oracle WebLogic Server" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

This chapter includes the following sections:

- [Section 5.1, "Oracle HTTP Server Processing Model"](#)
- [Section 5.2, "Monitoring Oracle HTTP Server Performance"](#)
- [Section 5.3, "Configuring Oracle HTTP Server Performance Directives"](#)
- [Section 5.4, "Understanding Process Security"](#)

## 5.1 Oracle HTTP Server Processing Model

The following sections explain the processing model for Oracle HTTP Server.

### 5.1.1 Request Process Model

After Oracle HTTP Server is started, it is ready to listen for and respond to HTTP(S) requests. The request processing model on Microsoft Windows systems differs from that on UNIX systems.

- On Microsoft 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 created threads is static and can be configured for performance.
- 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 child processes initially so that requests can be handled without having to spawn more child processes.

### 5.1.2 Single Unit Process Model

Oracle HTTP Server provides functionality that allows it to terminate as a single unit if the parent process fails. The parent process is responsible for starting and stopping all 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 shutting down all the orphaned child processes. Until all the child processes are closed, a new Oracle HTTP Server instance cannot be started because the orphaned child processes still occupy the ports the new Oracle HTTP Server instance needs to access.

To prevent this from occurring, the DMS instrumentation layer in child processes on UNIX and monitor functionality within WinNT MPM on Windows monitor the parent process. If they detect that the parent process has failed, then all of the remaining child processes are shut down.

When this functionality is combined with OPMN, it means that Oracle HTTP Server is easily restarted in case of a parent process failure. The DMS instrumentation layer on UNIX and a monitor within WinNT MPM on Windows ensures that all of the Oracle HTTP Server child processes are shut down, 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.

## 5.2 Monitoring Oracle HTTP Server Performance

Oracle Fusion Middleware automatically and continuously measures run-time performance for Oracle HTTP Server. The performance metrics are automatically enabled; you do not need to set options or perform any extra configuration to collect them. If you encounter a problem, such as an application that is running slowly or is hanging, you can view particular metrics to find out more information about the problem.

Note that Fusion Middleware Control provides real-time data. If you are interested in viewing historical data, consider using Grid Control.

### 5.2.1 Viewing Oracle HTTP Server Performance Metrics

You can view metrics from the Oracle HTTP Server home menu of Fusion Middleware Control:

1. Select the Oracle HTTP Server that you want to monitor.

The Oracle HTTP Server home page is displayed.

2. From the Oracle HTTP Server menu, choose **Monitoring**, and then select **Performance Summary**.

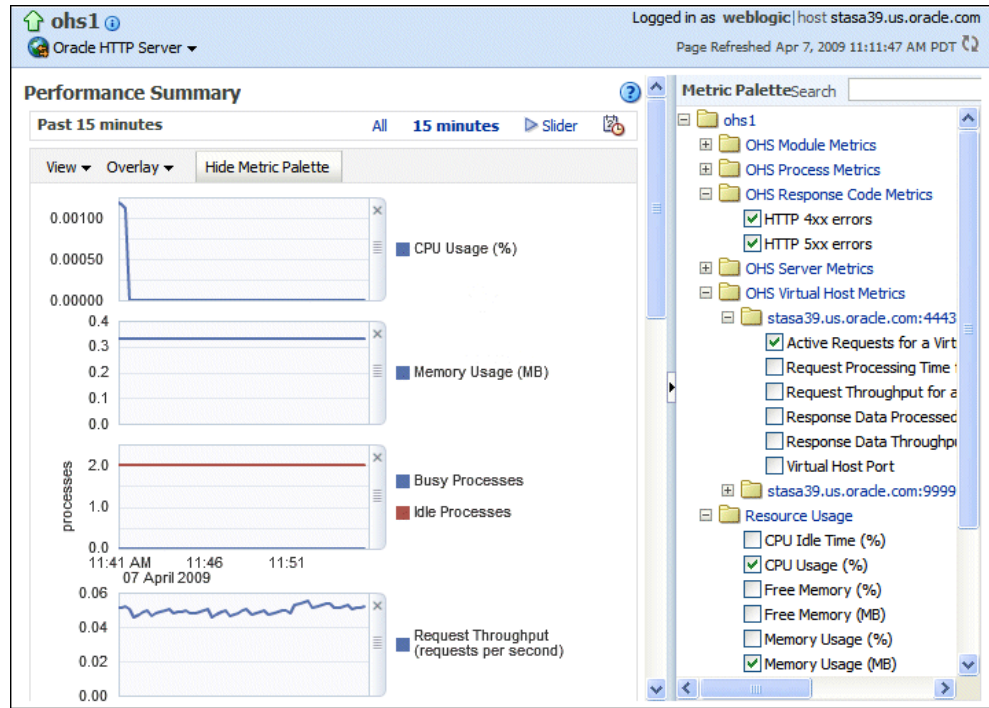
The Performance Summary page is displayed. It shows performance metrics, as well as information about response time and request processing time for the Oracle HTTP Server instance.

3. To see additional metrics, click **Show Metric Palette** and expand the metric categories.

**Tip:** Oracle HTTP Server port usage information is also available from the Oracle HTTP Server home menu.



The following figure shows the Oracle HTTP Server Performance Summary page with the Metric Palette displayed:



4. Select additional metrics to add them to the Performance Summary.

## 5.2.2 Understanding Oracle HTTP Server Performance Metrics

This section lists some of the most commonly-used metrics that can help you analyze Oracle HTTP Server performance.

### OHS Server Metrics

The OHS Server Metrics folder contains performance metric options for Oracle HTTP Server. The following table describes the metrics in the OHS Server Metrics folder:

Element	Description
CPU Usage	CPU usage and idle times
Memory Usage	Memory usage and free memory, in MB
Processes	Busy and idle process metrics
Request Throughput	Request throughput, as measured by requests per second
Request Processing Time	Request processing time, in seconds
Response Data Throughput	Response data throughput, in KB per second
Response Data Processed	Response data processed, in KB per second
Active HTTP Connections	Number of active HTTP connections
Connection Duration	Length of time for connections
HTTP Errors	Number of HTTP 4xx and 5xx errors

**OHS Virtual Host Metrics**

The OHS Virtual Host Metrics folder contains performance metric options for virtual hosts, also known as access points. The following table describes the metrics in the OHS Virtual Host Metrics folder:

Element	Description
Request Throughput for a Virtual Host	Number of requests per second for each virtual host
Request Processing Time for a Virtual Host	Time to process each request for each virtual host
Response Data Throughput for a Virtual Host	Amount of data being sent for each virtual host
Response Data Processed for a Virtual Host	Amount of data being processed for each virtual host

**OHS Module Metrics**

The OHS Module Metrics folder contains performance metric option for modules. The following table describes the metrics in the OHS Module Metrics folder.

Element	Description
Request Handling Throughput	Request handling throughput for a module, in requests per second
Request Handling Time	Request handling time for a module, in seconds
Module Metrics	Modules including active requests, throughput, and time for each module

## 5.3 Configuring Oracle HTTP Server Performance Directives

Oracle HTTP Server uses directives in `httpd.conf`. This configuration file specifies the maximum number of HTTP requests that can be processed simultaneously, logging details, and certain limits and timeouts. Oracle HTTP Server supports and ships with the following three Multi-Processing Modules (MPMs) which are responsible for binding to network ports on the machine, accepting requests, and dispatching children to handle the requests:

- **Worker** - This is the default MPM for Oracle HTTP Server on UNIX/Linux platforms. This MPM implements a hybrid multi-process multi-threaded server. By using threads to serve requests, it is able to serve a large number of requests with fewer system resources than a process-based server. However, it retains much of the stability of a process-based server by keeping multiple processes available, each with many threads.
- **WinNT** - This is the default MPM for Oracle HTTP Server on Windows platforms. It uses a single control process which launches a single child process which in turn creates threads to handle requests.
- **Prefork** - This MPM implements a non-threaded, pre-forking server that handles requests in a manner similar to Apache 1.3. It is appropriate for sites that need to avoid threading for compatibility with non-thread-safe libraries. It is also the best MPM for isolating each request, so that a problem with a single request will not affect any other.

The discussion and recommendations in this section are based on the use of Worker or WinNT MPM, which uses threads. The thread-related directives listed below are not applicable if you are using the Prefork MPM.

The Performance Directives page allows you to tune performance-related directives for Oracle HTTP Server, as illustrated in the following figure:

**Information**  
All fields on this page will require a restart to take effect.

**Performance Directives** Reset to Default Apply Revert

Tune performance-related directives for Oracle HTTP Server.

MPM Name: worker

**Request Configuration**

Maximum Requests:

Maximum Requests per Child Server Process:  No Limit  Limit To

Request Timeout (seconds):

**Connection Configuration**

Maximum Connection Queue Length:

Multiple Requests per Connection:  Not Allowed  Allow with Connection Timeout (seconds)

**Process Configuration**

Initial Child Server Processes:

Maximum Idle Threads:

Minimum Idle Threads:

Threads per Child Server Process:

Performance directives management consists of three areas: request configuration, connection configuration, and process configuration. You can set these configurations using the Performance Directive page of Fusion Middleware Control and by following the instructions in the following sections:

- [Using Fusion Middleware Control to Set the Request Configuration](#)
- ["Using Fusion Middleware Control to Set the Connection Configuration"](#)
- [Using Fusion Middleware Control to Set the Process Configuration](#)

### 5.3.1 Using Fusion Middleware Control to Set the Request Configuration

To specify the Oracle HTTP Server request configuration using Fusion Middleware Control, do the following:

1. Select **Administration** from the Oracle HTTP Server menu.
2. Select **Performance Directives** from the Administration menu. The Performance Directives page appears.
3. Enter the maximum number of requests in the Maximum Requests field (`MaxClients` directive). This setting limits the number of requests that can be dealt with at one time. The default and recommended value is 150. This is applicable for all Linux/Unix platforms.
4. Set the maximum requests per child process in the Maximum Request per Child Process field (`MaxRequestPerChild` directive). You can choose to have no limit, or a maximum number. If you choose to have a limit, enter the maximum number in the field.
5. Enter the request timeout value in the Request Timeout (seconds) field (`Timeout` directive). This value sets the maximum time, in seconds, Oracle HTTP Server waits to receive a GET request, the amount of time between receipt of TCP packets

on a POST or PUT request, and the amount of time between ACKs on transmissions of TCP packets in responses.

6. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
7. Restart Oracle HTTP Server. See [Section 4.1.4](#).

The request configuration settings are saved, and shown on the Performance Directives page.

### 5.3.2 Using Fusion Middleware Control to Set the Connection Configuration

To specify the connection configuration using Fusion Middleware Control, do the following:

1. Select **Administration** from the Oracle HTTP Server menu.
2. Select **Performance Directives** from the Administration menu. The Performance Directives page appears.
3. Enter the maximum connection queue length in the Maximum Connection Queue Length field (`ListenBacklog` directive). This is the queue for pending connections. This is useful if the server is experiencing a TCP SYN overload, which causes numerous new connections to open up, but without completing the pending task.
4. Set the Multiple Requests per Connection field (`KeepAlive` directive) to indicate whether or not to allow multiple connections. If you choose to allow multiple connections, enter the number of seconds for timeout in the Allow With Connection Timeout field.

The Allow With Connection Timeout value sets the number of seconds the server waits for a subsequent request before closing the connection. Once a request has been received, the specified value applies. The default is 15 seconds.

5. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
6. Restart Oracle HTTP Server. See [Section 4.1.4](#).

The connection configuration settings are saved, and shown on the Performance Directives page.

### 5.3.3 Using Fusion Middleware Control to Set the Process Configuration

The child process and configuration settings impact the ability of the server to process requests. You may need to modify the settings as the number of requests increase or decrease to maintain a well-performing server.

For UNIX, the default number of child server processes is 2. For Microsoft Windows, the default number of threads to handle requests is 150.

To specify the process configuration using Fusion Middleware Control, do the following:

1. Select **Administration** from the Oracle HTTP Server menu.
2. Select **Performance Directives** from the Administration menu. The Performance Directives page appears.

3. Enter the number for the initial child server processes in the Initial Child Server Processes field (`StartServers` directive). This is the number of child server processes created when Oracle HTTP Server is started. The default is 2. This is for UNIX only.
4. Enter the number for the maximum idle threads in the Maximum Idle Threads field (`MaxSpareThreads` directive). An idle thread is a process that is running, but not handling a request.
5. Enter the number for the minimum idle threads in the Minimum Idle Threads field (`MinSpareThreads` directive).
6. Enter the number for the threads per child server process in the Threads per Child Server Process field (`ThreadsPerChild` directive).
7. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
8. Restart Oracle HTTP Server. See [Section 4.1.4](#).

The process configuration settings are saved, and shown on the Performance Directives page.

## 5.4 Understanding Process Security

By default, Oracle HTTP Server runs as a non-root user (the user that installed Oracle Fusion Middleware). Therefore, on UNIX systems, if you plan on running Oracle HTTP Server on a privileged port (for example, port 80), you must enable Oracle HTTP Server to run as root. See "[Starting Oracle HTTP Server on a Privileged Port](#)" on page 4-3.

For additional security on UNIX, you can change the `User` directive in the `httpd.conf` configuration file to `nobody`. Be sure that the child processes can accomplish their tasks as the 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, specified through the parameter `PlsqlCacheDirectory` in `ORACLE_INSTANCE/config/OHS/component_name/mod_plsql/cache.conf`. By default, this parameter points to `ORACLE_INSTANCE/OHS/component_name`.

Finally, given that the cached content might contain sensitive data, the 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:** [Section 3.10, "mod\\_plsql"](#)



---

---

## Managing Connectivity

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier Without Oracle WebLogic Server" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

Oracle HTTP Server comes configured with two listen ports: a non-SSL port (http) and an SSL port (https). The default, non-SSL port is 7777. If port 7777 is occupied, the next available port number, within a range of 7777-7877, is assigned. The default SSL port is 4443. Similarly, if port 4443 is occupied, the next available port number, within a range of 4443-4543, is assigned.

You can set these ports during installation or when creating a new Oracle HTTP Server component. For specifying ports at installation time, see the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*. For information about specifying ports when creating a new Oracle HTTP Server component using `opmnctl`, see [Section 4.2, "Creating an Oracle HTTP Server Component"](#).

---

---

**Note:** An additional SSL port (9999) is configured to run out-of-the-box in the `admin.conf` file. It is called *Proxy MBean* or *Admin port* and is used internally by Oracle HTTP Server to communicate with Fusion Middleware Control.

---

---

This chapter includes the following sections:

- [Section 6.1, "Viewing Port Number Usage"](#)
- [Section 6.2, "Managing Ports"](#)
- [Section 6.3, "Configuring Virtual Hosts"](#)

### 6.1 Viewing Port Number Usage

This section describes how to view ports using Fusion Middleware Control.

### 6.1.1 Using the Fusion Middleware Control to View Port Number Usage

To view the port number usage using Fusion Middleware Control, do the following:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Port Usage** from the Oracle HTTP Server menu.

The Port Usage detail page shows the component, the ports that are in use, the IP address the ports are bound to, and the protocol being used, as illustrated in the following figure:

Port in Use	IP Address	Component	Protocol
8889	ALL	ohs1	https
4443	ALL	ohs1	https
8888	ALL	ohs1	http

## 6.2 Managing Ports

The ports used by Oracle HTTP Server can be set during and after installation. In addition, you can change the port numbers, as needed. This section describes how to create, edit, and delete ports using Fusion Middleware Control.

---

**Caution:** The Oracle HTTP Server administration (proxy MBean) virtual host and its configuration, defined in the `admin.conf` file, must not be edited with the WebLogic Scripting Tool (WLST).

---

**See Also:** "Changing the Oracle HTTP Server Listen Ports" in the *Oracle Fusion Middleware Administrator's Guide*.

- [Using Fusion Middleware Control to Create Ports](#)
- [Using Fusion Middleware Control to Edit Ports](#)
- [Updating OHS Registration with a WebLogic Domain](#)

**Ports Configuration**  
All ports configured for this system component are shown. Some ports configured of type Admin cannot be edited or deleted.

Port	Port Type	Host Name	IP Address	Protocol
7782	Listen	stadk58.us.oracle.com	ANY	HTTP
4443	Listen	stadk58.us.oracle.com	ANY	HTTPS
9999	Admin	stadk58.us.oracle.com	ANY	HTTPS
12345	Listen	stadk58.us.oracle.com	ANY	HTTP

---

**Note:** When deleting a port, if there is a virtual host configured to use the port you want to delete, you must first delete that virtual host before deleting the port.

---



## 6.2.1 Using Fusion Middleware Control to Create Ports

To create ports using Fusion Middleware Control, do the following:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Administration** from the Oracle HTTP Server menu.
3. Select **Ports Configuration** from the Administration menu.
4. Click **Create**.

5. Use the IP Address menu to select an IP address for the new port. Ports can listen on a local IP Address of an associated host or on any available network interfaces. SSL for a port can be configured on the Virtual Hosts page, as described in [Section 6.3.2, "Using Fusion Middleware Control to Configure Virtual Hosts"](#).
6. Use the **Port** field to enter the port number.
7. Click **OK**.
8. Restart Oracle HTTP Server. See [Section 4.1.4](#).

---

**Note:** If you change the port or make other changes that affect the URL, such as changing the host name, enabling or disabling SSL, you need to re-register partner applications with the SSO server using the new URL. For more information, see "Registering Oracle HTTP Server mod\_osso with OSSO Server 10.1.4" in the *Oracle Fusion Middleware Application Security Guide*.

---

## 6.2.2 Using Fusion Middleware Control to Edit Ports

To create the ports using Fusion Middleware Control, do the following:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Administration** from the Oracle HTTP Server menu.
3. Select **Ports Configuration** from the Administration menu.
4. Select the port for which you want to change the port number.

Note that the Admin port cannot be edited using Fusion Middleware Control. Although this is a port Oracle HTTP Server uses for its internal communication with Fusion Middleware Control, in most of the cases it does not need to be changed. If you really want to change it, manually edit the `ORACLE_INSTANCE/config/OHS/component_name/admin.conf` file. For information

about the additional steps to be performed after changing the Admin port, see [Section 6.2.3, "Updating OHS Registration with a WebLogic Domain"](#)

5. Click **Edit**.

Ports Configuration > Create Port

**Information**  
All fields on this page will require a restart to take effect.

**Edit Port : 8888** OK Cancel

Edit attributes of a port for this system component. Ports created can listen on local IP Address of associated host or any of available network interfaces. SSL for a port can be configured here :[Virtual Hosts](#)

**Endpoint Attributes**

Port Type Listen

Endpoint Name 8888

IP Address

\* Port 8888

6. Edit the IP Address and/or Port number for the port.

SSL for a port can be configured on the Virtual Hosts page, as described in [Section 6.3.2, "Using Fusion Middleware Control to Configure Virtual Hosts"](#).

7. Click **OK**.
8. Restart Oracle HTTP Server. See [Section 4.1.4](#).

---

**Note:** If you change the port or make other changes that affect the URL, such as changing the host name, enabling or disabling SSL, you need to re-register partner applications with the SSO server using the new URL. For more information, see "Registering Oracle HTTP Server mod\_osso with OSSO Server 10.1.4" in the *Oracle Fusion Middleware Application Security Guide*.

---

### 6.2.3 Updating OHS Registration with a WebLogic Domain

In an Oracle Instance that is registered with a WebLogic domain, if the Oracle HTTP Server administration port (proxy MBean port in the `admin.conf` file) is changed after creating the component, then you must update the component registration with the WebLogic domain using the `opmnctl updatecomponentregistration` command, as follows:

```
> $ORACLE_INSTANCE/bin/opmnctl updatecomponentregistration -componentType OHS
-componentName component_name -proxyPort port
```

For example, if the proxy port of an Oracle HTTP Server component named `ohs2` has been changed to 8787, then use the following command:

```
> $ORACLE_INSTANCE/bin/opmnctl updatecomponentregistration -componentType OHS
-componentName ohs2 -proxyPort 8787
```

## 6.3 Configuring Virtual Hosts

You can create virtual hosts to run more than one Web site (such as `www.company1.com` and `www.company2.com`) on a single machine. Virtual hosts can be *IP-based*, meaning that you have a different IP address for every Web site, or

*name-based*, meaning that you have multiple names running on each IP address. The fact that they are running on the same physical server is not apparent to the end user.

---

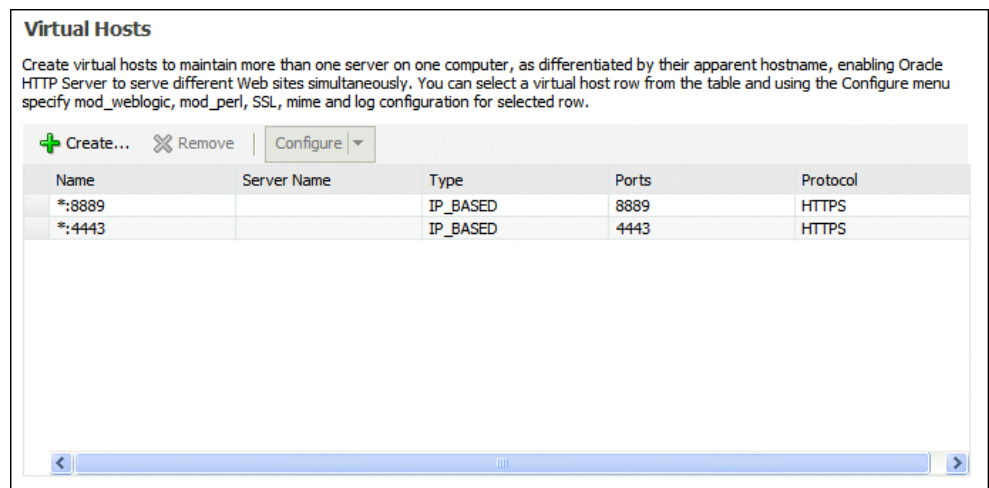
**Caution:** The Oracle HTTP Server administration (proxy MBean) virtual host and its configuration, defined in the `admin.conf` file, must not be edited with the WebLogic Scripting Tool (WLST).

---

This section describes how to create and edit virtual hosts using Fusion Middleware Control.

- [Using Fusion Middleware Control to Create Virtual Hosts](#)
- [Using Fusion Middleware Control to Configure Virtual Hosts](#)

**See Also:** For more information about virtual hosts, refer to the Apache HTTP Server documentation.



### 6.3.1 Using Fusion Middleware Control to Create Virtual Hosts

To create a virtual Host using Fusion Middleware Control, do the following:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Administration** from the Oracle HTTP Server menu.
3. Select **Virtual Hosts** from the Administration menu.
4. Click **Create**.

5. Enter a name for the virtual host field and then choose whether to enter a new listen address or to use an existing listen address.
  - New listen address - use this option when you want to create a virtual host that maps to a specific hostname or IP address, for example `mymachine.com:8080`. This will create following type `NameVirtualHost` and `VirtualHost` directives:
 

```
NameVirtualHost mymachine.com:8080
<VirtualHost mymachine.com:8080>
```
  - Use existing listen address - use this option when you want to create a virtual host using an existing listen port and the one that maps to all IP addresses. This will create following type `VirtualHost` directive:
 

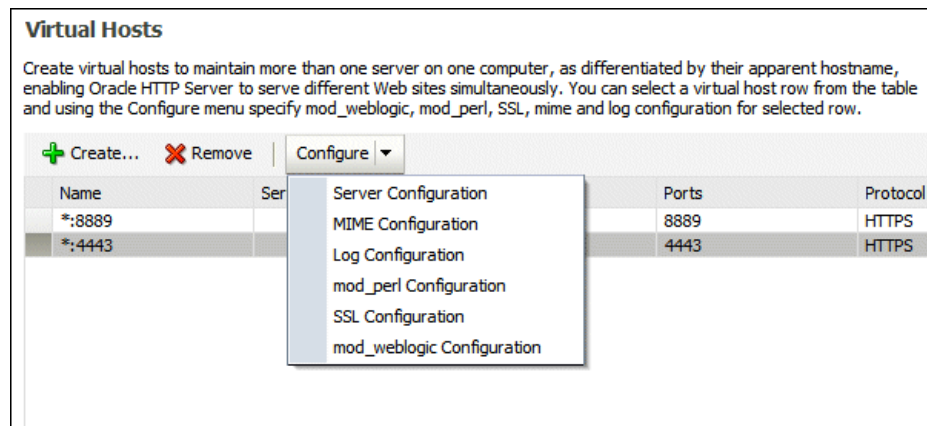
```
<VirtualHost *:8080>
```
6. Enter the remaining attributes for the new virtual host.
7. Use the Type field to select whether the virtual host will be IP-based or name-based.
8. Click **OK**.
9. Restart Oracle HTTP Server. See [Section 4.1.4](#).

### 6.3.2 Using Fusion Middleware Control to Configure Virtual Hosts

You can use the options on the Configure menu to specify Server, MIME, Log, `mod_perl`, `SSL`, and `mod_wl_ohs` configuration for a selected virtual host.

To configure a virtual host using Fusion Middleware Control, do the following:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Administration** from the Oracle HTTP Server menu.
3. Select **Virtual Hosts** from the Administration menu.
4. Highlight an existing virtual host in the table.
5. Click **Configure**.



6. Select one of the following options from Configure menu to open its corresponding configuration page. The values on these pages apply only to the virtual host. If the fields are blank, the virtual host uses the values configured at the server level.
  - Server Configuration – Configure basic virtual host properties, such as document root directory, installed modules, and aliases. See [Section 4.3.1, "Using Fusion Middleware Control to Specify Server Properties."](#)
  - MIME Configuration – Configure MIME settings, which are used by Oracle HTTP Server to interpret file types, encodings, and languages. [Section 4.4.2, "Configuring MIME Settings."](#)
  - Log Configuration – Configure access logs that will record all requests processed by the virtual host. The logs contain basic information about every HTTP transaction handled by the virtual host. See [Section 7.2, "Configuring Oracle HTTP Server Logs."](#)
  - mod\_perl Configuration – Configure the mod\_perl module to embed the Perl interpreter into the virtual host, thereby eliminating startup overhead and enabling you to write modules in Perl. This module is disabled, by default. See [Section 4.4.3, "Configuring the mod\\_perl Module."](#)
  - SSL Configuration – For instructions on configuring SSL using Fusion Middleware Control, see "Enabling SSL for Oracle HTTP Server Virtual Hosts" in the *Oracle Fusion Middleware Administrator's Guide*.
  - mod\_wl\_ohs Configuration – Configure the mod\_wl\_ohs module to allow requests to be proxied from an Oracle HTTP Server to Oracle WebLogic Server. See [Section 4.4.4, "Configuring mod\\_wl\\_ohs."](#)
7. Review the settings on each configuration page. If the settings are correct, click **OK** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Cancel** to return to the original settings.
8. Restart Oracle HTTP Server. See [Section 4.1.4](#).



---

---

# Managing Oracle HTTP Server Logs

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier Without Oracle WebLogic Server" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

Oracle HTTP Server generates log files containing messages that record all types of events, including startup and shutdown information, errors, warning messages, access information on HTTP requests, and additional information. This chapter describes how to find information about the cause of an error and its corrective action, to view and manage log files to assist in monitoring system activity and to diagnose problems.

This chapter includes the following sections:

- [Section 7.1, "Overview of Server Logs"](#)
- [Section 7.2, "Configuring Oracle HTTP Server Logs"](#)
- [Section 7.3, "Log Directives for Oracle HTTP Server"](#)
- [Section 7.4, "Viewing Oracle HTTP Server Logs"](#)

## 7.1 Overview of Server Logs

You can view Oracle Fusion Middleware log files using either Fusion Middleware Control or a text editor. The log files for Oracle HTTP Server are located in the following directory:

```
ORACLE_INSTANCE/diagnostics/logs/OHS/component_name
```

There are two types of logs for Oracle HTTP Server:

- Error logs, which record server problems.
- Access logs, which record which components and applications are being accessed and by whom.

This section contains the following topics:

- [Section 7.1.1, "About Error Logs"](#)
- [Section 7.1.2, "About Access Logs"](#)
- [Section 7.1.3, "Log Rotation"](#)

## 7.1.1 About Error Logs

Oracle HTTP Server enables you to choose the format in which you want to generate log messages. You can choose to generate log messages in the legacy Apache message format, or use Oracle Diagnostic Logging (ODL) to generate log messages in text or XML-formatted logs, which complies with Oracle standards for generating error log messages.

By default, Oracle HTTP Server error logs use ODL for generating diagnostic messages. It provides a common format for all diagnostic messages and log files, and a mechanism for correlating the diagnostic messages from various components across Oracle Fusion Middleware.

The default name of the error log file is `component_name.log`.

## 7.1.2 About Access Logs

Access logs record all requests processed by the server. The logs contain basic information about every HTTP transaction handled by the server. The access log contains the following information:

- Host name
- Remote log name
- Remote user and time
- Request
- Response code
- Number of transferred bytes

The default name of the access log file is `access_log`.

### Access Log Format

You can specify the information to include in the access log, and the manner in which it is written. The default format is the Common Log Format (CLF).

The CLF format contains the following fields:

```
host ident authuser date request status bytes
```

- `host`: This is the client domain name or its IP number. Use `%h` to specify the host field in the log.
- `ident`: If IdentityCheck is enabled and the client system runs `identd`, this is the client identity information. Use `%i` to specify the client identity field in the log.
- `authuser`: This is the user ID for the authorized user. Use `%a` to specify the authorized user field in the log.
- `date`: This is the date and time of the request in the `day/month/year:hour:minute:second` format. Use `%t` to specify date and time in the log.
- `request`: This is the request line, in double quotes, from the client. Use `%r` to specify request in the log.



- **status:** This is the three-digit status code returned to the client. Use `%s` to specify the status in the log. If the request will be forwarded from another server, use `%>s` to specify the last server in the log.
- **bytes:** This is the number of bytes, excluding headers, returned to the client. Use `%b` to specify number of bytes in the log. Use `%i` to include the header in the log.

**See Also:** Access Log in the Apache HTTP Server documentation.

### 7.1.3 Log Rotation

Oracle HTTP Server supports two types of log rotation policies: size-based and time-based. You can configure the Oracle HTTP Server logs to use either of the two rotation policies, by using the `odl_rotatelogs` command. By default, Oracle HTTP Server uses `odl_rotatelogs` for both error and access logs.

`odl_rotatelogs` supports all the features of Apache's `rotatelogs` command and the additional feature of log retention.

The following is the general syntax of the `odl_rotatelogs` command:

```
odl_rotatelogs [-u:offset] logfile {size-|time-based-rotation-options}
```

Table 7-1 describes the size- and time-based rotation options:

**Table 7-1 Options of the `odl_rotatelogs` command**

Option	Description
<code>-u</code>	The time (in seconds) to offset from UTC.
<code>logfile</code>	The path and name of the log file, followed by a hyphen (-) and then the timestamp format.  The following are the common timestamp format strings: <ul style="list-style-type: none"> <li>■ <code>%m</code>: Month as a two-digit decimal number (01-12)</li> <li>■ <code>%d</code>: Day of month as a two-digit decimal number (01-31)</li> <li>■ <code>%Y</code>: Year as a four-digit decimal number</li> <li>■ <code>%H</code>: Hour of the day as a two-digit decimal number (00-23)</li> <li>■ <code>%M</code>: Minute as a two-digit decimal number (00-59)</li> <li>■ <code>%S</code>: Second as a two-digit decimal number (00-59)</li> </ul> It should not include formats that expand to include slashes.
<code>frequency</code>	The time (in seconds) between log file rotations.
<code>retentionTime</code>	The maximum time for which the rotated log files are retained.
<code>startTime</code>	The time when time-based rotation should start.
<code>maxFileSize</code>	The maximum size (in MB) of log files.
<code>allFileSize</code>	The total size (in MB) of files retained.

#### Syntax and Examples for Time- and Size-Based Rotation

- Time-based rotation

##### Syntax:

```
$ odl_rotatelogs logfile frequency retentionTime startTime
```

##### Example:

```
$ odl_rotatelogs /varlog/error.log-%Y-%m-%d 21600 172800 2012-03-10T08:30:00
```

This command configures log rotation to be performed every 21600 seconds (6 hours) starting from 8:30 a.m. on March 10, 2012, and it specifies that the rotated log files should be retained for 172800 seconds (2 days).

- Size-based rotation

**Syntax:**

```
$ odl_rotatelogs logfile maxFileSize allFileSize
```

**Example:**

This command configures log rotation to be performed when the size of the log file reaches 10 MB, and it specifies the maximum size of all the rotated log files as 70 MB (up to 7 log files (=70/10) will be retained).

```
$ odl_rotatelogs /var/log/error.log-%Y-%m-%d 10M 70M
```

## 7.2 Configuring Oracle HTTP Server Logs

You can use Fusion Middleware Control to configure error and access logs. The following logging tasks can be set from the Log Configuration page:

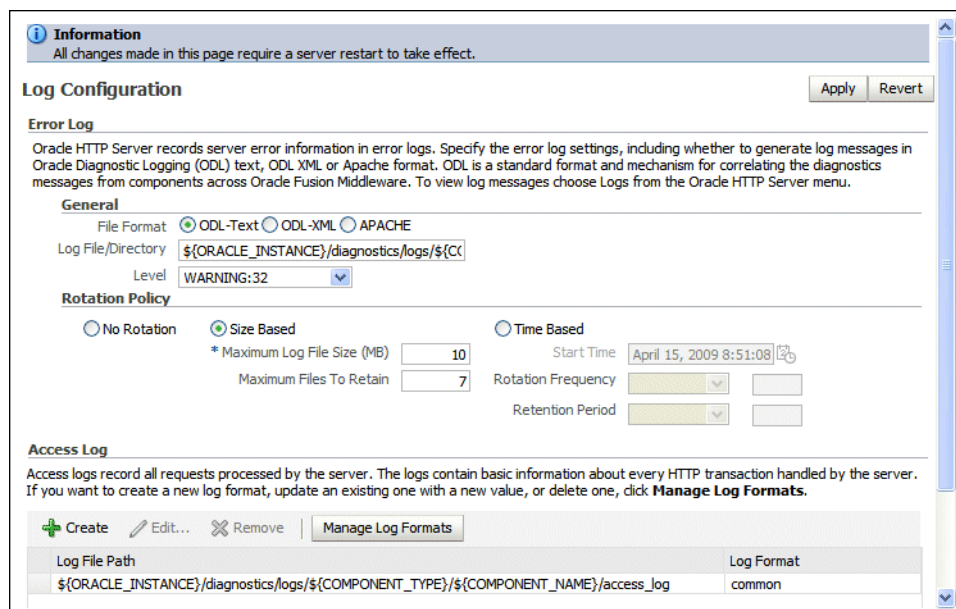
- [Using Fusion Middleware Control to Configure Error Logs](#)
- [Using Fusion Middleware Control to Configure Access Logs](#)

### 7.2.1 Using Fusion Middleware Control to Configure Error Logs

To configure an error log for Oracle HTTP Server using Fusion Middleware Control, do the following:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Log Configuration** from the Administration menu.

The Log Configuration page is displayed, as shown in the following figure.



3. The following error log configuration tasks can be set from this page:

- [Configuring the Error Log Format and Location](#)
- [Configuring the Error Log Level](#)
- [Configuring Error Log Rotation Policy](#)

### 7.2.1.1 Configuring the Error Log Format and Location

Oracle HTTP Server by default uses ODL-Text as the error log format and creates the log file with the name `component_name.log` under the `ORACLE_INSTANCE/diagnostics/logs/OHS/ component_name` directory. To use a different format or log location, do the following:

1. From the Log Configuration page, navigate to the General section under the Error Log section.
2. Select the desired file format. Although both ODL-Text and ODL-XML formats provide the same information, the ODL-XML file includes XML elements and wrappers, and so may be easier to read.
  - ODL-Text – the format of the diagnostic messages conform to an Oracle standard and are written in text format.
  - ODL-XML – the format of the diagnostic messages conform to an Oracle standard and are written in XML format.
  - Apache – the format of the diagnostic messages conform to the legacy Apache HTTP Server message format.
3. Enter a path for the error log in the Log File/Directory field. This directory must exist before you enter it here.

---

---

**Note:** You cannot configure `console~ohs~1.log` to be logged into another location.

---

---

4. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
5. Restart Oracle HTTP Server. See [Section 4.1.4](#).

### 7.2.1.2 Configuring the Error Log Level

You can configure the amount and type of information written to log files by specifying the message type and level. Error log level for Oracle HTTP Server by default is configured to `WARNING:32`. To use a different error log level do the following:

1. From the Log Configuration page, navigate to the General section under the Error Log section.
2. Select a level for the logging from the Level menu. The higher the log level, the more information that is included in the log.
3. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
4. Restart Oracle HTTP Server. See [Section 4.1.4](#).

---

---

**Note:** The log levels are different for the Apache log format from ODL-Text and the ODL-XML log format.

- For details on ODL log levels, refer to "Setting the Level of Information Written to Log Files" in the *Oracle Fusion Middleware Administrator's Guide*.
  - For details on Apache log levels, refer to the LogLevel Directive in the Apache HTTP Server documentation.
- 
- 

### 7.2.1.3 Configuring Error Log Rotation Policy

Log rotation policy for error logs can either be time-based, such as once a week, or sized-based, such as 120MB. By default, the error log file is rotated when it reaches 10 MB in size and a maximum of 7 error log files will be retained. To use a different rotation policy, do the following:

1. From the Log Configuration page, navigate to the General section under the Error Log section.
2. Select a rotation policy.
  - No Rotation – if you do not want to have the log file rotated ever.
  - Size Based – rotate the log file whenever it reaches a configured size. Set the maximum size for the log file in Maximum Log File Size (MB) field and the maximum number of error log files to retain in Maximum Files to Retain field.
  - Time Based – rotate the log file whenever configured time is reached. Set the start time, rotation frequency, and retention period.
3. Review the settings. If the settings are correct, click **Apply** to apply the changes. If the settings are incorrect, or you decide to not apply the changes, click **Revert** to return to the original settings.
4. Restart Oracle HTTP Server. See [Section 4.1.4](#).

## 7.2.2 Using Fusion Middleware Control to Configure Access Logs

To configure an access log for Oracle HTTP Server using Fusion Middleware Control, do the following:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Log Configuration** from the Administration menu.

The following access log configuration tasks can be set from this page:

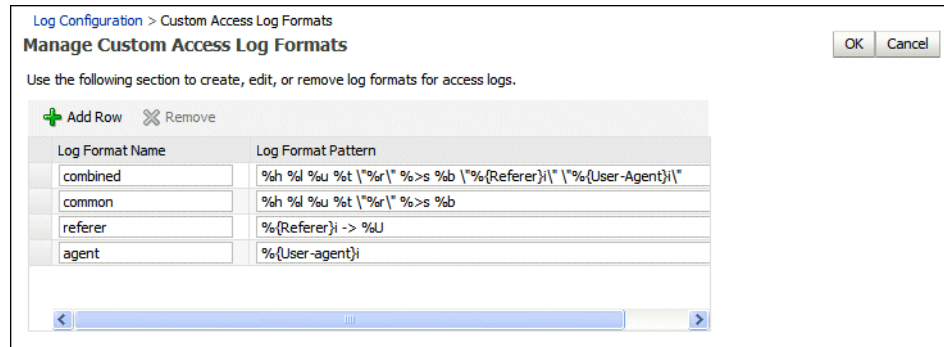
- [Configuring the Access Log Format](#)
- [Configuring the Access Log File](#)

### 7.2.2.1 Configuring the Access Log Format

Log format specifies the information included in the access log file and the manner in which it is written. To add a new access log format or to edit or remove an existing format, do the following:

1. From the Log Configuration page, navigate to the Access Log section.
2. Click **Manage Log Formats**.

The Manage Custom Access Log Formats page is displayed, as shown in the following figure.



3. Select an existing format to change or remove, or click **Add Row** to create a new format.
4. If you choose to create a new format, then enter the new log format in the Log Format Name field and the log format in the Log Format Pattern field.

**See Also:** Refer to the Apache HTTP Server documentation for information about log format directives.

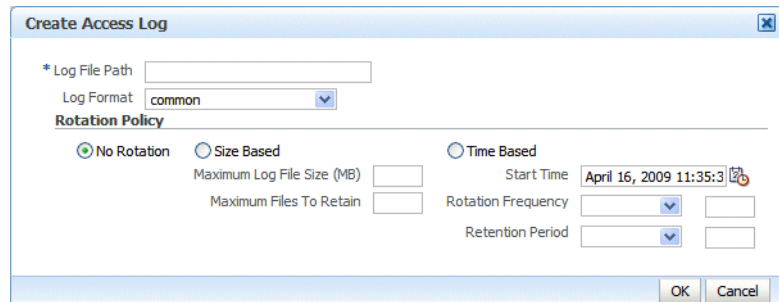
5. Click **OK** to save the new format.

### 7.2.2.2 Configuring the Access Log File

To configure an access log for file Oracle HTTP Server, do the following:

1. From the Log Configuration page, navigate to the Access Log section.
2. Click **Create** to create a new access log, or select a row from the table and click **Edit** button to edit an existing access log file.

The Create or Edit Access Log page is displayed.



3. Enter the path for the access log in the Log File Path field. This directory must exist before you enter it.
4. Select an existing access log format from the Log Format menu.
5. Select a rotation policy.
  - No Rotation – if you do not want to have the log file rotated ever.
  - Size Based – rotate the log file whenever it reaches a configured size. Set the maximum size for the log file in Maximum Log File Size (MB) field and the maximum number of error log files to retain in Maximum Files to Retain field.

- Time Based – rotate the log file whenever configured time is reached. Set the start time, rotation frequency, and retention period.
6. Click **OK** to continue.

Note that you can create multiple access log files.

## 7.3 Log Directives for Oracle HTTP Server

This section discusses Oracle HTTP Server error and access log-related directives in the `httpd.conf` file. The directives discussed are:

- [Oracle Diagnostic Logging Directives](#)
- [Apache HTTP Server Log Directives](#)

### 7.3.1 Oracle Diagnostic Logging Directives

Oracle HTTP Server by default uses Oracle Diagnostic Logging (ODL) for generating diagnostic messages. The following directives are used to setup logging using ODL:

- [OraLogMode](#)
- [OraLogDir](#)
- [OraLogSeverity](#)
- [OraLogRotationParams](#)

#### 7.3.1.1 OraLogMode

Enables you to choose the format in which you want to generate log messages. You can choose to generate log messages in the legacy Apache, ODL text, or ODL XML format.

`OraLogMode Apache | ODL-Text | ODL-XML`

Default value: `ODL-Text`

For example: `OraLogMode ODL-XML`

---

---

**Note:** The Apache log directives `ErrorLog` and `LogLevel` are only effective when `OraLogMode` is set to `Apache`. When `OraLogMode` is set to either `ODL-Text` or `ODL-XML`, the `ErrorLog` and `LogLevel` directives are ignored.

---

---

#### 7.3.1.2 OraLogDir

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

This directive is used only when `OraLogMode` is set to either `ODL-Text` or `ODL-XML`. When `OraLogMode` is set to `Apache`, `OraLogDir` is ignored and `ErrorLog` is used instead.

`OraLogDir <path>`

Default value: `ORACLE_INSTANCE/diagnostics/logs/OHS/component_name`

For example: `OraLogDir /tmp/logs`

### 7.3.1.3 OraLogSeverity

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

This directive is used only when `OraLogMode` is set to either `ODL-Text` or `ODL-XML`. When `OraLogMode` is set to `Apache`, `OraLogSeverity` is ignored and `LogLevel` is used instead.

```
OraLogSeverity <msg_type>[:msg_level]
```

Default value: `WARNING:32`

For example: `OraLogSeverity NOTIFICATION:16`

#### **msg\_type**

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

- INCIDENT\_ERROR
- ERROR
- WARNING
- NOTIFICATION
- TRACE

#### **msg\_level**

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

### 7.3.1.4 OraLogRotationParams

Enables you to choose the rotation policy for an error log file. This directive is used only when `OraLogMode` is set to either `ODL-Text` or `ODL-XML`. When `OraLogMode` is set to `Apache`, `OraLogRotationParams` is ignored.

```
OraLogRotationParams <rotation_type> <rotation_policy>
```

Default value: `S 10:70`

For example: `OraLogRotationParams T 43200:604800  
2009-05-08T10:53:29`

#### **rotation\_type**

This parameter can either be `S` (for sized-based rotation) or `T` (for time-based rotation).

#### **rotation\_policy**

When `rotation_type` is set to `S` (sized-based), set the `rotation_policy` parameter to:

```
maxFileSize:allFilesSize (in MB)
```

For example, when configured as `10:70`, the error log file is rotated whenever it reaches 10MB in size and a total of 70MB is allowed for all error log files (a maximum of  $70/10=7$  error log files will be retained).

When `rotation_type` is set to `T` (time-based), set the `rotation_policy` parameter to:

```
frequency(in sec) retentionTime(in sec) startTime(in  
YYYY-MM-DDThh:mm:ss)
```

For example, when configured as `43200:604800 2009-05-08T10:53:29`, the error log is rotated every 43200 seconds (that is, 12 hours), rotated log files are retained for maximum of 604800 seconds (7 days) starting from May 5, 2009 at 10:53:29.

## 7.3.2 Apache HTTP Server Log Directives

Although Oracle HTTP Server uses ODL by default for error logs, you can configure the `OraLogMode` directive to Apache to generate error log messages in the legacy Apache HTTP Server message format. The following directives are discussed in this section:

- [ErrorLog](#)
- [LogLevel](#)
- [LogFormat](#)
- [CustomLog](#)

### 7.3.2.1 ErrorLog

The `ErrorLog` directive sets the name of the file where the server logs any errors it encounters. If the file-path is not absolute then it is assumed to be relative to the `ServerRoot`.

This directive is used only when `OraLogMode` is set to `Apache`. When `OraLogMode` is set to either `ODL-Text` or `ODL-XML`, `ErrorLog` is ignored and `OraLogDir` is used instead.

**See Also:** For information about the `ErrorLog` directive, see the Apache HTTP Server documentation .

### 7.3.2.2 LogLevel

The `LogLevel` directive adjusts the verbosity of the messages recorded in the error logs.

This directive is used only when `OraLogMode` is set to `Apache`. When `OraLogMode` is set to either `ODL-Text` or `ODL-XML`, `LogLevel` is ignored and `OraLogSeverity` is used instead.

**See Also:** Refer to the Apache HTTP Server documentation for information about the `LogLevel` directive.

### 7.3.2.3 LogFormat

The `LogFormat` directive specifies the format of the access log file. By default, Oracle HTTP Server comes with the following four access log formats defined:

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
```

**See Also:** Refer to the Apache HTTP Server documentation for information about the `LogFormat` directive.

### 7.3.2.4 CustomLog

The `CustomLog` directive is used to log requests to the server. A log format is specified and the logging can optionally be made conditional on request characteristics



using environment variables. By default, the access log file is configured to use the common log format.

**See Also:** Refer to the Apache HTTP Server documentation for information about the `CustomLog` directive.

## 7.4 Viewing Oracle HTTP Server Logs

You can search, view, and list Oracle HTTP Server log files using Fusion Middleware Control, or you can download a log file to your local client and view the log files using another tool.

You can also use the text editor of your choice to view Oracle HTTP Server log files directly from the `ORACLE_INSTANCE` directory. By default, Oracle HTTP Server log files are located in the `ORACLE_INSTANCE/diagnostics/logs/OHS/component_name` directory.

As discussed in [Section 7.1, "Overview of Server Logs"](#), there are mainly two types of log files for Oracle HTTP Server: error logs and access logs. 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. 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.

**See Also:** For information about searching and viewing log files, see the *Oracle Fusion Middleware Administrator's Guide*



---

---

# Managing Application Security

This chapter contains an overview of Oracle HTTP Server security features, and provides configuration information for setting up a secure Web site.

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier Without Oracle WebLogic Server" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

This chapter includes the following sections:

- [Section 8.1, "About Oracle HTTP Server Security"](#)
- [Section 8.2, "Classes of Users and Their Privileges"](#)
- [Section 8.3, "Resources Protected"](#)
- [Section 8.4, "Authentication, Authorization and Access Control"](#)

## 8.1 About Oracle HTTP Server Security

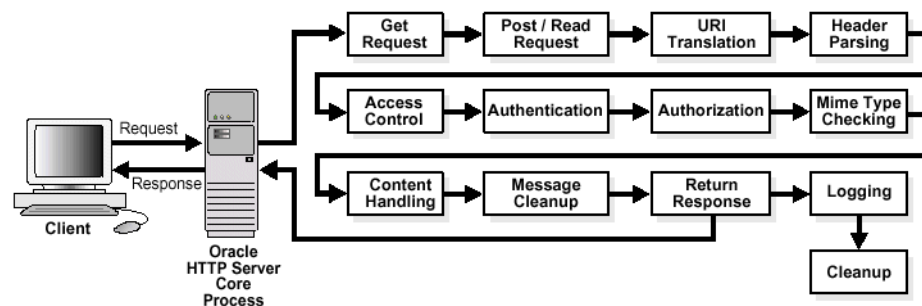
Security can be organized into the three categories of authentication, authorization, and confidentiality. Oracle HTTP Server provides support for all three of these categories. It is based on the Apache Web server, and its security infrastructure is primarily provided by the Apache modules, `mod_auth_basic`, `mod_authn_file`, `mod_auth_user`, and `mod_authz_groupfile`, and the Oracle modules, `mod_oss` and `mod_osso`. The `mod_auth_basic`, `mod_authn_file`, `mod_auth_user`, and `mod_authz_groupfile` modules provide authentication based on user name and password pairs, while `mod_authz_host` controls access to the server based on the characteristics of a request, such as host name or IP address, `mod_oss` provides confidentiality and authentication with X.509 client certificates over SSL, and `mod_osso` enables single sign-on authentication for Web applications.

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 8–1 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 8–1 represent the processing of the Apache `mod_auth_basic`, `mod_authn_file`, `mod_auth_user`, and `mod_authz_groupfile` modules.

**Figure 8–1 Steps for Handling URL Requests in Oracle HTTP Server**



## 8.2 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 Apache HTTP Server modules like `mod_auth_basic`, `mod_authn_file`, `mod_auth_user`, and `mod_authz_groupfile` modules and Oracle's `mod_oss1`. Such users have access to URLs defined in `http.conf` file.

**See Also:** [Section 8.4, "Authentication, Authorization and Access Control"](#).

- 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 Fusion Middleware Security Guide*

## 8.3 Resources Protected

Oracle HTTP Server may be configured since these resources are not protected out of the box:

- 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 Oracle WebLogic Server that are accessed through `mod_wl_ohs`. Oracle HTTP Server forms the first line of authentication and authorization for these components, although further authentication may occur at the component level.

## 8.4 Authentication, Authorization and Access Control

Oracle HTTP Server provides user authentication and authorization at two stages:

- **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.

### 8.4.1 Access Control

Access control refers to any means of controlling access to any resource.

**See Also:** Refer to the Apache HTTP Server documentation for more information on how to configure access control to resources.

### 8.4.2 User Authentication and Authorization

---

---

**Critical Note:** The FMW 10g based Single Sign-On Plug-Ins like OSSO Plug-In for Oracle HTTP Server and other web servers, such as IIS, has been deprecated and will be discontinued in future releases. You should upgrade to the Oracle Fusion Middleware 11g-based Single Sign-On Plug-In known as WebGate. For more information, see the *Oracle Fusion Middleware 11g Security Guide* at:

[http://docs.oracle.com/cd/E29505\\_01/core.1111/e10043/osso\\_a\\_intro.htm](http://docs.oracle.com/cd/E29505_01/core.1111/e10043/osso_a_intro.htm)

---

---

Authentication is any process by which you verify that someone is who they claim they are. Authorization is any process by which someone is allowed to be where they want to go, or to have information that they want to have.

#### 8.4.2.1 Using Apache HTTP Server Modules to Authenticate Users

Access control refers to any means of controlling access to any resource.

**See Also:** Refer to the Apache HTTP Server documentation for more information on how to authenticate users.

### 8.4.2.2 Using mod\_osso to Authenticate Users

---

**Note:** mod\_osso for use with Oracle Single Sign-On 10g is deprecated and is not recommended for use. Instead, you should use the Oracle Access Manager.

---

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 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 Oracle WebLogic Server, and it can also authenticate users for access to static files.

**See Also:** *Oracle Fusion Middleware Security Guide*

### 8.4.3 Support for FMW Audit Framework

Oracle HTTP Server supports a directive called OraAuditEnable. The purpose of this directive is to enable auditing of authentication and authorization using the FMW Common Audit Framework. This directive takes on two values, ON and OFF. When the value is set to ON, all the audit events reported by Oracle HTTP Server are recorded into an audit framework specific file. By default, the value is ON. If OraAuditEnable is set to OFF or is not present, then audit is disabled. The default httpd.conf file that is provided with Oracle HTTP Server sets it to ON. For more information, see

[http://download.oracle.com/docs/cd/E14571\\_01/core.1111/e10043/audintro.htm#JISEC2372](http://download.oracle.com/docs/cd/E14571_01/core.1111/e10043/audintro.htm#JISEC2372)

---

---

## Configuring mod\_oradav

This chapter describes distributed authoring and versioning (DAV) concepts, and explains how to configure OraDAV using the mod\_oradav module. The mod\_oradav module enables you to use OraDAV to access content in files from a Web browser or a WebDAV client.

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier Without Oracle WebLogic Server" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

This chapter includes the following sections:

- [Section 9.1, "Introduction to the mod\\_oradav Module"](#)
- [Section 9.2, "Configuring mod\\_oradav"](#)
- [Section 9.3, "WebDAV Security Considerations"](#)
- [Section 9.4, "OraDAV Performance Considerations"](#)
- [Section 9.5, "Globalization Support Considerations with OraDAV"](#)
- [Section 9.6, "Location of DAV Files"](#)

### 9.1 Introduction to the mod\_oradav Module

The mod\_oradav module is an extended implementation of the Apache implementation of the WebDAV specification. The mod\_oradav module is an OCI application written in C and is integrated with Oracle HTTP Server. The mod\_oradav module enables WebDAV clients to connect to files, read and write content, and query and lock documents.

This section includes the following subsections:

- [Section 9.1.1, "WebDAV"](#)
- [Section 9.1.2, "OraDAV"](#)
- [Section 9.1.3, "OraDAV Architecture"](#)

- [Section 9.1.4, "OraDAV Usage Model"](#)
- [Section 9.1.5, "PROPFIND Security"](#)

### 9.1.1 WebDAV

WebDAV is a protocol extension to HTTP 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 access and optionally change that content from various devices.

---

---

**Note:** When a WebDAV client first connects to Oracle HTTP Server, it must use the full ServerName string in the URL for the connection. Do not use an abbreviated form of the server name.

For example, if the server name value is `server1.example.com`, then connect to Oracle HTTP Server using the string `http://server1.example.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 run, and generate BAD\_GATEWAY errors.

---

---

### 9.1.2 OraDAV

OraDAV refers to the set of capabilities available through the mod\_oradav module to Oracle Fusion Middleware users. Some OraDAV-specific terms include:

- OraDAV: Code in the Oracle HTTP Server that supports file-based DAV access. Apache DAV directives can be used with OraDAV.

**See Also:** For more information about DAV directives, see the article written by Greg Stein ([gstein@lyra.org](mailto:gstein@lyra.org)) available at

[http://www.webdav.org/mod\\_dav/install.html#apache](http://www.webdav.org/mod_dav/install.html#apache)

- OraDAV API: Stored procedure calls that are used by the OraDAV driver to provide support for the following WebDAV functions over the Internet:
  - Reading and writing documents
  - Locking and unlocking documents
  - Managing, such as creating, populating, and deleting, hierarchies of information
  - Retrieving properties associated with documents
  - Associating properties with specific documents
- OraDAV driver: Stored procedure implementation of the OraDAV driver API that runs in Oracle and manages a repository.

The primary users of OraDAV are Oracle HTTP Server Web administrators and content editors. End users interact only indirectly with OraDAV through Web



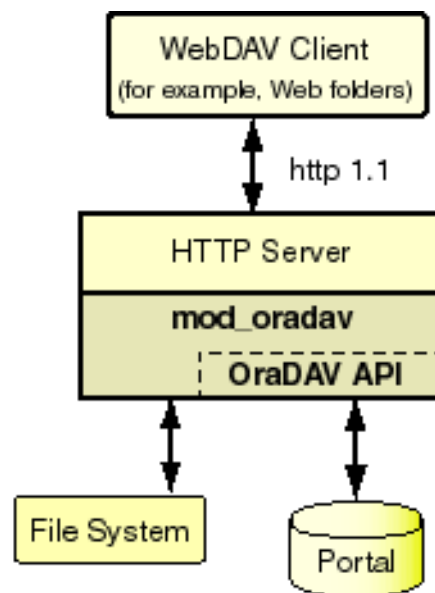
browsers or WebDAV client tools. OraDAV interaction requires the following proficiency:

- The Web administrator 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 content editors need to know how to connect to the server, and upload and retrieve files.

### 9.1.3 OraDAV Architecture

The mod\_oradav module, which includes OraDAV, is part of the Oracle HTTP Server architecture. 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 Web folders, passing HTTP requests to Oracle HTTP Server. If the request is for content stored in the file system, the mod\_oradav module handles the access. If the request is for content stored in Oracle Portal, the OraDAV API handles the access.

The OraDAV API capabilities are equivalent to using the mod\_oradav module running with a file system. The following HTTP methods are supported by the OraDAV API:

- COPY: Copies files within a Web site folder.
- DELETE: Deletes files within a Web site folder.
- MOVE: Moves files within a Web site folder.
- MKCOL: Makes a new directory.
- GET: Retrieves a file from the server. This method is not supported by Oracle Web Cache.
- PUT: Puts a file back to the server. This method is not supported by Oracle Web Cache.

- HEAD: Gets the header content of a file without retrieving the file.
- LOCK: Locks a file when the file is checked out. This method is not supported by Oracle Web Cache.
- UNLOCK: Unlocks a file after check in. This method is not supported by Oracle Web Cache.
- PROPFIND: Gets properties defined for a file.
- PROPPATCH: Sets the properties for a file.

The OraDAV API supports shared and exclusive locking, retrieving basic DAV properties, and defining and retrieving server-defined properties or client-defined properties. Set-based operations such as COPY, MOVE, DELETE can be done completely by a single call to an OraDAV driver.

### 9.1.4 OraDAV Usage Model

OraDAV usage can involve any combination of the following activities:

- Browsing: Read-only activity which uses WebDAV to access content on the file server. Its usage model is typical of a read-only Web site.

The DAVOraReadOnly directive specifies whether or not WebDAV should be used in a read-only mode by WebDAV clients. A value of Off specifies that WebDAV clients function normally. A value of On prevents WebDAV clients from performing write operations while using WebDAV. It does allow read-only activity by Web browsers and WebDAV clients. The default is Off.

- 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, then 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 use 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.

### 9.1.5 PROPFIND Security

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 to fetch the files.

## 9.2 Configuring mod\_oradav

Use the Advanced Server Configuration page of Fusion Middleware Control to configure the mod\_oradav module.

This section includes the following subsections:

- [Section 9.2.1, "OraDAV Configuration Directives"](#)
- [Section 9.2.2, "Using Fusion Middleware Control to Configure mod\\_oradav"](#)

### 9.2.1 OraDAV Configuration Directives

When Oracle Fusion Middleware is installed, all required OraDAV parameters, called "directives", are set to their default values. If the default values do not meet your needs, you can modify the values for required parameters and specify values for optional parameters. The OraDAV parameters in the mod\_oradav.conf file start with "DAV" and "DAVParam".

---



---

**Note:** To configure the parameters use Fusion Middleware Control. Do not edit the mod\_oradav.conf file directly. Doing so may harm your installation.

---



---

The DAV parameter indicates that a URL location is DAV-enabled. The DAV keyword is followed by one of the following values:

- `On` – indicates that mod\_oradav is to use the local file system for content.
- `Oracle` – indicates that mod\_oradav is to use OraDAV for all content.

The 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.

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 additional DAV parameters, such as DavMinTimeout.

[Example 9–1](#) shows the syntax to configure access to files on the local system. It specifies that the directory dav\_portal under the Web server documents directory is to be DAV-enabled, along with all directories under dav\_portal in the hierarchy. There must not be any symlinks defined on the dav\_portal directory or any of its subdirectories.

#### **Example 9–1 Configuring File System Access**

```
<Location /dav_portal>
  DAV On
</Location>
```

The following recommendations should be considered when mapping containers under the root location:

- Do not map the root itself. For example, do not specify `<Location / >` in the `mod_oradav.conf` file.
- Do not map a container as a subelement in the hierarchy to another container. For example, do not specify the containers `<Location /project1>` and `<Location /project1/project2>`. 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.

---

**Note:** The OraDAV directives are described in Appendix D, "OHS Module Directives". See [Section E.3, "mod\\_oradav."](#)

---

## 9.2.2 Using Fusion Middleware Control to Configure mod\_oradav

On the Advanced Server Configuration page of Fusion Middleware Control, you can enter parameters within a `<Location>` container directive in the `mod_oradav.conf` file. The `<Location>` container directive specifies the DAV-enabled URL. The DAV keyword is followed by the parameter `On`, which instructs `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) to be DAV-enabled, along with all directories under `myfiles` in the hierarchy. There must not be any symbolic links defined on the `myfiles` directory or any of its subdirectories.

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

## 9.2.3 Editing mod\_oradev.conf

Create the `mod_oradev.conf` file at the following location:

```
<oracle_instance>/config/OHS/ohs1/moduleconf/mod_oradev.conf
```

Insert the below mentioned entries in the `mod_oradev` file.

```
LoadModule oradav_module "${ORACLE_HOME}/ohs/modules/mod_oradav.so"
```

```
#<Location /dav/lsn>
#DAV oracle
#DAVDepthInfinity Off
#DavParam ORACONTAINERNAME LSNDV
#DavParam ORALockExpirationPad 0
#DavParam ORAException RAISE
#DavParam ORATraceLevel 0
#DavParam OraTraceEvents "request"
#DavParam ORASERVICE psgprod
#DavParam ORACONNECTSN db-host
#DAVParam ORAUser lsn
#DAVParam ORAPassword psg_lsn
#DAVParam ORAPackageName ordsys.dav_api_driver
#DAVParam OraWebCacheReadOnly On
#</Location>
```

Save the file, and restart Oracle HTTP Server as described in [Section 4.1.4, "Restarting Oracle HTTP Server."](#) The option for editing `mod_oradev.conf` file is displayed on the Advanced Server Configuration page of Fusion Middleware Control.

## 9.3 WebDAV Security Considerations

Because WebDAV enables read/write capabilities, Internet users can write to your Web site or to an Oracle repository. A major concern is preventing users from placing an inappropriate file, such as a Trojan horse, that can run 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 run. However, `mod_oradav` is disabled by default in new installations of Oracle HTTP Server so that your system is secure out-of-the-box.

**See Also:** Apache Module `mod_dav` Security Issues in the Apache HTTP Server documentation.

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 `mod_oradav.conf` file. This restricts who can use your system for remote storage, preventing unauthorized users from filling up your disks.

In addition, 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 runs. To do this, create an alias for the directory associated with the execution context, and then DAV-enable the aliased location.

## 9.4 OraDAV Performance Considerations

This section provides information that can help you optimize the performance of various operations. It contains the following subsections:

- [Section 9.4.1, "Using Disk Caching with OraDAV"](#)
- [Section 9.4.2, "Bypassing Oracle Web Cache for WebDAV Activities"](#)

### 9.4.1 Using Disk Caching with OraDAV

The performance benefit from disk caching is greatest with medium to large-size files (approximately 50 KB and larger). 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 server to the local system is very small compared to the time required for accessing the server to check if the file has changed. If disk caching is not used, there is no check of the server to see if the file has changed, and the file is copied 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. 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.

## 9.4.2 Bypassing Oracle Web Cache for WebDAV Activities

Oracle Web Cache enhances performance for most Web activity that involves client read-only operations of data on the Web server system. Oracle Web Cache does not cache OraDAV operations for GET, PUT, LOCK and UNLOCK, which are designed for read/write capability. For better performance, WebDAV clients can connect directly to Oracle HTTP Server.

To bypass Oracle Web Cache for WebDAV clients, you can send requests directly to the Oracle HTTP Server listen port, which is set in the `httpd.conf` file. By doing this, WebDAV clients will connect directly to Oracle HTTP Server, resulting in better performance than if Oracle Web Cache is used.

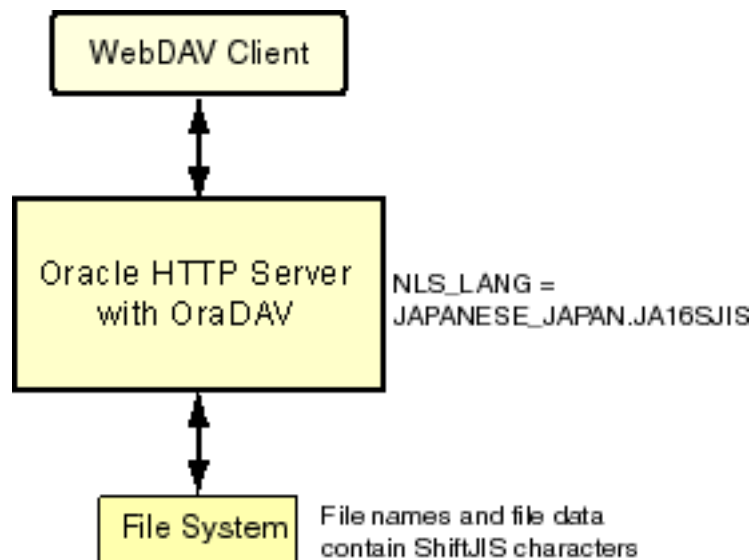
**See Also:** *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*

## 9.5 Globalization Support Considerations with OraDAV

The `DAVOraNLSLangClient` directive 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` specifies that no conversion is needed. A value of `On` specifies that the character set for the file system provides for conversion of all possible characters in client requests. The default is `Off`.

For access to the local file system, 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 `DAVOrAUseNLSLang`.

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**

You must do the following:

1. Set the NLS\_LANG value to JAPANESE\_JAPAN.JA16SJIS.
2. Include the following in the mod\_oradav.conf file:

```

<Location /dav_public>
  DAV On
  DAVOraNLSLangClient On
</Location>
  
```

---



---

**Note:** If you use Microsoft Internet Explorer with OraDAV and a multibyte character set, you must disable the Internet option Always send URLs as UTF-8, located under the Advanced tab in the Internet Options section. By default, this option is enabled. The requirement to disable this option applies to both database access and file system access.

---



---

## 9.6 Location of DAV Files

When the ORACLE\_HOME/ohs/cas/templates/default/moduleconf/mod\_oradav.conf file is configured to use file storage, it places the files by default in:

```
ORACLE_INSTANCE/config/OHS/component_name/moduleconf
```

Oracle Fusion Middleware Backup and Recovery Service backs up this default location. If you change the location where the files are stored, and you want Oracle Fusion Middleware Backup and Recovery Service to backup the files, then you must register the new location.





# Part III

---

## Appendices and Glossary

This part contains the following appendixes and a glossary:

- [Appendix A, "Using Oracle Plug-Ins for Third-Party Web Servers"](#)
- [Appendix B, "Frequently Asked Questions"](#)
- [Appendix C, "Troubleshooting Oracle HTTP Server"](#)
- [Section E, "OHS Module Directives"](#)
- ["Glossary"](#)



---

---

# Using Oracle Plug-Ins for Third-Party Web Servers

This appendix explains how the Oracle Proxy Plug-In and Oracle SSO Plug-In enable you to use Oracle Fusion Middleware components in conjunction with a third-party HTTP listener.

---

---

**Note:** Unless otherwise mentioned, the information in this document is applicable when Oracle HTTP Server is installed with Oracle WebLogic Server and Oracle Fusion Middleware Control. It is assumed that readers are familiar with the key concepts of Oracle Fusion Middleware, as described in the *Oracle Fusion Middleware Concepts Guide* and the *Oracle Fusion Middleware Administrator's Guide*.

For information about installing Oracle HTTP Server in standalone mode, see "Installing Oracle Web Tier in Stand-Alone Mode" in the *Oracle Fusion Middleware Installation Guide for Oracle Web Tier*.

---

---

This appendix includes the following sections:

- [Section A.1, "Using Oracle Proxy Plug-In"](#)
- [Section A.2, "Using Oracle SSO Plug-In"](#)

Documentation from the Apache Software Foundation is referenced when applicable.

## A.1 Using Oracle Proxy Plug-In

Oracle Proxy Plug-In enables you to proxy/send requests from a third-party HTTP listener to Oracle Fusion Middleware. The Oracle Proxy Plug-In is provided and certified to work with Sun Java System Web Server Enterprise Edition on UNIX and Microsoft Windows systems, or Microsoft Internet Information Server (IIS) v6.0 and 7.0.

For other third-party HTTP listeners, you can use the respective listener's native proxy functionality.

**See Also:**

[http://www.oracle.com/technology/software/products/ias/files/fusion\\_certification.html](http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html) for complete certification information.

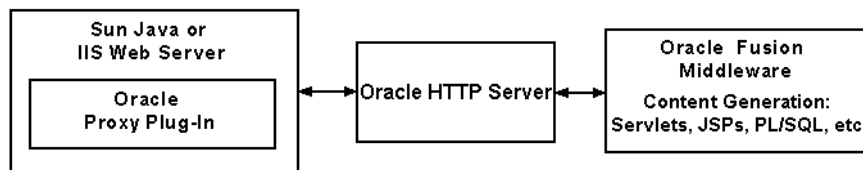
This section includes the following topics:

- [Section A.1.1, "Overview of Oracle Proxy Plug-In"](#)
- [Section A.1.2, "Installing Oracle Proxy Plug-In"](#)
- [Section A.1.3, "Configuring Oracle Proxy Plug-In"](#)
- [Section A.1.4, "Configuring Sun Java System Web Server to Use Oracle Proxy Plug-In"](#)
- [Section A.1.5, "Configuring Microsoft IIS 6.0 Listener to Use Oracle Proxy Plug-In"](#)
- [Section A.1.6, "Configuring Microsoft IIS 7.0 Listener to Use Oracle Proxy Plug-In"](#)
- [Section A.1.7, "Concurrently Serving Files From IIS While Proxying Requests to OHS"](#)
- [Section A.1.8, "Oracle Proxy Plug-In Usage Information"](#)
- [Section A.1.9, "Troubleshooting Oracle Proxy Plug-In"](#)

## A.1.1 Overview of Oracle Proxy Plug-In

Oracle Proxy Plug-In is a reverse HTTP proxy. The **plug-in** forwards incoming HTTP requests from a Sun Java System or Microsoft IIS server HTTP listener to an Oracle Fusion Middleware instance, as shown in [Figure A-1](#).

**Figure A-1 Oracle Proxy Plug-In**



The proxy logic is provided as a plug-in, 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 used with Oracle HTTP Server.

Oracle HTTP Server can mimic the address and port used by the third-party listener. When sending a request to Oracle HTTP Server, the proxy can be configured to send a different Host: HTTP header than the actual host name and port that the request is being sent to, so that other applications are safeguarded from the reverse proxy.

Oracle Proxy Plug-In does not perform load balancing; it only forwards requests to a single back-end Oracle HTTP Server for a given URL. Also, Oracle Proxy Plug-In does not support SSL between a third-party listener and Oracle HTTP Server.

## A.1.2 Installing Oracle Proxy Plug-In

The Oracle Proxy Plug-Ins for Sun Java System Web Server and Microsoft IIS are available in the Oracle Fusion Middleware Web Tier and Utilities installer (11.1.1.2 and later *full* installers, not on patch-set installers). Place the appropriate definition file and shared library in directories that are accessible to the third-party listener.

[Table A-1](#) contains information about the shared libraries for the OracleAS Proxy Plug-In.

**Table A–1 Oracle Proxy Plug-In Shared Libraries**

Listener	Location and Description
Sun Java System Web Server	<p>Location on CD-ROM:</p> <ul style="list-style-type: none"> <li>■ UNIX: /Disk1/plugins/sjsws/oracle_proxy.so</li> <li>■ Microsoft Windows: \Disk1\plugins\sjsws\oracle_proxy_nes.dll</li> </ul> <p>To install the plug-in for the listener, copy the .dll file to a directory that the listener has read and execute privileges for UNIX, or a directory that the listener can access for Microsoft Windows.</p>
Microsoft IIS	<p>Location on CD-ROM: \Disk1\plugins\iis\oracle_proxy.dll</p> <p>To install the plug-in for the listener, copy oracle_proxy.dll to a directory the listener can access.</p>

### A.1.3 Configuring Oracle Proxy Plug-In

There is one definition file for the Oracle Proxy Plug-In that controls the proxy functionality. The presence of the file in the Web server file system makes the proxy functionality active. You also need to modify the configuration file(s) specific to the third-party listener, such as the Sun Java System Web Server or Microsoft IIS configuration file, to enable the plug-in on these listeners. The definition file for the Oracle Proxy Plug-In can have any name.

#### A.1.3.1 Proxy Server Definition File

The [proxy server](#) definition file must reside in a directory that is readable by the third-party listener. For example, you could create a directory called `proxy` in a directory on your system, and place the proxy server definition file, the proxy shared library file, and proxy log files in it.

The proxy server definition file contains the following parameters:

- Name-value pairs that describe the servers that will be used to proxy requests to Oracle Fusion Middleware
- 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.

#### A.1.3.2 Proxy Definition File Parameters

The following proxy parameters are used in the proxy definition file:

- [oproxy.serverlist](#)
- [oproxy.servername.hostname](#)
- [oproxy.servername.port](#)
- [oproxy.servername.alias](#)
- [oproxy.servername.resolveall](#)
- [oproxy.servername.urlrule](#)

**A.1.3.2.1 [oproxy.serverlist](#)** Lists all the server names recognizable to the plug-in.

Category	Value
Allowable Values	Comma-separated list of server names, one for each Oracle HTTP Server to which requests will be sent. All servers in the server list must also be defined in the file.  <b>Note:</b> Oracle Proxy Plug-In does not do load balancing; therefore, if you define two servers with same <code>urlrule</code> , it will forward requests to only one of those servers.
Default Value	None. At least one server name must be provided for the proxy to be functional.
Example	<code>oproxy.serverlist=ohs1,ohs2</code>

**A.1.3.2.2 `oproxy.servername.hostname`** Defines the host name to use when communicating with a specific server.

Category	Value
Allowable Values	A valid host name
Default Value	None
Example	<code>oproxy.ohs1.hostname=host1.acme.com</code>

**A.1.3.2.3 `oproxy.servername.port`** Defines the port to use when communicating with a specific server.

Category	Value
Allowable Values	A port value
Default Value	80
Example	<code>oproxy.ohs1.port=7777</code>

**A.1.3.2.4 `oproxy.servername.alias`** Supports the alias feature of the proxy by defining the host name and port that clients use to access the third-party HTTP listener. If defined, then this value will be passed as the Host: HTTP header. If not defined, then the host name and port of the system actually being communicated with will be sent.

Category	Value
Allowable Values	host:port
Example	<code>oproxy.ohs1.alias=www.example.com:80</code>

**A.1.3.2.5 `oproxy.servername.resolveall`** Directs the proxy plug-in to resolve the host name 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 requires going to the DNS server for every incoming request, and should only be used if the mapping from host name to IP address will change dynamically.

Category	Value
Allowable Values	true   false
Default Value	false
Example	<code>oproxy.ohs1.resolveall=true</code>

**A.1.3.2.6 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
Example	<code>oproxy.ohs1.urlrule=/foo/*</code>

The following types of rules can be used:

- Exact matches: One URL is mapped to a server. For example:

```
oproxy.ohs1.urlrule=/foo/bar/foo.html
```

maps only the URL `/foo/bar/foo.html`. This would be the proxy for the server with the name `ohs1`.

- Context matches: A set of URLs with a common prefix or context are mapped to a server. For example:

```
oproxy.ohs1.urlrule=/foo/*
```

maps URLs beginning with `/foo` to the server with the name `ohs1`.

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.

Parameters	URL Request	Result
<code>oproxy.ohs1.urlrule=/foo/*</code> <code>oproxy.ohs1.stripcontext=true</code>	<code>http://hostname/foo/header1.gif</code>	<code>ORACLE_INSTANCE/config/OHS/component_name/htdocs/header1.gif</code>
<code>oproxy.ohs1.urlrule=/foo/*</code> <code>oproxy.ohs1.stripcontext=false</code>	<code>http://hostname/foo/header1.gif</code>	<code>ORACLE_INSTANCE/config/OHS/component_name/htdocs/foo/header1.gif</code>

- Suffix matches: All files with a common file extension are mapped to a server.

For example:

```
oproxy.ohs1.urlrule=/*.jsp
```

maps all the URLs that end in `.jsp` to the server `ohs1`. This can be combined with the context rule to have `/foo/bar/*.jsp` so that only URLs that start with `/foo/bar` and end in `.jsp` would be proxied.

---



---

**Note:** For `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 as follows:

- `/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/*`
- 
- 

### A.1.3.3 Defining Oracle Proxy Plug-In Behavior

In the proxy server definition file, you define which servers and URLs to use as 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 plug-ins, such as the following:

```
oproxy.serverlist=ohs1,ohs2
```

2. View the Oracle HTTP Server ports using Fusion Middleware Control or the `opmnctl` command-line utility. The port number will be used in the next step.
3. Set the relevant properties (host name, port, and server alias) for each server. For example:

```
oproxy.ohs1.hostname=host1.acme.com.us.example.com
oproxy.ohs1.port=7777
oproxy.ohs1.alias=www.example.com
```

The host name must be provided. If an alias value is not given, the combination of the host name and port is used. The alias enables the backend server to receive requests that have an HTTP Host header that looks exactly like the one the client delivers to the third-party listener.

4. Set the `urlrule` parameter to specify redirection between servers. For example, this rule maps all incoming requests to proxy to the Web server on the server `ohs1`:

```
oproxy.ohs1.urlrule=/*
```

[Example A-1](#) is 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=ohs1

# Hostname to use when communicating with a specific server.
oproxy.ohs1.hostname=host1.acme.com

# Port to use when communicating with a specific server.
oproxy.ohs1.port=7777

# URL(s) that will be redirected to this server.
```



```
oproxy.ohsl.urlrule=/*
```

## A.1.4 Configuring Sun Java System Web Server to Use Oracle Proxy Plug-In

This section provides proxy plug-in configuration information for Sun Java System Web Server on UNIX and Microsoft Windows platforms.

The default configuration files for Sun Java System Web Server route all incoming requests for the URI `/servlet` to the Sun Java System servlet handler. The Oracle Proxy Plug-In does not override the Sun Java System server configuration settings. You must ensure that the URL mappings to the Oracle Proxy Plug-In are distinct from the URL mappings to the Sun Java System servlet engine.

If you are configuring the Sun Java System Web Server on Microsoft Windows, use forward slashes (`/`) in all paths.

1. Open the `magnus.conf` file located in the `/config` directory for the Sun Java System Web Server version 6.1 or version 7.0.
2. Add the following load-modules line, depending on the operating system:

- For UNIX:

```
Init fn="load-modules" shlib="/path/oracle_proxy.so" funcs=op_init,op_
objecttype,op_service
```

- For Microsoft Windows:

```
Init fn="load-modules" shlib="/path/oracle_proxy_nes.dll" funcs=op_init,op_
objecttype,op_service
```

In the preceding lines, *path* is the path to the shared library for the plug-in. This tells the listener where the proxy shared library is, and which functions are exposed by this library.

3. Add the following configuration parameters:

```
Init fn="op_init" server_defs="/path/to/proxy/definition/file" log_
file="/path/to/proxy/log/file" log_level=loglevel
```

For example:

```
Init fn="op_init" server_defs="/oracle/proxyplugin/proxydefs" log_
file="/oracle/proxyplugin/oproxy.log" log_level=error
```

The proxy server definition file contains all the configuration information for the servers that can communicate with the proxy plug-in. The definition file can have any name.

A log file and log level can be specified to log messages from the plug-in. This is optional.

**See Also:** See [Section A.1.3.1, "Proxy Server Definition File"](#) 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. Save the file.
7. Start the listener using the Sun Java System Web Server GUI or the shell script.

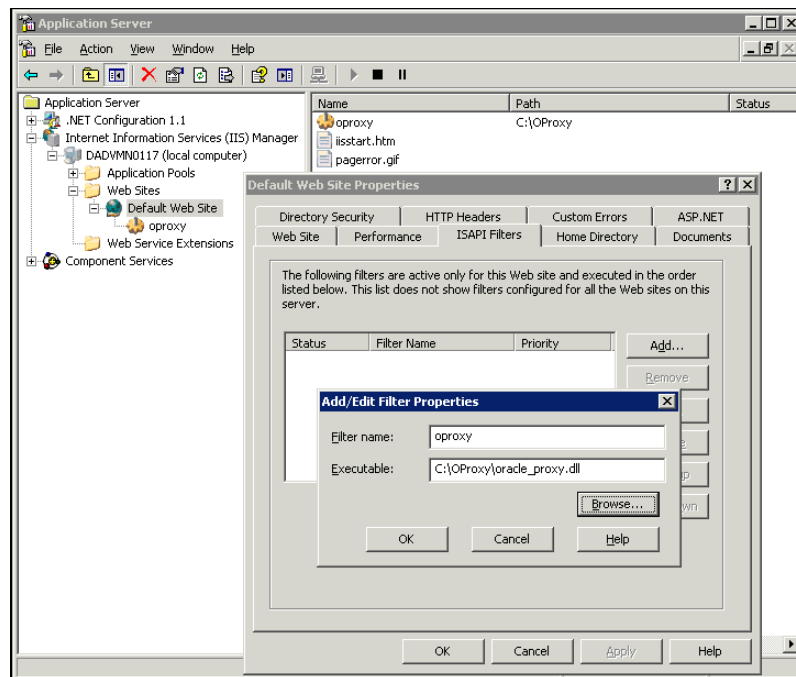
### A.1.5 Configuring Microsoft IIS 6.0 Listener to Use Oracle Proxy Plug-In

This section provides proxy plug-in configuration instructions for the Microsoft IIS 6.0 listener on Microsoft Windows platforms. The procedure involves creating Microsoft Windows registry entries and using the Microsoft IIS 6.0 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. Use the Microsoft Windows Registry Editor to create new registry entries.
  - a. From the Start menu, Select **Run**, and the type `regedit` in the dialog box, and click **OK**. The Registry Editor window opens
  - b. Expand the `HKEY_LOCAL_MACHINE` folder by clicking the plus sign (+) preceding its name In the Registry Editor window.
  - c. Expand the `SOFTWARE` folder by clicking the plus sign (+) preceding its name, and then Click the **ORACLE** folder.
  - d. From the Edit menu, select **New > Key**. A new folder is added under the `ORACLE` folder with the name `New Key #1`.
  - e. Enter `IIS Proxy Adapter` for the key name.
  - f. From the Edit menu, select **New > String Value**. A new value is added in the right window pane with the name `New Value #1`. Enter `server_defs` for the value name.
  - g. From the Edit menu, select **Modify**. The Edit String dialog box appears.
  - h. In the Value data field, enter the full path of your proxy server definition file, and then click **OK**.
  - i. Optionally, you can specify `log_file` and `log_level` using the procedure specified in steps **d.) – h.)**.
    - Add a string value with the name `log_file` and the desired location of the log file, such as `d:\proxy\proxy.log`.
    - 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`.
  - j. Close the Registry Editor window by selecting **Exit** from the File menu.
2. Use the Microsoft IIS management console to add a new virtual directory to your Microsoft IIS Web site with the same physical path as that of the `oracle_proxy.dll`.
  - a. Open the IIS Manager by clicking **Start -> Programs -> Administrative Tools -> Internet Information Services (IIS) Manager**.
  - b. Expand the server folder by clicking the plus sign (+) preceding the server name.
  - c. Right-click the **Default Sites** folder, and then select the **New -> Virtual Directory** option from the menu.

- d. In the Virtual Directory Creation Wizard window, enter **oproxy** for the Alias. Then, enter the path or browse to the directory containing the `oracle_proxy.dll` file (e.g., `c:\OProxy`) and select the **Execute (such as ISAPI applications or CGI)** check box.
  - e. Click **Finish** to close the Virtual Directory Creation Wizard.
3. Use the Microsoft IIS management console to add `oracle_proxy.dll` as a filter in your Microsoft IIS Web site. The name of the filter should be **oproxy** and its executable must point to the directory containing the `oracle_proxy.dll` file.
    - a. Right-click the **Default Sites** folder, and then select the **Properties** option from the menu.
    - b. In the Default Web Site Properties window, select the **ISAPI Filters** tab.
    - c. Click **Add** to add a new filter.  
The Add/Edit Filter Properties window is displayed.
      - d. Enter **oproxy** for the filter name. Then, enter the path or browse to the directory containing the `oracle_proxy.dll` file (e.g., `c:\OProxy\oracle_proxy.dll`).



- e. Click **OK** to close the Add/Edit Filter Properties window.
  - f. Click **OK** to close the Default Web Site Properties window.
4. Under the server name folder, click the **Web Service Extensions** folder to open the Web Service Extensions page. Select **All Unknown ISAPI Extensions** in the right panel and click **Allow**. The status of this item should change from *Prohibited* to *Allowed*.
  5. Select the **Application Pools** folder, and then right-click the **Properties** option from the menu. On the Application Pools Properties window, select the **Identity** tab and change the Predefined identity to **Local System**. A confirmation dialog box displays asking Are you sure you want to do this? Select **Yes**.
  6. Stop and restart the Microsoft IIS 6.0 server.

---

---

**Note:** To restart Microsoft IIS, you must stop all the Microsoft IIS services through the control panel or restart the computer. This is the only way to ensure that the .dll is reloaded. Restarting Microsoft IIS 6.0 through the management console is not sufficient.

---

---

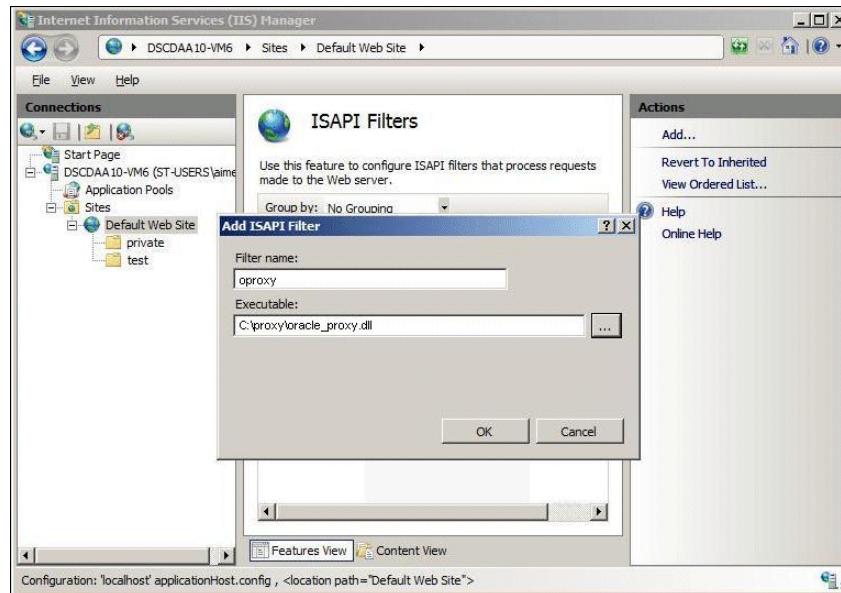
7. Make sure the newly created `oproxy` filter is marked with a green upward arrow.

## A.1.6 Configuring Microsoft IIS 7.0 Listener to Use Oracle Proxy Plug-In

This section provides proxy plug-in configuration instructions for the Microsoft IIS 7.0 Listener on Microsoft Windows platforms. The procedure involves creating Microsoft Windows registry entries and using the Microsoft IIS 7.0 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. Complete step 1 in [Section A.1.5, "Configuring Microsoft IIS 6.0 Listener to Use Oracle Proxy Plug-In"](#) to use the Registry Editor to create new registry entries for the plug-in.
2. Use the Microsoft IIS management console to add the proxy plug-in filter:
  - a. Open the IIS Manager by clicking **Start -> Programs -> Administrative Tools -> Internet Information Services (IIS) Manager**.
  - b. Expand the server folder by clicking the plus sign (+) preceding the server name (for example, DSCDAA10-VM6).
  - c. Expand the Sites folder by clicking the plus sign (+) preceding its name.
  - d. Click the **Default Web Site** icon to open the Default Web Site Home page.
  - e. Double-click the **ISAPI Filters** icon to open the ISAPI Filters page, and then complete the following tasks:
    - In the Actions pane, click **Add** to open the Add ISAPI Filter dialog box.
    - In the Filter Name field, enter a user-friendly name for the ISAPI filter.
    - In the Executable field, enter the file system path for the location of ISAPI filter file, or click the **ellipsis** button (...) to navigate to the folder that contains the ISAPI filter file.



– Click **OK**.

3. Follow these steps to configure the newly added ISAPI filter:
  - a. Click the **Default Web Site** icon in the navigation panel to view all the settings related to the application that can be modified.
  - b. Click the **Handler Mappings** option to set the mappings for the handler for a particular MIME type.
  - c. Click the **StaticFile** option in the Handler Mappings page, and in the Edit Module Mapping dialog box, change the Request path to `*.*`.
  - d. In the Actions area of the Handler Mappings page, click the **Add Script Map** option.
  - e. In the Edit Script Map dialog box: enter `*` for the Request path. Use the Executable field to browse to the `oracle_proxy.dll` file and add it as the executable. Name it as `proxy`.
  - f. Click the **Request Restrictions** button to open the Request Restrictions dialog box. Clear the **Invoke handler only if the request is mapped to** check box and then click **OK** to add this Handler mapping.
  - g. Click **Yes** on the Add Script Map dialog box.

When you click the Root node of the IIS manager tree, and then click on the ISAPI and CGI Restrictions, you should see an entry for the `oracle_proxy.dll`, as shown here:

```
[No Description] Allowed C:\proxy\oracle_proxy.dll
```

4. Restart the Microsoft IIS server by opening the Services Control Panel, and then stopping and restarting the World Wide Web Publishing Service.

---

**Note:** To restart Microsoft IIS, you must stop all the Microsoft IIS services through the control panel or restart the computer. This is the only way to ensure that the `.dll` is reloaded. Restarting Microsoft IIS 7.0 through the management console is not sufficient.

---

## A.1.7 Concurrently Serving Files From IIS While Proxying Requests to OHS

Configuring IIS7 for the Oracle Proxy Plugin by using the steps provided above will only proxy requests to OHS. In some circumstances, you might want to serve files from IIS at the same time you are proxying requests to OHS. To do so, after configuring the proxy plugin and confirming it works (Section A.1.6, "Configuring Microsoft IIS 7.0 Listener to Use Oracle Proxy Plug-In") do the following:

1. Right click Default web site and then click **Add Virtual Directory**.
2. In the Alias field, enter `static` and set the physical path to `c:\inetpub\wwwroot\static`. Click **OK**.

---

---

**Note:** Your physical path might be different if you put the files in a different location. You will need to modify the path accordingly.

---

---

A static folder under Default Web Site appears.

3. Click the static folder under Default Web Site.  
The static Home page appears.
4. On the static Home page click **Handler Mappings** then **View Ordered List** on the right pane. You will see ordered list of Handler Mappings.
5. Click **proxy** and then move it down below StaticFile handle mapping (so that the StaticFile handler mapping is above the proxy handler mapping).
6. Create a static folder under `c:\inetpub\wwwroot` and place some HTML file (for example, `index.html`) inside this folder.
7. Restart the IIS7 by restarting the World Wide Web publishing Service under services.
8. Now try to access the `index.html` by opening `http://localhost:80/static/index.html`.
9. Try to access the site that you have configured through the proxy plugin.

## A.1.8 Oracle Proxy Plug-In Usage Information

This section highlights development and usage practices to consider when developing an application that runs behind the Oracle Proxy Plug-In. Some of the practices also apply when enabling an application to run behind Oracle Web Cache.

- Check for configurations based on 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 Oracle HTTP Server. Specifically, look for dependencies on obtaining information about the client based on the connection made to Oracle HTTP Server, such as using the SSL certificate for authentication. Currently, Secure Socket Layer (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 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 Oracle Proxy Plug-In and Oracle Web Cache are used, but other client information may, in practice, represent the system on which the proxy resides rather than the actual client host. These behaviors must be discovered and eliminated in cases where Oracle HTTP Server is not the external listener for Oracle Fusion Middleware.

- 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 the code is not embedding the server name into pages that are sent back to the client. To test for this behavior, use a Web crawler application (also known as a spider) to traverse 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 host name 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.

To successfully proxy all requests for an application, the Oracle Proxy Plug-In must have a complete description of the URL space for that application. Each Oracle Fusion Middleware 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, then the request will be served by the third-party HTTP listener, and a "document not found" error may occur, or a different 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 the following:

- 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 directory structure. For performance reasons, it is a good idea to have the third-party listener handle static `.gif` and `.jpg` files, but it requires that the files be copied to the third-party listener.

## A.1.9 Troubleshooting Oracle Proxy Plug-In

This section describes common problems and solutions related to Oracle Proxy Plug-In.

### Listener Fails to Start

- Check for problems in the proxy server definition file. Each server in the server list line must be defined later in the file, and you must define at least one server. If a server name is listed but not defined, then the listener may not start. Ensure that there are no typographical errors or missing quotes in the proxy server definition file.
- For Sun Java System Web Server 6.1 or 6.0: Ensure that `Init` lines are added to the `magnus.conf` file, and the `ObjectType` and `Service` lines are added to the `obj.conf` file.

**Listener Returns Incorrect URLs**

- Verify that changes to the proxy server definition file were saved and the listener was restarted.
- Ensure that there are no typographical errors in the proxy server definition file.
- Ensure that the `urlrule` parameter is set up correctly, and determine 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, and the back-end server logs to verify that requests are getting through. The proxy log may need to be set to debug mode. You may need to restart the listener.

**Changes Made to Proxy Server Definition File are Not Reflected**

- Verify that changes to the proxy server definition file were saved and the listener was restarted.
- For Microsoft IIS, verify that WWW Publishing Service was stopped and started from the Control Panel. This may take a few minutes.

**Microsoft IIS Listener Displays Incomplete Pages or Garbled Characters**

Do not display Microsoft IIS pages with a Sun Java System browser.

**Parsing Error Occurs with Sun Java System Web Server 6.1 or 7.0**

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:** [Section A.1.3.2.6, "oproxy.servername.urlrule"](#)

**Sun Java System Web Server Returns Server Error with /servlet Request**

The default Sun Java System Web Server configuration maps any URL requests to `/servlet` to its own servlet handler. You must edit the proxy server definition file, or change the Sun Java System Web Server configuration to correct this.

**Partial URL Requests Return Unexpected Results**

The Microsoft IIS and Sun Java System servers auto-complete URLs differently than others. Requests to `http://example`, `http://example/`, and



`http://example/index.html` do not necessarily return the same results on different platforms. Use the `oproxy.servername.urlrule` parameter to work around this problem.

**See Also:** [Section A.1.3.2.6, "oproxy.servername.urlrule"](#)

### Server Returns Page with Broken Image Links

If you use an exact `urlrule` parameter in the proxy server definition file, such as `urlrule=/*.html`, the server retrieves the specified page. All other links are forbidden to the user, including inline images on 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 Java System and Microsoft Internet Explorer use cached pages even when set to retrieve the page every time and 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 are not expecting, try refreshing the browser or reloading the page.

### REMOTE\_ADDR Contains Unexpected IP Address

The `REMOTE_ADDR` field usually contains the IP address of the client system. 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, the host and port information should be updated. Choose one of the following options, based on your installation. The first option is the preferred method.

- Use the Advanced Server Configuration page of Fusion Middleware Control to set the following directives in the `httpd.conf` file:

```
UseCanonicalName On
ServerName name of listener host
Port port of listener host
```

- Use the Advanced Server Configuration page of Fusion Middleware Control to set the following directives in the `httpd.conf` file:

```
UseCanonicalName port
Port port of listener host
```

Edit the proxy plug-in server definition file to use the following:

```
oproxy.serverName.alias=name of listener host:port of listener host
```

## A.2 Using Oracle SSO Plug-In

---

---

**Critical Note:** The FMW 10g-based Proxy Plug-In known as the OProxy Plug-In for Oracle HTTP Server 11g and other Web Servers, such as IIS has been deprecated and will be discontinued in future release. This legacy proxy solution is designed to front-end applications deployed on Oracle Application Server 10g. Oracle Fusion Middleware 11g Applications are now deployed on WebLogic. Oracle HTTP Server 11g includes a built-in proxy module optimized to front-end this environment.

If you are using FMW 10g-based Proxy Plug-In, you should migrate to Oracle HTTP Server 11g based proxy solutions for a well integrated deployment or leverage the built-in proxy modules within other Web Servers

---

---

Oracle SSO Plug-In is designed to protect native third-party listener applications using the Oracle single sign-on (SSO) infrastructure. The Oracle SSO Plug-In is provided and certified to work with Microsoft Internet Information Server (IIS) v6.0 and v7.0 on Microsoft Windows systems.

**See Also:**

[http://www.oracle.com/technology/software/products/iis/files/fusion\\_certification.html](http://www.oracle.com/technology/software/products/iis/files/fusion_certification.html) for complete certification information

This chapter includes the following topics:

- [Section A.2.1, "Overview of Oracle SSO Plug-In."](#)
- [Section A.2.2, "Installing Oracle SSO Plug-In."](#)
- [Section A.2.3, "Registering with the Oracle Single Sign-On Server."](#)
- [Section A.2.4, "Configuring the Oracle SSO Plug-In."](#)
- [Section A.2.4.2, "Rules to Protect Resources."](#)
- [Section A.2.5, "Configuring Microsoft IIS 6.0 Listener to Use Oracle Single Sign-On."](#)
- [Section A.2.6, "Configuring Microsoft IIS 7.0 Listener to Use Oracle Single Sign-On."](#)
- [Section A.2.7, "Troubleshooting Oracle SSO Plug-In."](#)

### A.2.1 Overview of Oracle SSO Plug-In

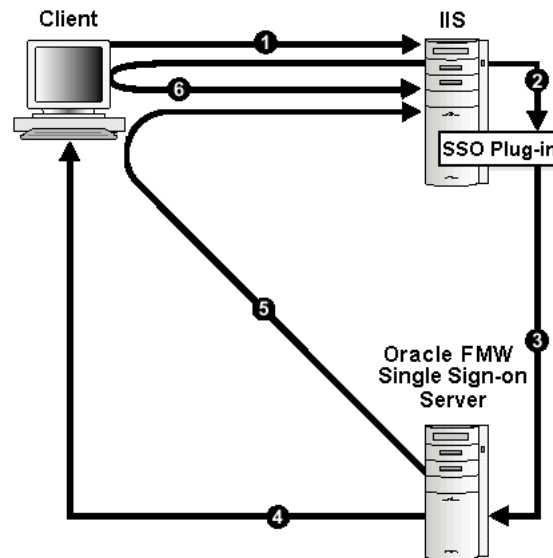
Oracle SSO Plug-In is the Oracle single sign-on (SSO) solution for Microsoft IIS. The **plug-in** is designed to protect native third-party listener applications using the SSO infrastructure. With the help of the Oracle SSO Plug-In, users can be authenticated to different third-party listener applications 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 applications as long as they are all protected on the same SSO server.

Oracle 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, global timeout, and basic authentication for legacy application are not implemented in the current Oracle SSO Plug-In release.

Figure A-2 illustrates the process when a user requests a URL protected by the Oracle SSO Plug-In.

Figure A-2 Oracle SSO Plug-In



1. The user requests a URL through a Web browser.
2. The Web server looks for an Oracle 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, the Oracle 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 a cookie exists, then the single sign-on server authenticates using the cookie. If authentication is successful, then the single sign-on server creates a cookie in the browser as a reminder that the user has been authenticated. If it finds none, it tries to authenticate the user with a user name and password.
5. The single sign-on server returns the user's encrypted information to the Oracle SSO Plug-In.
6. Oracle 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 Oracle SSO Plug-In session cookie.

**See Also:** *Oracle Fusion Middleware Security Guide*

## A.2.2 Installing Oracle SSO Plug-In

The Oracle SSO Plug-In for Microsoft IIS is available in the Oracle Fusion Middleware Web Tier and Utilities installer (11.1.1.2 and later *full* installers, not on patch-set installers).

Location: \Disk1\plugins\iis\oracle\_osso.dll

To install the plug-in for the listener, copy `oracle_osso.dll` to a directory the listener can access. For security reasons, ensure that the plug-in library is given minimum privileges.

Install Oracle SSO Plug-In on a system that has an Oracle Fusion Middleware installation. This installation is required only for the network and security dependent libraries and the single sign-on registration tool. It is not required to be running.

### A.2.3 Registering with the Oracle Single Sign-On Server

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 `osso_conf` is created. The `osso_conf` file is obfuscated for security purposes. Copy the file to an appropriate location so the listener can access it.

**See Also:** *Oracle Fusion Middleware Security Guide* for details on how to register with Oracle Single Sign-On.

### A.2.4 Configuring the Oracle SSO Plug-In

To configure Oracle SSO Plug-In, you must create a configuration file such as the `osso_plugin.conf` file. This file must reside in a directory that is readable by the third-party listener. You define all the plug-in functionality within the file. It can also be referred as the `osso` property file. The file contains the following:

- Plug-in directives such as `LoginServerFile` and `IpCheck`
- A set of rules that match resources to be protected.

#### A.2.4.1 Oracle SSO Plug-In Directives

[Table A-2](#) lists the configuration directives for the Oracle SSO Plug-In.

**Table A-2 Oracle SSO Plug-In Configuration Directives**

Directive	Function
<code>LoginServerFile</code>	<p>Specifies the location of the single sign-on server configuration file such as the <code>osso.conf</code> file that is attained from the SSO registration process.</p> <p>This is a global parameter and should not be used on a per-resource basis. You must provide one and only one single sign-on server configuration file.</p> <ul style="list-style-type: none"> <li>■ Value: The full path of your Single Sign-On Server configuration file</li> <li>■ Default: None</li> <li>■ Example: <code>LoginServerFile=c:\OSSO\config\osso.conf</code></li> </ul>
<code>IpCheck</code>	<p>Specifies whether the Oracle SSO plug-in should check the IP address of each request when it examines the cookie. Setting it to <code>true</code> prevents cookies from being accessed by another person.</p> <ul style="list-style-type: none"> <li>■ Values: <code>true</code>   <code>false</code></li> <li>■ Default: <code>false</code></li> <li>■ Example: <code>IpCheck=true</code></li> </ul> <p><b>Note:</b> Set it to <code>false</code> if you have a proxy server or firewall between your Sun Java System server and your client browsers.</p>

### A.2.4.2 Rules to Protect Resources

To ensure resource protection via the Oracle SSO Plug-In, a set of rules must be defined. The rules are defined according to the following format:

```
<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: <code>/examples/hello.html</code>
Context Match	This option identifies a directory as a protected resource, for example: <code>/examples/*</code>
Extension Match	This option identifies files with a certain extension in a particular directory as a protected resource, for example: <code>/examples/*.jsp</code>

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 A–2** shows a simple file with the commands for resource protection. In the example, the `IpCheck` directive is set to `false` for the `/private/hello.html` file, but it is set to `true` for `/private2/*.jsp`. This setting ensures the cookies used with requests to the `/private2/*.jsp` files are not accessed by another user.

#### **Example A–2 Simple Single Sign-on Configuration File, `osso_plugin.conf`**

```
LoginServerFile=c:\OSSO\conf\osso.conf
<OSSO /private/hello.html>
  IpCheck=false
</OSSO>
<OSSO /private1/*>
</OSSO>
<OSSO /private2/*.jsp>
  IpCheck=true
</OSSO>
```

## A.2.5 Configuring Microsoft IIS 6.0 Listener to Use Oracle Single Sign-On

This section provides instructions on configuring the Microsoft IIS 6.0 Listener to use Oracle SSO Plug-In. The plug-in consists of a single `.dll` file, `oracle_osso.dll`. To install the plug-in, copy the `.dll` to the host on which Microsoft IIS 6.0 resides and perform the following steps:

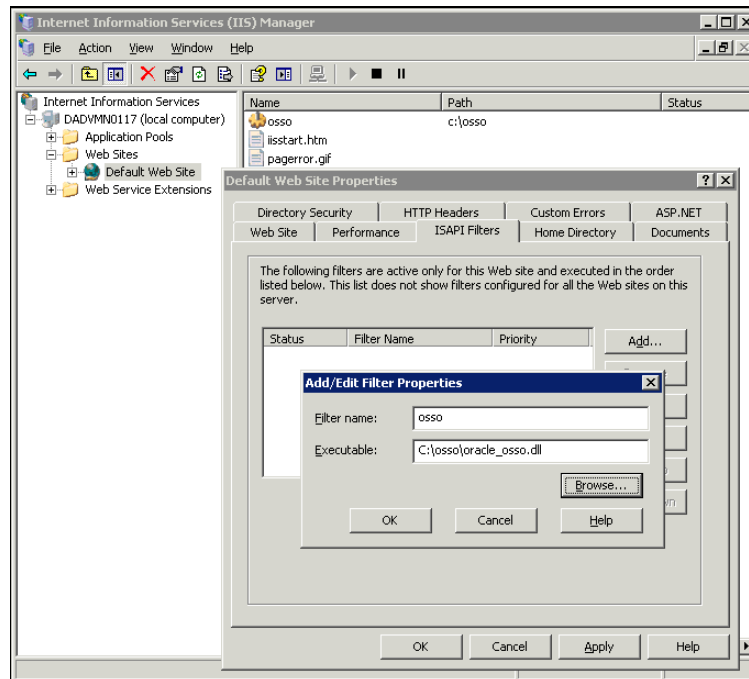
1. Use the Microsoft Windows Registry Editor to create new registry entries.

- a. From the Start menu, Select **Run**, and the type `regedit` in the dialog box, and click **OK**. The Registry Editor window opens
  - b. Expand the `HKEY_LOCAL_MACHINE` folder by clicking the plus sign (+) preceding its name In the Registry Editor window.
  - c. Expand the `SOFTWARE` folder by clicking the plus sign (+) preceding its name, and then Click the **ORACLE** folder.
  - d. From the Edit menu, select **New > Key**. A new folder is added under the `ORACLE` folder with the name `New Key #1`.
  - e. Enter `IIS_OSSO_Adapter` for the key name.
  - f. From the Edit menu, select **New > String Value**. A new value is added in the right window pane with the name `New Value #1`. Enter `cfg_file` for the value name.
  - g. From the Edit menu, select **Modify**. The Edit String dialog box appears.
  - h. In the Value data field, enter the full path of the OSSO plug-in configuration file you created (e.g., `c:\osso\osso_plugin.conf`).
- 
- Note:** This is the plug-in configuration file and not the encrypted `osso.conf` file generated by the SSO registration process.
- 
- i. Optionally, you can specify `log_file` and `log_level` using the procedure specified in steps **d.)** through **h.)**.
    - Add a string value with the name `log_file` and the desired location of the log file, such as `c:\osss\osso_plugin.log`.
    - 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`.
  - j. Close the Registry Editor window by selecting **Exit** from the File menu.
2. Use the Microsoft IIS management console to add a new virtual directory to your Microsoft IIS Web site with the same physical path as that of the `osso.dll`.
    - a. Open the IIS Manager by clicking **Start -> Programs -> Administrative Tools -> Internet Information Services (IIS) Manager**.
    - b. Expand the server folder by clicking the plus sign (+) preceding the server name.
    - c. Right-click the **Default Sites** folder, and then select the **New -> Virtual Directory** option from the menu.
    - d. In the Virtual Directory Creation Wizard window, enter `osso` for the Alias. Then, enter the path or browse to the directory containing the `oracle_osso.dll` file (e.g., `c:\osso`) and select the **Execute (such as ISAPI applications or CGI)** check box.
    - e. Click **Finish** to close the Virtual Directory Creation Wizard.
  3. Use the Microsoft IIS management console to add `oracle_osso.dll` as a filter in your Microsoft IIS Web site. The name of the filter should be `osso` and its executable must point to the directory containing the `oracle_osso.dll` file.
    - a. Right-click the **Default Sites** folder, and then select the **Properties** option from the menu.

- b. In the Default Web Site Properties window, select the **ISAPI Filters** tab.
- c. Click **Add** to add a new filter.

The Add/Edit Filter Properties window is displayed.

- d. In the Filter Name field, enter **osso** for the filter name.
- e. In the Executable field, enter the path for the location containing the `oracle_osso.dll` (e.g., `c:\osso\oracle_osso.dll`), or click the **ellipsis** button (...) to navigate to the folder that contains the `oracle_osso.dll` file.



- f. Click **OK** to close the Add/Edit Filter Properties window.
- g. Click **OK** to close the Default Web Site Properties window.
4. Configure security settings for Oracle Home directory. Make sure you login to machine as an administrator user.
  - a. In Windows Explorer, right-click the `ORACLE_HOME\bin` folder, select **Properties** from the menu, and then click the **Security** tab.
  - b. Add the `IIS_WPG`, `NETWORK` and `NETWORK SERVICE` groups with Read and Execute permissions.
  - c. Click **OK**.
5. Stop and restart the Microsoft IIS 6.0 Server.

---

**Notes:**

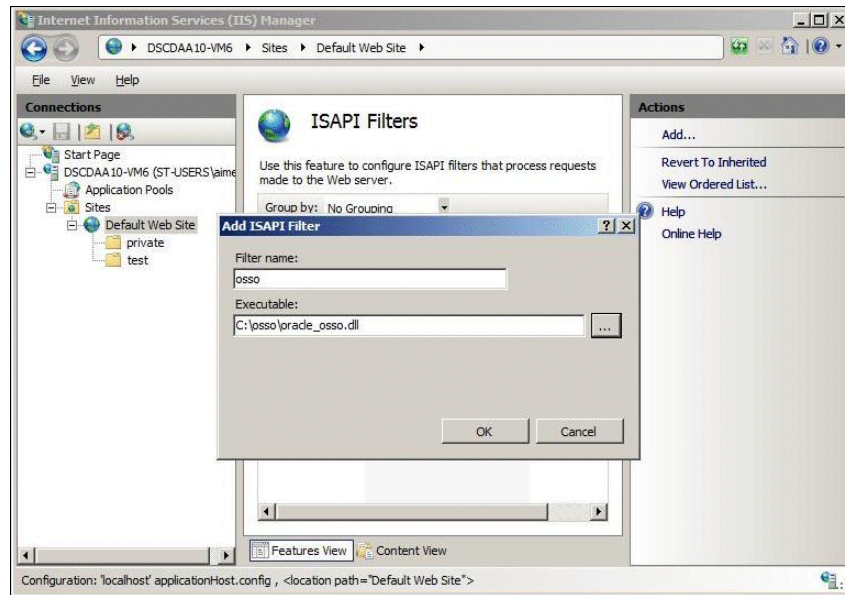
- To restart Microsoft IIS 6.0, you must stop all the Microsoft IIS 6.0 services through the control panel or restart the computer. This is the only way to ensure that the .dll file is reloaded. Restarting Microsoft IIS 6.0 through the management console is not sufficient.
  - If you want multiple Oracle installations on the same home, then the ORACLE\_HOME\bin PATH entry for the installation that you plan to use in conjunction with the Oracle SSO Plug-In must appear first in your PATH.
  - Make sure the newly added ISAPI filter is marked with a green upward arrow.
- 

## A.2.6 Configuring Microsoft IIS 7.0 Listener to Use Oracle Single Sign-On

This section provides instructions on configuring the Microsoft IIS 7.0 Listener to use Oracle SSO Plug-In. The plug-in consists of a single .dll file, `oracle_osso.dll`. To install the plug-in, copy the .dll to the host on which Microsoft IIS 7.0 resides and perform the following steps:

1. Complete step 1 in [Section A.2.5, "Configuring Microsoft IIS 6.0 Listener to Use Oracle Single Sign-On"](#) to use the Microsoft Windows Registry Editor to create new registry entries for the plug-in.
2. Use the Microsoft IIS management console to add the `oracle_osso.dll` as a filter in your Microsoft IIS Web site:
  - a. Open the IIS Manager by clicking **Start -> Programs -> Administrative Tools -> Internet Information Services (IIS) Manager**.
  - b. Expand the server folder by clicking the plus sign (+) preceding the server name (e.g, DSCDAA10-VM6).
  - c. Expand the Sites folder by clicking the plus sign (+) preceding its name.
  - d. Click the **Default Web Site** icon to open the Default Web Site Home page.
  - e. Double-click the **ISAPI Filters** icon to open the ISAPI Filters page, and then complete the following tasks:
    - In the Actions pane, click **Add** to open the Add ISAPI Filter dialog box.
    - In the Filter Name field, enter `osso`.
    - In the Executable field, enter the file system path for the location containing the `oracle_osso.dll` (e.g., `c:\osso\oracle_osso.dll`), or click the **ellipsis** button (...) to navigate to the folder that contains the `oracle_osso.dll` file.





- Click **OK**.
- 3. Configure security settings for Oracle Home directory. Make sure you log in to machine as an administrator user.
  - a. In Windows Explorer, right-click the `ORACLE_HOME\bin` folder, select **Properties** from the menu, and then click the **Security** tab.
  - b. Add the `IIS_WPG`, `NETWORK` and `NETWORK SERVICE` groups with Read and Execute permissions.
  - c. Click **OK**.
- 4. Restart the Microsoft IIS server by opening the Services Control Panel, and then stopping and restarting the World Wide Web Publishing Service.

---



---

#### Notes:

- To restart Microsoft IIS 7.0, you must stop all the Microsoft IIS 7.0 services through the Services Control Panel or restart the computer. This is the only way to ensure that the `.dll` file is reloaded. Restarting Microsoft IIS 7.0 through the management console is not sufficient.
  - If you want multiple Oracle installations on the same home, then the `ORACLE_HOME\bin` `PATH` entry for the installation that you plan to use in conjunction with the Oracle SSO Plug-In must appear first in your `PATH`.
- 
- 

## A.2.7 Troubleshooting Oracle SSO Plug-In

This section describes common problems and solutions.

### Oracle Dependency Libraries Not Found

You may not have included `ORACLE_HOME` in your path.

### **Solution**

Check to see that you have `ORACLE_HOME/lib` included in your library path variable on UNIX. On Microsoft Windows, ensure that you have `ORACLE_HOME\bin` in your path.

If you continue to receive this message in your `osso.log` file, then verify that all configuration files are properly configured, as described in [Section A.2.4, "Configuring the Oracle SSO Plug-In"](#).

### **Microsoft IIS Oracle SSO Plug-In Does not Work with HTML Authentication**

The Oracle SSO Plug-In is designed not to work with other authentication modules. Authentication is either a native listener authentication module or a third-party module.

---

---

## Frequently Asked Questions

This appendix provides answers to frequently asked questions about Oracle HTTP Server. It includes the following topics:

- [Section B.1, "How Do I Create Application-Specific Error Pages?"](#)
- [Section B.2, "What Type of Virtual Hosts Are Supported for HTTP and HTTPS?"](#)
- [Section B.3, "Can I Use Oracle HTTP Server As Cache?"](#)
- [Section B.4, "Can I Use Different Language and Character Set Versions of Document?"](#)
- [Section B.5, "Can I Apply Apache Security Patches to Oracle HTTP Server?"](#)
- [Section B.6, "Can I Upgrade the Apache Version of Oracle HTTP Server?"](#)
- [Section B.7, "Can I Compress Output From Oracle HTTP Server?"](#)
- [Section B.8, "How Do I Create a Namespace That Works Through Firewalls and Clusters?"](#)
- [Section B.9, "How do I Protect the Web Site from Hackers?"](#)
- [Section B.10, "Do I Need to Re-register Partner Applications with the SSO Server If I Disable or Enable SSL?"](#)
- [Section B.11, "Why REDIRECT\\_ERROR\\_NOTES is not set for file-not-found errors?"](#)
- [Section B.12, "How can I hide information about the Web Server Vendor and Version"](#)

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.

---

---

### B.1 How Do I Create 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 HTTP Server documentation

## B.2 What Type of Virtual Hosts Are Supported for HTTP and HTTPS?

For HTTP, Oracle HTTP Server supports both name-based and IP-based virtual hosts. Name-based virtual hosts are virtual hosts that share a common listening address (IP plus port combination), but route requests based on a match between the Host header sent by the client and the `ServerName` directive set within the `VirtualHost`. IP-based virtual hosts are virtual hosts that have distinct listening addresses. IP-based virtual hosts route requests based on the address they were received on.

For HTTPS, only IP-based virtual hosts are possible with Oracle HTTP Server. This is because for name-based virtual hosts, the request must be read and inspected to determine which virtual host is used to process the request. If HTTPS is used, an SSL handshake must be performed before the request can be read. In order to perform the SSL handshake, a server certificate must be provided. In order to have a meaningful server certificate, the host name in the certificate must match the host name the client requested, which implies a unique server certificate per virtual host. However, because the server cannot know which virtual host to route the request to until it has read the request, and it can't properly read the request unless it knows which server certificate to provide, there is no way to make name-based virtual hosting work with HTTPS.

**See Also:** Apache Virtual Host documentation at:  
<http://httpd.apache.org/docs/2.2/vhosts/>

## B.3 Can I Use Oracle HTTP Server As Cache?

Oracle recommends using Oracle Web Cache instead. Oracle Web Cache is a content-aware server accelerator and secure reverse proxy server that improves the performance, scalability, and availability of Web sites. For more details, refer to the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*.

## B.4 Can I Use Different Language and Character Set Versions of Document?

Yes, 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 HTTP Server documentation

## B.5 Can I Apply Apache Security Patches to Oracle HTTP Server?

No, you cannot apply the Apache HTTP Server 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, the Apache HTTP Server alerts, such as OpenSSL alerts, may not be applicable because Oracle has removed those components from the stack.

The latest security related fixes to Oracle HTTP Server are performed through the Oracle Critical Patch Update (CPU). For more details, refer to Oracle's Critical Patch Updates and Security Alerts Web page.

---



---

**Note:** After applying a CPU, the Apache HTTP Server-based version may stay the same, but the vulnerability will be fixed. There are third-party security detection tools that can check the version, but do not check the vulnerability itself.

---



---

## B.6 Can I Upgrade the Apache Version of Oracle HTTP Server?

No, you cannot upgrade only the Apache HTTP Server version inside Oracle HTTP Server. Oracle provides a newer version of Apache HTTP Server that Oracle HTTP Server is based on, which is part of either a patch update or the next major or minor release of Oracle Fusion Middleware.

## B.7 Can I Compress Output From Oracle HTTP Server?

In general, Oracle recommends using Oracle Web Cache for this purpose. Oracle Web Cache provides efficient delivery of contents by using on-the-fly compression, dynamically learning which MIME types are compressible, and throttling responses to slower network clients. Another compression solution is `mod_deflate`, which is included with Oracle HTTP Server. For more information pertaining to `mod_deflate` module, see [http://httpd.apache.org/docs/2.2/mod/mod\\_deflate.html](http://httpd.apache.org/docs/2.2/mod/mod_deflate.html)

For additional information, refer to the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*.

## B.8 How Do I Create a Namespace That Works Through Firewalls and Clusters?

The general idea is that all servers in a distributed Web site should use 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
```

You could initially map these name spaces to two Web servers by putting `app1` on `server1` and `app2` on `server2`. The configuration for `server1` might look like the following:

```
Redirect permanent /app2 http://server2/app2
Alias /app1 /myApps/application1
<Directory /myApps/application1>
    ...
</Directory>
```

The configuration for `Server2` is complementary.

If you decide to partition the namespace by content type (HTML on `server1`, and JSP on `server2`), then you can change server configuration and move files around, but you 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>
```

The amount of actual redirection can be minimized by configuring a hardware load balancer like F5 system BIG-IP to send requests to server1 or server2 based on the URL.

## B.9 How do I Protect the Web Site from Hackers?

There are many attacks by hackers, 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. Remember not all hackers are outside your organization.
- Use switched Ethernet to limit the amount of traffic a compromised server can detect. Use additional firewalls between Web server machines and highly sensitive internal servers running the database and enterprise applications.
- Remove unnecessary network services such as RPC, Finger, and telnet from your server.
- 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 to prevent a hacker from hijacking a valid session. For example, it should be difficult to guess a valid sessionID.
- 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 reliable sources. Only download patches 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 into your site. 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.

## B.10 Do I Need to Re-register Partner Applications with the SSO Server If I Disable or Enable SSL?

Yes, if you enable or disable SSL, you have to re-register partner applications with the SSO server. When you make any changes that affect the URL (for example, changing the host name or port, or enabling or disabling SSL), you have re-register partner applications with the SSO server because the old URL registered with the SSO server is no longer valid. You have to re-register the partner applications with the new URL.

## B.11 Why REDIRECT\_ERROR\_NOTES is not set for file-not-found errors?

The REDIRECT\_ERROR\_NOTES CGI environment variable is not set for file not found errors in Oracle HTTP Server 11g because the Apache HTTP Server 2.0 and

above do not make that information available to CGI and other applications for this condition.

## B.12 How can I hide information about the Web Server Vendor and Version

Specify "ServerSignature Off" to remove this information from web server generated responses. Specify "ServerTokens Custom some-server-string" to disguise the web server software when Oracle HTTP Server generates the web Server response header. (When a backend server generates the response, the Server response header may come from the backend server depending on the proxy mechanism.)

---

---

**Note:** `ServerTokens Custom some-server-string` is a replacement for the `ServerHeader Off` setting in Oracle HTTP Server 10g.

---

---





---

---

## 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 includes the following topics:

- [Section C.1, "Oracle HTTP Server Unable to Start Due to Port Conflict"](#)
- [Section C.2, "System Overloaded by Number of httpd Processes"](#)
- [Section C.3, "Permission Denied When Starting Oracle HTTP Server On a Port Below 1024"](#)
- [Section C.4, "Oracle HTTP Server May Fail To Start If PM Files Are Not Located Correctly"](#)
- [Section C.5, "Unsetting PerSetEnv and Removing a Previously Configured PerSetEnv variable in EM Throws an Exception"](#)
- [Section C.6, "Using Log Files to Locate Errors"](#)
- [Section C.7, "Client IP Address Not Used in a Configuration with Oracle Web Cache"](#)

### C.1 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:

```
[VirtualHost: main] (98)Address already in use: make_sock: could not bind to address [::]:7777
```

#### Solution

Determine what process is already using that port, and then either change the IP:port address of Oracle HTTP Server or the port of the conflicting process.

### C.2 System Overloaded by Number of httpd Processes

When too many httpd processes are running on a system, the response time degrades because there are insufficient resources for normal processing.

#### Solution

Lower the value of `MaxClients` to a value the machine can accommodate.

## C.3 Permission Denied When Starting Oracle HTTP Server On a Port Below 1024

You will get the following error if you try to start Oracle HTTP Server on a port below 1024:

```
[VirtualHost: main] (13)Permission denied: make_sock: could not bind to address [::]:443
```

Oracle HTTP Server will not start on ports below 1024 because root privileges are needed to bind these ports.

### Solution

Follow the steps in [Section 4.1.2.3, "Starting Oracle HTTP Server on a Privileged Port"](#) to start Oracle HTTP Server on a Privileged Port.

## C.4 Oracle HTTP Server May Fail To Start If PM Files Are Not Located Correctly

If Oracle HTTP Server is not able to locate Perl module (PM) files in the path defined in the `PERL5LIB` variable, Oracle HTTP Server may encounter the following errors, 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/perl/...)
```

### Solution

Check that `ORACLE_HOME/ohs/bin/apachectl` is correctly defined in the `PERL5LIB` variable. It should point to the path(s) containing the PM files. By default, it points to PM files in the following directories:

```
ORACLE_HOME/ohs/mod_perl/lib/site_perl/5.10.0
ORACLE_HOME/perl/lib/5.10.0
ORACLE_HOME/perl/lib/site_perl/5.10.0
```

## C.5 Unsetting PerSetEnv and Removing a Previously Configured PerSetEnv variable in EM Throws an Exception

If you configure `mod_perl` by using the EM `mod_perl` configuration page and try to remove a previously configured `PerSetEnv` variable from the Environment field, this error is thrown:

```
Failed to invoke operation save on MBean
oracle.as.management.mbeans.register:type=component,name=ohs1,instance=webtier

_inst7971,Location=AdminServer
Apply failed, modify required parameters and save again. Validation of
configuration trying to apply failed
.
.
.
```

**Solution**

To correct this situation:

1. Close the pop-up error and click **Revert**.
2. Remove the PerSetEnv by doing one of the following:
  - Go to the Advanced Configuration page of EM and modify the mod\_perl.conf file directly.  
OR
  - Go to the instance home/config/OHS/component\_name/moduleconf/mod\_perl.conf and edit the configuration file directly to remove the PerSetEnv value.
3. Restart OHS.

## C.6 Using Log Files to Locate Errors

You can use the following log files to help locate errors:

- [Rewrite Log](#)
- [Script Log](#)
- [Error Log](#)

### C.6.1 Rewrite Log

This log file is necessary for debugging when mod\_rewrite is used. The 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 HTTP Server documentation.

### C.6.2 Script Log

This log file enables you to record the input to and output from the CGI scripts. This should only be used in testing, and not for production servers.

**See Also:** Script Log in the Apache HTTP Server documentation.

### C.6.3 Error Log

This log file records overall server problems. Refer to [Chapter 7, "Managing Oracle HTTP Server Logs"](#) for details on configuring and viewing error logs.

## C.7 Client IP Address Not Used in a Configuration with Oracle Web Cache

The UseWebCacheIp directive allows Oracle HTTP Server to use the Client IP address for logging and mod\_authz\_host access control when the client connects to Oracle HTTP Server through Oracle Web Cache. This feature may be usable with other front-end proxy servers, if the proxy sets the ClientIP request header to the Client IP address. When UseWebCacheIp is not enabled and a client connects to Oracle HTTP

Server through Oracle Web Cache or other proxy, the client address used for logging and `mod_authz_host` access control will be that of Oracle Web Cache or other proxy.

**Solution**

Set `UseWebCacheIp` to `ON` in your `httpd.conf` file.

Also, if you do not set `UseWebCacheIp` to `ON`, the address of the host connecting to Oracle HTTP Server will be used for logging and host-based access control. In some cases this will be a proxy instead of the client.

---

---

**Note:** The `UseWebCacheIp` directive is not available in Oracle HTTP Server versions 11.1.1.4.0 and 11.1.1.5.0.

---

---

---

---

## Configuring mod\_security

This appendix contains a usable example ([Example D-1](#)) of the `mod_security.conf` file, including the `loadModule` statement.

By default, this file does not exist, so you need to create it, preferably by using the template here. Copy and paste the sample into a text editor and read the entire file, editing it for your system. Then save it as your own `mod_security.conf` and include it from your `httpd.conf`. If you implement `mod_security.conf` as described in this appendix, it will use the `LoadModule` directive to load `mod_security2.so` into the run time environment.

For more information on `mod_security`, see [Section 3.11, "mod\\_security"](#).

### **Example D-1** *mod\_security.conf* Sample

```
#Load module
LoadModule security2_module "${ORACLE_HOME}/ohs/modules/mod_security2.so"
# -- Rule engine initialization -----

# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-installation
# disruption.
#
SecRuleEngine DetectionOnly

# -- Request body handling -----

# Allow ModSecurity to access request bodies. If you don't, ModSecurity
# won't be able to see any POST parameters, which opens a large security
# hole for attackers to exploit.
#
SecRequestBodyAccess On

# Enable XML request body parser.
# Initiate XML Processor in case of xml content-type
#
SecRule REQUEST_HEADERS:Content-Type "text/xml" \
    "id:'200000',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=XML"

# Maximum request body size we will accept for buffering. If you support
# file uploads then the value given on the first line has to be as large
# as the largest file you are willing to accept. The second value refers
# to the size of data, with files excluded. You want to keep that value as
# low as practical.
#
```

---

```

SecRequestBodyLimit 13107200
SecRequestBodyNoFilesLimit 131072
# Store up to 128 KB of request body data in memory. When the multipart
# parser reaches this limit, it will start using your hard disk for
# storage. That is slow, but unavoidable.
#
SecRequestBodyInMemoryLimit 131072

# What do do if the request body size is above our configured limit.
# Keep in mind that this setting will automatically be set to ProcessPartial
# when SecRuleEngine is set to DetectionOnly mode in order to minimize
# disruptions when initially deploying ModSecurity.
#
SecRequestBodyLimitAction Reject

# Verify that we've correctly processed the request body.
# As a rule of thumb, when failing to process a request body
# you should reject the request (when deployed in blocking mode)
# or log a high-severity alert (when deployed in detection-only mode).
#
SecRule REQBODY_ERROR "!@eq 0" \
  "id:'200001', phase:2,t:none,log,deny,status:400,msg:'Failed to parse request
  body.',logdata:'{%reqbody_error_msg}',severity:2"

# By default be strict with what we accept in the multipart/form-data
# request body. If the rule below proves to be too strict for your
# environment consider changing it to detection-only. You are encouraged
# _not_ to remove it altogether.
#
SecRule MULTIPART_STRICT_ERROR "!@eq 0" \
  "id:'200002',phase:2,t:none,log,deny,status:44, \
  msg:'Multipart request body failed strict validation: \
  PE {%REQBODY_PROCESSOR_ERROR}, \
  BQ {%MULTIPART_BOUNDARY_QUOTED}, \
  BW {%MULTIPART_BOUNDARY_WHITESPACE}, \
  DB {%MULTIPART_DATA_BEFORE}, \
  DA {%MULTIPART_DATA_AFTER}, \
  HF {%MULTIPART_HEADER_FOLDING}, \
  LF {%MULTIPART_LF_LINE}, \
  SM {%MULTIPART_MISSING_SEMICOLON}, \
  IQ {%MULTIPART_INVALID_QUOTING}, \
  IP {%MULTIPART_INVALID_PART}, \
  IH {%MULTIPART_INVALID_HEADER_FOLDING}, \
  FL {%MULTIPART_FILE_LIMIT_EXCEEDED}"

# Did we see anything that might be a boundary?
#
SecRule MULTIPART_UNMATCHED_BOUNDARY "!@eq 0" \
  "id:'200003',phase:2,t:none,log,deny,status:44,msg:'Multipart parser detected a possible unmatched
  boundary.'"

# PCRE Tuning
# We want to avoid a potential RegEx DoS condition
#
SecPcreMatchLimit 1000
SecPcreMatchLimitRecursion 1000

# Some internal errors will set flags in TX and we will need to look for these.
# All of these are prefixed with "MSC_". The following flags currently exist:
#

```

---

```
# MSC_PCRE_LIMITS_EXCEEDED: PCRE match limits were exceeded.
#
SecRule TX:/^MSC_/ "!@streq 0" \
    "id:'200004',phase:2,t:none,deny,msg:'ModSecurity internal error flagged: %{MATCHED_VAR_
NAME}'"

# -- Response body handling -----

# Allow ModSecurity to access response bodies.
# You should have this directive enabled in order to identify errors
# and data leakage issues.
#
# Do keep in mind that enabling this directive does increases both
# memory consumption and response latency.
#
SecResponseBodyAccess On

# Which response MIME types do you want to inspect? You should adjust the
# configuration below to catch documents but avoid static files
# (e.g., images and archives).
#
SecResponseBodyMimeType text/plain text/html text/xml

# Buffer response bodies of up to 512 KB in length.
SecResponseBodyLimit 524288

# What happens when we encounter a response body larger than the configured
# limit? By default, we process what we have and let the rest through.
# That's somewhat less secure, but does not break any legitimate pages.
#
SecResponseBodyLimitAction ProcessPartial

# -- Filesystem configuration -----

# The location where ModSecurity stores temporary files (for example, when
# it needs to handle a file upload that is larger than the configured limit).
#
# This default setting is chosen due to all systems have /tmp available however,
# this is less than ideal. It is recommended that you specify a location that's private.
#
SecTmpDir /tmp/

# The location where ModSecurity will keep its persistent data. This default setting
# is chosen due to all systems have /tmp available however, it
# too should be updated to a place that other users can't access.
#
SecDataDir /tmp/

# -- File uploads handling configuration -----

# The location where ModSecurity stores intercepted uploaded files. This
# location must be private to ModSecurity. You don't want other users on
# the server to access the files, do you?
#
#SecUploadDir /opt/modsecurity/var/upload/

# By default, only keep the files that were determined to be unusual
# in some way (by an external inspection script). For this to work you
# will also need at least one file inspection rule.
```

---

```

#
#SecUploadKeepFiles RelevantOnly
# Uploaded files are by default created with permissions that do not allow
# any other user to access them. You may need to relax that if you want to
# interface ModSecurity to an external program (e.g., an anti-virus).
#
#SecUploadFileMode 0600

# -- Debug log configuration -----
# The default debug log configuration is to duplicate the error, warning
# and notice messages from the error log.
#
#SecDebugLog /opt/modsecurity/var/log/debug.log
#SecDebugLogLevel 3

# -- Audit log configuration -----
# Log the transactions that are marked by a rule, as well as those that
# trigger a server error (determined by a 5xx or 4xx, excluding 404,
# level response status codes).
#
SecAuditEngine RelevantOnly
SecAuditLogRelevantStatus "^(?:5|4?!04)"

# Log everything we know about a transaction.
SecAuditLogParts ABIJDEFHZ

# Use a single file for logging. This is much easier to look at, but
# assumes that you will use the audit log only occasionally.
#
SecAuditLogType Serial
SecAuditLog "${ORACLE_INSTANCE}/diagnostics/logs/${COMPONENT_TYPE}/${COMPONENT_NAME}/modsec_
audit.log"

# Specify the path for concurrent audit logging.
SecAuditLogStorageDir "${ORACLE_INSTANCE}/diagnostics/logs/${COMPONENT_TYPE}/${COMPONENT_NAME}"
#Simple test
SecRule ARGS "\.\\.\/" "t:normalisePathWin,id:99999,severity:4,msg:'Drive Access'"

# -- Miscellaneous -----
# Use the most commonly used application/x-www-form-urlencoded parameter
# separator. There's probably only one application somewhere that uses
# something else so don't expect to change this value.
#
SecArgumentSeparator &

# Settle on version 0 (zero) cookies, as that is what most applications
# use. Using an incorrect cookie version may open your installation to
# evasion attacks (against the rules that examine named cookies).
#
SecCookieFormat 0

# Specify your Unicode Code Point.
# This mapping is used by the t:urlDecodeUni transformation function
# to properly map encoded data to your language. Properly setting
# these directives helps to reduce false positives and negatives.

```



---

```
#  
#SecUnicodeCodePage 20127  
#SecUnicodeMapFile unicode.mapping
```



---



---

## OHS Module Directives

This appendix describes the directives available in the Oracle-developed modules supported by OHS. It contains these sections:

- [Section E.1, "mod\\_certheaders"](#)
- [Section E.2, "mod\\_onsint"](#)
- [Section E.3, "mod\\_oradav"](#)
- [Section E.4, "mod\\_oss1"](#)
- [Section E.5, "mod\\_osso"](#)
- [Section E.6, "mod\\_plsql"](#)
- [Section E.7, "mod\\_wl\\_ohs"](#)

---



---

**Note:** The directives described in this appendix are only for modules developed by Oracle. OHS also includes many Apache HTTP Server and third-party modules out-of-the-box. See "[Apache HTTP Server and Third-party Modules in Oracle HTTP Server](#)" in Chapter 3 for a list of these modules and a link to their documentation.

---



---

### E.1 mod\_certheaders

mod\_certheaders accepts the following directives:

- [AddCertHeader](#)
- [SimulateHttps](#)

#### E.1.1 AddCertHeader

Specifies 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. AddCertHeader is generally used for Web cache in front of OHS.

Category	Value
Syntax	AddCertHeader <i>environment_variable</i>
Example	AddCertHeader SSL_CLIENT_CERT

Category	Value
Default	None

---

**Note:** If a simulated SSL connection needs to be passed to WebLogic Server, `AddCertHeader` can be used with `mod_wl_ohs` as well. That module's directive, `WLProxySSLPassThrough`, ensures that the certificate is passed to the backend (irrespective of the real or simulated SSL connection).

---

### E.1.2 SimulateHttps

`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. and `SimulateHttps` is used for load balancer type of devices.

Category	Value
Syntax	<code>SimulateHttps on off</code>
Example	<code>SimulateHttps on</code>
Default	<code>off</code>

## E.2 mod\_onsint

`mod_certheaders` accepts the `OpmnHostPort` directive.

### E.2.1 OpmnHostPort

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. However, in certain circumstances, you may want to choose a specific HTTP port and hostname that OPMN should use to ping the listener with.

Category	Value
Syntax	<code>OpmnHostPort host:Port</code>
Example	<code>OpmnHostPort localhost:7778</code>
Default	None

## E.3 mod\_oradav

`mod_oradav` accepts the following directives:

- [ORAAllowIndexDetails](#)
- [ORAAltPassword](#)
- [ORACacheDirectory](#)
- [ORACacheMaxResourceSize](#)

- [ORACachePrunePercent](#)
- [ORACacheTotalSize](#)
- [ORAConnect](#)
- [ORAConnectSN](#)
- [ORAContainerName](#)
- [ORAException](#)
- [ORAGetSource](#)
- [ORALockExpirationPad](#)
- [ORAPackageName](#)
- [ORAPassword](#)
- [ORARootPrefix](#)
- [ORAService](#)
- [ORATraceEvents](#)
- [ORATraceLevel](#)
- [ORAUser](#)

### E.3.1 ORAAallowIndexDetails

In an Oracle HTTP Server environment that is not OraDAV-enabled, mod\_dav does not respond to HTTP GET requests. Instead, normal Oracle HTTP Server mechanisms are used to respond to GET requests.

The ORAAallowIndexDetails parameter controls how OraDAV responds when a GET request is performed on a DAV collection and no index.html file is found in the 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.

Category	Value
Syntax	ORAAallowIndexDetails true   false
Example	ORAAallowIndexDetails true
Default	false

### E.3.2 ORAAltPassword

Specifies the password for the user specified by the [ORAUser](#) parameter. The ORAAltPassword uses a base-64 encoded character string. This parameter provides an alternative if you do not want the password to appear in unencoded plain text with the ORAUser parameter. If the ORAPassword parameter is not specified, this value is used for the password.

Category	Value
Syntax	ORAAltPassword <i>password</i>
Example	ORAAltPassword myPassword
Default	None

### E.3.3 ORACacheDirectory

Specifies the directory to use for disk caching operations, per these requirements:

- 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 be located on a file system that supports a last accessed time. On Microsoft Windows systems, this means using NTFS, not FAT, formatted partitions.
- You cannot 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.

Category	Value
Syntax	<code>ORACacheDirectory <i>directoryName</i></code>
Example	<code>ORACacheDirectory</code>
Default	If not used, disk caching is not performed for OraDAV operations

**See Also:** [Section 9.4.1, "Using Disk Caching with OraDAV"](#)

### E.3.4 ORACacheMaxResourceSize

Specifies the maximum cacheable resource size for disk caching operations. You can specify kilobytes (KB) or megabytes (MB) after an integer. If you do not specify a unit after the integer, then the default unit is bytes.

This parameter enables Web administrators to prevent large media files from dominating the cache. The performance benefit of caching a large file is greater than from caching a small file.

Category	Value
Syntax	<code>ORACacheMaxResourceSize <i>n</i>nKB MB</code>
Example	<code>ORACacheMaxResourceSize 1024KB</code>
Default	None

**See Also:** [Section 9.4.1, "Using Disk Caching with OraDAV"](#)

### E.3.5 ORACachePrunePercent

Specifies the percentage of disk cache usage to be freed when the cache is full. When the disk cache is full, the oldest files in the cache are deleted until the cache disk usage is reduced by the `ORACachePrunePercent` value.

Category	Value
Syntax	<code>ORACachePrunePercent <i>nn</i></code>
Example	<code>ORACachePrunePercent 40</code>
Default	25

**See Also:** [Section 9.4.1, "Using Disk Caching with OraDAV"](#)

### E.3.6 ORACacheTotalSize

Specifies the size of the cache to use for disk caching operations. You can specify MB (for megabytes) or GB (for gigabytes) after an integer. If you do not specify a unit after the integer, the default unit is bytes.

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 amount of your Web site, or all of the 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 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.

Category	Value
Syntax	DAVParam ORACacheTotalSize <i>nn</i> MB GB
Example	DAVParam ORACacheTotalSize 1GB
Default	None

**See Also:** [Section 9.4.1, "Using Disk Caching with OraDAV"](#)

### E.3.7 ORAConnect

Specifies the Oracle database to connect to. The [ORAConnect](#) parameter lets you connect to a database that is not included in the `tnsnames.ora` file. The value must use the following format:

To connect to an Oracle database, you must specify one, and no more than one, of the parameters [ORAConnect](#), [ORAConnectSN](#), or [ORAService](#). To connect to a database included in the `tnsnames.ora` file, use the [ORAService](#) parameter.

Category	Value
Syntax	<a href="#">ORAConnect</a> <i>database_host:database_port:database_sid</i>
Example	<a href="#">ORAConnect</a> my-pc.example.com:1521:mysid
Default	None

### E.3.8 ORAConnectSN

Specifies the Oracle database to connect to. The [ORAConnectSN](#) parameter lets you connect to a database that is not included in the `tnsnames.ora` file. The value for this parameter is a character string. The value must use the following format:

To connect to an Oracle database, you must specify one, and no more than one, of the parameters [ORAConnect](#), or [ORAService](#). To connect to a database included in the `tnsnames.ora` file, use the [ORAService](#) parameter.

Category	Value
Syntax	ORAConnectSN <i>database-host:database-port:database-service-name</i>
Example	ORAConnectSN my-pc.example.com:1521:mysid
Default	None

### E.3.9 ORAContainerName

Within the database user (schema) specified by the [ORAUser](#) parameter, there must exist a container, which is a set of PL/SQ packages and database tables that allow the storage of files in the database within a hierarchical structure. The `ORAContainerName` parameter specifies the name of the container to use for the location. The value for this parameter is a character string, up to 20 characters. For example, `<Location/project1>`.

Category	Value
Syntax	ORAContainerName <i>Location/project1</i>
Example	ORAContainerName
Default	None

### E.3.10 ORAException

Writes PL/SQL stack dumps in the Oracle HTTP Server log file, `error_log`, in the event of an exception in the PL/SQL package.

Category	Value
Syntax	ORAException RAISE   NORAISE
Example	ORAException RAISE
Default	NORAISE

---

**Note:** 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.

---

### E.3.11 ORAGetSource

Specifies one or more file extensions to identify types of files that are not to be run, but rather opened for editing. This allows you to open files for editing that are usually run as a result of a GET operation. When entering filetypes, include dot separators (.) and use a comma to separate file extensions. The value for this parameter is enclosed within double quotation marks. For example:

```
".htm, .html, .jsp1, .jsp2"
```

This directive applies only to file system access.

Category	Value
Syntax	ORAGetSource ".type[, .type, .type, .type]"



Category	Value
Example	ORAGetSource ".htm, .html, .jsp1, .jsp2"
Default	.JSP .SQLJSP

---

**Note:** The .jsp and .sqljsp files are by default opened for editing; you do not need to specify them in the ORAGetSource parameter.

---

### E.3.12 ORALockExpirationPad

Intended to be used in high-latency network environments to adjust the refresh lock behavior in Microsoft Office. Microsoft Office attempts to refresh locks on DAV resources just before the lock is set to expire. If the Microsoft Office client and the DAV server experience network congestion between the refresh, the request might arrive 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 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 for at least two minutes after the expiration time.

Category	Value
Syntax	ORALockExpirationPad <i>nn</i>
Example	ORALockExpirationPad 90
Default	0

### E.3.13 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
Syntax	ORAPackageName <i>name</i>
Example	ORAPackageName
Default	The OraDAV driver

### E.3.14 ORAPassword

Specifies the password for the user specified by the [ORAUser](#) parameter.

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.

Category	Value
Syntax	ORAPassword <i>password</i>
Example	ORAPassword myPassword

Category	Value
Default	None

### E.3.15 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 able to see the repository directories that lead up to it. Do not include a trailing slash (/) in the value.

WebDAV clients can view the /third directory and can navigate to the /third/fourth directory, but will not be able to view or navigate to the /first or /first/second directories.

Category	Value
Syntax	ORARootPrefix /dirName1[/dirName1/dirName1/dirName1]
Example	ORARootPrefix /first/second
Default	None

### E.3.16 ORAService

Specifies the Oracle database to connect to. The specified value must match a SID value in the tnsnames.ora file.

To connect to an Oracle database, you must specify one, and no more than one, of the parameters [ORACONNECT](#), [ORACONNECTSN](#), or [ORASERVICE](#). To connect to a database that is not included in the tnsnames.ora file, use the [ORACONNECT](#) or [ORACONNECTSN](#) parameter.

Category	Value
Syntax	ORAService <i>databaseName</i>
Example	ORAService
Default	None

### E.3.17 ORATraceEvents

Specifies the types of events to be recorded in the Oracle HTTP Server error log for debugging. The value for this parameter is one of the following:

- `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

Category	Value
Syntax	ORATraceEvents <code>getsource</code>   <code>hreftoutf8</code>   <code>request</code>
Example	ORATraceEvents <code>getsource</code>
Default	None

---



---

**Note:** 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.

---



---

### E.3.18 ORATraceLevel

Specifies the level of debugging (trace statements) that will be entered in the Oracle HTTP Server error log. The lowest level is 0 (the default), which performs no tracing. The highest level is 4, which performs maximum tracing. The value for this parameter is an integer between 0 and 4.

The higher the number for the debugging level, the more information is written to the error log file.

Category	Value
Syntax	ORATraceLevel 0   1   2   3   4
Example	ORATraceLevel 2
Default	0

---



---

**Note:** 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.

---



---

### E.3.19 ORAUser

Specifies the database user (schema) to use when connecting to the service specified by the [ORACONNECT](#), [ORACONNECTSN](#), or [ORASERVICE](#) parameter.

This user must have the following privileges:

- CONNECT
- RESOURCE
- CREATE TABLESPACE
- DROP TABLESPACE
- CREATE ANY TRIGGER

Category	Value
Syntax	ORAUser
Example	ORAUser
Default	None

## E.4 mod\_oss1

To configure SSL for your Oracle HTTP Server, enter the `mod_oss1` 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

#### E.4.1 SSLAccelerator

Specifies if SSL accelerator is used. Currently only nFast card is supported.

Category	Value
Syntax	SSLAccelerator yes no
Example	SSLAccelerator yes
Default	SSLAccelerator no

---

**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>.

---

#### E.4.2 SSLCARevocationFile

Specifies the file where you can assemble the Certificate Revocation Lists (CRLs) from CAs (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 <i>file_name</i>

Category	Value
Example	SSLCARevocationFile /ORACLE_ HOME/Apache/Apache/conf/ssl.crl/ca_bundle.crl
Default	None

### E.4.3 SSLCARevocationPath

Specifies the directory where **PEM**-encoded Certificate Revocation Lists (CRLs) are stored. These CRLs come from the **CAs** (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 <i>path/to/CRL_directory/</i>
Example	SSLCARevocationPath /ORACLE_ HOME/Apache/Apache/conf/ssl.crl/
Default	None

### E.4.4 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. SSLCipherSuite accepts the following values:

- none: Adds the cipher to the list
- + : Adds the cipher to the list and place it in the correct location in the list
- - : Remove the cipher from the list (can be added later)
- ! : Remove the cipher from the list permanently

Tags are joined together with prefixes to form a cipher specification string. Cipher suite tags are listed in [Table E–1](#) and [Table E–2](#).

Category	Value
Example	SSLCipherSuite ALL:!LOW:!DH In this example, all ciphers are specified except low strength ciphers and those using the <b>Diffie-Hellman key negotiation algorithm</b> .
Syntax	SSLCipherSuite <i>cipher-spec</i>
Default	ALL:!ADH:!EXPORT56:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP

**Table E–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

**Table E-1 (Cont.) SSLCipher Suite Tags**

Function	Tag	Meaning
Authentication	aDH	Diffie-Hellman authentication
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_oss1`, supports RC4-40 encryption only when the server uses 512 bit key size wallets.

---

**Table E-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

**Table E-2 (Cont.) Cipher Suites Supported in Oracle Advanced Security 10i**

Cipher Suite	Authentication	Encryption	Data Integrity
SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DH DSS	3DES (168)	SHA
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

### E.4.5 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.

Category	Value
Syntax	SSLEngine on off
Example	SSLEngine on
Default	off

### E.4.6 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	SSLLog <i>path/to/file</i>
Example	SSLLog
Default	None

### E.4.7 SSLLogLevel

Specifies the verbosity degree of the SSL engine log file. The degrees, or 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.

Category	Value
Syntax	<code>SSLLogLevel none   error   warn   info   trace   debug</code>
Example	<code>SSLLogLevel error</code>
Default	None

#### E.4.8 SSLMutex

Type of semaphore (lock) for SSL engine's mutual exclusion of operations that have to be synchronized between Oracle HTTP Server processes. Accepted values are:

- `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.

Category	Value
Syntax	<code>SSLMutex none   file   sem</code>
Example	<code>SSLMutex file:/usr/local/apache/logs/ssl_mutex</code>
Default	None

#### E.4.9 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.

Accepted values are:

- `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  $n = 0, 1, 2, \dots$ )

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.
- `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_oss1` 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.

Category	Value
Syntax	<code>SSLOptions [+ -] StdEnvVars   ExportCertData   FakeBasicAuth   StrictRequire   CompatEnvVars   OptRenegotiate</code>
Example	<code>SSLOptions -StdEnvVars</code>
Default	None

## E.4.10 SSLPassPhraseDialog

Type of pass phrase dialog for wallet access. `mod_oss1` asks the administrator for a pass phrase in order to access the wallet. Accepted if values are:

- `builtin`: when the server is started, `mod_oss1` 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_oss1` calls an external program configured for each wallet. This program is invoked with two arguments: `servername:portnumber` and `RSA` or `DSA`.

Category	Value
Syntax	<code>SSLPassPhraseDialog builtin   exec</code>
Example	<code>SSLPassPhraseDialog exec:/usr/local/apache/sbin/pfilter</code>
Default	<code>builtin</code>

### E.4.11 SSLProtocol

Specifies SSL protocol(s) for `mod_oss1` to use when establishing the server environment. Clients can only connect with one of the specified protocols. Accepted values are:

- `SSLv2`
- `SSLv3`
- `TLSv1`
- `All`

Category	Value
Syntax	<code>SSLProtocol [+ -] SSLv2   SSLv3   TLSv1   All</code>
Example	<code>SSLProtocol +SSLv3</code>
Default	<code>ALL</code>

### E.4.12 SSLRequire

Denies access unless an arbitrarily complex boolean expression is true.

Category	Value
Syntax	<code>SSLRequire expression</code> (see <a href="#">Understanding the Expression</a> )
Example	<code>SSLRequire word "&gt;=" word   word "ge" word</code>
Default	<code>None</code>

#### Understanding the Expression

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 E-3](#) and [Table E-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 E-3](#) lists the standard variables for `SSLRequire` `varname`.

**Table E-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 E-4](#) lists the SSL variables for `SSLRequire` `varname`.

**Table E-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

**Table E-4 (Cont.) SSL Variables for SSLRequire Varname**

SSL Variables	SSL Variables	SSL Variables
SSL_CIPHER_USEKEYSIZE	SSL_VERSION_LIBRARY	SSL_SESSION_ID
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_SERVER_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	SSL_SERVER_I_DN_T	SSL_SERVER_I_DN_OU
SSL_SERVER_I_DN_G	SSL_SERVER_I_DN_I	

### E.4.13 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
Example	SSLRequireSSL
Default	None

### E.4.14 SSLSessionCache

Specifies the global/interprocess session cache storage type. The cache provides an optional way to speed up parallel request processing. The accepted values are:

- 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`.

Category	Value
Syntax	<code>SSLSessionCache none   shmht: /path/to/datafile[bytes]   shmcb: /path/to/datafile[bytes]</code>
Examples	<code>SSLSessionCache shmht: /ORACLE_HOME/Apache/Apache/logs/ssl_scache(512000)</code> <code>SSLSessionCache shmcb: /ORACLE_HOME/Apache/Apache/logs/ssl_scache(512000)</code>
Default	<code>SSLSessionCache none</code>

#### E.4.15 SSLSessionCacheTimeout

Specifies the number of seconds before a SSL session in the session cache expires.

Category	Value
Syntax	<code>SSLSessionCacheTimeout seconds</code>
Example	<code>SSLSessionCacheTimeout 120</code>
Default	300

#### E.4.16 SSLVerifyClient

Specifies whether or not a client must present a certificate when connecting. The accepted values are:

- `none`: No client certificate is required
- `optional`: Client may present a valid certificate
- `require`: Client must present a valid certificate

Category	Value
Syntax	<code>SSLVerifyClient none   optional   require</code>
Example	<code>SSLVerifyClient optional</code>
Default	None

---

**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_oss1`.

---

## E.4.17 SSLWallet

Specifies the location of the wallet with its **WRL**, specified as a filepath.

Category	Value
Syntax	SSLWallet <i>file:path to wallet</i>
Example	SSLWallet <i>file:/etc/ORACLE/WALLETS/server</i>
Default	None

## E.4.18 SSLWalletPassword

---

**Note:** SSLWalletPassword has been deprecated. A warning message is generated in the Oracle HTTP Server log if this directive is used. For secure wallets, you should get a SSO wallet, with auto-login enabled, instead.

---

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. If no password is required do not set this directive.

---

**Note:** If you are using a wallet created with the Auto Login feature of Oracle Wallet Manager, do not set this directive because these wallets do not require passwords.

---

Category	Value
Syntax	SSLWalletPassword <i>password</i>
Example	SSLWalletPassword <i>myWalletPassword</i>
Default	None

**See Also:** "Using the `iasobf` Utility" on page 11-15

## E.5 mod\_osso

---

**Note:** mod\_osso is deprecated. You should use the Webgate plugin to achieve single sign on. For more information, see "Installing and Configuring Oracle HTTP Server 11g Webgate for OAM" in *Oracle® Fusion Middleware Installation Guide for Oracle Identity Management*.

---

mod\_ossdo accepts the following directives:

- [OssoConfigFile](#)
- [OssoIdleTimeout](#)
- [OssoIgnoreUri](#)

- [OssoIpCheck](#)
- [OssoLegacyApp](#)
- [OssoProtectedOnly](#)
- [OssoRedirectByForm](#)
- [OssoSendCacheHeaders](#)
- [OssoHTTPOnly](#)

### E.5.1 OssoConfigFile

Specifies the path to `osso.conf`.

Category	Value
Syntax	<code>OssoConfigFile path/to/osso.conf</code>
Example	<code>OssoConfigFile OracleOHS_Home/ohs/conf/osso/osso.conf</code>
Default	None

### E.5.2 OssoldleTimeout

Specifies whether or not to enable cookie idle timeout.

Category	Value
Syntax	<code>OssoIdleTimeout ON   OFF</code>
Example	<code>OssoIdleTimeout OFF</code>
Default	OFF

### E.5.3 OssolgnoreUri

Identifies which URLs OHS should ignore.

- If ALL is specified, then all URLs are ignored.
- If querystring is specified, URLs matching the query string are ignored.

Category	Value
Syntax	<code>OssoIgnoreUri ALL   querystring string</code>
Example	<code>OssoIgnoreUri ALL</code> or <code>OssoIgnoreUri querystring foo=bar,bar=baz\</code>
Default	No URLs are ignored.

### E.5.4 OssolpCheck

When set to On, the directive enables a weak form of IP spoof checking. The incoming IP address is compared against the ClientIP header value.

Category	Value
Syntax	OssoIpCheck ON   OFF
Example	OssoIpCheck ON
Default	OFF

### E.5.5 OssoLegacyApp

Specifies whether or not to enable URI authentication for the location where it was set. When On, URI authentication is enabled.

Category	Value
Syntax	OssoLegacyApp ON   OFF
Example	OssoLegacyApp ON
Default	OFF

### E.5.6 OssoProtectedOnly

Specifies whether or not to create a cookie for an application that protects itself via dynamic sso directives. When ON this cookie is created.

Category	Value
Syntax	OssoProtectedOnly ON   OFF
Example	OssoProtectedOnly ON
Default	OFF

### E.5.7 OssoRedirectByForm

Specifies whether or not to enable redirect by form. When On, the redirect is enabled.

Category	Value
Syntax	OssoRedirectByForm ON   OFF
Example	OssoRedirectByForm ON
Default	OFF

### E.5.8 OssoSecureCookies

Specifies whether or not to enable secure cookies. When set to On, these cookies are enabled.

Category	Value
Syntax	OssoSecureCookies ON   OFF
Example	OssoSecureCookies OFF
Default	ON



## E.5.9 OssoSendCacheHeaders

For statically protected pages, this directive specifies whether or not the following headers are sent:

- Pragma: no-cache
- Cache-Control: no-store
- Surrogate-Control: no-store
- Expires: Thu, 01 Jan 1970 12:00:00 GMT

Category	Value
Syntax	OssoSendCacheHeaders ON   OFF
Example	OssoSendCacheHeaders OFF
Default	ON

## E.5.10 OssoHTTPOnly

Specifies whether or not to set HTTPOnly in the cookies created by the module. If ON, HTTPOnly is set.

Category	Value
Syntax	OssoHttpOnly ON   OFF
Example	OssoHttpOnly OFF
Default	ON

## E.6 mod\_plsql

The mod\_plsql configuration parameters are described in these sections:

- [Section E.6.1, "plsql.conf"](#)
- [Section E.6.2, "dads.conf"](#)
- [Section E.6.3, "cache.conf"](#)

### E.6.1 plsql.conf

The following parameters are used with the `plsql.conf` file:

- [PlsqlDMSEnable](#)
- [PlsqlLogEnable](#)
- [PlsqlLogDirectory](#)
- [PlsqlIdleSessionCleanupInterval](#)

#### E.6.1.1 PlsqlDMSEnable

Enables Dynamic Monitoring Service (DMS) for the mod\_plsql module.

Category	Value
Syntax	PlsqlDMSEnable On   Off

Category	Value
Example	PlsqlDMSEnable On
Default	On

### E.6.1.2 PlsqlLogEnable

Enables debug level logging for the mod\_plsql module. Debug level logging is meant to be used for debugging purposes only.

When logging is enabled, Oracle HTTP Server log files are typically created in the `ORACLE_INSTANCE/diagnostics/logs/OHS/config/OHS/component_name` directory. However, the location specified in [PlsqlLogDirectory](#) determines the final location.

This parameter should be set to `Off` unless recommended by Oracle support to debug problems with the mod\_plsql module.

To view more details about the internal processing of the mod\_plsql module, set this directive to `On`. This causes the mod\_plsql module to start logging every request that is processed. The log files are generated as specified by the [PlsqlLogDirectory](#) directive.

Category	Value
Syntax	PlsqlLogEnable On   Off
Example	PlsqlLogEnable Off
Default	Off

### E.6.1.3 PlsqlLogDirectory

Specifies the directory where debug level logs are written.

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 <i>directory</i>
Example	PlsqlLogDirectory ORACLE_INSTANCE/diagnostics/logs/OHS/ <i>component_name</i>
Default	None

### E.6.1.4 PlsqlIdleSessionCleanupInterval

Specifies the time (in minutes) in which the idle database sessions should be closed and cleaned by the mod\_plsql module.

This directive is used in conjunction with connection pooling of database connections and sessions in the mod\_plsql module. 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. If this number is too low, then this may adversely affect the performance benefits of connection pooling in the mod\_plsql module.

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 value is adequate.

Category	Value
Syntax	PlsqlIdleSessionCleanupInterval <i>number</i>
Example	PlsqlIdleSessionCleanupInterval 10
Default	15 (minutes)

## E.6.2 dads.conf

The `dads.conf` file contains the configuration parameters for the PL/SQL database access descriptor. (See [Table E-1](#) for the file location.) A DAD is a set of values that specifies how the mod\_plsql module connects to a database server to fulfill a HTTP request.

The following parameters are used with the `dads.conf` file:

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

### E.6.2.1 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 is made after running every procedure.

Category	Value
Syntax	PlsqlAfterProcedure <i>string</i>
Example	PlsqlAfterProcedure portal.mypkg.myafterproc
Default	None

---

**Note:** This parameter should only be used for debugging purposes. In addition, you could use this parameter to stop SQL trace/SQL profiling.

---

### E.6.2.2 PlsqlAlwaysDescribeProcedure

Specifies whether the mod\_plsql module should describe a procedure before trying to run it. If this is set to On, then the mod\_plsql module will always describe a procedure before invoking it. Otherwise, the mod\_plsql module will only describe a procedure when its internal heuristics have interpreted a parameter type incorrectly.

Category	Value
Syntax	PlsqlAlwaysDescribeProcedure On   Off
Example	PlsqlAlwaysDescribeProcedure On
Default	Off

---

**Note:** This parameter should only be used for debugging purposes.

---

### E.6.2.3 PlsqlAuthenticationMode

Specifies the authentication mode to use for allow access through the DAD. The accepted values for PlsqlAuthenticationMode are Basic, SingleSignOn, GlobalOwa, CustomOwa, PerPackageOwa.

Category	Value
Syntax	PlsqlAuthenticationMode Basic   SingleSignOn   GlobalOwa   CustomOwa   PerPackageOwa
Example	PlsqlAuthenticationMode CustomOwa
Default	Basic

- Basic is the default mode and determines whether or not to ask for username and password if they are not provided with PlsqlDatabaseUsername and PlsqlDatabasePassword. This setting is required for WebDB 2.x applications. If the DAD is not using the Basic authentication, then you must include a valid username/password in the DAD configuration.
- SingleSignOn specifies that you want to use Single Sign-On server. This is required for DADs using Oracle9iAS Portal. As already stated the provided username and password need to be the one from your single signon server.
- GlobalOwa, CustomOwa, and PerPackageOwa are used only by very few PL/SQL applications. Custom authentication enables applications to authenticate users within the application itself, not at the database level.

Authorization is performed by invoking a user-written authorization function. Custom authentication uses a static username/password that is stored in the DAD. It cannot be combined with dynamic username/password authentication. To enable custom authentication, set the level of authentication for `PlsqlAuthenticationMode` and implement the `authorize` function.

You should also be aware of the following:

- 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, to perform dynamic authentication, the DAD username/password parameters must be omitted.
- The `SingleSignOn` mode is supported only for Oracle Fusion Middleware releases, and is used by Oracle Portal and Oracle Single Sign-On. Most customer applications use Basic authentication. Custom authentication modes (`GlobalOwa`, `CustomOwa`, and `PerPackageOwa`) are used by very few PL/SQL applications.

#### E.6.2.4 `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	<code>PlsqlBeforeProcedure string</code>
Example	<code>PlsqlBeforeProcedure portal.mypkg.mybeforeproc</code>
Default	None

---

**Note:** This parameter should only be used for debugging purposes. In addition, you could use this parameter to start SQL Trace/SQL Profiling.

---

#### E.6.2.5 `PlsqlBindBucketLengths`

---

**Note:** This configuration property is rarely ever changed, and system defaults suffice in almost all cases.

---

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 run 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 the `mod_plsql` module 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 <i>number multiline</i>
Example	PlsqlBindBucketLengths 4 PlsqlBindBucketLengths 25 PlsqlBindBucketLengths 125
Default	4,20,100,400

- 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.
- After the higher configured value, mod\_plsql starts auto-generating bucket sizes of larger values by doubling the last value, as needed. Therefore, after 400, the next bucket value becomes 800, then 1600, and so on.
- Consider using flexible parameter passing to reduce the problem.

### E.6.2.6 PlsqlBindBucketWidths

---



---

**Note:** This configuration property is rarely ever changed, and system defaults suffice in almost all cases.

---



---

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 run 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 the mod\_plsql module 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 <i>number multiline</i>
Example	PlsqlBindBucketWidths 40 PlsqlBindBucketWidths 400 PlsqlBindBucketWidths 2000

Category	Value
Default	32, 128, 1450, 2048, 4000

- 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.
- After the higher configured value, `mod_plsql` starts auto-generating bucket sizes of larger values by doubling the last value, as needed. Therefore, after 400, the next bucket value becomes 800, then 1600, and so on.
- Consider using flexible parameter passing to reduce the problem.

### E.6.2.7 PlsqlCGIEnvironmentList

Specifies overrides and additions of CGI environment variables to the default set of environment variables passed 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 blank. To add your own name-value pair, use the syntax `myname=myvalue`.

Category	Value
Syntax	<code>PlsqlCGIEnvironmentList string multiline</code>
Default	None
Example	<ul style="list-style-type: none"> <li>■ To add a new environment variable from the Oracle HTTP Server environment: <code>PlsqlCGIEnvironmentList DOCUMENT_ROOT</code></li> <li>■ To remove an environment variable: <code>PlsqlCGIEnvironmentList MYENVAR2=</code></li> <li>■ To override from the Oracle HTTP Server environment: <code>PlsqlCGIEnvironmentList REQUEST_PROTOCOL=HTTPS</code></li> <li>■ To add your own environment variable: <code>PlsqlCGIEnvironmentList MY_VARNAME=MY_VALUE</code></li> </ul>

- Environment variables added here are available in the PL/SQL application through the function `owa_util.get_cgi_env`.

### E.6.2.8 PlsqlConnectionTimeout

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

Category	Value
Syntax	<code>PlsqlConnectionTimeout number</code>
Example	<code>PlsqlConnectionTimeout 5000</code>

Category	Value
Default	10000 (milliseconds)

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

### E.6.2.9 PlsqlConnectionValidation

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

---



---

**Note:** This configuration property is rarely ever changed, and system defaults suffice in almost all cases.

---



---

For performance reasons, the `mod_plsql` module pools database connections. If a database instance goes down, and the `mod_plsql` module 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 such as RAC where even if one node goes down, other nodes servicing the database might have been able to service the request successfully. The `mod_plsql` module 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`: The `mod_plsql` module tests all pooled database connections which were created prior to the detection of a failure that could mean an instance failure.
- `ThrowAwayOnFailure`: The `mod_plsql` module throws away all pooled database connections which were created prior to the detection of a failure that could mean an instance failure.
- `AlwaysValidate`: The `mod_plsql` module 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`: The `mod_plsql` module never pings any pooled database connection.

Category	Value
Syntax	<code>PlsqlConnectionValidation Automatic   ThrowAwayOnFailure   AlwaysValidate   NeverValidate</code>
Example	<code>PlsqlConnectionValidation ThrowAwayOnFailure</code>
Default	<code>Automatic</code>

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

- 00443 – background process <string> did not start
- 00444 – background process <string> failed while starting



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

#### **E.6.2.10 PlsqlDatabaseConnectionString**

Specifies the connection to an Oracle database.

Category	Value
Syntax	<p><code>PlsqlDatabaseConnectionString string {ServiceNameFormat   SIDFormat   TNSFormat   NetServiceNameFormat}</code></p> <p>The <i>string</i> parameter depends on the second argument:</p> <ul style="list-style-type: none"> <li>■ If the second argument is <code>ServiceNameFormat</code>, <i>string</i> is <code>HOST:PORT:SERVICE_NAME</code>, where <i>HOST</i> is the host name running the database, <i>PORT</i> is the port number the TNS listener is listening at, and <i>SERVICE_NAME</i> is the database service name.</li> </ul> <p>An IPv6 address can be specified using the format <code>[IPv6_ADDRESS]:PORT:SERVICE_NAME</code>.</p> <ul style="list-style-type: none"> <li>■ If the second argument is <code>SIDFormat</code>, <i>string</i> is <code>HOST:PORT:SID</code> where <i>HOST</i> is the host name running the database, <i>PORT</i> is the port number the TNS listener is listening at, and <i>SID</i> is the database SID.</li> </ul> <p>An IPv6 address can be specified using the format <code>[IPv6_ADDRESS]:PORT:SID</code>.</p> <ul style="list-style-type: none"> <li>■ If the second argument is <code>TNSFormat</code>, <i>string</i> is a valid TNS alias that can be resolved using Net8 utilities like <code>tnsping</code> and <code>SQL*Plus</code>.</li> <li>■ If the second argument is <code>NetServiceNameFormat</code>, <i>string</i> is a valid net service name that can be resolved 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.</li> </ul> <p>If the format argument is not specified, then the <code>mod_plsql</code> module assumes the <i>string</i> is either in the <code>HOST:PORT:SID</code> format, or resolvable by Net8. The differentiation between the two is made by the presence of the colon in the specified string.</p> <p>It is recommended that newer DADs do not use the <code>SIDFormat</code> syntax. This exists only for backward compatibility reasons. Use the new two argument format for newly created DADs.</p>
Example	<ul style="list-style-type: none"> <li>■ <code>PlsqlDatabaseConnectionString example.com:1521:myhost.iasdb.inst ServiceNameFormat</code></li> <li>■ <code>PlsqlDatabaseConnectionString [2001:DB8:f1ff:f1ff]:1521:myhost.iasdb.inst ServiceNameFormat</code></li> <li>■ <code>PlsqlDatabaseConnectionString example.com:1521:iasdb SIDFormat</code></li> <li>■ <code>PlsqlDatabaseConnectionString [2001:DB8:ff1ff:f1ff]:1521:iasdb SIDFormat</code></li> <li>■ <code>PlsqlDatabaseConnectionString myhost_tns TNSFormat</code></li> <li>■ <code>PlsqlDatabaseConnectionString cn=oracle,cn=iasdb NetServiceNameFormat</code></li> <li>■ <code>PlsqlDatabaseConnectionString (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=example.com)(Port= 1521))(CONNECT_DATA=(SID=iasdb))) TNSFormat</code></li> <li>■ <code>PlsqlDatabaseConnectionString myhost_tns</code></li> <li>■ <code>PlsqlDatabaseConnectionString example.com:1521:iasdb</code></li> </ul>
Default	None
	<ul style="list-style-type: none"> <li>■ If the database is running in the same Oracle home, or the environment variable <code>TWO_TASK</code> is set, then this parameter need not be specified.</li> <li>■ If the database is running in a separate Oracle home, then this parameter is mandatory.</li> </ul>

- If you have problems connecting to the database:
  - Check the username and password information in the DAD.
  - Make sure that you run `tnsping db_connect_string`, and commands such as:
 

```
sqlplus DADUsername/DADPassword@db_connect_string
```
  - Ensure that TNS\_ADMIN is configured properly.
  - Verify that the HOST:PORT:SERVICE\_NAME format works correctly.
  - Ensure that the TNS listener and database are up and running.
  - Ensure that you can ping the host from this machine.
- From a the mod\_plsql module perspective, `TNSFormat` and `NetServiceNameFormat` are synonymous and denote connect descriptors that are resolved by Net8. 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, Oracle Real Application Clusters 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 the mod\_plsql module by changing Oracle Internet Directory with the new or 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.

### E.6.2.11 PlsqlDatabasePassword

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

Category	Value
Syntax	<code>PlsqlDatabasePassword string</code>
Example	<code>PlsqlDatabasePassword tiger</code>
Default	None

- 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 uses the name of the schema owner.

After making manual configuration changes to DAD passwords, you should obfuscate the DAD passwords by running the `dadTool.pl` script, located in `ORACLE_HOME/bin`.

To obfuscate DAD passwords:

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

```
$ su - oracle
```

- 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.

Bourne, Bash, or Korn shell:

```
$ ORACLE_HOME=new_ORACLE_HOME_path;export ORACLE_HOME
$ PATH=ORACLE_HOME/bin:ORACLE_HOME/perl/bin:$PATH;export PATH
```

C or tcsh shell:

```
% setenv ORACLE_HOME new_ORACLE_HOME_PATH
% setenv PATH ORACLE_HOME/bin:ORACLE_HOME/perl/bin:PATH
```

On Microsoft Windows, set the `PATH` and `PERL5LIB` environment variable:

```
set PATH=ORACLE_HOME\bin;ORACLE_HOME\perl\bin;%PATH%
set PERL5LIB=ORACLE_HOME\perl\lib
```

- On UNIX platforms, set the shared library path environment variable.

Include the `ORACLE_HOME/lib` or `lib32` directory in your shared library path. [Table E-5](#) shows the appropriate directory and environment variable for each platform.

**Table E-5 Shared Library Path Environment Variable**

Platform	Environment Variable	Include Directory
AIX Based Systems	LIBPATH	ORACLE_HOME/lib
HP-UX PA-RISC	SHLIB_PATH	ORACLE_HOME/lib
Solaris Operating System	LD_LIBRARY_PATH	ORACLE_HOME/lib32
Other UNIX platforms, including Linux and HP Tru64 UNIX	LD_LIBRARY_PATH	ORACLE_HOME/lib

For example, on HP-UX PA-RISC systems, set the `SHLIB_PATH` environment to include the `ORACLE_HOME/lib` directory:

```
$SHLIB_PATH=$ORACLE_HOME/lib:$SHLIB_PATH;export SHLIB_PATH
```

- Change directory to the `mod_plsql` configuration directory for the current release of Oracle HTTP Server:

```
cd ORACLE_HOME/bin
```

- Invoke the following Perl script to obfuscate DAD password:

```
perl dadTool.pl -f dadfilename
```

where `dadfilename` is the filename for `dads.conf`, which includes the full path to the DAD file.

For example:

```
perl dadTool.pl -f /u01/app/oracle/as11gr1/ORACLE_
INSTANCE/config/OHS/component_name/mod_plsql/dads.conf
```

### E.6.2.12 PlsqlDatabaseUserName

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

Category	Value
Syntax	PlsqlDatabaseUsername <i>string</i>
Example	PlsqlDatabaseUsername scott
Default	None

- 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.

### E.6.2.13 PlsqlDefaultPage

Specifies the default procedure to call if none is specified in the URL.

Category	Value
Syntax	PlsqlDefaultPage <i>string</i>
Example	PlsqlDefaultPage myschema.mypackage.home
Default	None

You can also use Oracle HTTP Server Rewrite rules to achieve the same effect as you get by setting this configuration parameter.

### E.6.2.14 PlsqlDocumentPath

Specifies a virtual path in the URL that initiates document download from 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 <i>string</i>
Example	PlsqlDocumentPath docs
Default	docs

Omit this parameter for applications that do not perform document uploads or downloads.

### E.6.2.15 PlsqlDocumentProcedure

Specifies the procedure to call when a document download is initiated. This procedure is called to process the download.

Category	Value
Syntax	PlsqlDocumentProcedure <i>string</i>
Example	PlsqlDocumentProcedure portal.wwdoc_process.process_download

Category	Value
Default	None

Omit this parameter for applications that do not perform document uploads or downloads.

### E.6.2.16 PlsqlDocumentTablename

Specifies the table in the database to which all documents are uploaded.

Category	Value
Syntax	<code>PlsqlDocumentTablename <i>string</i></code>
Example	<code>PlsqlDocumentTablename myschema.document_table</code>
Default	None

Omit this parameter for applications that do not perform document uploads or downloads.

### E.6.2.17 PlsqlErrorStyle

Specifies the error reporting mode for mod\_plsql errors.

Category	Value
Syntax	<code>PlsqlErrorStyle {ApacheStyle   ModplsqlStyle   DebugStyle}</code> <ul style="list-style-type: none"> <li>▪ <b>ApacheStyle:</b> The mod_plsql module 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 <code>ErrorDocument</code> directive to produce customized error messages.</li> <li>▪ <b>ModplsqlStyle:</b> The mod_plsql module generates the error pages, usually a short message indicating the PL/SQL error encountered and PL/SQL exception stack, if any. For example: <code>scott.foo PROCEDURE NOT FOUND</code></li> <li>▪ <b>DebugStyle:</b> This mode provides more details than <code>ModplsqlStyle</code>. The mod_plsql module provides more details about the URL and 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.</li> </ul>
Example	<code>PlsqlErrorStyle ModplsqlStyle</code>
Default	<code>ApacheStyle</code>

### E.6.2.18 PlsqlExclusionList

Specifies a pattern for procedures, packages, or schema names which are forbidden to be directly run from a browser. This is a multi-line directive in which each pattern is on a separate line. The pattern is not case sensitive and can accept a wildcard such as an asterisk (\*). The default patterns disallowed from direct URL access are as follows:

- `sys.*`
- `dbms_*`

- utl\_\*
- owa\_util\*
- owa.\*
- http.\*
- htf.\*
- wpg\_docload.\*

Setting this directive to #NONE# will disable all protection. This is strongly discouraged for an active site and should not be done. It may be 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}
Example	<pre>PlsqlExclusionList myschema.private.* PlsqlExclusionList myschema.private1.*</pre> <p>will disallow access to URLs which contain one of:</p> <pre>sys.*,dbms_*,utl_*,owa_util*, owa.*, http.*,htf.*, wpg_docload.*,myschema.private.*, myschema.private1.*</pre> <pre>PlsqlExclusionList "#NONE#"</pre> <p>will disable all protection. Its use is strongly discouraged for an active site.</p>
Default	<pre>sys.* dbms_* utl_* owa_util* owa.* http.* htf.* wpg_docload.*</pre>

- In addition to the patterns specified with this parameter, the mod\_plsql module disallows any procedure name which contains the following special characters:
  - tabs
  - new lines
  - carriage-return
  - single quotation mark
  - reverse slash
  - form feed
  - left parenthesis
  - right parenthesis
  - space

This cannot be changed.

### E.6.2.19 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, the `mod_plsql` module 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  $255 \times 200 = 51000$  bytes for each round trip. 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. The memory usage for the `mod_plsql` module will increase.

Category	Value
Syntax	<code>PlsqlFetchBufferSize number</code>
Example	<code>PlsqlFetchBufferSize 256</code>
Default	200

- 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 the `mod_plsql` module 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.

### E.6.2.20 PlsqlInfoLogging

Specifies what mode the `mod_plsql` module should use to do extra performance logging.

InfoDebug mode: 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	<code>PlsqlInfoLogging InfoDebug</code>
Example	<code>PlsqlInfoLogging InfoDebug</code>
Default	Empty

The logging setting is useful for debugging problems in your PL/SQL application.



### E.6.2.21 PlsqlMaxRequestsPerSession

Specifies the maximum number of requests a pooled database connection should service before it is closed and re-opened.

Category	Value
Syntax	PlsqlMaxRequestsPerSession <i>number</i>
Example	PlsqlMaxRequestsPerSession 500
Default	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.

### E.6.2.22 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 the mod\_plsql module.

Category	Value
Syntax	PlsqlNLSLanguage <i>string</i>
Example	PlsqlNLSLanguage America_America.UTF8
Default	None

- 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 the mod\_plsql module 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 module character set.
- If PlsqlTransferMode is set to RAW, then this parameter can be ignored.

### E.6.2.23 PlsqlPathAlias

Specifies a virtual path alias to map to a procedure call. This is application-specific. This directive is used with [PlsqlPathAliasProcedure](#).

Category	Value
Syntax	PlsqlPathAlias <i>string</i>
Example	PlsqlPathAlias url
Default	None

For applications that do not use path aliasing, this parameter may be omitted.

#### E.6.2.24 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 <i>string</i>
Example	PlsqlPathAliasProcedure portal.wwpth_api_alias.process_download
Default	None

For applications that do not use path aliasing, this parameter may be omitted.

#### E.6.2.25 PlsqlRequestValidationFunction

Specifies an application-defined PL/SQL function which gives you the opportunity to allow and disallow further processing of the requested procedure. This is useful in implementing tight security for your PL/SQL application by blocking out package and procedure calls that should not be allowed to run from a DAD.

The function defined by this parameter must have the following prototype:

```
boolean function_name (procedure_name IN varchar2)
```

The *procedure\_name* parameter will contain the name of the procedure that the request is trying to run.

For example, if all the PL/SQL application procedures callable from a browser are inside the package mypkg, then an implementation of this function can be as follows:

```
boolean my_validation_check (procedure_name varchar2)
is
begin
  if (upper (procedure_name) like upper ('myschema.mypkg%')) then
    return TRUE
  else
    return FALSE
  end if;
end;
```

Category	Value
Syntax	PlsqlRequestValidationFunction <i>string</i>
Example	PlsqlRequestValidationFunction myschema.mypkg.my_validation_check
Default	none

- By default, the mod\_plsql module already disallows direct URL access to certain schemas and 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 optimized 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 previous example.
  - If your implementation performs a table lookup to determine what packages and procedures should be allowed, then performance can be improved if you pin the cursor in the shared pool.

### E.6.2.26 PlsqlSessionCookieName

Specifies the cookie name when [PlsqlAuthenticationMode](#) is set to SingleSignOn. This parameter is supported only for Oracle Fusion Middleware releases, and is used by Oracle Portal and Oracle Single Sign-On.

Category	Value
Syntax	PlsqlSessionCookieName <i>cookie_name</i>
Example	PlsqlSessionCookieName mycookie
Default	Same as DAD name

- 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 Portal instances that need to participate in a distributed Oracle Portal environment. For those Oracle Portal nodes you want to seamlessly participate as a federated cluster, ensure that the session cookie name for all the participating nodes is the same.
- Independent Oracle Portal nodes need to use distinct session cookie names.

### E.6.2.27 PlsqlSessionStateManagement

Specifies how package and session state should be cleaned up at the end of each the mod\_plsql request.

- StatelessWithResetPackageState causes the mod\_plsql module to call `dbms_session.reset_package_state` at the end of each mod\_plsql request. This is the default.
- StatelessWithPreservePackageState causes the mod\_plsql module to call `http.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.
- StatelessWithFastResetPackageState causes the mod\_plsql module to call `dbms_session.modify_package_state(dbms_session.reinitialize)` at the end of each mod\_plsql request. This API is faster than the mode of `StatelessWithResetPackageState`, and avoids some latch contention issues, but exists only in Oracle database releases 8.1.7.2 and later. This mode uses slightly more memory than the default mode.

Category	Value
Syntax	PlsqlSessionStateManagement {StatelessWithResetPackageState   StatelessWithFastResetPackageState   StatelessWithPreservePackageState}
Example	PlsqlSessionStateManagement StatelessWithPreservePackageState
Default	StatelessWithResetPackageState

- The earlier values of `stateful=no` or `stateful=STATELESS_RESET` corresponds to `StatelessWithResetPackageState`.
- The earlier value of `stateful=STATELESS_FAST_RESET` corresponds to `StatelessWithFastResetPackageState`.
- The earlier value of `stateful=STATELESS_PRESERVE` corresponds to `StatelessWithPreservePackageState`.

The `mod_plsql` module does not support stateful mode of operation. To allow PL/SQL applications stateful behavior, save the state in cookies and/or in the database.

#### E.6.2.28 PlsqlTransferMode

Specifies the transfer mode for data from the database back to the `mod_plsql` module. Most applications use the default value of `CHAR`.

Category	Value
Syntax	PlsqlTransferMode {CHAR   RAW}
Example	PlsqlTransferMode CHAR
Default	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.

#### E.6.2.29 PlsqlUploadAsLongRaw

Specifies the file 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 asterisk (\*) in this field causes all documents to be uploaded as `LONGRAW`.

Category	Value
Syntax	PlsqlUploadAsLongRaw <i>string multiline</i>
Example	PlsqlUploadAsLongRaw jpg PlsqlUploadAsLongRaw gif
Default	None

For applications that do not upload or download documents, this parameter may be omitted.

## E.6.3 cache.conf

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

The following parameters are specified in the `cache.conf` file:

- [PlsqlCacheCleanupTime](#)
- [PlsqlCacheDirectory](#)
- [PlsqlCacheEnable](#)
- [PlsqlCacheMaxAge](#)
- [PlsqlCacheMaxSize](#)
- [PlsqlCacheTotalSize](#)

### E.6.3.1 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 every day at the time defined. For example, `Everyday 2:00` causes the cleanup to happen everyday at 2:00 a.m. (local time).
- To define weekly frequency, the days of the week, (`Sunday`, `Monday`, `Tuesday`, `Wednesday`, `Thursday`, `Friday`, `Saturday`) are used. For example, `Wednesday 15:30` causes the cleanup to happen every Wednesday at 3:30 p.m. (local time).
- To define monthly frequency, the keyword `Everymonth` is used. The cleanup starts on the Saturday of the month at the time defined. For example, `Saturday Everymonth 23:00` causes the cleanup to happen the first Saturday of every month at 11:00 p.m. (local time).

Category	Value
Syntax	<code>PlsqlCacheCleanupTime {Sunday-Saturday   Everyday   Everymonth} {hh:mm}</code>
Example	<code>PlsqlCacheCleanupTime Monday 20:00</code>
Default	<code>Saturday 23:00</code>

### E.6.3.2 PlsqlCacheDirectory

Specifies the directory where cache files are written out by the `mod_plsql` module. This directory must exist or Oracle HTTP Server will not start.

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

Category	Value
Syntax	<code>PlsqlCacheDirectory <i>directory</i></code>
Example	<code>PlsqlCacheDirectory ORACLE_INSTANCE/OHS/component_name</code>

Category	Value
Default	none

### E.6.3.3 PlsqlCacheEnable

Enables mod\_plsql caching.

Category	Value
Syntax	PlsqlCacheEnable {On   Off}
Example	PlsqlCacheEnable On
Default	Off

If an application does not make use of the OWA\_CACHE package in the PL/SQL Web Toolkit, then you can choose to disable caching. In such situations, there will be a minor performance benefit.

### E.6.3.4 PlsqlCacheMaxAge

Specifies the maximum time, in days, a cache file can 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 <i>number</i>
Example	PlsqlCacheMaxAge 20
Default	30 (days)

### E.6.3.5 PlsqlCacheMaxSize

Specifies the maximum possible size of a cache file.

This setting prevents 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, which is specified by [PlsqlCacheTotalSize](#).

Category	Value
Syntax	PlsqlCacheMaxSize <i>number</i>
Example	PlsqlCacheMaxSize 1048576
Default	1048576

### E.6.3.6 PlsqlCacheTotalSize

Specifies the total size of the cache directory. The default is 20 MB.

This setting limits the amount of space the cache is allowed to use. Both PL/SQL cache and Session Cookie cache share this cache space. 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 <i>number</i>
Example	PlsqlCacheTotalSize 20971520
Default	20971520 (bytes)

## E.7 mod\_wl\_ohs

mod\_wl\_ohs accepts the following directives:

- [ConnectRetrySecs](#)
- [ConnectTimeoutSecs](#)
- [Debug](#)
- [DebugConfigInfo](#)
- [DefaultFileName](#)
- [DynamicServerList](#)
- [ErrorPage](#)
- [FileCaching](#)
- [Idempotent](#)
- [KeepAliveEnabled](#)
- [KeepAliveSecs](#)
- [MatchExpression](#)
- [MaxPostSize](#)
- [MaxSkipTime](#)
- [PathPrepend](#)
- [PathTrim](#)
- [QueryFromRequest](#)
- [WebLogicCluster](#)
- [WebLogicHost](#)
- [WebLogicPort](#)
- [WebLogicSSLVersion](#)
- [WLCookieName](#)
- [WLDNSRefreshInterval](#)
- [WLExcludePathOrMimeType](#)
- [WLForwardUriUnparsed](#)
- [WLIOTimeoutSecs](#)
- [WLocalIP](#)
- [WLogFile](#)
- [WLProxyPassThrough](#)
- [WLProxySSL](#)
- [WLProxySSLPassThrough](#)
- [WLServerInitiatedFailover](#)
- [WLSocketTimeoutSecs](#)
- [WLSRequest](#)
- [WTempDir](#)

---

**Note:** mod\_wl\_ohs directives are the same as those accepted by mod\_wl.

---

### E.7.1 ConnectRetrySecs

Interval in seconds that the plug-in should sleep between attempts to connect to the WebLogic Server host (or all of the servers in a cluster). Make this number less than the [ConnectTimeoutSecs](#). The number of times the plug-in tries to connect before returning an HTTP 503/Service Unavailable response to the client is calculated by dividing ConnectTimeoutSecs by ConnectRetrySecs.

To specify no retries, set `ConnectRetrySecs` equal to `ConnectTimeoutSecs`. However, the plug-in attempts to connect at least twice.

You can customize the error response by using the [ErrorPage](#) directive.

Category	Value
Syntax	<code>ConnectRetrySecs secs</code>
Example	<code>ConnectRetrySecs 10</code>
Default	2

## E.7.2 ConnectTimeoutSecs

Maximum time in seconds that the plug-in should attempt to connect to the WebLogic Server host. Make the value greater than `ConnectRetrySecs`. If `ConnectTimeoutSecs` expires without a successful connection, even after the appropriate retries (see `ConnectRetrySecs`), an HTTP 503/Service Unavailable response is sent to the client.

You can customize the error response by using the [ErrorPage](#) directive.

Category	Value
Syntax	<code>ConnectTimeoutSecs secs</code>
Example	<code>ConnectTimeoutSecs 120</code>
Default	10

## E.7.3 Debug

Sets the type of logging performed for debugging operations. The debugging information is written to the `/tmp/wlproxy.log` file on UNIX systems and `c:\TEMP\wlproxy.log` on Windows NT/2000 systems.

Override this location and filename by setting the [WLLogFile](#) parameter to a different directory and file. (See the `WLTempDir` parameter for an additional way to change this location.)

Ensure that the `tmp` or `TEMP` directory has write permission assigned to the user who is logged in to the server. Set any of the following logging options (`HFC`, `HTW`, `HFW`, and `HTC` options may be set in combination by entering them separated by commas, for example `"HFC,HTW"`):

- `ON` - The plug-in logs informational and error messages.
- `OFF` - No debugging information is logged.
- `HFC` - The plug-in logs headers from the client, informational, and error messages.
- `HTW` - The plug-in logs headers sent to WebLogic Server, and informational and error messages.
- `HFW` - The plug-in logs headers sent from WebLogic Server, and informational and error messages.
- `HTC` - The plug-in logs headers sent to the client, informational messages, and error messages.
- `ERR` - Prints only the Error messages in the plug-in.



- ALL - The plug-in logs headers sent to and from the client, headers sent to and from WebLogic Server, information messages, and error messages.

Category	Value
Syntax	Debug ON   OFF   HFC   HTW   HFW   HTC   ERR   ALL
Example	Debug HFC, HTW, HTC
Default	OFF

### E.7.4 DebugConfigInfo

Enables the special query parameter “\_\_WebLogicBridgeConfig”. Use it to get details about configuration parameters from the plug-in.

For example, if you enable “\_\_WebLogicBridgeConfig” by setting DebugConfigInfo and then send a request that includes the query string ?\_\_WebLogicBridgeConfig, then the plug-in gathers the configuration information and run-time statistics and returns the information to the browser. The plug-in does not connect to WebLogic Server in this case.

This parameter is strictly for debugging and the format of the output message can change with releases. For security purposes, keep this parameter turned OFF in production systems.

Category	Value
Syntax	DebugConfigInfo OFF   ON
Example	DebugConfigInfo ON
Default	OFF

### E.7.5 DefaultFileName

Sets the default welcome page of the Web Application in WebLogic Server to which requests are being proxied.

If the URI is “/” then the plug-in performs the following steps:

1. Trims the path specified with the PathTrim parameter.
2. Appends the value of DefaultFileName.
3. Prepends the value specified with PathPrepend.

This procedure prevents redirects from WebLogic Server. For example, If the DefaultFileName is set to welcome.html, an HTTP request like “http://somehost/weblogic” becomes “http://somehost/weblogic/welcome.html”. For this parameter to function, the same file must be specified as a welcome file in all the Web Applications to which requests are directed. For more information, see Configuring Welcome Pages.

**Apache users:** If you are using Stronghold or Raven versions, define this parameter inside of a Location block, and not in an IfModule block.Oracle

Category	Value
Syntax	DefaultFileName <i>"http://path/to/file"</i>
Example	DefaultFileName <i>"http://somehost/weblogic"</i>
Default	none

### E.7.6 DynamicServerList

When set to OFF, the plug-in ignores the dynamic cluster list used for load balancing requests proxied from the plug-in and only uses the static list specified with the WebLogicCluster parameter. Normally this parameter should remain set to ON.

There are some implications for setting this parameter to OFF:

- If one or more servers in the static list fails, the plug-in could waste time trying to connect to a dead server, resulting in decreased performance.
- If you add a new server to the cluster, the plug-in cannot proxy requests to the new server unless you redefine this parameter. WebLogic Server automatically adds new servers to the dynamic server list when they become part of the cluster.

Category	Value
Syntax	DynamicServerList ON   OFF
Example	DynamicServerList OFF
Default	ON

### E.7.7 ErrorPage

Sets the error page that appears whenever your Web server is unable to forward requests to WebLogic Server. You can create your own error page. The plug-in redirects to an error page when the back-end server returns an HTTP 503/Service Unavailable response and there are no servers for failover.

Category	Value
Syntax	ErrorPage
Example	ErrorPage
Default	none

### E.7.8 FileCaching

Specifies whether or not file caching is enabled. When set to ON, and the size of the POST data in a request is greater than 2048 bytes, the POST data is first read into a temporary file on disk and then forwarded to the WebLogic Server in chunks of 8192 bytes. This preserves the POST data during failover, allowing all necessary data to be repeated to the secondary if the primary goes down.

Note that when FileCaching is ON, any client that tracks the progress of the POST will see that the transfer has completed even though the data is still being transferred between the WebServer and WebLogic. So, if you want the progress bar displayed by a browser during the upload to reflect when the data is actually available on the WebLogic Server, you might not want to have FileCaching ON.

When set to OFF and the size of the POST data in a request is greater than 2048 bytes, the reading of the POST data is postponed until a WebLogic Server cluster member is identified to serve the request. Then the plug-in reads and immediately sends the POST data to the WebLogic Server in chunks of 8192 bytes.

Note that turning FileCaching OFF limits failover. If the WebLogic Server primary server goes down while processing the request, the POST data already sent to the primary cannot be repeated to the secondary.

Finally, regardless of how FileCaching is set, if the size of the POST data is 2048 bytes or less the plug-in will read the data into memory and use it if needed during failover to repeat to the secondary.

Category	Value
Syntax	FileCaching ON   OFF
Example	FileCaching OFF
Default	ON

### E.7.9 Idempotent

Specifies whether or not plug-ins fail over.

- When set to ON and if the servers do not respond within WLIOTimeoutSecs, the plug-ins fail over if the method is idempotent.
- The plug-ins also fail over if Idempotent is set to ON and the servers respond with an error such as READ\_ERROR\_FROM\_SERVER.
- If set to OFF the plug-ins do not fail over. If you are using the Apache HTTP Server you can set this parameter differently for different URLs or MIME types.

Category	Value
Syntax	Idempotent ON   OFF
Example	Idempotent OFF
Default	ON

### E.7.10 KeepAliveEnabled

Enables pooling of connections between the plug-in and WebLogic Server.

If you are using Apache prefork mpm, Apache web server might crash. Turn KeepAliveEnabled to OFF when using prefork mpm or use worker mpm in Apache.

Category	Value
Syntax	KeepAliveEnabled ON(TRUE)   OFF(FALSE)
Example	KeepAliveEnabled OFF
Default	ON(TRUE)

### E.7.11 KeepAliveSecs

Specifies how long to maintain an inactive connection between the plug-in and WebLogic Server. Once the time set here has elapsed, the connection is closed. You

must set `KeepAliveEnabled` to true (ON when using the Apache HTTP Server) for this parameter to be effective.

The value of this parameter must be less than or equal to the value of the `Duration` field set in the Administration Console on the Server/HTTP tab, or the value set on the server Mbean with the `KeepAliveSecs` attribute.

Category	Value
Syntax	<code>KeepAliveSecs secs</code>
Example	<code>KeepAliveSecs 120</code>
Default	20

## E.7.12 MatchExpression

Specifies the filename pattern to use when proxying. This directive is entered inside of an `IfModule` block. For example, if you wanted to proxy by MIME type, you would use `MatchExpression` as shown here:

```
<IfModule weblogic_module>
  MatchExpression *.jsp
  WebLogicHost=myHost|paramName=value
</IfModule>
```

Or, if you wanted to proxy by path, you would use `MatchExpression` like this:

```
<IfModule weblogic_module>
  MatchExpression /weblogic
  WebLogicHost=myHost|paramName=value
</IfModule>
```

You can define a new parameter for `MatchExpression` by using the following syntax:

```
MatchExpression *.jsp PathPrepend=/test PathTrim=/foo
```

Category	Value
Syntax	<code>MatchExpression expression</code>
Example	<code>MatchExpression /weblogic</code>
Default	none

## E.7.13 MaxPostSize

Maximum allowable size of POST data, in bytes. If the content-length exceeds `MaxPostSize`, the plug-in returns an error message. If set to -1, the size of POST data is not checked. This is useful for preventing denial-of-service attacks that attempt to overload the server with POST data.

Category	Value
Syntax	<code>MaxPostSize nn</code>
Example	<code>MaxPostSize 64</code>
Default	-1

### E.7.14 MaxSkipTime

If a WebLogic Server listed in either the WebLogicCluster parameter or a dynamic cluster list returned from WebLogic Server fails, the failed server is marked as “bad” and the plug-in attempts to connect to the next server in the list.

MaxSkipTime sets the amount of time after which the plug-in will retry the server marked as “bad.” The plug-in attempts to connect to a new server in the list each time a unique request is received (that is, a request without a cookie).

Category	Value
Syntax	MaxSkipTime <i>sec</i>
Example	MaxSkipTime 120
Default	10

### E.7.15 PathPrepend

Specifies the path that the plug-in prepends to the {PATH} portion of the original URL, after PathTrim is trimmed and before the request is forwarded to WebLogic Server.

Note that if you need to append File Name, use [DefaultFileName](#) parameter instead of PathPrepend.

Category	Value
Syntax	PathPrepend [PROTOCOL]://[HOSTNAME]:{PORT}/{PATH}/{FILENAME};{PATH_PARAMS}/{QUERY_STRING}
Example	PathPrepend http://myHost:1234/myPath/myfile;
Default	null

### E.7.16 PathTrim

PathTrim specifies the string trimmed by the plug-in from the {PATH}/{FILENAME} portion of the original URL, before the request is forwarded to WebLogic Server. For example, if the URL

http://myWeb.server.com/weblogic/foo

is passed to the plug-in for parsing and if PathTrim has been set to strip off /weblogic before handing the URL to WebLogic Server, the URL forwarded to WebLogic Server is:

http://myWeb.server.com:7001/foo

Note that if you are newly converting an existing third-party server to proxy requests to WebLogic Server using the plug-in, you will need to change application paths to /foo to include weblogic/foo. You can use PathTrim and PathPrepend in combination to change this path.

Category	Value
Syntax	PathTrim [PROTOCOL]://[HOSTNAME]:{PORT}/{PATH}/{FILENAME};{PATH_PARAMS}/{QUERY_STRING}
Example	PathTrim http://myWeb.server.com/weblogic/foo

Category	Value
Default	null

### E.7.17 QueryFromRequest

When set to ON, specifies that the Apache HTTP Server use

```
(request_rec *)r->the_request
```

to pass the query string to WebLogic Server. (For more information, see the Apache documentation.) This behavior is useful when a Netscape version 4.x browser makes requests that contain spaces in the query string

When set to OFF, the Apache HTTP Server uses (request\_rec \*)r->args to pass the query string to WebLogic Server.

Category	Value
Syntax	QueryFromRequest ON   OFF
Example	QueryFromRequest ON
Default	OFF

### E.7.18 WebLogicCluster

Specifies the list of WebLogic Servers that can be used for load balancing. The server or cluster list is a list of host:port entries. If a mixed set of clusters and single servers is specified, the dynamic list returned for this parameter will return only the clustered servers. This derivative is required to proxy a list of clustered back-end servers or to perform load balancing among non-clustered managed server instances.

If you are using SSL between the plug-in and WebLogic Server, set the port number to the SSL listen port and set the SecureProxy parameter to ON.

The plug-in does a simple round-robin between all available servers. The server list specified in this property is a starting point for the dynamic server list that the server and plug-in maintain. WebLogic Server and the plug-in work together to update the server list automatically with new, failed, and recovered cluster members.

You can disable the use of the dynamic cluster list by setting the [DynamicServerList](#) parameter to OFF.

The plug-in directs HTTP requests containing a cookie, URL-encoded session, or a session stored in the POST data to the server in the cluster that originally created the cookie.

Category	Value
Syntax	The syntax for specifying the value of this parameter varies depending on the web server for which you are configuring the plug-in, as described in <i>Using Web Server 1.1 Plug-Ins with Oracle WebLogic Server</i> . For more information, see the following: <ul style="list-style-type: none"> <li>■ Chapter 2, "Configuring the mod_wl_ohs Plug-In for Oracle HTTP Server"</li> <li>■ Chapter 3, "Installing and Configuring the Oracle iPlanet Web Server Plug-In"</li> <li>■ Chapter 4, "Installing and Configuring the Apache HTTP Server Plug-In"</li> <li>■ Chapter 5, "Installing and Configuring the Microsoft IIS Plug-In"</li> </ul>
Example	WebLogicCluster w1s1.com:7001,w1s2.com:7001,w1s3.com:7001
Default	none

### E.7.19 WebLogicHost

Specifies the WebLogic Server host (or virtual host name as defined in WebLogic Server) to which HTTP requests should be forwarded. If you are using a WebLogic cluster, use the WebLogicCluster parameter instead of WebLogicHost. This directive is required when proxying to a single WebLogic Server.

Category	Value
Syntax	WebLogicHost <i>hostname</i>
Example	WebLogicHost myHost
Default	none

### E.7.20 WebLogicPort

Specifies the port at which the WebLogic Server host is listening for connection requests from the plug-in (or from other servers). This directive is required when proxying to a single WebLogic Server.

- If you are using SSL between the plug-in and WebLogic Server, set this parameter to the SSL listen port and set the SecureProxy parameter to ON.
- If you are using a WebLogic Cluster, use the WebLogicCluster parameter instead of WebLogicPort.

Category	Value
Syntax	WebLogicPort <i>portNo</i>
Example	WebLogicPort 1234
Default	none

### E.7.21 WebLogicSSLVersion

Selects the SSL protocol version to use for communication between the plug-in and the WebLogic Server. The following values are accepted:

- SSLv3 : Uses SSL v3
- TLSv1 : Uses TLS v1

The SSL protocol version chosen is used for all the connections from the plug-in to WebLogic Server. Hence define this parameter at the global scope.

Category	Value
Syntax	WebLogicSSLVersion <i>versionNumber</i>
Example	WebLogicSSLVersion SSLv3
Default	If not configured, the best protocol supported by both the plug-in and WebLogic Server is used.

## E.7.22 WLCookieName

---



---

**Note:** WLCookieName replaces CookieName, which is deprecated.

---



---

If you change the name of the WebLogic Server session cookie in the WebLogic Server Web application, you need to change this parameter in the plug-in to the same value. The name of the WebLogic session cookie is set in the WebLogic-specific deployment descriptor, in the <session-descriptor> element.

Category	Value
Syntax	WLCookieName <i>cookieName</i>
Example	WLCookieName
Default	JSESSIONID

## E.7.23 WLDNSRefreshInterval

If defined in the proxy configuration, this directive specifies the interval duration, in seconds, at which WebLogic Server refreshes DNS name to IP mapping for a server. This can be used in the event that a WebLogic Server instance is migrated to a different IP address, but the DNS name for that server's IP remains the same. In this case, at the specified refresh interval the bi-directional DNS-to-IP mapping will be updated.

Category	Value
Syntax	WLDNSRefreshInterval <i>nn</i>
Example	WLDNSRefreshInterval 10
Default	Lookup once, during startup

## E.7.24 WExcludePathOrMimeType

This parameter allows you to exclude certain requests from proxying.

This parameter can be defined locally at the Location tag level as well as globally. When the property is defined locally, it does not override the global property but defines a union of the two parameters.



Category	Value
Syntax	WLEXcludePathOrMimeType <i>value</i>
Example	WLEXcludePathOrMimeType
Default	none

### E.7.25 WLForwardUriUnparsed

When set to ON, the WLS plug-in will forward the original URI from the client to WebLogic Server. When set to OFF (default), the URI sent to WebLogic Server is subject to modification by mod\_rewrite or other web server plug-in modules.

Category	Value
Syntax	WLForwardUriUnparsed ON   OFF
Example	WLForwardUriUnparsed ON
Default	OFF

### E.7.26 WLIOTimeoutSecs

---

**Note:** WLIOTimeoutSecs is the new name for HungServerRecoverSecs.

---

Defines the amount of time the plug-in waits for a response to a request from WebLogic Server. The plug-in waits for WLIOTimeoutSecs for the server to respond and then declares that server dead, and fails over to the next server. The value should be set to a very large value. If the value is less than the time the servlets take to process, then you may see unexpected results.

- Minimum value: 10
- Maximum value: Unlimited

Category	Value
Syntax	WLIOTimeoutSecs <i>nn</i>
Example	WLIOTimeoutSecs 128
Default	300

### E.7.27 WLLocalIP

Defines the IP address (on the plug-in's system) to which to bind when the plug-in connects to a WebLogic Server instance running on a multihomed machine.

If WLLocalIP is not set, the TCP/IP stack will choose the source IP address.

Category	Value
Syntax	WLLocalIP <i>ipAddress</i>
Example	WLLocalIP 1001.1111.1234.4567

Category	Value
Default	none

### E.7.28 WLogFile

Specifies path and file name for the log file that is generated when the [Debug](#) parameter is set to ON. You must create this directory before setting this parameter.

Category	Value
Syntax	WLogFile <i>path/to/file.ext</i>
Example	WLogFile <code>http://myFolder/mySubfolder/myLogfile.log</code>
Default	

### E.7.29 WLProxyPassThrough

If you have a chained proxy setup, where a proxy plug-in or `HttpClusterServlet` is running behind some other proxy or load balancer, you must explicitly enable the `WLProxyPassThrough` parameter. This parameter allows the header to be passed through the chain of proxies.

Category	Value
Syntax	WLProxyPassThrough ON   OFF
Example	<i>WLProxyPassThrough ON</i>
Default	OFF

### E.7.30 WLProxySSL

Specifies whether or not to maintain SSL communication between the plug-in and WebLogic Server when the following conditions exist:

- An HTTP client request specifies the HTTPS protocol.
- The request is passed through one or more proxy servers (including the WebLogic Server proxy plug-ins).
- The connection between the plug-in and WebLogic Server uses the HTTP protocol.
- When `WLProxySSL` is set to ON, the location header returned to the client from WebLogic Server specifies the HTTPS protocol.

Category	Value
Syntax	WLProxySSL ON   OFF
Example	WLProxySSL ON
Default	OFF

### E.7.31 WLProxySSLPassThrough

If a load balancer or other software deployed in front of the web server and plug-in is the SSL termination point, and that product sets the `WL-Proxy-SSL` request header to true or false based on whether or not the client connected to it over SSL, set

WLProxySSLPassThrough to ON so that the use of SSL is passed on to the Oracle WebLogic Server.

If the SSL termination point is in the web server where the plug-in operates, or the load balancer does not set WL-Proxy-SSL, set WLProxySSLPassThrough to OFF (default).

Category	Value
Syntax	WLProxySSLPassThrough ON   OFF
Example	WLProxySSLPassThrough ON
Default	OFF

### E.7.32 WLServerInitiatedFailover

Specifies whether or not a 503 error response from Oracle WebLogic Server triggers a failover to another server. Normally, the plug-in will attempt to failover to another server when a 503 error response is received. When WLServerInitiatedFailover is set to OFF, the 503 error response will be returned to the client immediately.

Category	Value
Syntax	WLServerInitiatedFailover ON   OFF
Example	WLServerInitiatedFailover OFF
Default	ON

### E.7.33 WLSocketTimeoutSecs

Sets the timeout, in seconds, for the socket while connecting. See [ConnectTimeoutSecs](#) and [ConnectRetrySecs](#) for additional details. This value must be greater than 0

Category	Value
Syntax	WLSocketTimeoutSecs <i>nn</i>
Example	WLSocketTimeoutSecs 10
Default	2

### E.7.34 WLSRequest

This is an alternative to the SetHandler weblogic-handler mechanism of identifying requests to be forwarded to Oracle WebLogic Server. For example,

```
<Location /weblogic>
  WLSRequest ON
  PathTrim /weblogic
</Location>
```

The use of WLSRequest ON instead of SetHandler weblogic-handler has the following advantages:

- Lower web server processing overhead in general
- Resolves substantial performance degradation when the web server DocumentRoot is on a slow file system

- Resolves 403 errors for URIs which cannot be mapped to the file system due to the file system length restrictions

Category	Value
Syntax	WLSRequest ON   OFF
Example	WLSRequest ON
Default	OFF

### E.7.35 WLTempDir

Specifies:

- The directory where a wlproxy.log will be created. If the location fails, the Plug-In resorts to creating the log file under C:/temp in Windows and /tmp in all Unix platforms.
- The location of the \_wl\_proxy directory for POST data files.

When both WLTempDir and [WLogFile](#) are set, WLogFile will override as to the location of wlproxy.log. WLTempDir will still determine the location of \_wl\_proxy directory.

Category	Value
Syntax	WLTempDir <i>dirName</i>
Example	WLTempDir MyLogDir
Default	See the <a href="#">Debug</a> parameter.

---

---

# Glossary

## **Apache**

Apache HTTP Server is an open source web server originally derived from the National Center for Supercomputing Applications (NCSA).

## **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.

## **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. It also 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. Ciphertext is unreadable until it has been converted to plain text (decrypted) with a key. See [decryption](#).

**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.

**decryption**

The process of converting the contents of an encrypted message ([ciphertext](#)) back into its original readable format ([plaintext](#)).

**digital certificate**

See [certificate](#).

**digital wallet**

See [wallet](#).

**encryption**

The process of converting a message thereby rendering it unreadable to any but the intended recipient. Encryption is performed by converting data into code that cannot be understood by unauthorized people or systems. There are two main types of encryption: [public-key encryption](#) (also known as 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.

**Fusion Middleware Control**

See [Oracle Enterprise Manager Fusion Middleware Control](#).

**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 Fusion Middleware 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.

**modules**

Modules extend the basic functionality of a Web server, and support integration between Oracle HTTP Server and other Oracle Fusion Middleware components.

**Oracle Enterprise Manager Fusion Middleware Control**

Oracle Enterprise Manager Fusion Middleware Control (Fusion Middleware Control) provides Web-based management tools designed specifically for Oracle Fusion Middleware. Using Fusion Middleware Control, you can monitor and configure the components of your application server, such as deploy applications, manage security, and create and manage Oracle Fusion Middleware clusters.

**PEM**

Privacy-enhanced Electronic Mail. An [encryption](#) technique that provides encryption, authentication, message integrity, and key management.

**PL/SQL**

PL/SQL is the Oracle 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 Proxy Plug-in or Oracle SSO 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 resides 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 systems, 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.



**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](#).

**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 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**

A standard for creating digital certificates.



---

---

# Index

## A

---

access log, 7-2  
accessing  
    Fusion Middleware Control, 2-1  
All16UTF-16, 3-7  
Apache, Glossary-1  
    security patches, B-2  
    version, 1-1  
Apache HTTP Server, 1-1  
Apache OraDAV, 9-2  
apachectl, 4-1  
ApacheStyle, E-36  
application-specific error pages, A-1, B-1  
authentication, 8-1, Glossary-1  
authorization, 8-1  
availability, Glossary-1

## B

---

browsing database content  
    with OraDAV, 9-4

## C

---

cache, B-2  
cache.conf, 3-11, E-43  
caching  
    disk  
        OraDAV and, 9-7  
certificate, Glossary-1  
    digital, Glossary-2  
    X.509, E-15  
certificate authority, Glossary-1  
certificate revocation list, E-10  
CGI, Glossary-2  
ciphertext, Glossary-2  
cleartext, Glossary-2  
CompatEnvVars, E-15  
confidentiality, 8-1  
configuration files  
    cache.conf, 3-11, E-43  
    dads.conf, 3-11, E-25  
    oracle\_osso.dll, A-19, A-22  
    osso\_plugin.conf, A-18  
    plsqli.conf, 3-10

    syntax, 1-5  
configuring  
    mod\_oradav, 9-1  
    Sun Java System  
        proxy plug-in, A-7  
creating  
    DAD, 3-8  
cryptography, Glossary-2

## D

---

DAD, Glossary-2  
    creating, 3-8  
    password  
        obfuscation, E-33  
dads.conf, 3-11, E-25  
dadTool.pl, E-33  
database access descriptor, 3-11, E-25, Glossary-2  
database connection  
    OraDAV and, E-5  
database usage notes, 3-6  
DAV, 9-1  
DAV parameter, 9-5  
DAVParam parameter  
    for OraDAV, 9-5  
DebugStyle, E-36  
decryption, Glossary-2  
Diffie-Hellman key negotiation algorithm, E-11  
digital certificate, Glossary-2  
digital wallet, Glossary-2  
directives  
    create name space, B-3  
    RewriteLogLevel, C-3  
    UseWebCacheIp, C-3  
directory structure, 1-5  
disk caching, E-4  
    OraDAV and, 9-7  
    size, E-4, E-5  
distinguished name, E-15  
distributed authoring and versioning, 9-1  
Dynamic Monitoring Service, E-23

## E

---

encryption, 1-3, Glossary-2  
entry, Glossary-2

- error log, C-3
  - mod\_oradav and, E-8, E-9
- events
  - OraDAV and, E-8
- exceptions
  - OraDAV and, E-6
  - ExportCertData, E-15

## F

---

- failover, Glossary-3
- FakeBasicAuth, E-15
- FAQ, A-1, B-1
  - Apache security patches, B-2
  - compressing
    - output, B-3
  - offering HTTPS to ISP customers, A-16, B-2
  - protecting Web site
    - hackers, B-4
- features, 1-1
- file system access
  - OraDAV and, 9-5
- frequently asked questions, A-1, B-1
- Fusion Middleware Control, Glossary-3
  - accessing, 2-1
  - managing, 2-1
    - Oracle HTTP Server, 2-2
  - Oracle HTTP Server Home page, 2-2

## G

---

- globalization support
  - OraDAV considerations, 9-8

## H

---

- hackers, B-4
- HTTP, Glossary-3
- HTTP listener, 1-1
- Hypertext Transfer Protocol, Glossary-3

## I

---

- identd, 7-2
- IdentityCheck, 7-2
- IIS
  - proxy plug-in, A-1
  - SSO plug-in, A-16
- InfoDebug, E-38
- IpCheck, A-18

## L

---

- LDAP, Glossary-3
- lightweight directory access protocol, Glossary-3
- listener addresses, 6-1
- listener ports, 6-1
- LoadModule directive, 3-10
- locking
  - OraDAV and, E-7
- log files, C-3

- locations, C-3
- log formats
  - authuser, 7-2
  - bytes, 7-3
  - Common Log Format, 7-2
  - data, 7-2
  - host, 7-2
  - ident, 7-2
  - request, 7-2
  - status, 7-3
- log rotation, 7-6
- LoginServerFile, A-18

## M

---

- managing
  - Fusion Middleware Control, 2-1
  - Oracle HTTP Server, 2-2
- mod\_certheaders, 3-2
- mod\_dav
  - OraDAV
    - disk caching, 9-7
    - Oracle Web Cache, 9-8
    - performance considerations, 9-7
  - usage notes
    - globalization support, 9-8
- mod\_dms, 3-3, 8-3
- mod\_onsint, 3-3
- mod\_oradav, 3-4, 9-1
  - concepts
    - OraDAV, 9-2
    - WebDAV, 9-2
  - error log, E-8, E-9
  - Get requests and, E-3
  - OraDAV
    - administration, 9-3
    - Apache OraDAV, 9-2
    - architecture, 9-3
    - configuration parameters, 9-5
    - OraDAV driver, 9-2
    - OraDAV driver API, 9-2
    - usage model, 9-4
- parameters
  - ORAAllowIndexDetails, E-3
  - ORAAltPassword, E-3
  - ORACacheDirectory, E-4
  - ORACacheMaxResourceSize, E-4
  - ORACachePrunePercent, E-4
  - ORACacheTotalSize, E-5
  - ORAConnect, E-5
  - ORAConnectSN, E-5
  - ORAContainerName, E-6
  - ORAException, E-6
  - ORAGetSource, E-6
  - ORALockExpirationPad, E-7
  - ORAPackageName, E-7
  - ORAPassword, E-7
  - ORARootPrefix, E-8
  - ORAService, E-8
  - ORATraceEvents, E-8

- ORATraceLevel, E-9
- ORAUser, E-9
- mod\_ossl, 3-4, 8-1
  - directives
    - SSLAccelerator, E-10
    - SSLCARevocationFile, E-10
    - SSLCARevocationPath, E-11
    - SSLCipherSuite, E-11
    - SSLEngine, E-13
    - SSLLog, E-13
    - SSLLogLevel, E-13
    - SSLMutex, E-14
    - SSLOptions, E-14
    - SSLPassPhraseDialog, E-15
    - SSLProtocol, E-16
    - SSLRequire, E-16
    - SSLRequireSSL, E-18
    - SSLSessionCache, E-18
    - SSLSessionCacheTimeout, E-19
    - SSLVerifyClient, E-19
    - SSLWallet, E-20
    - SSLWalletPassword, E-20
- mod\_osso, 3-5, 8-1, 8-4, A-16
- mod\_perl, 1-1, 3-5, 8-3
  - database usage notes, 3-6
  - testing database connection, 3-7
- mod\_plsql, 3-8
  - configuration files, 3-10, E-23
    - cache.conf, 3-11, E-43
    - dads.conf, 3-11, E-25
    - plsql.conf, 3-10
  - configuration parameters, 3-11
    - CustomOwa, E-27
    - PerPackageOwa, E-27
- mod\_reqtimeout, 3-2
- mod\_ssl, 3-4
- mod\_wl\_ohs, 8-3
- ModplsqlStyle, E-36
- modules, 1-1, Glossary-3
  - mod\_certheaders, 3-2
  - mod\_dms, 3-3
  - mod\_onsint, 3-3
  - mod\_oradav, 3-4
  - mod\_ossl, 3-4
  - mod\_osso, 3-5
  - mod\_perl, 3-5
  - mod\_plsql, 3-8
  - mod\_ssl, 3-4
- Multipurpose Internet Mail Extension, 4-9
- multiviews, B-2

## N

---

- nFast, E-10
- NLS\_LANG environment variable
  - OraDAV considerations, 9-8

## O

---

- oproxy.serverlist, A-3

- oproxy.servername.alias, A-4
- oproxy.servername.hostname, A-4
- oproxy.servername.port, A-4
- oproxy.servername.urlrule, A-5
  - matches
    - context, A-5
    - exact, A-5
    - suffix, A-5
- OptRenegotiate, E-15
- ORA\_IMPLICIT, 3-7
- ORA\_NCHAR, 3-7
- ORAAllowIndexDetails parameter, E-3
- ORAAltPassword parameter, E-3
- ORACacheDirectory parameter, 9-7, E-4
- ORACacheMaxResourceSize parameter, 9-7, E-4
- ORACachePrunePercent parameter, 9-7, E-4
- ORACacheTotalSize parameter, 9-7, E-5
- Oracle Enterprise Manager Application Server
  - Control, Glossary-3
- Oracle HTTP Server
  - cache, B-2
  - C/C++, 1-4
  - components
    - HTTP listener, 1-1
    - modules, 1-1
    - Perl interpreter, 1-1
  - compressing
    - output, B-3
  - configuration files syntax, 1-5
  - directory structure, 1-5
  - Distributed Authoring and Versioning
    - Support, 1-3
  - FAQ, A-1, B-1
  - features, 1-1
  - load balancing, 1-5
  - managing, 2-2
  - OPMN, 1-4
  - overview, 1-1
  - Perl, 1-4
  - PHP, 1-4
  - PL/SQL server pages, 1-4
  - process model
    - security considerations, 5-7
  - proxy plug-in, 1-4
  - restarting, 4-4
  - security, 1-3
  - server side include, 1-4
  - single sign-on, 1-3
  - sso plug-in, 1-4
  - starting, 4-3
  - stopping, 4-4
  - support, 1-6
  - URL rewriting and proxy server, 1-3
- Oracle HTTP Server Home page, 2-2
- Oracle Web Cache
  - OraDAV and, 9-8
  - WebDAV, 9-8
- oracle\_osso.dll, A-19, A-22
- oracle\_proxy\_sunone.dll, A-3
- oracle\_proxy.dll, A-3, A-18

- oracle\_proxy.so, A-3
- ORACONNECT parameter, E-5
- ORACONNECTSN parameter
  - database connection
  - OraDAV and, E-5
- ORACONTAINERNAME parameter, E-6
- OraDAV, 9-1, 9-2
  - administration, 9-3
  - description, 9-2
  - globalization support considerations, 9-8
  - usage model, 9-4
  - WebDAV
    - security considerations, 9-7
- OraDav, 3-4
- OraDAV and, E-4
- OraDAV configuration parameters, 9-5
- OraDAV driver, 9-2
- OraDAV driver API, 9-2
- OraDAV users, E-9
- ORAEXCEPTION parameter, E-6
- ORAGETSOURCE parameter, E-6
- ORALOCKEXPIRATIONPAD parameter, E-7
- ORAPACKAGENAME parameter, E-7
- ORAPASSWORD parameter, E-7
- ORAROOTPREFIX parameter, E-8
- ORASERVICE parameter, E-8
- ORATRACEEVENTS parameter, E-8
- ORATRACELEVEL parameter, E-9
- ORAUER parameter, E-9
- osso\_plugin.conf, A-18
- overview, 1-1

## P

---

- passwords
  - OraDAV and, E-3, E-7
- PEM, Glossary-3
- Perl
  - access database, 3-6
- Perl interpreter, 1-1
- PID file, 4-2
- plaintext, Glossary-3
- PL/SQL, Glossary-3
- plsql.conf, 3-10
- PlsqlErrorStyle
  - ApacheStyle, E-36
  - DebugStyle, E-36
  - ModplsqlStyle, E-36
- PlsqlInfoLogging
  - InfoDebug, E-38
- plug-in, A-2, A-16, Glossary-3
- port, Glossary-4
- private key, Glossary-4
- privileges
  - ORAUER, E-9
- property management
  - with OraDAV, 9-4
- PROPFIND directive, 9-4
- PROPFIND method
  - security considerations, 9-4

- PROPPATCH directive, 9-4
- protecting
  - Web site, B-4
- proxy plug-in
  - behavior, A-6
  - configuring
    - oproxy.serverlist, A-3
    - oproxy.servername.alias, A-4
    - oproxy.servername.hostname, A-4
    - oproxy.servername.port, A-4
    - oproxy.servername.urlrule, A-5
    - proxy configuration file parameters, A-3
    - proxy server definition file, A-3
  - IIS, A-1
  - Sun Java System Web Server
    - proxy plug-in, A-1
  - troubleshooting
    - "file not found" error, A-14
    - broken image links page, A-15
    - garbled characters, A-14
    - incomplete pages, A-14
    - incorrect URLs, A-14
    - listener fails to start, A-13
    - parsing error, A-14
    - partial URL requests errors, A-14
    - proxy server definition file, A-14
    - redirects, A-15
    - REMOTE\_ADDR, A-15
    - Sun Java One "server error", A-14
    - unexpected pages displayed, A-15
- proxy server, A-3, Glossary-4
- proxy server definition file, A-3
- public key, Glossary-4
- public-key cryptography, Glossary-4
- public-key encryption, Glossary-4
- public/private key pair, Glossary-4

## R

---

- restarting, 4-4
- restructuring content
  - with OraDAV, 9-4
- rewrite log, C-3
- RewriteLogLevel, C-3
- root prefix
  - OraDAV and, E-8
- RSA, Glossary-4

## S

---

- scalability, Glossary-4
- script log, C-3
- Secure Sockets Layer, Glossary-5
- secure sockets layer, 4-9
- security
  - authentication, 8-1
  - authorization, 8-1
  - confidentiality, 8-1
  - PROPFIND method, 9-4
  - WebDAV, 9-7

- service name
  - OraDAV and, E-8
- SID value
  - OraDAV and, E-8
- single sign-on, 8-1, Glossary-5
  - partner application, 8-4
  - sso\_conf, A-18
- specifying
  - listener addresses, 6-1
  - listener ports, 6-1
  - log file locations, C-3
  - log files, C-3
    - access log, 7-2
    - error log, C-3
    - lot rotation, 7-6
    - PID file, 4-2
    - rewrite log, C-3
    - script log, C-3
- SQL NCHAR datatypes, 3-7
- SSL, 4-9, Glossary-5
- SSL HW Acceleration Support, 1-3
- SSLAccelerator, E-10
  - nFast, E-10
- SSLCARevocationFile, E-10
- SSLCARevocationPath, E-11
- SSLCipherSuite, E-11
  - tags, E-11
- SSLEngine, E-13
- SSLLog, E-13
- SSLLogLevel, E-13
- SSLMutex, E-14
- SSLOptions, E-14
  - CompatEnvVars, E-15
  - ExportCertData, E-15
  - FakeBasicAuth, E-15
  - OptRenegotiate, E-15
  - StdEnvVars, E-14
  - StrictRequire, E-15
- SSLPassPhraseDialog, E-15
- SSLProtocol, E-16
- SSLRequireSSL, E-18
- SSLRequire, E-16
  - variables
    - SSL, E-17
    - standard, E-17
- SSLSessionCache, E-18
- SSLSessionCacheTimeout, E-19
- SSLVerifyClient, E-19
- SSLWallet, E-20
- SSLWalletPassword, E-20
- SSO plug-in
  - configuring
    - directives, A-18
  - directives
    - IpCheck, A-18
    - LoginServerFile, A-18
  - IIS, A-16
  - troubleshooting
    - HTML authentication, A-24
    - Oracle dependency libraries, A-23

- sso\_conf, A-18
- starting, 4-3
- StdEnvVars, E-14
- stopping, 4-4
- StrictRequire, E-15
- support, 1-6
- symbolic links
  - avoiding use with containers, 9-6
- symlinks, avoiding use with containers, 9-6

## T

---

- trace levels
  - OraDAV and, E-9
- troubleshooting, C-1
  - Oracle HTTP Server may fail to start if PM files are not located correctly, C-2
  - permission denied, C-2

## U

---

- urlrule, A-5
- users
  - OraDAV, E-9
- UseWebCacheIp, C-3
- UTF8, 3-7

## W

---

- wallet, Glossary-5
  - digital, Glossary-2
- Wallet Resource Locator, Glossary-5
- WebDAV, 9-1
  - connecting to HTTP Server, 9-2
  - protocol, 9-2
  - security considerations, 9-7
- WRL, Glossary-5

## X

---

- X.509, Glossary-5

