Oracle® Application Server 10*g*

Security Guide

10*g* (9.0.4)

Part No. B10377-01

September 2003

ORACLE®

Oracle Application Server 10*g* Security Guide, 10*g* (9.0.4)

Part No. B10377-01

Primary Authors: Aimee Reyes, Elizabeth Hanes Perry

Contributing Authors: John Heimann, Uppili Srinivasan

Contents

2 Oracle Application Server Security Architecture

3 Oracle Identity Management

List of Figures

Send Us Your Comments

Oracle Application Server 10*g* Security Guide, 10*g* (9.0.4)

Part No. B10377-01

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

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

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

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

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

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

Preface

This document presents basic Web security concepts and describes the Oracle Application Server security framework and how to use it. First, it provides a survey of security issues and requirements that arise when operating private business systems in the public Internet environment. Then it introduces the security features of Oracle Application Server and provides configuration information for setting up a secure middle tier.

This preface contains these topics:

Audience

The *Oracle Application Server 10g Security Guide* is intended for security administrators, application developers, database administrators, system operators, and other Oracle users who perform the following tasks:

- Configure middle-tier system security
- Analyze application security requirements
- Implement security technologies
- Administer middle-tier system security

To use this document, you need to have general knowledge of Web server administration, Internet concepts, and networking concepts.

Documentation Accessibility

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

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

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

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

Organization

This document contains:

Chapter 1, "Oracle Application Server Overview"

This chapter provides a basic overview of Oracle Application Server.

Chapter 2, "Oracle Application Server Security Architecture"

This chapter discusses the Oracle Application Server security framework, including its architecture. It describes each element and how they work together.

Chapter 3, "Oracle Identity Management"

This chapter presents Oracle Application Server deployment options.

Chapter 4, "Recommended Deployment Topologies"

This chapter provides details on the recommended security topologies for Oracle Application Server.

Chapter 5, "Privilege Delegation"

This chapter covers common security considerations for Oracle Application Server administrators.

Appendix A, "Managing PKI Credentials with Oracle Wallet Manager"

This appendix describes Oracle Wallet Manager and managing PKI credentials.

Glossary

This glossary contains terms that are pertinent to Web security and Oracle environments.

Related Documentation

For Oracle Application Server Application Administrators

This section lists common administration tasks and the manuals that describe them.

Task	Read...
General administration tasks	*Oracle Application Server 10g Administrator's Guide*
Managing static content	*Oracle HTTP Server Administrator's Guide*
Controlling user access to Web content using portals	*Oracle Application Server Portal Configuration Guide*
Managing Oracle Application Server Web Cache	*Oracle Application Server Web Cache Administrator's Guide*
Writing and deploying secure OC4J applications	*Oracle Application Server Containers for J2EE Security Guide*
Managing Oracle Application Server Wireless for security mechanisms	*Oracle Application Server Wireless Administrator's Guide*
Managing users, passwords, and privileges	*Oracle Internet Directory Administrator's Guide*
Managing application, resource, and data source security using Oracle Application Server Reports Services	*Oracle Application Server Reports Services Publishing Reports to the Web*
Managing user access and internalization	*Oracle Application Server Personalization Administrator's Guide*
Configuring security for Oracle Application Server Workflow	*Oracle Workflow Administrator's Guide*
Administering SSO	*Oracle Application Server Single Sign-On Administrator's Guide*
Managing certificate issues	*Oracle Application Server Certificate Authority Administrator's Guide*

For Oracle Identity Management Infrastructure Administrators

For all tasks pertaining to administering and deploying Oracle Identity Management, see the *Oracle Identity Management Concepts and Deployment Planning Guide.*

For Oracle Application Server Application Developers

This section lists common development tasks and the manuals that describe them.

Task	Go to...
Configuring SSO	*Oracle Application Server Single Sign-On Administrator's Guide*
Using `mod_osso` or the Oracle Application Server Single Sign-On SDK to enable applications for SSO	*Oracle Application Server Single Sign-On Application Developer's Guide*
Configuring Web Services	*Oracle Application Server Web Services Developer's Guide*
Configuring Syndication Services	*Oracle Application Server Syndication Services Developer's and Administrator's Guide*
Configuring BC4J	*Oracle Business Component for Java Developing Business Components*
Using keys and certificates for SSL communication in OC4J	*Oracle Application Server Containers for J2EE Servlet Developer's Guide*

For Oracle Application Server Application Deployers

This section lists common deployment tasks and the manuals that describe them.

Task	Go to...
Configuring SSO	*Oracle Application Server Single Sign-On Administrator's Guide*
Configuring Forms with HTTP listener, OC4J, SSO, and OID	*Oracle Application Server Forms Services Deployment Guide*
Configuring security mechanisms in Oracle Application Server Discoverer	*Oracle Application Server Discoverer Configuration Guide*

See Also: *Oracle Application Server 10g Release Notes* in the Oracle Application Server Platform-specific documentation for any security issues that are not addressed here.

For Oracle Application Server Application Users

Component	Go to...
Using Oracle Ultra Search	*Oracle Ultra Search User's Guide*
Using Oracle Application Server ProcessConnect	*Oracle Application Server ProcessConnect User's Guide*
Setting up the database and PL/SQL to avoid known security problems	*Oracle Application Server 10g mod_plsql User's Guide*

Guide to Oracle Documentation

For more information, see these Oracle resources. Descriptions of documents have been added to some listings to guide you to where specific security information can be found. Where document titles are self-explanatory, no description is provided.

The **Oracle Application Server Documentation Library** contains the following documents:

- *Oracle Application Server 10g Quick Tour*

  A brief graphical overview of the application server.

- *Oracle Application Server 10g Concepts*

  An overview of the application server features.

- *Oracle Identity Management Concepts and Deployment Planning Guide*

  An overview of the Identity Management features.

- *Oracle Internet Directory Administrator's Guide*

  Detailed description of Oracle Internet Directory, including Delegated Administration Service and Directory Integration Service, and how to use them.

- *Oracle Internet Directory Application Developer's Guide*

  Detailed description of how to enable applications to access Oracle Internet Directory by using the C API and the PL/SQL API.

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

  Detailed description of how to enable single sign-on for Oracle Application Server.

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

  Detailed description of how to enable applications to use Oracle Application Server Single Sign-On.

- *Oracle HTTP Server Administrator's Guide*
- *Oracle Application Server Portal Configuration Guide*
- *Oracle Application Server Containers for J2EE Services Guide*

  Discusses how to make effective use of the Oracle Application Server Containers for J2EE security features.

- *JAAS Provider API Reference*
- *Oracle Application Server Containers for J2EE User's Guide*
- *Oracle Application Server Web Cache Administrator's Guide*
- *Oracle Application Server 10g mod_plsql User's Guide*

Detailed description of how to configure and use Oracle HTTP Server plug-in `mod_plsql`, which enables communication between the middle tier and an Oracle database.

Oracle Application Server Platform-Specific Documentation contains the following documents:

- *Oracle Application Server 10g Installation Guide*

  Detailed description of what you must install to get the security functionality you require.

- *Oracle Application Server 10g Release Notes*
- *Oracle Application Server 10g Upgrading to 10g (9.0.4)*

  Detailed description of what you must do if you are migrating from a previous version of Oracle Application Server, such as migrating digital certificates.

- *Oracle Application Server 10g Performance Guide*
- *Oracle Application Server 10g Best Practices*

  Detailed description of Oracle Application Server best practices, including security best practices.

Oracle Database Documentation Library contains the following documents:

- *Oracle Advanced Security Administrator's Guide*

  Detailed description of how to configure and use Oracle Advanced Security, the Oracle database option that provides encryption, integrity protection, and advanced authentication to Oracle database clients and servers.

- *Oracle9i Database Administrator's Guide*

  Description of the Oracle9*i* Database Server feature proxy authentication, which allows Oracle Application Server to establish an authenticated session with the database.

- *Oracle9i Application Developer's Guide - Fundamentals*

  Detailed description of how to enable Oracle Application Server to use database proxy authentication.

Printed documentation is available for sale in the Oracle Store at

`http://oraclestore.oracle.com/`

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free of charge and can be done at:

`http://otn.oracle.com/membership/`

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

`http://otn.oracle.com/documentation/content.html`

Conventions

This manual uses the following conventions:

Convention	Meaning
. . .	Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.
. . .	Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted
boldface text	Boldface type in text indicates a term defined in the text, the glossary, or in both locations.
italic text	Italicized text indicates placeholders or variables for which you must supply particular values.
[ ]	Brackets enclose optional clauses from which you can choose one or none.

1

Oracle Application Server Overview

Oracle Application Server provides a comprehensive security framework supporting all Oracle Application Server components, as well as third-party and custom applications deployed on the application server. The framework is based on Oracle Application Server Single Sign-On for authentication, Oracle Internet Directory for authorization and centralized user provisioning, Oracle HTTP Server for Web access, and OracleAS JAAS Provider for security in Java2 Enterprise Edition (J2EE) applications.

This chapter provides an overview of the security architecture and features of Oracle Application Server. It contains the following topics:

Introduction to Oracle Application Server

Oracle Application Server is a reliable, scalable, secure middle-tier application server designed to support a company's evolution into e-business. With this product, the technological complexity of assembling a complete middle-tier Internet foundation is managed for you. The technological foundation that Oracle Application Server provides can grow with your business. Your application can start small and support growing numbers of users and sophisticated functionality on all of your Web sites.

Oracle Application Server components provide a general framework for development and deployment of applications, as well as specific application services and functionality. This chapter focuses on the security services provided by OracleAS Infrastructure 10*g*, which includes Oracle Application Server Single Sign-On and Oracle Internet Directory, an LDAP version 3-compliant directory service. This chapter also provides an overview of the security services provided by Oracle HTTP Server, OracleAS Web Cache, OracleAS Portal, and OracleAS JAAS Provider (Java Authentication and Authorization Service), which provide support for a broad range of application development and deployment strategies.

System Security and Non-Oracle Components

Security is a system issue, not a single-product issue. Each component of your computer application affects the security of the entire system. Proper security requires careful configuration of the following non-Oracle system components:

- Web Browsers
- Firewalls
- Load Balancers
- Virtual Private Networks (VPNs)

Oracle Application Server was designed and coded to integrate smoothly with all these external components.

Web Browsers

In the overall system security picture, the Web browser is the component over which e-business sites have least control. When running a Web storefront, for example, you may not be able to control the browser that customers use. The customer's browser nonetheless impacts the security of your system, and must be taken into consideration. To securely implement Web transactions, your application

must support specific communications and security technologies, including HTTP, LDAP, SSL, x.509 certificates, and Java.

Most commercially available Web browsers support several of these security-related features. However, users must configure the browser properly to take advantage of its security capabilities.

By default, information sent to and from a Web browser is transmitted in the clear; any intermediate site can read the data and potentially alter it in midstream. Web browsers and servers partially address this problem by using the Secure Sockets Layer to encrypt HTTP transmissions (referred to as HTTP/SSL or HTTPS). This ensures the security of data transmitted between the client to the server. However, because commercially available Web browsers do not ship with client certificates, most HTTP/SSL transmissions are authenticated in only one direction, from server to client; the client does not authenticate itself to the server.

Because the HTTP protocol does not support sessions, many e-commerce applications use cookies to store session data for individual customers. These cookies are transmitted as cleartext; this means that they can be intercepted by a third party. For this reason, it is wise for the application to encrypt or obfuscate information that is stored in cookies.

> **Note:** The W3C has a useful discussion of cookie security issues at `http://www.w3.org/Security/Faq/wwwsf2.html#CLT-Q10`.

Firewalls

Firewalls control access between the full Internet and a corporation's internal network. A firewall defines which sorts of Internet communications will be permitted into the corporate network, and which will be blocked. A well-designed firewall can foil many common Internet-based security attacks. However, a firewall is only as secure as its maintenance. New Internet-based attacks are constantly being designed, and firewall configurations must constantly be updated to keep abreast of these attacks.

Firewalls monitor communications methods, not communications content. Therefore, firewalls cannot protect your application against misuse of permitted communications channels. For instance, to permit the use of the Web, a firewall must permit HTTP communication. Because firewalls do not monitor content, a firewall cannot protect against security attacks transmitted within valid HTTP messages. Similarly, because a firewall does not monitor the content of e-mail messages, it cannot prevent the transmission of e-mail viruses.

Load Balancers

Load balancing distributes an application's load over many identically configured servers. This distribution ensures consistent application availability, even when one or more server fails. Load balancing has a significant impact on security design, especially on encryption issues. For instance, in many installations, SSL keys are unique to a particular server in a cluster, and are not necessarily shared with other servers. This sharing complicates moving an SSL session from one server to another.

Virtual Private Networks (VPNs)

A Virtual Private Network (VPN) allows applications to use the public Internet to communicate securely with the corporate LAN. All IP communications between the application and the corporate LAN are encrypted so that they cannot be read or altered by intermediate sites. A VPN prevents a third party from monitoring or altering communications. Like other network-based security solutions, VPNs cannot prevent the transmission of viruses, nor can they control the content of the information being transmitted.

Security Objectives

The security objectives for Oracle Application Server derive from the overall architecture and functions of the product, as well as the range of operational environments and risk scenarios in which Oracle anticipates the product will be deployed.

Oracle Application Server was designed to meet the following objectives:

- Providing Basic Security Services
- Supporting Standards
- Ensuring Deployment and Configuration Flexibility
- Minimizing Application Development and Deployment Cost
- Providing Security In Depth

Providing Basic Security Services

Certain security services are fundamental to providing security in a multi-user, networked environment. Oracle Application Server has been designed to provide all these services, including:

- **Authentication**. Allows a system to verify the identity of users and other systems that request access to services or data. Authentication is a prerequisite for many other security services, including access control, authorization, and accountability.
- **Authorization**. Allows a system to determine the privileges which users and other systems have for accessing resources on that system. Authorization is generally required for effective access control.
- **Access Control**. Ensures that a system grants access to resources only in ways that are consistent with security policies defined for those resources. Access decisions are based on the authenticated identity and/or authorization of the requesting user, and on what type of access that user is requesting.
- **Accountability and Intrusion Detection.** Ensures that users who access the system can be held accountable for their usage of the system and its resources. The purpose of these services is to ensure that system usage contrary to system security policy is detected and recorded, so that the offending user can be identified and disciplined if necessary. Intrusion detection services are related to accountability, but attempt to detect and react to unauthorized usage (including unauthorized usage by authorized users) in real time.
- **Data Protection**. Protects sensitive data against access by those who are not authorized users of the system. For example, encryption mechanisms can protect data sent through a public network from interception. Encryption can also protect highly sensitive data (such as passwords) stored on a disk from users who bypass system access control mechanisms, such as by exploiting a vulnerability in the underlying operating system or by stealing the physical disk storage medium.

Supporting Standards

Oracle Application Server is an open standards-based product. It complies with the J2EE framework and supports standard protocols, such as HTTP, and markup languages, such as HTML and XML. Corresponding Oracle Application Server security services also comply with relevant standards, facilitating interoperation with third-party products. For example, most Oracle Application Server applications support browser-based clients, typically Internet Explorer or Netscape

Navigator. Oracle Application Server therefore supports the security standards that these browsers implement, including SSL for encryption, and X.509v3 when certificates are used to authenticate users. Similarly, OC4J supports the J2EE security standards such as the Java Authentication and Authorization Service (JAAS), so that customers can deploy third-party Java applications securely.

Ensuring Deployment and Configuration Flexibility

Oracle Application Server supports a wide range of potential configurations and deployment options. These configurations span the range from standalone developer installations of Oracle Application Server Java Edition on a small desktop computer to large, distributed, multi-server deployments of Oracle Application Server serving hundreds of thousands of users in a worldwide enterprise.

Oracle Application Server security services have been designed to support the full range of product deployment options. In particular, the security services deployed on each edition of Oracle Application Server have been chosen to support the particular deployment scenarios and types of applications for which that edition of Oracle Application Server is targeted. Moreover, security mechanisms in Oracle Application Server have been designed to ensure that practical, real-world constraints on deployment can be met, such as the need to deploy certain components of Oracle Application Server in the DMZ, other components in the corporate intranet, and allow those components to communicate through firewalls.

> **See Also:** Chapter 4, "Recommended Deployment Topologies" for more information about deployment options, typical configurations for Oracle Application Server, and specific examples of real-world constraints and how to deploy Oracle Application Server in the face of them.

Minimizing Application Development and Deployment Cost

Oracle Application Server serves as a development and deployment environment for web applications. Oracle Application Server is designed to provide services and tools that reduce the time, effort, and expense to develop and deploy such applications. Because security is an important part of deploying applications in a production environment, Oracle Application Server has been designed to provide the essential security services common to most web applications. Individual components work together with your application and the application server to furnish a complete assortment of security services.

Working in cooperation, the security services provided in Oracle Application Server ensure the following:

- **Easy development and deployment of secure applications**. Oracle Application Server provides the basic, easy-to-use services required to deploy applications. These basic security services are discussed in "Providing Basic Security Services".
- **Scalability, supporting complex deployments that support large numbers of users and servers**. Oracle Application Server provides additional security services that reduce cost and complexity for large or complex deployments. These services include centralized user provisioning, single sign-on, and authorization, so that customers do not need to develop or purchase and integrate these services themselves.
- **Protection of existing investments in third-party technology**. Oracle Application Server protects your existing investment through compliance with security standards and support for specific third-party security mechanisms and infrastructure where required.

Providing Security In Depth

An important design objective for Oracle Application Server is to provide security in depth, meaning that:

- **Security mechanisms are implemented with high assurance, so that the probability of failure of any given security mechanism is low**. This is achieved through secure coding practices, developer security education and training, secure coding compliance checklist/testing, independent evaluations, independent security assessments and penetration testing, and security incident response.
- **Security must degrade gracefully, and there must be no single points of failure**. Failure of any single security mechanism should cause only incremental loss of security, not compromise the entire system.
- **Privileges are minimized by default**. You must explicitly grant permission to perform sensitive functions or access sensitive data.
- **Intrusions are contained**. The system should detect and limit damage from security breaches.

Oracle Application Server Middle-Tier Components

This section gives a brief overview of the Oracle Application Server middle-tier components. You should be aware of three important points about application servers and the middle tier:

- An application server is a deployment environment for business applications and provides a standard Web interface to these applications. The development environment Oracle supports is based on Java 2, Enterprise Edition. Standard interfaces and standard development and deployment environments are important for interoperability, so that your investment in standards-based technology is protected and the costs to develop and deploy applications are reduced.
- An application server typically provides common integration and management functions, such as application monitoring, application and resource access control, user authentication, and centralized authorization. These functions reduce costs to develop, manage, and deploy applications.
- An application server typically supplies specific services, such as business functions and presentation and UI services, which are commonly needed when developing applications. This improves productivity and reduces deployment cost and time.

Oracle Application Server provides the following middle-tier components that are particularly important in developing secure applications:

- Oracle Application Server Web Cache
- Oracle HTTP Server
- Oracle Application Server Containers for J2EE (OC4J) and OracleAS JAAS Provider
- OracleAS Portal

Oracle Application Server Web Cache

OracleAS Web Cache can be configured to receive HTTPS browser requests and send HTTPS requests to origin servers. OracleAS Web Cache caches frequently accessed Web pages or partial pages.

Oracle HTTP Server

Oracle HTTP Server is the Web server component of Oracle Application Server. It is based on the Apache HTTP Server. The Apache open source Web server is among

the most widely adopted Web server products; it supports a rich set of existing applications, and provides a flexible and well-understood security model. Apache is a very well-tested platform on which to deploy secure applications. Customers familiar with Apache should find it easy to build and deploy secure Web applications using Oracle HTTP Server.

Oracle HTTP Server Security Services Overview

Oracle HTTP Server extends Apache with several standard enhancements, called `mods` (a shortened form of "modules"), as well as with mods developed by Oracle Corporation. Oracle HTTP Server allows users with Web browsers to access Oracle Application Server using standard Web protocols. Oracle HTTP Server provides an HTTP listener that supports HTTP and HTTPS and serves up information to users in standard HTML format. Oracle HTTP Server provides access to both static Web pages and dynamic content.

Oracle HTTP Server security services include the ability to restrict or allow access to files and services based on the identity of users established by means of basic authentication, by client- supplied X.509 certificates, and by IP or hostname addresses.

Another important feature of Oracle HTTP Server security is protection of data exchanged between clients and the server. This is provided by means of the SSL protocol, which also provides data integrity and strong authentication of both users and HTTP servers.

> **Note:** At this release, Oracle HTTP Server is not installed with SSL enabled. You must explicitly create a security certificate and enable SSL. For details, see the *Oracle HTTP Server Administrator's Guide*.

In addition, Oracle HTTP Server supplies logging and other facilities needed to detect and resolve intrusion attempts. It provides integration with the other Oracle Application Server components, such as `mod_osso`, which enables the HTTP server to receive and route requests for single sign-on services to Oracle Application Server Single Sign-On server. Oracle HTTP Server is also well integrated with other Oracle products such as Oracle applications and the database. In this way, the Oracle HTTP Server offers a comprehensive set of security services for building and deploying Web applications.

> **See Also:** *Oracle HTTP Server Administrator's Guide* for detailed information about configuring and using the HTTP server

Oracle Application Server Containers for J2EE (OC4J) and OracleAS JAAS Provider

Oracle Application Server Containers for J2EE provides the Java runtime environment for Oracle Application Server components. Oracle Application Server Java Authentication and Authorization Service ensures secure access to and execution of Java applications, as well as integration of Java-based applications with Oracle Application Server Single Sign-On.

Applications and Tools

The following products may also be installed with Oracle Application Server:

- OracleAS Reports Services
- OracleAS Forms Services
- OracleAS Discoverer
- Oracle BI Beans

These products have their own product-specific security features, which are discussed in their individual documentation.

OracleAS Portal

Enterprise portals are specifically designed to be the single source of interaction with corporate information and to be the focal point for conducting day-to-day business. OracleAS Portal is a complete and integrated solution for building, deploying, and maintaining a world-class enterprise portal. It combines a rich, declarative environment for creating a portal Web interface, publishing and managing information, accessing dynamic data, and customizing the portal experience with an extensible framework for J2EE-based application access. Using OracleAS Portal, e-businesses have the power to connect employees, partners, and suppliers with the information they need and the flexibility to create views tailored to each community.

In addition to core security capabilities, OracleAS Portal leverages Oracle Identity Management to manage and provide secure access to content and applications.

Identity Management Infrastructure

Oracle Identity Management is an integrated infrastructure on which Oracle products rely for distributed security. Oracle Identity Management ships with Oracle Application Server but it also ships as part of the infrastructure of other Oracle products. The Oracle Identity Management infrastructure is discussed in detail in Chapter 3, "Oracle Identity Management".

> **See Also:** *Oracle Identity Management Concepts and Deployment Planning Guide.*

Repositories

An Oracle Application Server application uses at least two different data repositories: one or more Metadata Repositories and the repository for your application data.

- The Oracle Application Server Metadata Repository stores configuration information about the application and about Oracle Application Server itself.
- The application repository stores the data manipulated by the application: names, addresses, invoices, and so on.

These repositories can be housed on the same server, and indeed in the same database, but should not be stored in the same database tables. In particular, your application must not store its data in the Metadata Repository.

Configuration Options and Common Topologies

The following are common installation and configuration options for Oracle Application Server. For full information on these topologies, see Chapter 4, "Recommended Deployment Topologies", and the *Oracle Application Server 10g Installation Guide*.

- Java Developer Topology

  This is a single-computer development topology on which you can build, run, and test J2EE applications. It does not have an OracleAS Infrastructure 10*g*; it includes Oracle HTTP Server, Oracle Application Server Containers for J2EE, and Oracle Application Server Web Cache.

- Portal and Wireless Developer Topology

  This is a single-computer development topology containing an OracleAS Infrastructure 10*g* and a OracleAS Portal and Oracle Application Server Wireless middle tier. The OracleAS Infrastructure 10*g* installation creates a new Oracle Database and Oracle Internet Directory.

- Forms, Reports, and Discoverer Developer Topology

  This topology enables Oracle Application Server Forms Services and Oracle Application Server Reports Services developers to build and test their applications. Developers use Forms Builder and Reports Builder to develop their applications. This is a single-computer development topology containing:

  - OracleAS Infrastructure 10*g* (a new Oracle Database and Oracle Internet Directory)
  - Business Intelligence and Forms middle tier
  - Forms Builder (part of Developer Suite 10*g*)
  - Reports Builder (part of Developer Suite 10*g*)
  - Oracle Application Server Discoverer Administrator (part of Developer Suite 10*g*)

- Integration Architect and Process Modeler Topology

  This development topology enables Oracle Application Server ProcessConnect architects and modelers to design applications that can communicate with external applications using Oracle Application Server and Oracle Application Server ProcessConnect. This development topology includes:

  - OracleAS Infrastructure 10*g*
  - J2EE and Oracle Application Server Web Cache middle tier
  - Oracle Application Server ProcessConnect

- Departmental Topology

  This topology consists of an OracleAS Infrastructure 10*g* with two metadata repositories and multiple middle tiers, including at least one Portal and Wireless middle tier. This topology uses two metadata repositories:

  - One for Oracle Identity Management services; all the middle tiers use this metadata repository for Oracle Identity Management services.
  - One for product metadata; the OracleAS Portal and OracleAS Wireless middle tier uses this metadata repository.

- Enterprise Data Center Topology: J2EE Applications

  This deployment topology is optimized to support J2EE applications. It contains the components required to run J2EE applications in a secure, high availability environment. This topology is intended for enterprises that have users internal as well as external to the organization. Requests from external users go through firewalls.

- Enterprise Data Center Topology: Portal, Wireless, and Business Intelligence Applications

  This deployment topology supports J2EE applications as well as applications that use components in the OracleAS Portal and OracleAS Wireless, and the Business Intelligence and OracleAS Forms Services middle tiers. This topology is intended for enterprises that have users internal as well as external to the organization. Requests from external users go through firewalls.

- Development Life Cycle Support Topology

  This topology is a combination of other topologies to support moving applications from test to stage to production environments.

  - Test environment: Application developers test their applications in their own environments.
  - Stage environment: QA personnel test all applications before deploying them to the production environment.
  - Production environment: Applications are ready for use by users internal and external to the enterprise.

- Oracle Application Server Certificate Authority Topology

  In this topology, Oracle Application Server Certificate Authority has its own Oracle Application Server Metadata Repository, and both these components run on a computer separate from other infrastructure components. The other components use a different metadata repository.

New Security Platform Capabilities in Oracle Application Server 10*g*

Oracle Identity Management is a new security solution for Oracle Application Server 10*g*. In addition, security enhancements have been made across the entire product.

This section discusses the following security enhancements:

- Oracle Identity Management Enhancements
- General Security Enhancements

Oracle Identity Management Enhancements

Oracle Identity Management is an integrated package of directory, security and user management functionality. Oracle Identity Management provides the integrated infrastructure on which Oracle products rely for distributed security.

Oracle Identity Management includes the following components:

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

Oracle Identity Management Components

The following new features and capabilities for Oracle Identity Management components are described:

- Oracle Internet Directory
- OracleAS Single Sign-On
- Oracle Application Server Certificate Authority (OCA)

Oracle Internet Directory

Oracle Internet Directory introduces several new features and capabilities with Oracle Application Server 10*g*,including Windows integration, new password policy options, and partial replication features.

- Windows Integration Capabilities—Oracle Internet Directory now provides a preconfigured directory synchronization solution for Windows Active Directory Services. This feature allows users to have a single identity and password credential across the Oracle and Windows environments. It also includes directory plug-ins that support mastering and changing passwords stored in the Windows environment, relieving customers of overhead and potential security concerns associated with synchronizing passwords across the two environments.
- Flexible Password Policy—Oracle Internet Directory supports new password policy options. In addition, a new Oracle Internet Directory plug-in support allows customers to implement an almost unlimited variety of site-specific password policies.
- Partial Replication—-Oracle Internet Directory now supports new replication models, enabling improved scalability and performance in large network configurations.
- Other Features—Other new features include support for dynamic groups, an expanded Oracle Internet Directory Self-Service Console, easy synchronization of directory data with database tables, and features to permit user identity synchronization with the Oracle e-Business Suite Release 11*i*.

OracleAS Single Sign-On

New features for OracleAS Single Sign-On include support for:

- Federated Identity Management—OracleAS Single Sign-On can obtain user identities from one or more trusted authentication sources, and proxy these identities into the Oracle Application Server environment. This feature supports federated identity management scenarios.

  For example, customers could configure Oracle Application Server to obtain and accept authenticated user identities from the identity management systems of business partners.
- Multilevel Authentication—OracleAS Single Sign-On allows customers to establish more than one authentication mechanism, and to indicate the way in which a user authenticated to single sign-on enabled applications. Applications can take advantage of this to grant different degrees of privilege to users, depending on how they authenticated.

  For example, users may get partial privileges if they authenticate using password, but more complete privileges if they use stronger authentication, such as X.509v3.

Oracle Application Server Certificate Authority (OCA)

OracleAS Certificate Authority is a new component in 10*g* (9.0.4). It completes the Oracle public key infrastructure (PKI) offering by allowing customers to create and manage X.509v3 digital certificates for use in Oracle or third-party software. OracleAS Certificate Authority is fully standards compliant and is seamlessly integrated with Oracle Application Server Single Sign-On and Oracle Internet Directory. It provides an out-of- the-box PKI solution for Oracle customers that is easy to use and manage. OracleAS Certificate Authority provides Web-based certificate management and administration, as well as XML-based configuration. It leverages the identity management infrastructure, high availability, and scalability of the Oracle Application Server platform.

General Security Enhancements

Oracle Application Server has added many other security enhancements across the entire product, including:

- Oracle HTTP Server Enhancements
- Privilege Delegation
- Enterprise Integration
- Oracle Workflow
- Oracle Business Components for Java (BC4J)

Oracle HTTP Server Enhancements

To incorporate the latest optimizations and security features of Apache, the Oracle HTTP Server uses Apache v1.3.28. In addition, Oracle HTTP Server has the following security enhancements:

- Session Renegotiation support—This feature allows individual directories to be protected by different strength encryption, some with weaker encryption, while others with stronger encryption.
- SSL HW Acceleration support (for nCipher)—SSL encryption is slower when performed in software. Oracle HTTP Server now supports dedicated nCipher hardware for SSL encryption.

- Port Tunneling—Oracle9*i*AS 9.0.2 introduced the AJP protocol for routing between Oracle HTTP Server and Oracle Application Server Containers for J2EE (OC4J). The firewall configuration required knowledge of several ports—especially for deployments that had several OC4J instances behind a firewall being routed to and from a front-end Oracle HTTP Server. This is now simplified with the Port Tunnel, which lets all communication between Oracle HTTP Server and OC4J happen on a limited number of designated ports. The port tunnel daemon routes the requests to the appropriate OC4J. Therefore, only one port (possibly more, depending on configuration) has to be opened through the firewall, regardless of the number of back-end OC4J instances.
- Oracle HTTP Server to OC4J SSL Support—Oracle HTTP Server and OC4J communication can now occur over AJP/SSL, providing end-to-end SSL support for OC4J requests.

Privilege Delegation

This release of Oracle Application Server provides fine-grained control over system administration and management privileges, allowing you to:

- Delegate only the privileges necessary for installation and administration
- Grant application administration permissions without making the application administrator an Oracle Internet Directory superuser
- Isolate application installation privileges from application administration privileges
- Encapsulate privileges for each application, so that permission to deploy one component does not grant the right to deploy or administer other components

Enterprise Integration

Oracle Application Server Integration adds robust secure communication, including SSL encryption, digital certificates, and digital signatures. The product ensures guaranteed exactly once delivery, provides end-to-end auditing and tracing, and supports non-repudiation. It also supports Oracle Wallet Manager for management of digital credentials.

Oracle Workflow

With Oracle Application Server 10*g* (9.0.4), Oracle Workflow supports Oracle Application Server Single Sign-On. All users can be authenticated using Oracle Application Server Single Sign-On technology with the users stored in Oracle Internet Directory. As a result, the default Oracle Workflow directory service is based on users stored in Oracle Internet Directory. Oracle Workflow also provides fine-grained security using VPD, which can be used in a hosted environment. Each subscriber's or organization's data is secured from other subscribers or organizations. The subscribers in the hosted environment are stored in Oracle Internet Directory.

Oracle Business Components for Java (BC4J)

Oracle Business Components for Java has added support for implementing application-level security using J2EE security standards (Oracle Application Server Java Authentication and Authorization Service).

2

Oracle Application Server Security Architecture

This chapter provides an overview of the security architecture of Oracle Application Server in the following topics:

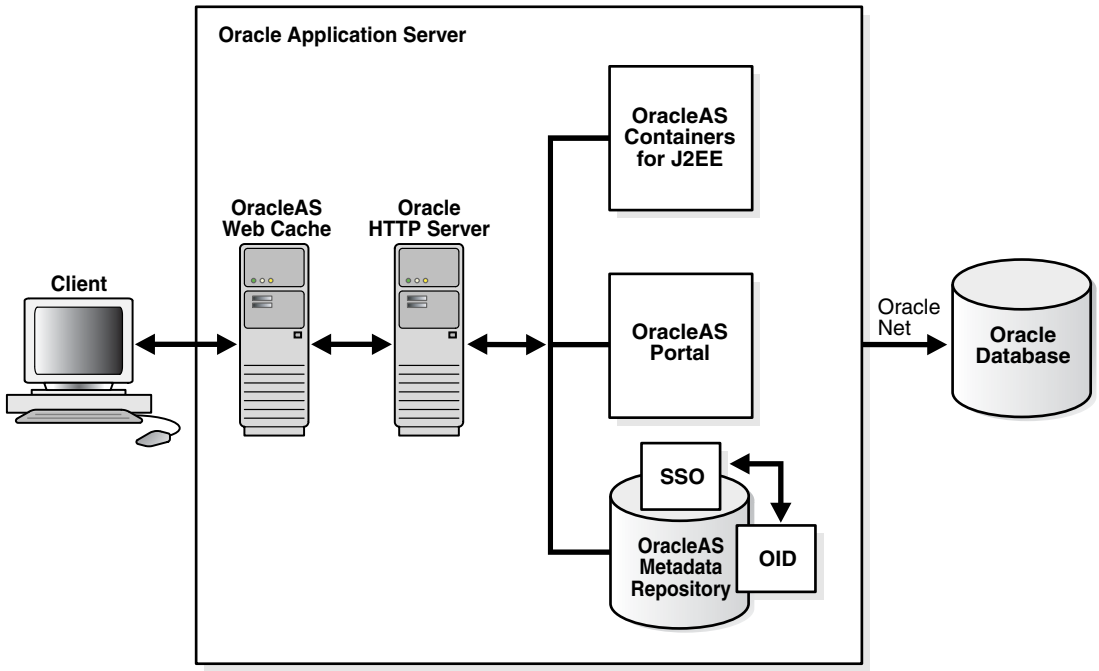
Security Architecture of Oracle Application Server

Oracle Application Server provides a solid framework for building and deploying Web applications using the Apache-based Oracle HTTP Server, Oracle Application Server Containers for J2EE, and OracleAS Portal, which use the advanced security functionality provided by OracleAS Infrastructure 10*g*. OracleAS Infrastructure 10*g* consists of Oracle Application Server Metadata Repository and Oracle Identity Management. Oracle Application Server security starts from the well-tested and highly configurable Web security services provided by Oracle HTTP Server, adds a comprehensive set of Web single sign-on services, and extends them further with centralized user provisioning that is available in Oracle Internet Directory, an LDAP, version 3-compliant directory service. In addition, Oracle Application Server provides the Oracle implementation of Java Authentication and Authorization Services (JAAS) for J2EE application security, and extensive portal authorization and application integration mechanisms. Oracle Application Server also supports secure access to Oracle database systems using Oracle Advanced Security.

Elements of Oracle Application Server Security Architecture

Figure 2–1 illustrates the flow of information among the elements of Oracle Application Server.

Figure 2–1 Security Architecture of Oracle Application Server



The remainder of this chapter discusses each element in greater detail.

Oracle HTTP Server Security

The Oracle HTTP Server provides the first line of defense in Oracle Application Server security. The Oracle HTTP Server makes data available to users through a standard Web interface. Oracle HTTP Server mediates user access to both static and dynamic content by restricting access to URLs and directories on the server. Dynamic content is provided by applications running natively on Oracle HTTP Server, such as CGI, or in other Oracle Application Server components. These components include J2EE applications deployed on Oracle Application Server Containers for J2EE (OC4J) and accessed through `mod_oc4j`, as well as PL/SQL applications deployed on an Oracle Database and accessed through `mod_plsql`. You configure access to resources on Oracle HTTP Server using the standard Apache directive model; see the *Oracle HTTP Server Administrator's Guide* for details.

The Oracle HTTP Server controls access to resources based on user identity. Identity is established through one of a number of authentication mechanisms. These include standard Apache authentication mechanisms such as basic authentication and SSL with client certificate. Users can also be authenticated through OracleAS Single Sign-On, using `mod_osso`; this is described in detail in the *Oracle Application Server Single Sign-On Application Developer's Guide* and the *Oracle Identity Management Concepts and Deployment Planning Guide*. Applications running on Oracle Application Server can obtain OracleAS Single Sign-On user identity from Oracle HTTP Server using the Apache header created by `mod_osso`.

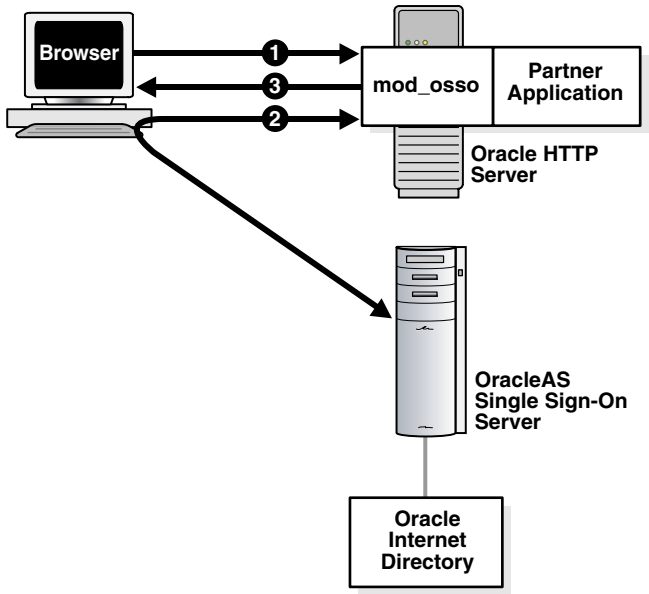
In Oracle Application Server 10*g*, when users are authenticated by `mod_osso`, control of user access on Oracle HTTP Server is limited to specifying whether a user may have access to server resources (URLs, directories) or not. Applications accessible through Oracle HTTP Server can use the SSO-authenticated user identity to enforce fine-grained control of user access to resources that are managed by those applications. The Oracle HTTP Server does not itself provide fine-grained access control of users to static content on the HTTP Server when users are authenticated using SSO.

Oracle HTTP Server can be configured to protect data exchanged between the server and Web clients using the Secure Sockets Layer (SSL) cryptographic protocol. The SSL protocol is an industry-accepted standard for network transport layer security. SSL provides encryption and data integrity, and support for digital certificate authentication using a public key infrastructure (PKI). Digital certificates for SSL authentication require use of an Oracle Wallet; for more information, see Appendix A.

Message Flow With Single Sign-On

Figure 2–2 shows the flow of information when a user requests the URL for a partner application using the Oracle HTTP authentication module `mod_osso`.

Figure 2–2 Single Sign-On With mod_osso



1. The user tries to access a partner application.
2. The user is redirected to the single sign-on server. The server challenges her for her credentials. After verifying these credentials in Oracle Internet Directory, it passes them on to the partner application
3. The application serves up the requested content.

Authenticating To an External Application For the First Time

OracleAS Single Sign-On uses the following process if the user is accessing an external application for the first time.

1. The external application login procedure checks the single sign-on server password store for the user's credentials for the requested external application. If it finds that the user has no such credentials, the single sign-on server prompts the user for them.
2. The user enters the user name and password.
3. If the user elects to save the credentials in the single sign-on server password store, the server uses these credentials to construct a login form to submit to the login processing routine for the external application. This routine has been preconfigured by the single sign-on server administrator and is associated with the requested application.
4. The single sign-on server sends the form to the client browser, with a directive to post it immediately to the external application.
5. The client posts the form to the external application and logs the user in.

If the user declines to save her credentials in the single sign-on password store, she must enter a user name and password each time she logs in to the application.

SSL Acceleration

In addition to offboard SSL acceleration solutions, Oracle Application Server now supports BHAPI-compliant hardware for deployment on servers running Oracle Application Server Web Cache and/or Oracle HTTP Server. When executed in software, SSL operations place a strain on server CPU resources, causing a reduction in throughput and slower overall performance. The hardware offloads the SSL key exchange processing from a server's CPUs, increasing the number of concurrent SSL connections and improving response times for SSL-protected content.

J2EE Security and JAAS

J2EE is the primary application development and deployment environment supported by Oracle Application Server. Oracle fully supports the Java2 Security model for managing access to applications, resources and data, and specifically provides Oracle Application Server Java Authentication and Authorization Service to authenticate users and manage their access privileges.

> **See Also:** *Oracle Application Server Containers for J2EE Security Guide*

The OracleAS JAAS Provider allows user authentication and authorization information to be managed in two ways. For Oracle Application Server Java Edition deployments that do not use Oracle Identity Management, user information can be managed in a flat file in XML format. For Oracle Application Server deployments in which Oracle Identity Management is installed, the OracleAS JAAS Provider can also take advantage of Oracle Identity Management. In this case, user authentication and authorization information is managed in Oracle Internet Directory, and the OracleAS JAAS Provider can leverage Oracle Application Server Single Sign-On for user authentication. In this case, users can be provisioned using the Oracle Delegated Administration Services component of Oracle Identity Management, or can be managed in and provisioned from a third-party repository using the Directory Integration and Provisioning component. Directory Integration and Provisioning specifically allows OracleAS JAAS Provider user information to be managed in a third-party LDAP directory, using a connector to Oracle Internet Directory.

A major benefit of using the OracleAS JAAS Provider with Oracle Identity Management is that any J2EE applications deployed on Oracle Application Server, whether developed by Oracle, a customer, or a third party, can share a common framework for user authentication and authorization. This framework is integrated with every component of Oracle Application Server, as well as with other Oracle products, such as the Oracle Database, Oracle Collaboration Suite, and Oracle E-Business Suite. Another benefit of using Oracle Identity Management is that it can scale to support millions of users, and manages their information in a reliable, highly available, and secure directory. For more information on Oracle Internet Directory, Oracle Application Server Single Sign-On, and other components of Oracle Identity Management, see the *Oracle Identity Management Concepts and Deployment Planning Guide.*

A new feature of Oracle Application Server 10*g* is that OC4J exposes a `LoginModule` API, allowing customers to deploy custom JAAS LoginModules.

These can be used to authenticate users with third-party authentication mechanisms, or to manage JAAS user information in third-party directories when Oracle Identity Management has not been installed (for example, in Oracle Application Server Java Edition).

Another new feature of Oracle Application Server 10*g* is that the Apache Java Protocol (AJP) can now be used with SSL encryption. AJP is used when Oracle HTTP Server and OC4J are deployed on physically separate servers. SSL protection of AJP ensures that any sensitive data exchanged between Oracle HTTP Server and OC4J is protected against disclosure or modification in the communication network.

Oracle Application Server Portal Security

The OracleAS Portal allows customers to organize Web content and applications in a logical and consistent Web portal format. OracleAS Portal provides a flexible, sophisticated model for managing user access to OracleAS Portal resources based on user identity and privilege. It supports a hierarchical, group-based model for aggregating privileges. A collection of privileges is associated with each group, and users who are members of that group inherit the appropriate privileges. The model is hierarchical: groups may be defined as subgroups of other groups. In this case, users who belong to the subgroup inherit all the privileges of the larger group in addition to privileges unique to the subgroup.

As do other components of Oracle Application Server, OracleAS Portal uses Oracle Identity Management for user management, authentication, and authorization. After users have been provisioned in the Oracle Internet Directory component of Oracle Identity Management, they can authenticate themselves to OracleAS Portal using Oracle Application Server Single Sign-On.

See Also: *Oracle Application Server Portal Configuration Guide.*

Oracle Application Server Web Cache Security

OracleAS Web Cache serves as a caching front end to Oracle HTTP Server. When used, it intercepts HTTP requests sent to Oracle HTTP Server, and proxies them to Oracle HTTP Server if necessary. Because it acts as a proxy, OracleAS Web Cache necessarily terminates any SSL connections established by a client system to Oracle Application Server. If the SSL connection uses client certificate authentication, then the client certificate identity is provided to OracleAS Web Cache, and not to Oracle HTTP Server, because the SSL connection is established between the client and OracleAS Web Cache.

A new feature of Oracle Application Server 10*g* is that OracleAS Web Cache can proxy the contents of a client certificate, when used in an SSL connection, to Oracle HTTP Server. In this way, a client's SSL authenticated identity can be obtained and used by Oracle HTTP Server, even if OracleAS Web Cache is used in front of Oracle HTTP Server.

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

Security for Other Oracle Application Server Components

Oracle Application Server supports Oracle products for data reporting (OracleAS Forms Services, OracleAS Reports Services), business intelligence (OracleAS Discoverer), and other purposes. These applications may have application-specific privileges and data security models, but all of them leverage Oracle Identity Management for user management, authentication, and authorization. For more information on security associated with these products, please refer to the security sections in the individual product documentation.

Oracle Advanced Security

When Oracle Application Server accesses an Oracle database, customers may wish to protect data exchanged between Oracle Application Server and the database using a cryptographically protected network protocol. Network encryption is one of the features offered in the Oracle Advanced Security option available with the Oracle Database. Please refer to the *Oracle Advanced Security Administrator's Guide* for the available algorithms and configuration details.

3

Oracle Identity Management

This chapter outlines the dependency of Oracle Application Server on Oracle Identity Management and the role that Oracle Identity Management infrastructure plays in Oracle Application Server deployments. This chapter contains the following topics:

The Role Of Oracle Identity Management

Oracle Identity Management is a key deployment platform capability of Oracle Application Server. The Oracle Identity Management infrastructure centralizes management of security across the enterprise, simplifying management and reducing administrative overhead. This capability increases security while reducing administrative costs and enhancing the end-user experience.

Oracle Identity Management is a well-integrated suite of services that all Oracle products, including Oracle Database, Oracle Collaboration Suite, and Oracle E-Business Suite, can leverage out of the box. This allows rapid deployment of Oracle products in the enterprise without the cost and complexity associated with integrating disparate systems. Oracle Identity Management also serves as a single point of integration between the Oracle environment and any third-party Identity Management environments.

Oracle Identity Management infrastructure is not required in all Oracle Application Server deployments. The components of Oracle Application Server involved in the deployment, and the nature of the deployment, determine the need for an Oracle Identity Management infrastructure. Some components, such as OracleAS Portal, require the Oracle Identity Management infrastructure for their operation. A simple OC4J customer application might not have any need or awareness for such an

infrastructure. It is also possible to design an OC4J application to leverage an enterprise Oracle Identity Management infrastructure for its authentication and authorization services.

Dependencies on Oracle Identity Management

For some Oracle Application Server components, such as OracleAS Portal, the Oracle Identity Management infrastructure is always required. However, Oracle Identity Management is not mandatory for all Oracle Application Server components. Many Oracle Application Server components, such as OracleAS Forms Services and OracleAS Reports Services, can be deployed with or without leveraging the Oracle Identity Management infrastructure. When deployed without the Oracle Identity Management infrastructure, these services would rely on their own standalone interfaces for user management and security.

Leveraging Third-Party Identity Management Services

OC4J applications developed by ISVs and customers need not rely on Oracle Identity Management or any other infrastructure. These applications can instead use third-party identity management services, such as SunOne (formerly iPlanet) Directory, Microsoft Active Directory, or Netegrity SiteMinder. Thanks to configurable OracleAS JAAS Provider `LoginModules`, OC4J applications can also be integrated with any other custom user management and authentication services in the customer environment.

All Oracle products that rely on centralized user management and single sign-on services, including products such as OracleAS Portal, Oracle Application Server Forms Services, and Oracle Application Server Reports Services, require Oracle Identity Management infrastructure for their operation. If you have already deployed a non-Oracle Identity Management infrastructure, the Oracle products can be deployed to fully leverage your investment in such infrastructure. In such environments, Oracle product security still depends on Oracle Identity Management infrastructure, but that infrastructure is configured to fully utilize your existing infrastructure. For instance, you need not reimplement or alter your implementation of directory tree structure, practices, and policies for user management, password management, and so on. Oracle Identity Management integration services transparently adopt your existing policies without requiring any additional implementation effort.

Features and Benefits Of Oracle Identity Management

This section outlines the various capabilities offered by Oracle Identity Management and the benefits that enterprise applications based on Oracle Application Server can leverage.

These benefits include:

- Centralized User Management
- Password Management Policies

Centralized User Management

Oracle Internet Directory, a key component of the Oracle Identity Management infrastructure, facilitates centralized user management for the Oracle technology environment, as well as for the rest of the enterprise. Users are defined centrally in Oracle Internet Directory; all other Oracle Identity Management and security services, as well as all applications that in turn rely on these services, share this single definition of user identity, credentials, profiles and preferences. This centralized management not only facilitates administrative convenience, it also enhances security for applications that share this infrastructure.

Password Management Policies

Password policies help strengthen the security of password-based authentication environments. Password policies allow an enterprise to establish rules that users must follow while setting and using passwords to authenticate themselves to the applications on the network. Oracle Identity Management password policies can be customized at deployment.

Oracle Identity Management supports complex password policies that enterprises can leverage to make the user passwords more secure. Oracle Internet Directory and the OracleAS Single Sign-On services support value-based as well as state-based password policies.

- Value-based password policies make it difficult to guess passwords. These policies enforce the password values to be arbitrarily complex, such as minimum lengths, presence of minimum number of special characters, and so on.
- State-based password policies help enforce user discipline, such as periodically resetting password values. State-based password policies also facilitate detection and prevention of malicious attempts to break into these

environments. Password expiration policies and lockout policies based on maximum number of retries are examples of such state-based password policies.

The Oracle Internet Directory plug-in capability can be exploited by customers to implement custom password policies.

Changing Instance Passwords in Oracle Internet Directory

Each application server instance that uses an infrastructure has an entry in Oracle Internet Directory. The instance uses this entry to manage configuration information in Oracle Internet Directory.

Oracle Application Server generates random passwords for the instances in Oracle Internet Directory. You do not need to know what the passwords are, because there are no procedures that you need to run that require the passwords.

However, if your corporate security policy requires that passwords be changed on a regular basis, you can use the `resetiASpasswd` tool to change the password.

> **Note:** You cannot use Oracle Directory Manager, Oracle Delegated Administration Services, or `ldapmodify` to change the instance passwords; you can only use `resetiASpasswd`. The reason for this is that the password needs to be synchronized on the instance host and on Oracle Internet Directory.

To reset the password to a new randomly generated password, execute the following command in the Oracle home of the application server instance whose password you would like to change:

```
(UNIX) ORACLE_HOME/bin/resetiASpasswd.sh cn=orcladmin password ORACLE_HOME
(Windows) ORACLE_HOME\bin\resetiASpasswd cn=orcladmin password ORACLE_HOME
```

password is the `orcladmin` password.

ORACLE_HOME is the full path of the Oracle home for the application server instance. Note that this is identical to the Oracle home in which you run the command.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for full details on password policies and their configuration.

OracleAS Single Sign-On for Authentication

OracleAS Single Sign-On allows users to sign on to the enterprise network once instead of being prompted for sign-on credentials each time they access other Web applications. When you deploy an application with OracleAS Single Sign-On, after the first sign-on, a user's identity is validated by the OracleAS Single Sign-On only once, no matter how many different Oracle Application Server applications the user invokes during a session.

Transparent Sign-On To Non-Oracle Environments

OracleAS Single Sign-On provides two interfaces to transparently integrate with non-Oracle environments in two modes:

- OracleAS Single Sign-On is certified for integration and interoperation with leading third-party authentication services, such as Microsoft Windows and Netegrity SiteMinder.
- OracleAS Single Sign-On supports transparent sign-on to non-Oracle web sites and external applications. In this mode, users can configure their account names and passwords for external applications; OracleAS Single Sign-On uses this information to transparently connect the users to the applications.

In typical enterprise deployments involving numerous Web applications and portals, OracleAS Single Sign-On greatly enhances end-user ease of use.

> **See Also:** *Oracle Application Server Single Sign-On Administrator's Guide* for details on single sign-on.

Secure and Transparent Sign-On To Oracle Database

Middle-tier business intelligence components, such as OracleAS Reports Services and OracleAS Forms Services, must access Oracle Database schema resources on behalf of users who have signed on to the middle-tier. To do so, the components must acquire the end user's account name and password information for relevant database resources. To facilitate this acquisition, Oracle Internet Directory supports an LDAP structure called Resource Access Descriptors, as well as APIs and Oracle Delegated Administration Services interfaces to securely administer this information. This ensures that access is restricted to the end user who owns it and to the applications that need it.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for full details on Resource Access Descriptors.

Delegated Administration and Self-Service Interfaces

Although centralized management of user identities and other security information has its obvious benefits, the process of administration could become unscalable without the means to delegate administration to different sets of administrators for different real-world administrative functions. To support this delegation, the Oracle Delegated Administration Services component of Oracle Identity Management infrastructure defines a delegation model based on Role-Based Access Control (RBAC).

The infrastructure also supports necessary interfaces to implement this model not only for Oracle Identity Management, but also within applications that rely on Oracle Identity Management.

Oracle Delegated Administration Services consists of the following:

- Interfaces for enabling end-user self-service, such as:
  - User password updates, reset, and recovery
  - User preferences and profile management
  - Directory white page lookups
- Interfaces for enabling directory administrator self service such as:
  - Creating and managing users
  - Creating and managing groups
  - Customizing Oracle Delegated Administration Services user and group management interfaces
  - Customizing end-user self-service interface characteristics
  - Oracle Identity Management service-related administration roles

Oracle Delegated Administration Services also supports APIs that applications can use to integrate all the above services in their application-specific administration tools.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for full details on Oracle Delegated Administration Services.

Role-Based Access Control and Privilege Delegation

Many Oracle Application Server components, such as OracleAS Portal, support the Role-Based Access Control (RBAC) model to control access to their resources and operations. The associated application roles are implemented by using the

underlying support of Oracle Internet Directory for managing groups and roles. APIs and Delegated Administration Services interfaces are leveraged by the Oracle Application Server components for managing these objects that represent their application-specific administrative roles.

Installation and Deployment Privileges

Installing and deploying Oracle Application Server components involves creating identities for the applications being deployed and granting them run-time privileges to necessary resources, such as Oracle Application Server infrastructure database schema, and access to other application components. Without proper delegation, deployment of any application would require the directory administrator to be involved. On the other hand, with excessive privilege delegation, an administrator with privileges to deploy one application will also have unwarranted privileges over other applications. With proper delegation, specific administrators can be de granted privileges to specific applications.

The Oracle Application Server installation process supports many predefined roles to streamline the process of deploying Oracle Application Server components by enabling delegation of deployment privileges to application-specific administrators.

> **See Also:** Chapter 5, "Privilege Delegation", and the *Oracle Application Server 10g Installation Guide.*

Provisioning Integration

Provisioning Integration refers to integrating user account creation and privilege assignment tasks for all applications across the enterprise, based on Oracle Identity Management events. These activities are governed by application-specific rules, as well as by enterprise deployment policies. Oracle Identity Management infrastructure supports a feature called Provisioning Integration to facilitate both integration and automation of such provisioning related tasks.

Oracle Application Server components, such as OracleAS Portal and OracleAS Wireless, leverage this capability to be notified of events involving changes to user objects and specific group objects that have direct impact on user accounts and privileges within their environments.

To leverage this service, applications subscribe to directory events that have direct mappings to their application accounts and privileges. Provisioning Integration monitors change events in the directory and notifies applications whose registered interests match this change event.

APIs and configuration interfaces are available for integrating third-party enterprise applications with OracleAS Integration platform.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for full details on application provisioning integration.

Public Key Infrastructure (PKI) and OracleAS Certificate Authority

Oracle Application Server Certificate Authority (OCA) exposes a simple self-service interface for OracleAS Single Sign-On users to provision their own X.509 certificates. With OCA, customers who want to deploy PKI to enable higher levels of security for their environment can do so without incurring significant overhead.

Integrating Third-Party Identity Management Solutions

Oracle Identity Management supports interfaces and procedures to integrate Oracle products with existing third-party identity management solutions in a customer environment. There are three categories of Identity Management integration considerations:

- Integrating Third-Party LDAP Directories and Other Directory Sources
- Integrating Third-Party Single Sign-On Services
- Integrating Third-Party Provisioning Solutions

Integrating Third-Party LDAP Directories and Other Directory Sources

The Directory Integration and Provisioning platform of Oracle Identity Management includes connectors for integration with common commercial LDAP directories, such as Sun iPlanet and Microsoft Active Directory. In addition, interfaces are available to develop custom connectors to any other third-party LDAP directories. The Directory Integration and Provisioning platform also supports connectors for user information stored within SQL-accessible RDBMS tables.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for full details about available connectors and integration methodologies.

Integrating Third-Party Single Sign-On Services

Oracle Identity Management supports certified integration with major single sign-on vendor solutions, such as Netegrity SiteMinder. In addition, OracleAS Single Sign-On provides APIs for seamless single sign-on integration with any third-party authentication service.

> **See Also:** *Oracle Application Server Single Sign-On Administrator's Guide* for full details on third-party single sign-on integration.

Integrating Third-Party Provisioning Solutions

Oracle Identity Management supports certified integration with major third-party provisioning integration solutions. In addition, the Directory Integration and Provisioning platform provides interfaces for integrating with third-party provisioning platforms as well as automating the account provisioning of users for any application in the network.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for full details on supported interfaces for application provisioning integration.

4

Recommended Deployment Topologies

This chapter describes recommended architectures for deploying the Oracle Application Server 10*g* products to secure Internet access. These recommendations have been considerably changed since prior releases. For this reason, this chapter also includes significant detail regarding the criteria used to develop these new architectures.

This chapter presents both the criteria for configuration of firewalls and load balancers and recommended example architectures. You should focus on the criteria rather than the example architectures; although the example architectures will satisfy most customers, the criteria will help you understand how architectures are designed.

This chapter contains the following sections:

The Need for Firewalls and Hardware Load Balancers

Security is becoming increasingly important as more and more Internet-accessible applications are deployed. In the past, nearly all applications were accessible only from intranets whose attackers were limited to employees or contractors. Compared to intranet-only accessible applications, Internet-accessible applications have far larger numbers of potential attackers, who have less to lose and who enjoy a greatly reduced chance of apprehension and punishment.

Internet-accessible sites must now defend themselves against attackers whom they have little chance of locating or punishing. These sites must therefore deploy firewalls and other measures to defend against determined attacks by highly skilled and knowledgeable people.

In addition to enhanced security requirements, Internet-accessible applications often have much higher scale and availability requirements than do intranet-only applications. Internet applications may be accessed by thousands of times more users, while requiring 24x7 operation to accommodate worldwide access. In response to these requirements, hardware load balancers have been developed to meet both the scale and high availability requirements of Internet-accessible applications.

This chapter presents recommended architectures for secure deployment of the core Oracle Application Server products. Although Oracle believes that these configurations will satisfy a large percentage of Oracle's customer base, Oracle makes no claims regarding the suitability of these architectures for specific customer situations. Site managers should use this chapter, especially the criteria noted for particular architectural decisions, as a guide in configuring appropriate architectures for their Internet-accessible applications.

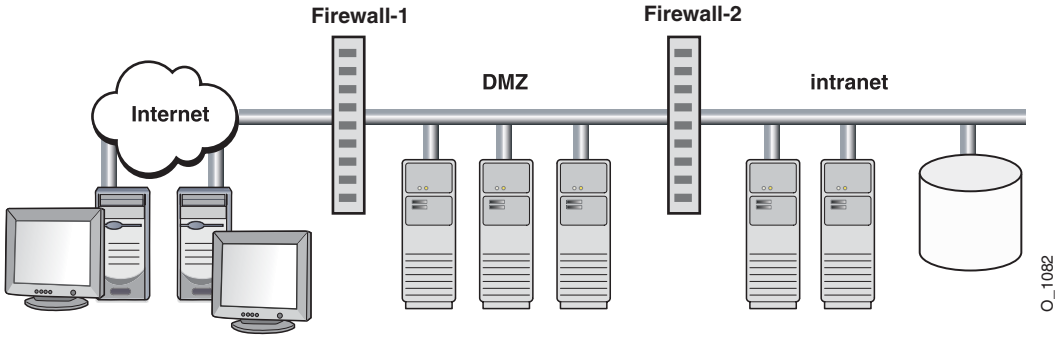
This chapter addresses only application access originating from the Internet. It does not address test, development, or intranet-only applications configurations.

General Architecture and Concepts

The general architectural recommendation is to use the well-known and generally accepted Internet-Firewall-DMZ-Firewall-Internet architecture shown in Figure 4–1.

> **Note:** DMZ stands for De-Militarized Zone, an industry-standard term referring from the Korean War. A DMZ is a server that is isolated by firewalls from both the Internet and the intranet, thus forming a buffer between the two.

Figure 4–1 Traditional DMZ View



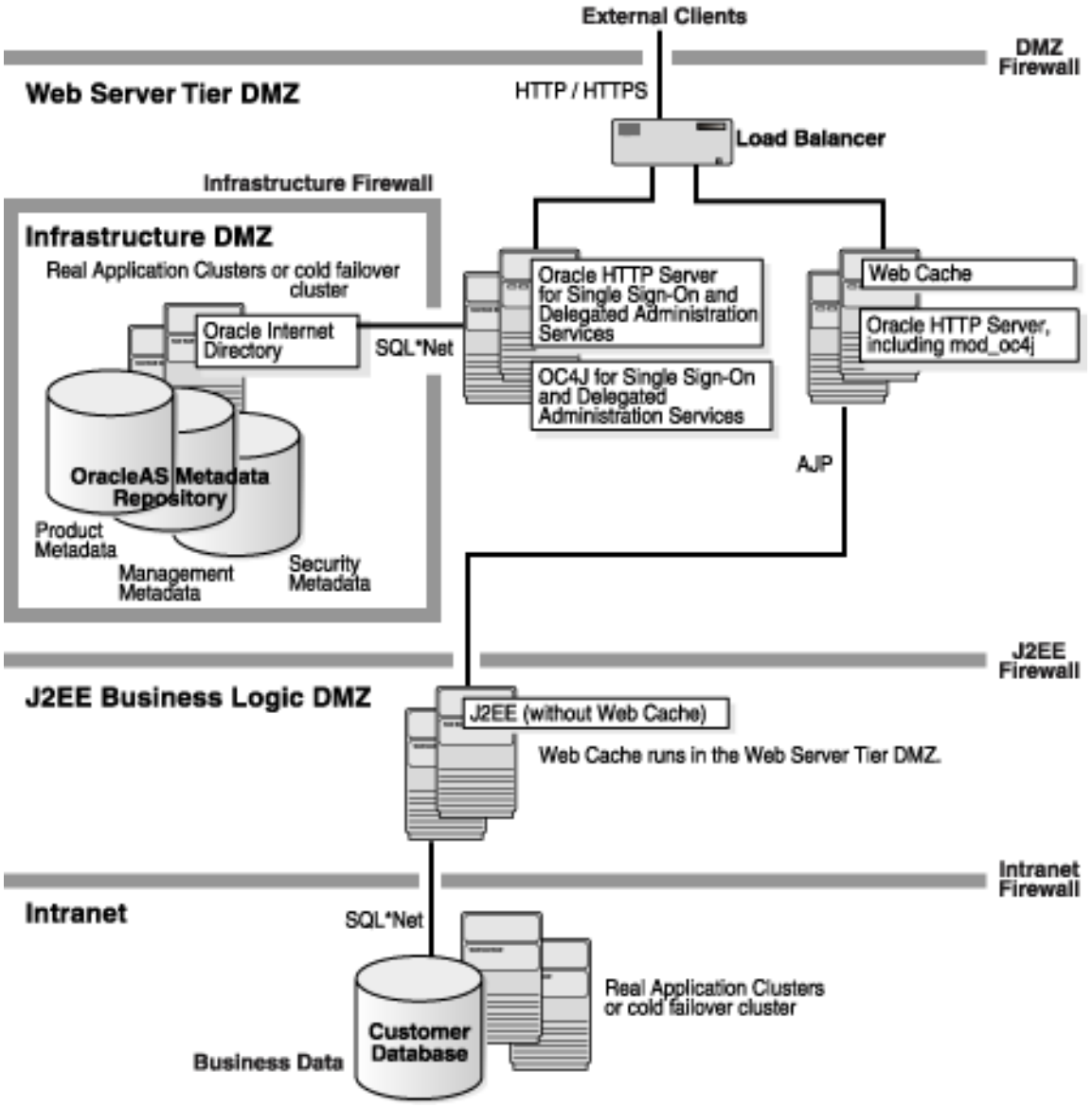
DMZ Zones

In Oracle Application Server 10*g*, the concept of DMZ zones is introduced. In this architecture, the DMZ includes all the zones between the Internet and the intranet. These zones are separated by firewalls. This chapter names these firewalls to indicate the zone they protect from messages arriving from the Internet. Thus, the firewall between the DMZ and the Internet is called the DMZ firewall; the firewall between the DMZ and the Infrastructure databases and meta data is called the Infrastructure Firewall, and so on. (See Figure 4–2, following.)

Firewalls separating DMZ zones provide two essential functions:

- Blocking any traffic types that are known to be illegal
- Providing intrusion containment, should successful intrusions take over processes or processors

Figure 4–2 DMZ Zones



We recommend that your DMZ zones satisfy the following criteria:

- **All incoming Internet HTTP traffic must be processed by HTTP servers in the DMZ zone connected to the Internet**. Because HTTP proxies do not fully process messages and are not a defense against cross-site scripting, directory traversal, and many other attacks, this means that all HTTP servers must reside in this zone, which is called the Web Server Tier zone. Thus, all OracleAS Web Cache (which is a proxy), Oracle HTTP Server and OracleAS Single Sign-On HTTP servers, HTTP load balancers, and HTTPS to HTTP appliances must reside in the DMZ zone.

  Note: If direct Oracle Internet Directory access is required from the Internet, then Oracle Internet Directory servers must reside in the DMZ zone.

- **CPUs that contain HTTP servers should not have direct access to the intranet if possible**. HTTP servers are at most risk for intrusion because of their complexity, because they first process incoming messages, and because hackers tend to focus efforts on these servers. As a result, we recommend a J2EE Business Logic DMZ zone, where OC4J processes that must access the intranet are run. Thus, incoming messages are first processed in the Web Server zone and then forwarded using the AJP protocol to the J2EE zone for processing. OC4J processes may then call business databases in the intranet using SQL*Net.

  We recommend that OC4J processors accessed from the Internet **not** be attached to the intranet. This provides intrusion containment in the event that an OC4J process is taken over. If an OC4J process were taken over, an OC4J processor attached to the intranet would have access to the entire intranet, since there would be no firewall protection.

- **Databases containing various types of metadata and the Oracle Internet Directory database are segregated in an *Infrastructure DMZ zone***. In previous releases, we recommended that processors containing this data reside in the intranet or in the same DMZ zone as HTTP servers. We now recommend placing these processors behind the Infrastructure DMZ firewall in the Infrastructure DMZ zone to protect their sensitive data in the event of Webserver CPU takeover.

Other metadata files have been moved from the intranet to eliminate the requirement of direct HTTP Server-to--intranet access.

> **Note:** Oracle Internet Directory servers should be placed in the Infrastructure DMZ zone if they are not directly accessed from the Internet. If directly accessed from the Internet, they should be placed in the Web Server Tier zone.

Some notes are appropriate:

- Applications that access the business database using `mod_plsql` in Oracle HTTP Server require direct intranet access from the HTTP servers. In this case, the J2EE firewall is eliminated because, with `mod_plsql` access to the business data, that firewall must be configured to allow SQL*Net traffic in any case (see Figure 4–3).
- From a security sense, it is acceptable for intranet-originating traffic to access HTTP servers in the Web Server Tier Zone of the DMZ. This is also true of intranet access to the Oracle Internet Directory servers, either in the Web Server Tier zone or in the Infrastructure DMZ zone. (The general rule is that outgoing messages can always go from more secure to less secure regions.)
- These rules can be used for placing all components. For example, the OracleAS Portal Parallel Page Engine runs as a servlet in the OC4J process. Therefore, it should run in the J2EE business logic DMZ zone.

Configuring DMZ-Based Architectures

In DMZ architectures, firewalls are deployed to ensure that only the traffic that the architecture expects is allowed to cross firewall boundaries. Firewalls also ensure that if intrusion attempts against DMZ processors are successful, the intrusion is contained within the DMZ and to as few holes in the intranet as possible. To achieve this, the component configuration must adhere to the following rules:

- *No* site processors are directly connected to the Internet. All incoming traffic must first be processed by DMZ-attached devices.
- DMZ-attached devices are attached using switched connections, not bussed connections. This ensures that DMZ processes cannot view traffic that does not concern them. Switches that allow IP port and protocol restrictions between each pair of processors, as well as to the Internet and intranet, are best.

- The Internet-to-DMZ firewall does not allow incoming Internet traffic that has sender addresses of DMZ hardware.
- The Internet-to-DMZ firewall prohibits all traffic that does not match the IP port and protocol types expected by the site applications.
- The DMZ-to-intranet firewall allows incoming DMZ-to-intranet messages only if they have DMZ sender addresses.
- The DMZ-to-intranet firewall allows only expected traffic from specific DMZ IP addresses to specific intranet IP port addresses, using the correct protocols for each port.

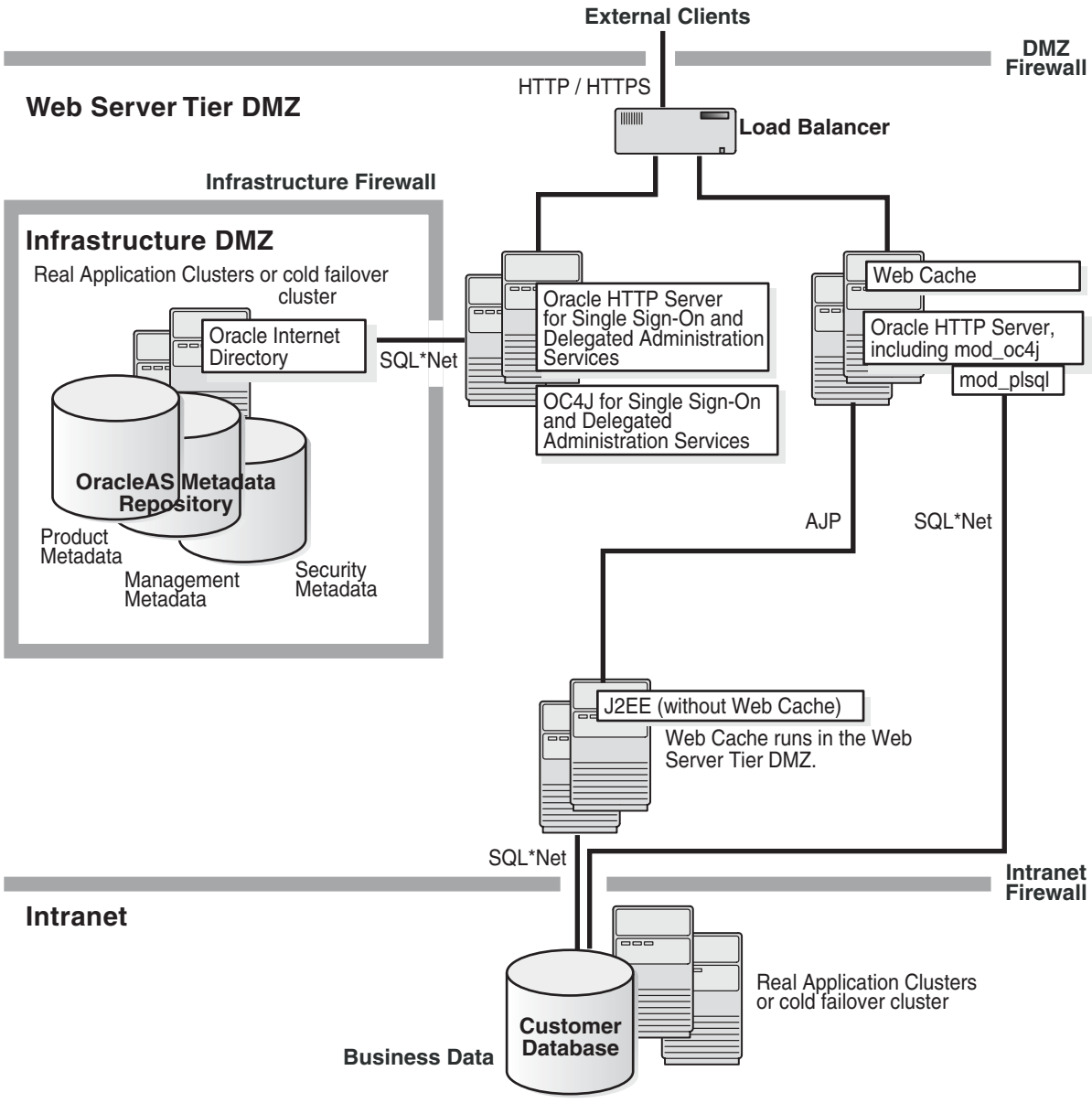
Hardware Load Balancers and HTTPS to HTTP Appliances

Hardware load balancers provide both scalability and high availability and are highly recommended when either of these requirements exists. Because load balancers and HTTPS-to-HTTP appliances are required in a high percentage of production sites, they are described in this chapter.

Generally, load balancers are needed **only** in front of OracleAS Web Cache, non-cached HTTP servers (including the OracleAS Single Sign-On Webserver), and Oracle Internet Directory processes. This is because the Oracle infrastructure provides high scalability and high availability elsewhere, as shown in Figure 4–2 and Figure 4–3.

Load balancers are often used with or contain HTTPS-to-HTTP protocol-converting appliances. These devices can be purchased from a number of vendors and can achieve rates of thousands of SSL key exchange sessions per second or higher. (By comparison, 500MHz Intel/UNIX systems can achieve only 20-30 SSL key exchanges per second, 60-90 exchanges if cryptography accelerator boards are used.) We strongly recommend HTTPS-to-HTTP protocol converting devices. Without these devices, as much as two-thirds of the CPU of a site's HTTP CPU cycles can be consumed by SSL operations—see the results of the SPECweb99_SSL benchmarks.

Figure 4–3 mod_plsql Access to Business Data



Enterprise Data Center Topologies

This section focuses on Enterprise Data Center topologies. These are topologies that are appropriate for production use of Internet-accessible applications. This discussion assumes that security is important and that protection of the intranet and its corporate data is essential.

J2EE Applications

J2EE applications form the heart of many production sites. A recommended architecture for J2EE applications, including Java Beans, servlets and JSPs, is shown in Figure 4–2.

The recommended architecture protects the intranet, because the only incoming access to the intranet is through OC4J processes. This discussion assumes that:

- The load balancer includes any HTTPS-to-HTTP appliances.
- No applications require `mod_plsql` access to the intranet. (Applications are allowed `mod_plsql` access to the Infrastructure DMZ zone.)
- If X.509 client certificates are used, OracleAS Web Cache and Oracle HTTP Server are configured to permit passing certificate information from OracleAS Web Cache to Oracle HTTP Server. The exact configuration differs if OracleAS Web Cache is included in the Oracle HTTP Server processor boxes, as opposed to housing OracleAS Web Cache and Oracle HTTP Server in different processor boxes. For details, see the *Oracle Application Server 10g Installation Guide*.

Mod_plsql Applications

Some applications require access to the corporate data on the intranet using `mod_plsql` modules in Oracle HTTP Server. For these applications, the J2EE zone does not provide any significant added security; the J2EE zone can be combined with the Web Server Tier zone. The reason a J2EE zone provides little added security is that its firewall must allow SQL*Net traffic through from the Web Server Tier. Because the reason for the J2EE zone's firewall is to block SQL*Net traffic, the justification for the firewall is eliminated.

Figure 4–3 provides a recommended architecture for applications that require `mod_plsql` access to the corporate data. This architecture is less secure than the architecture described in Figure 4–2, so application designers should consider alternatives where possible. One alternative is to access J2EE applications, which then make their own calls to SQL*Net. Where `mod_plsql` is used to access intranet

metadata, customers might consider placing such metadata in the Infrastructure DMZ zone.

Figure 4–3 assumes the following:

- Any HTTPS-to-HTTP appliances are included in the load balancers.
- If X.509 client certificates are used, OracleAS Web Cache and Oracle HTTP Server are configured to permit passing certificate information from OracleAS Web Cache to Oracle HTTP Server. The exact configuration differs if OracleAS Web Cache is housed in the same processor box as Oracle HTTP Server, as opposed to housing OracleAS Web Cache and Oracle HTTP Server in different processor boxes. For details, see the *Oracle Application Server 10g Installation Guide*.

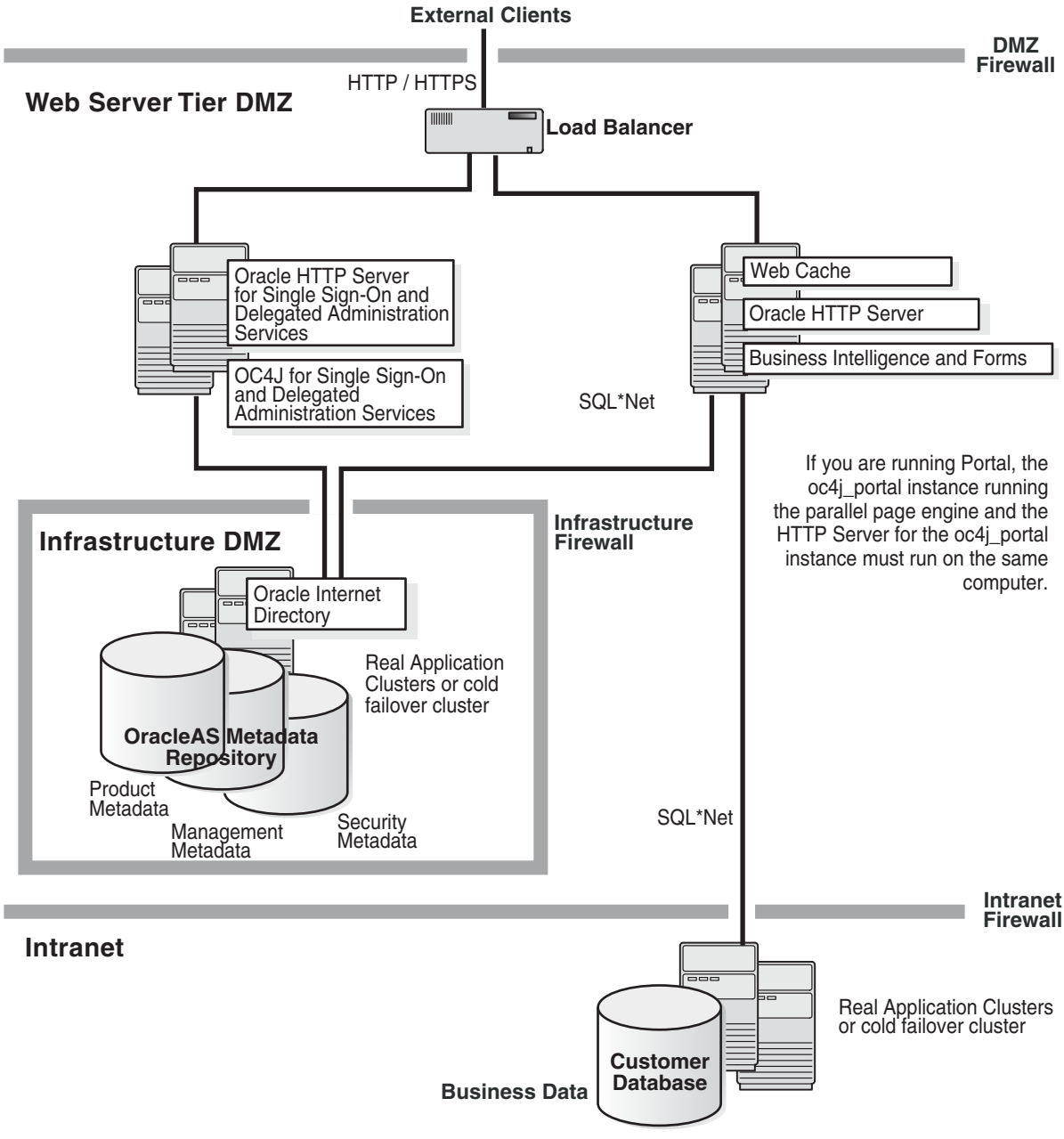
OracleAS Portal, OracleAS Wireless, and Business Intelligence Applications

OracleAS Portal has special requirements because its Oracle HTTP Server process must be housed in the same processor box as its OC4J processes; this technical requirement is unique to OracleAS Portal. Figure 4–4 shows a recommended architecture for OracleAS Portal, as well as OracleAS Wireless and Business Intelligence Applications based on OracleAS Portal. Figure 4–4 assumes the following:

- Any HTTPS-to-HTTP appliances are included in the load balancers.
- If X.509 client certificates are used, OracleAS Web Cache and Oracle HTTP Server are configured to permit passing certificate information from OracleAS Web Cache to Oracle HTTP Server. The exact configuration differs if OracleAS Web Cache is housed in the same processor as Oracle HTTP Server, as opposed to housing OracleAS Web Cache and Oracle HTTP Server in different processors. For details, see the *Oracle Application Server 10g Installation Guide*.
- OracleAS Portal metadata is housed within the Infrastructure DMZ zone.

Note: Because Oracle HTTP Server and OC4J are housed in the same processor box, a separate zone for OC4J processes is impossible.

Figure 4–4 Portal, Wireless and Business Intelligence



OracleAS Forms Services, OracleAS Reports Services, and OracleAS Discoverer Developer Topology

This section discusses deployment architectures for OracleAS Reports Services, OracleAS Forms Services and OracleAS Discoverer. All three of these products have similar architectures: they all use HTTP servers as listeners, use Java Servlets for control, and must communicate to C programs which must be co-located in the same CPU boxes as are the servlets.

In Figure 4–5 and Figure 4–6, the boxes labeled HTTP servers include Oracle HTTP Server, OracleAS Single Sign-On HTTP servers, and Oracle Application Server Web Cache. Load balancers, when required, would be included in front of the HTTP servers. The infrastructure databases, including meta-data repositories, are also part of the architecture.

Note: Although the diagrams do not show this, AJP is used for communication between HTTP servers and OC4J. In the Oracle Application Server 10*g* release, AJP can be SSL-protected with both client-side and server-side authentication.

OracleAS Reports Services Recommended Topology

The reports topology is represented in Figure 4–5. This shows the general infrastructure for J2EE applications, with the addition of the C reports engines.

There are two types of servlets. The first, the Reports servlet, provides control; the second, the Reports Server, communicates with the C Reports engines, which must be co-located with their associated servlets. The Reports engines prepare reports using input from various sources, including the Corporate Database, and then deliver the finished reports to the configured destinations using e-mail, printing, or HTTP.

Figure 4–5 OracleAS Reports Services

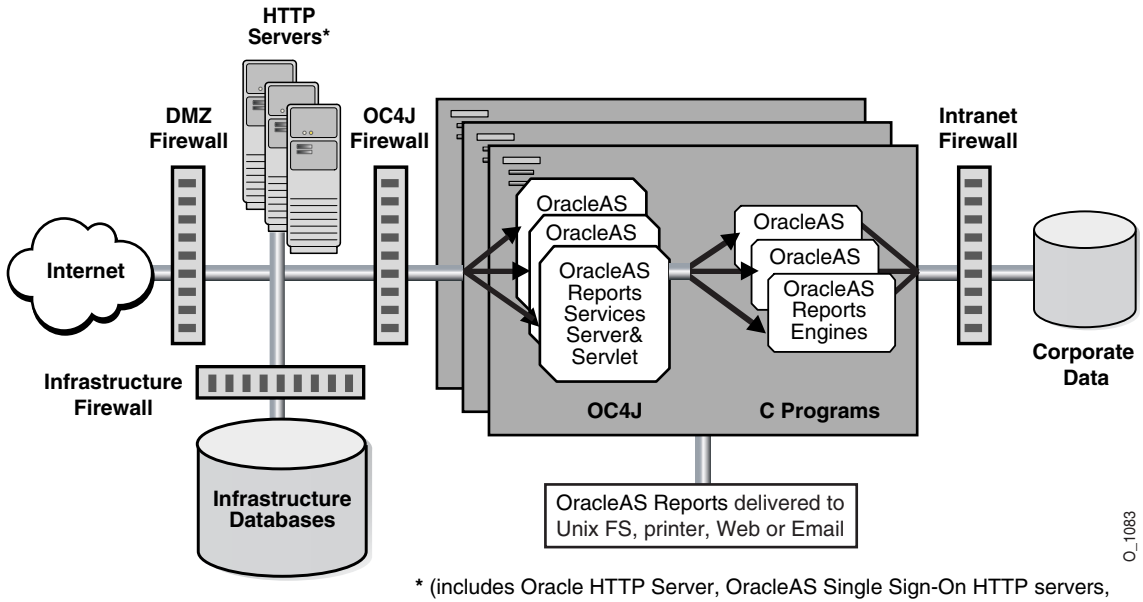


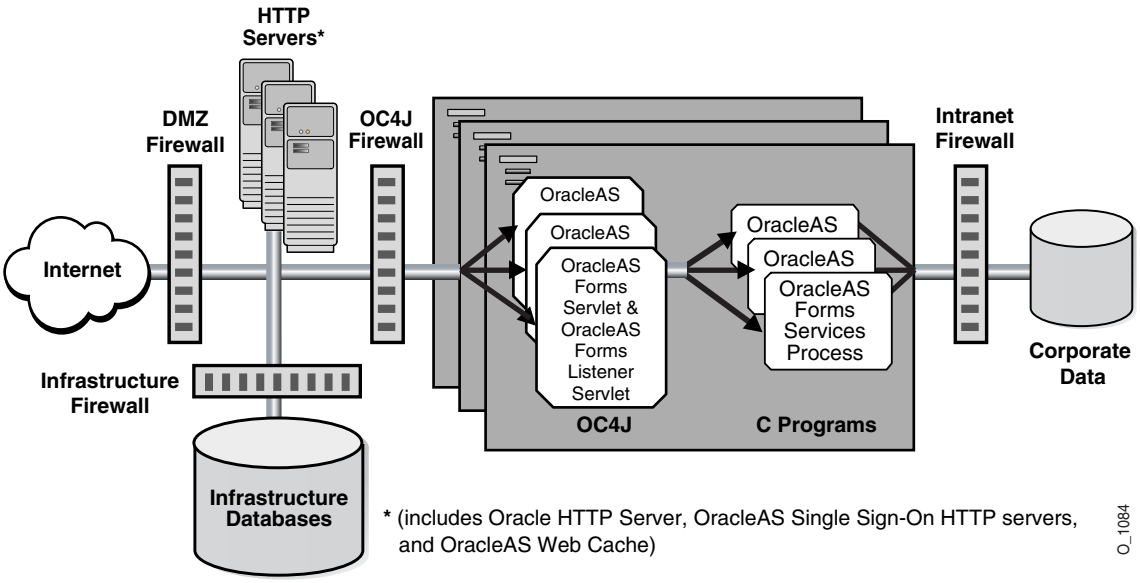
Figure 4–5 shows a recommended architecture. Some configurations of OracleAS Reports Services include CGI processes rather than servlets. Because these configurations are for upward compatibility and not otherwise recommended, they are not shown here.

OracleAS Forms Services Recommended Topology

The OracleAS Forms Services recommended topology is similar to those for both OracleAS Reports Services and OracleAS Discoverer. In the case of OracleAS Forms Services, there are two types of servlet running in OC4J. One is the OracleAS Forms Services servlet, used for control. The second is the OracleAS Forms Services Listener servlet, used in the main data path to communicate with the OracleAS Forms Services process, which is written in C.

Figure 4–6 illustrates a recommended architecture.

Figure 4–6 Forms

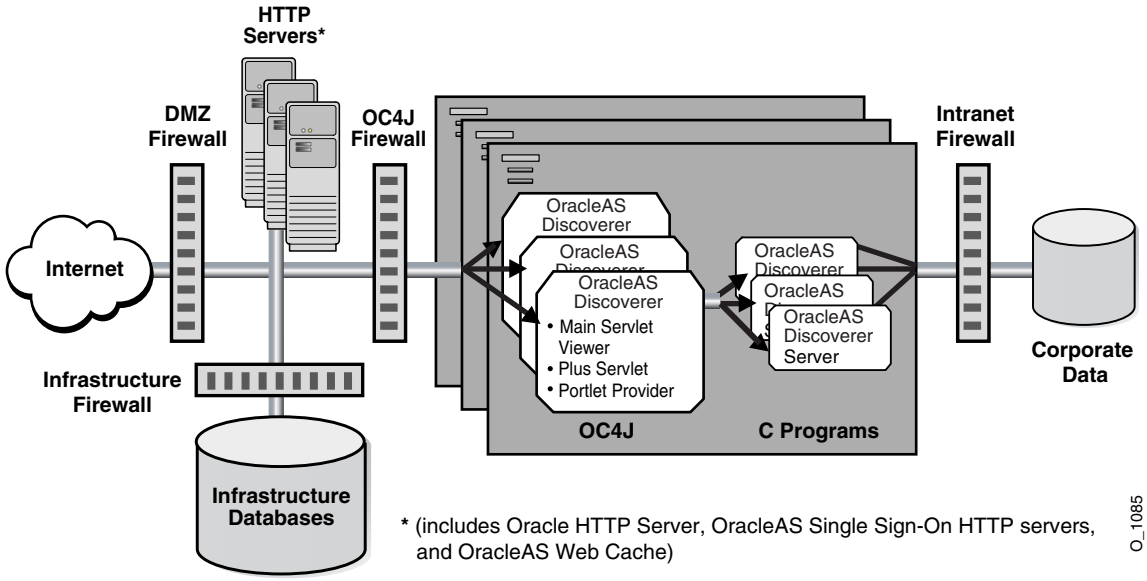


OracleAS Forms Services requires that the Java and C programs must be co-located in the same CPU. However, there can be multiple CPUs containing the Java and associated C processes.

OracleAS Discoverer Recommended Topology

OracleAS Discoverer architecture has the same model as both OracleAS Forms Services and OracleAS Reports Services. However, it is a bit more complex because its Portlet Provider Servlet is used to provide Portal content. In addition to the Portlet Provider, OracleAS Discoverer also has a Main Servlet Viewer for browser clients and a Plus Servlet to accommodate browser palettes. All these servlets communicate to the OracleAS Discoverer server, a C program that generates the content.

Figure 4–7 Oracle Application Server Discoverer



OracleAS Discoverer allows both OracleAS Single Sign-On authentication and OracleAS Discoverer native authentication. OracleAS Single Sign-On authentication is recommended for OracleAS Discoverer, as it is for all Oracle Application Server products. The metadata, including the OracleAS Portal information for the Portlet Provider Servlet, is included in the Infrastructure Databases.

OracleAS Single Sign-On and OracleAS Web Cache Considerations

This section contains considerations for specific Oracle Application Server 10*g* components that were not covered in the previous discussions.

Oracle Application Server Single Sign-On Considerations

The Oracle Application Server Single Sign-On architecture consists of several components. These include Oracle Internet Directory; an Oracle HTTP Server which accommodates OracleAS Single Sign-On requests; OC4J processes where some of the logic is run; `mod_plsql` calls to an infrastructure database; and an infrastructure database. Because OracleAS Single Sign-On is often a resource shared by multiple applications, many organizations may separate OracleAS Single

Sign-On infrastructure from other applications. Where this is not done, OracleAS Single Sign-On should probably have its own processor box, so that it does not share a processor box with higher risk components, such as Oracle HTTP Server, OC4J, or the Oracle Internet Directory server, if that server is in the Web Server Tier DMZ zone.

We recommend that OracleAS Single Sign-On be configured for high availability, especially where it protects multiple applications. Although this is not a security concern, the failure of OracleAS Single Sign-On means that no OracleAS Single Sign-On-protected application can be accessed by new requestors. You should provide fault-tolerant load balancers for OracleAS Single Sign-On's Oracle HTTP Server processes, and configure Oracle Internet Directory and other infrastructure for high availability.

Oracle Application Server Web Cache Considerations

OracleAS Web Cache provides caching, proxy, and load balancing facilities. OracleAS Web Cache should forward HTTP and HTTPS requests only to HTTP servers within the Web Server Tier. The OracleAS Web Cache proxy facility does not protect HTTP servers against many of the common HTTP server attacks, such as cross-site scripting, double encoding, and directory traversal.

5

Privilege Delegation

This chapter discusses Oracle Application Server support for privilege delegation. It contains the following topics:

Introduction

In an enterprise environment, you often deploy multiple applications against a shared infrastructure. For instance, you may have both your HR application and your sales application hosted in the same application server. These separate applications have separate administrators, but both depend on the security infrastructure supplied by the Oracle Internet Directory server.

How Delegation Works

Using the delegation model, a global administrator can delegate to realm administrators the privileges to create and manage the identity management realms for hosted companies. Realm administrators can, in turn, delegate to end users and groups the privileges to change their application passwords, personal data, and preferences. Each type of user can thus be given the appropriate level of privileges.

To delegate the necessary privileges, you assign the user to the appropriate administrative group. For example, suppose that you store data for both enterprise users and the e-mail service in the directory, and need to specify a unique administrator for each set of data. To specify a user as the administrator of enterprise users, you assign that user to, say, the Enterprise User Administrators Group. To specify a user as the administrator of the e-mail services, you assign that user to, say, the E-mail Service Administrators Group.

Delegating Privileges

As Figure 5–1 on page 5-4 shows, in an Oracle Application Server environment the directory super user creates:

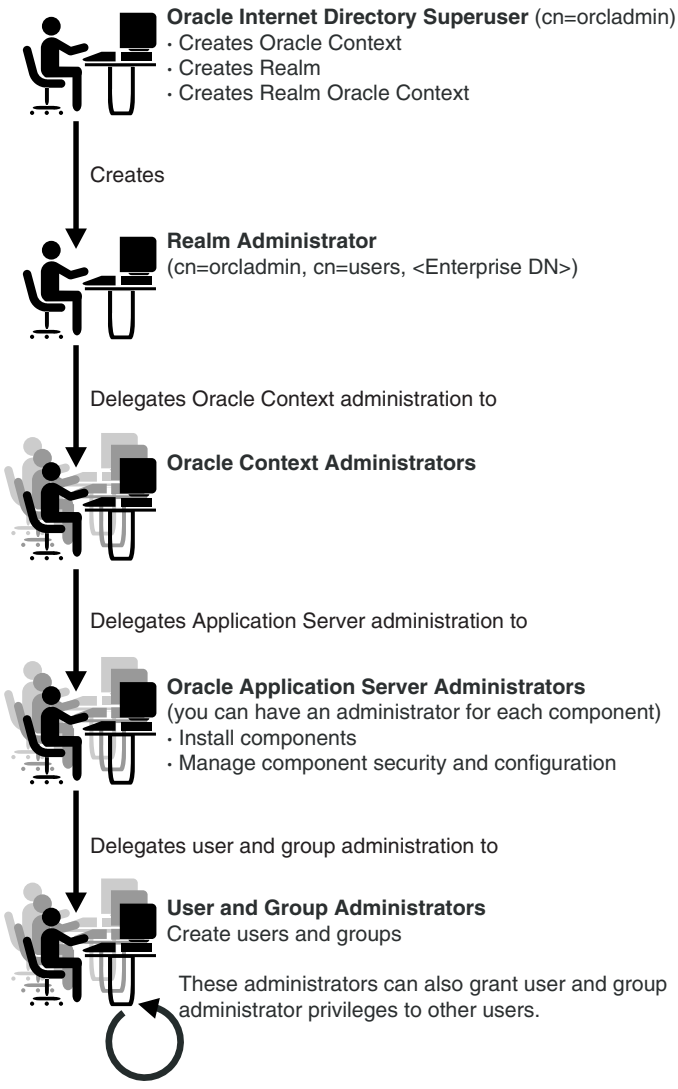
- The Oracle Context
- The realm
- The realm-specific Oracle Context
- The entry for the realm administrator

The realm administrator, in turn, delegates administration of the Oracle Context to specific users by assigning those users to the Oracle Context Administrators Group. Oracle Context Administrators then delegate administration of the Oracle Application Server to one or more users by assigning them to the Oracle Application Server Administrators Group. These administrators install and administer Oracle Application Server components and delegate administration of user and group data to other administrators. The latter can, in turn, delegate others to administer user and group data.

If you are working in an existing Oracle Internet Directory, you must work with the Oracle Internet Directory administrator to ensure that you have the following privileges:

- Administration privileges for Oracle Application Server. This enables you to install and configure Oracle Application Server components.
- Privileges to delegate privileges to other users: This enables you to delegate privileges to application administrators (for example, the OracleAS Portal administrator).

Figure 5–1 Delegation Flow



How Privileges Are Granted for Managing User and Group Data

To delegate administrative privileges, the Oracle Internet Directory super user does the following:

1. Creates an identity management realm
2. Identifies a special user in that realm, the realm administrator
3. Delegates all privileges to that realm administrator

This realm administrator, in turn, delegates certain privileges that Oracle components require to the Oracle defined roles—for example, Oracle Application Server administrators. The Oracle components receive these roles when they are deployed.

In addition to delegating privileges to roles specific to Oracle components, the realm administrator can also define roles specific to the deployment—for example, a role for help desk administrators—and grant privileges to those roles. These delegated administrators can, in turn, grant these roles to end users. In fact, because a majority of user management tasks involve self-service—like changing a phone number or specifying application-specific preferences—these privileges can be delegated to end users by both the realm administrator and Oracle component administrators.

In the case of a group, one or more owners—typically end users—can be identified. If they are granted the necessary administrative privileges, then these owners can manage the group by using Oracle Internet Directory Self-Service Console, Oracle Directory Manager, or command-line tools.

Security Goals for Privilege Model

This release of Oracle Application Server provides fine-grained control over system administration and management privileges. This allows you to:

- Delegate only the privileges necessary for installation and administration
- Grant application administration permissions without making the application administrator an Oracle Internet Directory superuser
- Isolate application installation privileges from application administration privileges
- Encapsulate privileges for each application, so that permission to deploy one component does not grant the right to deploy or administer other components

Roles and Responsibilities

The new privilege model supports the following user roles:

- Oracle Application Server Installation Administrator

  Responsible for installing and uninstalling applications. This administrative privilege is distinct from the next privilege, Oracle Application Server Application Administrator.

- Oracle Application Server Application Administrator

  Responsible for managing the roles and privileges used within an application.

- Oracle Identity Management Infrastructure Administrator

  Responsible for managing Oracle Internet Directory and other Identity Management technologies.

- Oracle Application Server Application User

  Has no responsibilities; runs the application and has only the permissions granted by the application.

Note: The same user may perform multiple roles.

Delegation of Privileges for Component Runtime

Many Oracle components administer user entries in Oracle Internet Directory and need the corresponding privileges. For example:

- When the Oracle Application Server Single Sign-On server authenticates a user, that server:
  - Connects to Oracle Internet Directory using its own identity
  - Verifies that the password entered by the user matches that user's password stored in the directory

  To do this, the Oracle Application Server Single Sign-On server needs permission to compare user passwords. To set up the Oracle Application Server Single Sign-On cookie, it needs permission to read user attributes.
- To grant access to a user, OracleAS Portal must retrieve that user's attributes. To do this, it logs in to Oracle Internet Directory as a proxy user, impersonating the user seeking access. It therefore needs the privileges of a proxy user.

In general, Oracle components can require these privileges:

- Read and modify user passwords
- Compare user passwords
- Proxy on behalf of users accessing applications
- Administer the Oracle Context where all Oracle components store their metadata

> **See Also:** For a comprehensive discussion of privilege delegation, see the *Oracle Internet Directory Administrator's Guide*.

A

Managing PKI Credentials with Oracle Wallet Manager

Security administrators use Oracle Wallet Manager to manage public key security credentials on Oracle clients and servers. The wallets it creates can be read by the Oracle database, Oracle Application Server, and the Oracle Identity Management infrastructure.

This appendix describes Oracle Wallet Manager, and contains the following topics:

Oracle Wallet Manager Overview

Oracle Wallet Manager is an application that wallet owners use to manage and edit the security credentials in their Oracle wallets. A wallet is a password-protected container that is used to store authentication and signing credentials, including private keys, certificates, and trusted certificates needed by SSL. You can use Oracle Wallet Manager to perform basic tasks such as creating wallets, generating certificate requests, and opening wallets to access PKI-based services. In addition, Oracle Wallet Manager can be used to upload wallets to and download them from an LDAP directory. Oracle Wallet Manager can also be used to import third-party **PKCS #12**-format wallets, and export Oracle wallets to a third-party environment.

Oracle Wallet Manager provides the following features:

- Wallet Password Management
- Strong Wallet Encryption
- Microsoft Windows Registry Wallet Storage
- Backward Compatibility
- Public-Key Cryptography Standards (PKCS) Support
- Multiple Certificate Support
- LDAP Directory Support

Wallet Password Management

Oracle wallets are password protected. Oracle Wallet Manager includes an enhanced wallet password management module that enforces Password Management Policy guidelines, including the following:

- Minimum password length (8 characters)
- Maximum password length unlimited
- Alphanumeric character mix required

Strong Wallet Encryption

Oracle Wallet Manager stores private keys associated with X.509 certificates, requiring strong encryption, and uses Triple-DES encryption—a substantially stronger encryption algorithm.

Microsoft Windows Registry Wallet Storage

Oracle Wallet Manager lets you optionally store multiple Oracle wallets in the user profile area of the Microsoft Windows system registry or in a Windows file management system. Storing your wallets in the registry provides the following benefits:

- **Better Access Control.** Wallets stored in the user profile area of the registry are only accessible by the associated user. User access controls for the system thus become, by extension, access controls for the wallets. In addition, when a user logs out of a system, access to that user's wallets is effectively precluded.
- **Easier Administration.** Since wallets are associated with specific user profiles, no file permissions need to be managed, and the wallets stored in the profile are automatically deleted when the user profile is deleted. Oracle Wallet Manager can be used to create and manage the wallets in the registry.

Options Supported:

- Open wallet from the registry
- Save wallet to the registry
- Save As to a different registry location
- Delete wallet from the registry
- Open wallet from the file system and save it to the registry
- Open wallet from the registry and save it to the file system

See Also: *Oracle10i Database Getting Started for Windows*

Backward Compatibility

Oracle Wallet Manager is backward-compatible to Release 8.1.5.

Public-Key Cryptography Standards (PKCS) Support

RSA Laboratories, a division of RSA Security, Inc., has developed, in cooperation with representatives from industry, academia, and government, a family of basic cryptography standards called Public-Key Cryptography Standards, or PKCS for short. These standards have been developed to establish interoperability between computer systems that use public-key technology to secure data across intranets and the Internet.

Oracle Wallet Manager stores X.509 certificates and **private key**s in PKCS #12 format, and generates certificate requests according to the PKCS #10 specification. This makes the Oracle wallet structure interoperable with supported third-party PKI applications, and provides wallet portability across operating systems.

Note: Although Oracle Advanced Security and Oracle Wallet Manager fully comply with PKCS #12, there may be some compatibility issues using third-party products such as Netscape Communicator and Microsoft Internet Explorer.

See Also:

- "Importing Third-Party Wallets" on page A-13
- "Exporting Oracle Wallets to Third-Party Environments" on page A-14
- To view PKCS standards documents, navigate to the following URL: `http://www.rsasecurity.com/rsalabs/PKCS`

Multiple Certificate Support

Oracle Wallet Manager allows you to store multiple **certificate**s for each wallet, supporting the following **Oracle PKI certificate usages**:

- SSL
- S/MIME signature
- S/MIME encryption
- Code-Signing
- CA Certificate Signing

Oracle Wallet Manager supports multiple certificates for a single digital entity, where each certificate can be used for a set of Oracle PKI certificate usages, but the same certificate cannot be used for all such usages (See Table A–2 and Table A–3 for legal usage combinations). There must be a one-to-one mapping between certificate requests and certificates. The same certificate request can be used to obtain multiple certificates; however, more than one certificate for each certificate request cannot be installed in the same wallet at the same time.

Oracle Wallet Manager uses the X.509 Version 3 `KeyUsage` extension to define Oracle PKI certificate usages (Table A–1):

Table A–1 KeyUsage Values

Value	Usage
0	digitalSignature
1	nonRepudiation
2	keyEncipherment
3	dataEncipherment
4	keyAgreement
5	keyCertSign
6	cRLSign
7	encipherOnly
8	decipherOnly

When installing a certificate (user certificate or **trusted certificate**), Oracle Wallet Manager maps the `KeyUsage` extension values to Oracle PKI certificate usages, as specified in Table A–2 and Table A–3.

Table A–2 Oracle Wallet Manager Import of User Certificates to an Oracle Wallet

KeyUsage Value	Critical?[1]	Usage
none	na	Certificate is importable for SSL or S/MIME encryption use.
0 alone, or any combination including 0 but excluding 5 and 2	na	Accept certificate for S/MIME signature or code-signing use.
1 alone	Yes	Not importable.
	No	Accept certificate for S/MIME signature or code-signing use.
2 alone, or 2 + any combination excluding 5	na	Accept certificate for SSL or S/MIME encryption use.
5 alone, or any combination including 5	na	Accept certificate for CA certificate signing use.
Any settings not listed previously	Yes	Not importable.
	No	Certificate is importable for SSL or S/MIME encryption use.

[1] If the `KeyUsage` extension is *critical*, the certificate cannot be used for other purposes.

Table A–3 Oracle Wallet Manager Import of Trusted Certificates to an Oracle Wallet

KeyUsage Value	Critical?[1]	Usage
none	na	Importable.
Any combination excluding 5	Yes	Not importable.
	No	Importable.
5 alone, or any combination including 5	na	Importable.

[1] If the `KeyUsage` extension is *critical*, the certificate cannot be used for other purposes.

You should obtain certificates from the certificate authority with the correct `KeyUsage` value for the required Oracle PKI certificate usage. A single wallet can

contain multiple **key pairs** for the same usage. Each certificate can support multiple Oracle PKI certificate usages, as indicated by Table A–2 and Table A–3. Oracle PKI applications use the first certificate containing the required PKI certificate usage.

For example: For SSL usage, the first certificate containing the SSL Oracle PKI certificate usage is used.

LDAP Directory Support

Oracle Wallet Manager can upload wallets to and retrieve them from an LDAP-compliant directory. Storing wallets in a centralized LDAP-compliant directory lets users access them from multiple locations or devices, ensuring consistent and reliable user authentication while providing centralized wallet management throughout the wallet life cycle. To prevent accidental over-write of functional wallets, only wallets containing an installed certificate can be uploaded.

Enterprise users must be defined and configured in the LDAP directory before Oracle Wallet Manager can be used to upload or download wallets for a user. If a directory contains Oracle8*i* (or prior) users, they are automatically upgraded to use the wallet upload and download feature on first use.

Oracle Wallet Manager downloads a user wallet by using a simple password-based connection to the LDAP directory. However, for uploads it uses an SSL connection if the open wallet contains a certificate with SSL Oracle PKI certificate usage. If an SSL certificate is not present in the wallet, password-based authentication is used.

Note: The directory password and the wallet password are independent, and can be different. Oracle Corporation recommends that these passwords be maintained to be consistently different, where neither one can logically be derived from the other.

See Also:

- Uploading a Wallet to an LDAP Directory on page A-15.
- Downloading a Wallet from an LDAP Directory on page A-16
- Multiple Certificate Support on page A-4, for more information about Oracle PKI certificate usage.

Starting Oracle Wallet Manager

To start Oracle Wallet Manager:

- (Windows) Select **Start** > **Programs** > **Oracle-HOME_NAME** > **Network Administration** > **Wallet Manager**
- (UNIX) At the command line, enter `owm`.

Managing Wallets

This section describes how to create a new wallet and perform associated wallet management tasks, such as generating certificate requests, exporting certificate requests, and importing certificates into wallets, in the following subsections:

- Required Guidelines for Creating Wallet Passwords
- Creating a New Wallet
- Opening an Existing Wallet
- Closing a Wallet
- Importing Third-Party Wallets
- Exporting Oracle Wallets to Third-Party Environments
- Exporting Oracle Wallets to Tools that Do Not Support PKCS #12
- Uploading a Wallet to an LDAP Directory
- Downloading a Wallet from an LDAP Directory
- Saving Changes
- Saving the Open Wallet to a New Location
- Saving in System Default
- Deleting the Wallet
- Changing the Password
- Using Auto Login

Required Guidelines for Creating Wallet Passwords

Because an Oracle wallet contains user credentials that can be used to authenticate the user to multiple databases, it is especially important to choose a strong wallet password. A malicious user who guesses the wallet password can access all the databases to which the wallet owner has access.

Passwords must contain at least eight characters that consist of alphabetic characters combined with numbers or special characters.

Caution: We strongly recommend that users avoid choosing easily guessed passwords based on user names, phone numbers, or government identification numbers, such as "admin0," "oracle1," or "2135551212A." This prevents a potential attacker from using personal information to deduce the users' passwords. It is also a prudent security practice for users to change their passwords periodically, such as once in each month or once in each quarter.

When you change passwords, you must regenerate auto login wallets.

See Also:

- "Wallet Password Management" on page A-2.
- "Using Auto Login" on page A-19

Creating a New Wallet

To create a PKCS #12 wallet that stores credentials in a directory on your file system, perform the following tasks:

1. Choose **Wallet** > **New** from the menu bar. The New Wallet dialog box appears.
2. Follow the "Required Guidelines for Creating Wallet Passwords" on page A-9 and enter a password in the **Wallet Password** field. This password protects unauthorized use of your credentials.
3. Re-enter that password in the **Confirm Password** field.

4. Click **OK** to continue. If the entered password does not conform to the required guidelines, then the following message appears:

   ```
   Password must have a minimum length of eight characters, and contain
   alphabetic characters combined with numbers or special characters. Do you
   want to try again?
   ```

5. An alert is displayed, and informs you that a new empty wallet has been created. It prompts you to decide whether you want to add a certificate request. See "Adding a Certificate Request" on page A-21.

   If you choose **No**, you are returned to the Oracle Wallet Manager main window. The new wallet you just created appears in the left window pane. The certificate has a status of **[Empty]**, and the wallet displays its default trusted certificates.

6. Select **Wallet** > **Save In System Default** to save the new wallet.

   If you do not have permission to save the wallet in the system default, you can save it to another location. This location must be used in the SSL configuration for clients and servers.

   A message at the bottom of the window confirms that the wallet was successfully saved.

Enabling Wallets to Open on Windows

On Windows platforms, if you are using Oracle Application Server Web Cache in a standalone environment, you must take special actions to enable wallets to open.

Oracle Application Server Web Cache attempts to open wallets at startup. On Windows, wallets are protected so that only the user that created them can open and use them. In a standalone environment, the Oracle Application Server Web Cache admin and cache processes are Windows services. By default, Oracle Application Server services, including Oracle Application Server Web Cache, are associated with the local system account, which does not have permission to open wallets.

To enable Oracle Application Server Web Cache to open wallets at startup:

1. Create a wallet with an administrator account.
2. Change the system account information for the Oracle Application Server services, as described in Table A–4.

Table A–4 Wallet Configuration on Windows

Windows NT	Windows 2000
1. Choose the **Services** icon from the Control Panel window. The Services window appears. 2. Select the `Oracle`*`HOME_NAME`*`WebCache` service. The Service dialog appears. 3. Choose **This Account**. By default, the LocalSystem user account is associated with the service. 4. Choose the ellipse (**...**) next to **This Account**. The Add User dialog box appears. 5. Select the user that created the wallet from the Names list, and then click **Add**. 6. Click **OK** to close the Add User dialog box. 7. In the Service dialog box, provide the password for the wallet administrator in the **Password** field, and then confirm the password in the **Confirm Password** field. 8. In the Services dialog box, click **OK**. 9. Repeat Steps 3 - 8 for the `Oracle`*`HOME_NAME`*`WebCacheAdmin service`. 10. In the Services window, click **Close**.	1. Choose **Administrative Tools** > **Services** from the Control Panel window. The Services window appears. 2. Select the `Oracle`*`HOME_NAME`*`WebCache` service. The Oracle*HOME_NAME*WebCache Properties dialog appears. 3. Choose the **Log On** tab. 4. In the Log On tab, choose **This account**. By default, the LocalSystem user account is associated with the service. 5. Choose **Browse** next to **This Account**. The Select User dialog box appears. 6. Select the user that created the wallet from the list, and then click **OK**. 7. Click **OK** to close the Add User dialog box. 8. In the Oracle*HOME_NAME*WebCache Properties dialog box, provide the password for the wallet administrator in the **Password** field, and then confirm the password in the **Confirm Password** field. 9. In the Services dialog box, click **OK**. 10. Repeat Steps 3 - 9 for the `Oracle`*`HOME_NAME`*`WebCacheAdmin` service.

On Windows NT, additionally grant the wallet administrator the right to run Oracle Application Server Web Cache as a service:

1. Choose **Start** > **Programs** > **Administrative Tools** > **User Manager**.

   The User Manager window appears.

2. Select the wallet administration, and then choose **Policies** > **User Rights**.

   The User Rights Policy dialog box appears.

3. Choose the **Show Advanced User Rights** check box, and then select **Log on as a service** from the **Right** list.

4. Select **Users** from the **Grant To** list.

   If **Users** does not exist, create it:

   a. Click **Add**.

      The Add Users and Groups dialog box appears:

   b. Select the name of the local host computer from the **List Names From** list.

   c. Select **Users** from the **Names** list, and then click **Add**.

   d. Click **OK**.

      Users appears in the **Grant To** list.

5. Click **OK** in the User Rights Policy dialog box.

   The User Manager window reappears.

6. Choose **User** > **Exit**.

Opening an Existing Wallet

Open a wallet that already exists in the file system directory as follows:

1. Choose **Wallet** > **Open** from the menu bar. The Select Directory dialog box appears.
2. Navigate to the directory location in which the wallet is located, and select the directory.
3. Choose **OK**. The Open Wallet dialog box appears.
4. Enter the wallet password in the **Wallet Password** field.
5. Choose **OK**.

   You are returned to the main window and a message appears at the bottom of the window indicating the wallet was opened successfully. The wallet's certificate and its trusted certificates are displayed in the left window pane.

Closing a Wallet

To close an open wallet in the currently selected directory:

Choose **Wallet** > **Close**.

A message appears at the bottom of the window to confirm that the wallet is closed.

Importing Third-Party Wallets

Third-party wallets are those where the certificate requests have been generated without using Oracle Wallet Manager. Oracle Wallet Manager can import and support the following PKCS #12-format wallets, subject to procedures and limitations specific to the program you use:

- Netscape Communicator 4.x
- Microsoft Internet Explorer 5.x and later
- OpenSSL

To import a third-party wallet, perform the following tasks:

1. Follow the procedures for your particular product to export the wallet.
2. Save the exported wallet to a file name appropriate for your operating system in a directory expected by Oracle Advanced Security.

   For UNIX and Windows, the appropriate file name is `ewallet.p12`.

   For other operating systems, see the Oracle documentation for that specific operating system.

Note: Because browsers typically do not export **trusted certificates** under PKCS #12 (other than the signer's own certificate), you may need to add trust points to authenticate the other party in the SSL connection. You can use Oracle Wallet Manager to import trusted certificates.

See Also: "Importing a Trusted Certificate" on page A-25

Exporting Oracle Wallets to Third-Party Environments

Oracle Wallet Manager can export its own wallets to third-party environments.

To export a wallet to third-party environments:

1. Use Oracle Wallet Manager to save the wallet file.
2. Follow the procedure specific to your third-party product to import an operating system PKCS #12 wallet file created by Oracle Wallet Manager (called `ewallet.p12` on UNIX and Windows platforms).

Note:

- Oracle Wallet Manager supports multiple certificates for each wallet, yet current browsers typically support import of single-certificate wallets only. For these browsers, you must export an Oracle wallet containing a single key-pair.
- Oracle Wallet Manager supports wallet export to only Netscape Communicator 4.7.2 and later, OpenSSL, and Microsoft Internet Explorer 5.0 and later.

Exporting Oracle Wallets to Tools that Do Not Support PKCS #12

You can export a wallet to a text-based PKI format if you want to put a wallet into a tool that does not support PKCS #12. Individual components are formatted according to the standards listed in Table A–5. Within the wallet, only those certificates with SSL key usage are exported with the wallet.

To export a wallet to text-based PKI format:

1. Choose **Operations** > **Export Wallet...**. The Export Wallet dialog box appears.
2. Enter the destination file system directory for the wallet, or navigate to the directory structure under **Folders**.
3. Enter the destination file name for the wallet.
4. Choose **OK** to return to the main window.

Table A–5 PKI Wallet Encoding Standards

Component	Encoding Standard
Certificate chains	X509v3
Trusted certificates	X509v3
Private keys	PKCS #8

Uploading a Wallet to an LDAP Directory

To upload a wallet to an LDAP directory, Oracle Wallet Manager uses SSL if the specified wallet contains an SSL certificate. Otherwise, it lets you enter the directory password.

To prevent accidental destruction of your wallet, Oracle Wallet Manager will not permit you to execute the upload option unless the target wallet is currently open and contains at least one user certificate.

To upload a wallet:

1. Choose **Wallet** > **Upload Into The Directory Service...**. If the currently open wallet has not been saved, a dialog box appears with the following message:

   Wallet needs to be saved before uploading.

   Choose **Yes** to proceed.

2. Wallet certificates are checked for `SSL` key usage. Depending on whether a certificate with `SSL` key usage is found in the wallet, one of the following results occur:

   - **If at least one certificate has `SSL` key usage:** When prompted, enter the LDAP directory server hostname and port information, then click **OK**. Oracle Wallet Manager attempts connection to the LDAP directory server using SSL.A message appears indicating whether the wallet was uploaded successfully or it failed.
   - **If no certificates have `SSL` key usage:** When prompted, enter the user's **distinguished name (DN)**, the LDAP server hostname and port information, and click **OK**. Oracle Wallet Manager attempts connection to the LDAP directory server using simple password authentication mode, assuming that the wallet password is the same as the directory password.

     If the connection fails, a dialog box prompts for the directory password of the specified DN. Oracle Wallet Manager attempts connection to the LDAP directory server using this password and displays a warning message if the

attempt fails. Otherwise, Oracle Wallet Manager displays a status message at the bottom of the window indicating that the upload was successful.

Downloading a Wallet from an LDAP Directory

When a wallet is downloaded from an LDAP directory, it is resident in working memory. It is not saved to the file system unless you expressly save it using any of the Save options described in the following sections.

See Also:

- "Saving Changes" on page A-17
- "Saving the Open Wallet to a New Location" on page A-17
- "Saving in System Default" on page A-18

To download a wallet from an LDAP directory:

1. Choose **Wallet** > **Download From The Directory Service...**.
2. A dialog box prompts for the user's distinguished name (DN), and the LDAP directory password, hostname, and port information. Oracle Wallet Manager uses simple password authentication to connect to the LDAP directory.

   Depending on whether the downloading operation succeeds or not, one of the following results occurs:

   - **If the download operation fails:** Check to make sure that you have correctly entered the user's DN, and the LDAP server hostname and port information.
   - **If the download is successful:** Choose **OK** to open the downloaded wallet. Oracle Wallet Manager attempts to open that wallet using the directory password. If the operation fails after using the directory password, then a dialog box prompts for the wallet password.

     If Oracle Wallet Manager cannot open the target wallet using the wallet password, then check to make sure you entered the correct password. Otherwise a message displays at the bottom of the window, indicating that the wallet was downloaded successfully.

Saving Changes

To save your changes to the current open wallet:

Choose **Wallet** > **Save**.

A message at the bottom of the window confirms that the wallet changes were successfully saved to the wallet in the selected directory location.

Saving the Open Wallet to a New Location

To save open wallets to a new location, use the **Save As...** menu option:

1. Choose **Wallet** > **Save As...**. The Select Directory dialog box appears.
2. Select a directory location in which to save the wallet.
3. Choose **OK**.

   The following message appears if a wallet already exists in the selected location:

   ```
   A wallet already exists in the selected path. Do you want to overwrite
   it?
   ```

   Choose **Yes** to overwrite the existing wallet, or **No** to save the wallet to another location.

   A message at the bottom of the window confirms that the wallet was successfully saved to the selected directory location.

Saving in System Default

To save wallets in the default directory location, use the **Save In System Default** menu option:

Choose **Wallet** > **Save In System Default**.

A message at the bottom of the window confirms that the wallet was successfully saved in the system default wallet location as follows for UNIX and Windows platforms:

- (UNIX) *ORACLE_HOME*/admin/*ORACLE_SID*
- (Windows) *ORACLE_BASE**ORACLE_HOME*\rdbms\admin

Note:

- SSL uses the wallet that is saved in the system default directory location
- Some Oracle applications are not able to use the wallet if it is not in the system default location. Check the Oracle documentation for your specific application to determine whether wallets must be placed in the default wallet directory location.

Deleting the Wallet

To delete the current open wallet:

1. Choose **Wallet** > **Delete**. The Delete Wallet dialog box appears.
2. Review the displayed wallet location to verify you are deleting the correct wallet.
3. Enter the wallet password.
4. Choose **OK**. A dialog panel appears to inform you that the wallet was successfully deleted.

Note: Any open wallet in application memory will remain in memory until the application exits. Therefore, deleting a wallet that is currently in use does not immediately affect system operation.

Changing the Password

A password change is effective immediately. The wallet is saved to the currently selected directory, with the new encrypted password.

> **Note:** If you are using a wallet with auto login enabled, you must regenerate the auto login wallet after changing the password. See "Using Auto Login" on page A-19

To change the password for the current open wallet:

1. Choose **Wallet** > **Change Password**. The Change Wallet Password dialog box appears.
2. Enter the existing wallet password.
3. Enter the new password.
4. Re-enter the new password.
5. Choose **OK**.

A message at the bottom of the window confirms that the password was successfully changed.

See Also:

- "Required Guidelines for Creating Wallet Passwords" on page A-9
- "Wallet Password Management" on page A-2, for password policy restrictions.

Using Auto Login

The Oracle Wallet Manager auto login feature creates an obfuscated copy of the wallet and enables PKI-based access to services without a password until the auto login feature is disabled for the wallet. The file system permissions provide the necessary security for auto login wallets. When auto login is enabled for a wallet, it is only available to the operating system user who created that wallet.

You must enable auto login if you want single sign-on access to multiple Oracle databases, which is disabled by default. Sometimes these are called “OracleAS Single Sign-On wallets” because they provide single sign-on capability.

Enabling Auto Login

To enable auto login:

1. Choose **Wallet** from the menu bar.
2. Check **Auto Login**. A message at the bottom of the window indicates that auto login is enabled.

Disabling Auto Login

To disable auto login:

1. Choose **Wallet** from the menu bar.
2. Uncheck **Auto Login**. A message at the bottom of the window indicates that auto login is disabled.

Managing Certificates

Oracle Wallet Manager uses two kinds of certificates: user certificates and trusted certificates. All certificates are signed data structures that bind a network identity with a corresponding public key. User certificates are used by end entities, including server applications, to validate an end entity's identity in a public key/private key exchange. In comparison, trusted certificates are any certificates that you trust, such as those provided by **CA**s to validate the user certificates that they issue.

This section describes how to manage both certificate types, in the following subsections:

- Managing User Certificates
- Managing Trusted Certificates

> **Note:** You must first install a trusted certificate from the certificate authority before you can install a user certificate issued by that authority. Several trusted certificates are installed by default when you create a new wallet.

Managing User Certificates

User certificates can be used by end users, smart cards, or applications, such as Web servers. Server certificates are a type of user certificate. For example, if a CA issues a certificate for a Web server, placing its **distinguished name (DN)** in the Subject

field, then the Web server is the certificate owner, thus the "user" for this user certificate. User certificates do not validate other user certificates, except when they are used as a trusted certificate in a *user-centric* trust model.

> **See Also:** *Understanding Public-Key Infrastructure*, a third-party publication, listed in the Preface under "Related Documentation" on page -xiv, for a discussion of user-centric and other trust models.

Managing user certificates involves the following tasks:

- Adding a Certificate Request
- Importing the User Certificate into the Wallet
- Removing a User Certificate from a Wallet
- Removing a Certificate Request
- Exporting a User Certificate
- Exporting a User Certificate Request

Adding a Certificate Request

You can add multiple certificate requests with Oracle Wallet Manager. When adding multiple requests, Oracle Wallet Manager automatically populates each subsequent request dialog box with the content of the initial request that you can then edit.

The actual certificate request becomes part of the wallet. You can reuse any certificate request to obtain a new certificate. However, you cannot edit an existing certificate request. Store only a correctly filled out certificate request in a wallet.

To create a PKCS #10 certificate request:

1. Choose **Operations** > **Add Certificate Request**. The Add Certificate Request dialog box appears.
2. Enter the information specified in Table A–6.
3. Choose **OK**. A message informs you that a certificate request was successfully created. You can either copy the certificate request text from the body of this dialog panel and paste it into an e-mail message to send to a certificate authority, or you can export the certificate request to a file.
4. Choose **OK** to return to the Oracle Wallet Manager main window. The status of the certificate changes to **[Requested]**.

> **See Also:** "Exporting a User Certificate Request" on page A-25

Table A–6 Certificate Request: Fields and Descriptions

Field Name	Description
Common Name	Mandatory. Enter the name of the user's or service's identity. Enter a user's name in first name /last name format. Example: Eileen.Sanger
Organizational Unit	Optional. Enter the name of the identity's organizational unit. Example: Finance.
Organization	Optional.Enter the name of the identity's organization. Example: XYZ Corp.
Locality/City	Optional. Enter the name of the locality or city in which the identity resides.
State/Province	Optional. Enter the full name of the state or province in which the identity resides. Enter the full state name, because some certificate authorities do not accept two–letter abbreviations.
Country	Mandatory. Choose to view a list of country abbreviations. Select the country in which the organization is located.
Key Size	Mandatory. Choose to view a list of key sizes to use when creating the public/private key pair. See Table A–7 to evaluate key size.
Advanced	Optional. Choose **Advanced** to view the Advanced Certificate Request dialog panel. Use this field to edit or customize the identity's distinguished name (DN). For example, you can edit the full state name and locality.

Table A–7 lists the available key sizes and the relative security each size provides. Typically, CAs use key sizes of 1024 or 2048. When certificate owners wish to keep their keys for a longer duration, they choose 3072 or 4096 bit keys.

Table A–7 Available Key Sizes

Key Size	Relative Security Level
512 or 768	Not regarded as secure.
1024 or 2048	Secure.
3072 or 4096	Very secure.

Importing the User Certificate into the Wallet

The certificate authority sends you an e-mail notification when your certificate request has been fulfilled. Import the certificate into a wallet in either of two ways: copy and paste the certificate from the certificate authority's e-mail, or import the user certificate from a file.

To copy and paste the user certificate from the certificate authority's e-mail:

1. Copy the certificate text from the e-mail message or file you receive from the certificate authority. Include the lines `Begin Certificate` and `End Certificate.`
2. Choose **Operations** > **Import User Certificate...**. The Import Certificate dialog box appears.
3. Choose **Paste the certificate**, and then click **OK**. Another Import Certificate dialog box appears with the following message:

   ```
   Please provide a base64 format certificate and paste it below.
   ```

4. Paste the certificate into the dialog box, and choose **OK**. A message at the bottom of the window confirms that the certificate was successfully installed. You are returned to the Oracle Wallet Manager main panel, and the status of the corresponding entry in the left panel subtree changes to **[Ready]**.

Keyboard shortcuts for copying and pasting certificates:

Use Ctrl+c to copy, and use Ctrl+v to paste.

To import a file that contains the user certificate:

The file containing the user certificate should have been saved in BASE64 format.

1. Choose **Operations** > **Import User Certificate...**. The Import Certificate dialog box appears.
2. Choose **Select a file that contains the certificate**, and click **OK**. Another Import Certificate dialog box appears.
3. Enter the path or folder name of the certificate file location.
4. Select the name of the certificate file (for example, `cert.txt`).
5. Choose **OK**. A message at the bottom of the window confirms that the certificate was successfully installed. You are returned to the Oracle Wallet

Manager main panel, and the status of the corresponding entry in the left panel subtree changes to **[Ready]**.

Removing a User Certificate from a Wallet

To remove a user certificate from a wallet:

1. In the left panel subtree, select the certificate that you want to remove.
2. Choose **Operations** > **Remove User Certificate...**. A dialog panel appears and prompts you to verify that you want to remove the user certificate from the wallet.
3. Choose **Yes** to return to the Oracle Wallet Manager main panel. The certificate displays a status of **[Requested]**.

Removing a Certificate Request

You must remove a certificate before removing its associated request.

To remove a certificate request:

1. In the left panel subtree, select the certificate request that you want to remove.
2. Choose **Operations** > **Remove Certificate Request...**.
3. Click **Yes**. The certificate displays a status of **[Empty]**.

Exporting a User Certificate

To save the certificate in a file system directory, export the certificate by using the following steps:

1. In the left panel subtree, select the certificate that you want to export.
2. Choose **Operations** > **Export User Certificate...** from the menu bar. The Export Certificate dialog box appears.
3. Enter the file system directory location where you want to save your certificate, or navigate to the directory structure under **Folders**.
4. Enter a file name for your certificate in the **Enter File Name** field.
5. Choose **OK**. A message at the bottom of the window confirms that the certificate was successfully exported to the file. You are returned to the Oracle Wallet Manager main window.

Exporting a User Certificate Request

To save the certificate request in a file system directory, export the certificate request by using the following steps:

1. In the left panel subtree, select the certificate request that you want to export.
2. Choose **Operations** > **Export Certificate Request...**. The Export Certificate Request dialog box appears.
3. Enter the file system directory location where you want to save your certificate request, or navigate to the directory structure under **Folders**.
4. Enter a file name for your certificate request, in the **Enter File Name** field.
5. Choose **OK**. A message at the bottom of the window confirms that the certificate request was successfully exported to the file. You are returned to the Oracle Wallet Manager main window.

Managing Trusted Certificates

Managing trusted certificates includes the following tasks:

- Importing a Trusted Certificate
- Removing a Trusted Certificate
- Exporting a Trusted Certificate
- Exporting All Trusted Certificates

Importing a Trusted Certificate

You can import a trusted certificate into a wallet in either of two ways: paste the trusted certificate from an e-mail that you receive from the certificate authority, or import the trusted certificate from a file.

Oracle Wallet Manager automatically installs trusted certificates from VeriSign, RSA, Entrust, and GTE CyberTrust when you create a new wallet.

To paste the trusted certificate:

1. Copy the trusted certificate from the body of the e-mail message you received that contained the user certificate. Include the lines `Begin Certificate` and `End Certificate.`
2. Choose **Operations** > **Import Trusted Certificate...** from the menu bar. The Import Trusted Certificate dialog panel appears.

3. Choose **Paste the Certificate**, and click **OK**. Another Import Trusted Certificate dialog panel appears with the following message:

   ```
   Please provide a base64 format certificate and paste it below.
   ```

4. Paste the certificate into the window, and click **OK**. A message at the bottom of the window informs you that the trusted certificate was successfully installed.
5. Choose **OK**. You are returned to the Oracle Wallet Manager main panel, and the trusted certificate appears at the bottom of the Trusted Certificates tree.

Keyboard shortcuts for copying and pasting certificates:

Use Ctrl+c to copy, and use Ctrl+v to paste.

To import a file that contains the trusted certificate:

1. Choose **Operations** > **Import Trusted Certificate...**. The Import Trusted Certificate dialog panel appears.
2. Enter the path or folder name of the trusted certificate location.
3. Select the name of the trusted certificate file (for example, `cert.txt`).
4. Choose **OK**. A message at the bottom of the window informs you that the trusted certificate was successfully imported into the wallet.
5. Choose **OK** to exit the dialog panel. You are returned to the Oracle Wallet Manager main panel, and the trusted certificate appears at the bottom of the Trusted Certificates tree.

Removing a Trusted Certificate

You cannot remove a trusted certificate if it has been used to sign a user certificate still present in the wallet. To remove such trusted certificates, you must first remove the certificates it has signed. Also, you cannot verify a certificate after its trusted certificate has been removed from your wallet.

To remove a trusted certificate from a wallet:

1. Select the trusted certificate listed in the Trusted Certificates tree.
2. Choose **Operations** > **Remove Trusted Certificate...** from the menu bar.

   A dialog panel warns you that your user certificate will no longer be verifiable by its recipients if you remove the trusted certificate that was used to sign it.

3. Choose **Yes**. The selected trusted certificate is removed from the Trusted Certificates tree.

Exporting a Trusted Certificate

To export a trusted certificate to another file system location:

1. In the left panel subtree, select the trusted certificate that you want to export.
2. Select **Operations** > **Export Trusted Certificate...**. The Export Trusted Certificate dialog box appears.
3. Enter a file system directory in which you want to save your trusted certificate, or navigate to the directory structure under **Folders**.
4. Enter a file name to save your trusted certificate.
5. Choose **OK**. You are returned to the Oracle Wallet Manager main window.

Exporting All Trusted Certificates

To export all of your trusted certificates to another file system location:

1. Choose **Operations** > **Export All Trusted Certificates...**. The Export Trusted Certificate dialog box appears.
2. Enter a file system directory location where you want to save your trusted certificates, or navigate to the directory structure under **Folders**.
3. Enter a file name to save your trusted certificates.
4. Choose **OK**. You are returned to the Oracle Wallet Manager main window.

Using OracleAS Certificate Authority Certificates

Oracle Wallet Manager can be used to manage certificates issued by OracleAS Certificate Authority like any other credentials that comply with PKCS #12.

To use certificates from OracleAS Certificate Authority:

1. Copy the OracleAS Certificate Authority trusted certificate (certificate authority certificate) and import it into Oracle Wallet Manager by using the method described in "Importing a Trusted Certificate" on page A-25.
2. Create a wallet and add a certificate request as described in "Adding a Certificate Request" on page A-21.

3. Copy the certificate request text, including the lines `BEGIN NEW CERTIFICATE REQUEST` and `END NEW CERTIFICATE REQUEST` to your system's clipboard.
4. Log in to OracleAS Certificate Authority User pages, and navigate to the Server/SubCA tab.
5. Paste your certificate request into the Server/SubCA certificate request form.
6. Click **Submit**.

When the OracleAS Certificate Authority administrator notifies you that your certificate has been issued, download the file to your file system, and import it into Oracle Wallet Manager by using the method described in "Importing the User Certificate into the Wallet" on page A-23.

Glossary

authentication

The process of verifying the identity of a user, device, or other entity in a computer system, often as a prerequisite to granting access to resources in a system. A recipient of an authenticated message can be certain of the message's origin (its sender). Authentication is presumed to preclude the possibility that another party has impersonated the sender.

availability

The percentage or amount of scheduled time that a computing system provides application service.

CA

See **certificate authority**.

certificate

Also called a digital certificate. An ITU x.509 v3 standard data structure that securely binds an identity to a public key.

A certificate is created when an entity's public key is signed by a trusted identity, a certificate authority. The certificate ensures that the entity's information is correct and that the public key actually belongs to that entity.

A certificate contains the entity's name, identifying information, and public key. It is also likely to contain a serial number, expiration date, and information about the rights, uses, and privileges associated with the certificate. Finally, it contains information about the certificate authority that issued it.

certificate authority

A trusted third party that certifies that other entities—users, databases, administrators, clients, servers—are who they say they are. When it certifies a user, the certificate authority first seeks verification that the user is not on the certificate revocation list (CRL), then verifies the user's identity and grants a certificate, signing it with the certificate authority's private key. The certificate authority has its own certificate and public key which it publishes. Servers and clients use these to verify signatures the certificate authority has made. A certificate authority might be an external company that offers certificate services, or an internal organization such as a corporate MIS department.

ciphertext

Data that has been encrypted. Cipher text is unreadable until it has been converted to plain text (decrypted) with a key. See **decryption**.

cipher suite

A set of authentication, encryption, and data integrity algorithms used for exchanging messages between network nodes. During an SSL handshake, for example, the two nodes negotiate to see which cipher suite they will use when transmitting messages back and forth.

cleartext

See **plaintext**.

cryptography

The art of protecting information by transforming it (encrypting) into an unreadable format (**ciphertext**). See **encryption**.

decryption

The process of converting the contents of an encrypted message (**ciphertext**) back into its original readable format (**plaintext**).

DES

Data Encryption Standard. A commonly used symmetric key **encryption** method that uses a 56-bit key.

de-militarized zone (DMZ)

A DMZ is a set of machines that are isolated from the internet by a firewall on one side, and from a company's intranet by a firewall on the other side. This set of machines are viewed as semi-secure. They are protected from the open Internet, but

are not completely trusted like machines that are inside the second firewall and part of the company's intranet. In a typical application server configuration with a DMZ, only the Web listener and the static content for the Web site are placed in the DMZ. All business logic, databases, and other critical data and systems in the intranet are protected.

Diffie-Hellman key negotiation algorithm

This is a method that lets two parties communicating over an insecure channel to agree upon a random number known only to them. Though the parties exchange information over the insecure channel during execution of the Diffie-Hellman key negotiation algorithm, it is computationally infeasible for an attacker to deduce the random number they agree upon by analyzing their network communications. Oracle Advanced Security uses the Diffie-Hellman key negotiation algorithm to generate session keys.

digital certificate

See **certificate**.

digital wallet

See **wallet**.

directory information tree (DIT)

A hierarchical tree-like structure consisting of the DNs of the directory entries. See **distinguished name (DN)**.

distinguished name (DN)

The unique name of a directory entry. It comprises all of the individual names of the parent entries back to the root.

encryption

The process of disguising a message thereby rendering it unreadable to any but the intended recipient. Encryption is performed by translating data into secret code. There are two main types of encryption: **public-key encryption** (or asymmetric-key encryption) and symmetric-key encryption. See **symmetric-key cryptography**.

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.

fault tolerance

The ability of a computing system to withstand faults and errors while continuing to provide the required services.

hot standby

A second running computing system that is ready to pick up application processing in the event that the primary computing system fails. That is, the secondary system takes over the processing at the point where the original computing system stopped and the secondary system continues the processing.

HTTPS protocol

Secure Hypertext Transfer Protocol. A protocol that uses the **Secure Sockets Layer (SSL)** to encrypt and decrypt user page requests as well as the pages that are returned by the origin server.

key

A password or a table needed to decipher encoded data.

key pair

A public key and its associated private key.

LDAP

See **Lightweight Directory Access Protocol (LDAP)**

LDAP Data Interchange Format (LDIF)

The set of standards for formatting an input file for any of the LDAP command-line utilities.

LDIF

See **LDAP Data Interchange Format (LDIF)**

Lightweight Directory Access Protocol (LDAP)

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.

localhost

Localhost is a special TCP/IP interface provided by the operating system which can only be used to communicate with processes that reside on the same machine. Because these connections do not need to leave a host, the information that is sent on such connections is never sent over the network. They are handled in a special manner by the operating system which guarantees that data that is sent on such connections originated from the local machine. These connections are considered immune to such attacks as IP spoofing, where a client fools the operating system into thinking that its IP address is different than it really is.

man-in-the-middle

A security attack characterized by the third-party, surreptitious interception of a message, wherein the third-party, the *man-in-the-middle*, decrypts the message, re-encrypts it (with or without alteration of the original message), and re-transmits it to the originally intended recipient—all without the knowledge of the legitimate sender and receiver. This type of security attack works only in the absence of **authentication**.

MD5

A hashing algorithm intended for use on 32-bit machines to create digital signatures. MD5 is a **one-way hash function**, meaning that it converts a message into a fixed string of digits that form a **message digest**.

message digest

Representation of text as a string of single digits. It is created using a formula called a **one-way hash function**.

mission critical

See fault tolerance.

one-way hash function

An algorithm that turns a message into a single string of digits. “One way” means that it is almost impossible to derive the original message from the string of digits.

The calculated **message digest** can be compared with the message digest that is decrypted with a **public key** to verify that the message has not been tampered with.

Oracle Net

An Oracle product that enables two or more computers that run an Oracle database server or Oracle tools, such as Designer/2000 to exchange data through a third-party network. Oracle Net supports distributed processing and distributed databases. Oracle Net is an open system because it is independent of the communication protocol, and users can interface Oracle Net to many network environments.

Oracle PKI certificate usages

Defines Oracle application types that a **certificate** supports.

PEM

Privacy-Enhanced Electronic Mail. An **encryption** technique that provides encryption, authentication, message integrity, and **key** management.

PGP

Pretty Good Privacy. An **encryption** technique that is based on **public key** cryptography. The PGP encryption package is free.

PKCS #12

A **public-key encryption** standard (PKCS). RSA Data Security, Inc., PKCS #12 is an industry standard for storing and transferring personal authentication credentials—typically in a format called a **wallet**.

PKI

Public Key Infrastructure. The basis for managing **public key**s used to provide **encryption**.

plaintext

Also called cleartext. Unencrypted data in ASCII format.

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**.

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 (**key**s). 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.

relative distinguished name (RDN)

The leftmost component in a directory entry's distinguished name (DN). See **distinguished name (DN)**.

reliability

The ability of a computing system to operate without failing. Reliability is measured by mean-time-between-failures (MTBF).

redundant

Duplicate or extra computing components that protect a computing system.

RSA

A **public-key encryption** technology developed by RSA Data Security. The RSA algorithm is based on the fact that it is computationally expensive to factor very

large numbers. This makes it mathematically unfeasible, because of the computing power and time required, to decode an RSA key.

scalability

A measure of how well the software or hardware product is able to adapt to future business needs.

SHA

See **Secure Hash Algorithm**.

Secure Hash Algorithm

An algorithm that assures data integrity by generating a 160-bit cryptographic message digest value from given data. If as little as a single bit in the data is modified, the Secure Hash Algorithm checksum for the data changes. Forgery of a given data set in a way that will cause the Secure Hash Algorithm to generate the same result as that for the original data is considered computationally infeasible.

An algorithm that takes a message of less than 264 bits in length and produces a 160-bit message digest. The algorithm is slightly slower than MD5, but the larger message digest makes it more secure against brute-force collision and inversion attacks.

Secure Shell (SSH)

SSH is a well-known protocol and has widely available implementations that provide a secure connection tunneling solution, very similar to what port tunneling offers. SSH provides a daemon on both the client and server sides of a connection. Clients connect to the local daemon rather than connecting directly to the server. The local SSH daemon then establishes a secure connection to the daemon on the server side. Communication is then routed from the client, through the client side daemon to the server side daemon and then on to the actual server. This allows a client/server program that uses an insecure protocol to be tunneled through a secure channel. For our purposes, the disadvantage of SSH is that it requires two hops to occur and that the implementations available do not perform and scale well enough. More information on SSH can be obtained from the sites `http://www.ssh.com` and `http://www.openssh.com`.

Secure Sockets Layer (SSL)

A protocol developed by Netscape Corporation. SSL is an industry-accepted standard for network transport layer security. SSL provides authentication, encryption, and data integrity, in a public key infrastructure (PKI). By supporting SSL, OracleAS Web Cache is able to cache pages for **HTTPS protocol** requests.

single key-pair wallet

A **PKCS #12**-format **wallet** that contains a single user **certificate** and its associated **private key**. The **public key** is embedded in the certificate.

single sign-on

The ability of a user to authenticate once, combined with strong authentication occurring transparently in subsequent connections to other databases or applications. Single sign-on lets a user access multiple accounts and applications with a single password, entered during a single connection. Single password, single authentication.

symmetric-key cryptography

Encryption method that uses the same **key** to encrypt and decrypt data using a mathematical formula.

trusted certificate

A trusted certificate, sometimes called a root key certificate, is a third-party identity that is qualified with a level of trust. The trusted certificate is used when an identity is being validated as the entity it claims to be. Typically, the certificate authorities you trust are called trusted certificates. If there are several levels of trusted certificates, a trusted certificate at a lower level in the certificate chain does not need to have all of its higher level certificates verified again.

wallet

Also called a digital wallet. A wallet is a data structure used to store and manage security credentials for an individual entity. It implements the storage and retrieval of credentials for use with various cryptographic services. A **wallet resource locator** (WRL) provides all the necessary information to locate the wallet.

wallet resource locator

A wallet resource locator (WRL) provides all necessary information to locate a wallet. It is a path to an operating system directory that contains a wallet.

WRL

See **wallet resource locator**.

X.509

Public keys can be formed in various data formats. The X.509 v3 format is one such popular format.

Index

E

F

H

I

J

K

L

M

O

P

R

S