Oracle® Application Server

Security Guide

10*g* Release 2 (10.1.2)

Part No. B13999-01

December 2004

This book gives an overview of security issues that affect Oracle Application Server.

Oracle Application Server Security Guide, 10*g* Release 2 (10.1.2)

Part No. B13999-01

Primary Author: Elizabeth Hanes Perry

Contents

Glossary

Index

Send Us Your Comments

Oracle Application Server Security Guide, 10*g* Release 2 (10.1.2)

Part No. B13999-01

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. 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 about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation 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: 650-506-7225. Attn: Oracle Application Server Documentation
- Postal service:

  Oracle Corporation
  Attention: Java Platform Group, Information Development Manager
  500 Oracle Parkway 4OP9
  Redwood Shores, CA 94065
  USA

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

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

List of Figures

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 the following sections:

- Documentation Accessibility
- Audience
- Organization
- Related Documentation
- Conventions

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

Audience

The *Oracle Application Server 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.

Organization

This document contains:

- Chapter 1, "Oracle Application Server Security Overview"—Basic overview of Oracle Application Server.
- Chapter 2, "Oracle Application Server Security Architecture"—Discussion of the Oracle Application Server security framework, including its architecture. It describes each element and how they work together.
- Chapter 3, "Recommended Deployment Topologies"—Recommended security topologies for Oracle Application Server.
- Chapter 4, "Oracle Identity Management"—Oracle Application Server deployment options.
- Chapter 5, "Privilege Delegation"—Common security considerations for Oracle Application Server administrators.
- Chapter 6, "Security Best Practices"—Best practices for developing secure applications.
- Glossary—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.

- General administration tasks

  Oracle Application Server 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
- 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.

- Configuring SSO

  Oracle Application Server Single Sign-On Administrator's Guide
- Configuring Web Services

  Oracle Application Server Web Services Developer's Guide
- 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.

- Configuring SSO

  Oracle Application Server Single Sign-On Administrator's Guide
- Configuring security mechanisms in Oracle Business Intelligence Discoverer

  Oracle Business Intelligence Discoverer Configuration Guide

For further information on security issues that are not addressed here, see the *Oracle Application Server Release Notes* in the Oracle Application Server Platform-specific documentation.

For Oracle Application Server Application Users

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

- Using Oracle Ultra Search

  Oracle Ultra Search Administrator's Guide

- Using Oracle Application Server Integration BAM

  Oracle Application Server Integration BAM User's Guide

- Setting up the database and PL/SQL to avoid known security problems

  Oracle Application Server 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 Quick Tour*

  A brief graphical overview of the application server.

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

  Discuss 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 mod_plsql User's Guide*

  Detailed descriptions 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 Installation Guide*

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

- *Oracle Application Server Release Notes*
- *Oracle Application Server Upgrade and Compatibility Guide*

  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 Performance Guide*
- *Oracle Application Server Best Practices*

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

Oracle Database Documentation Library contains the following documents:

- *Oracle Database 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.

- *Oracle Database Administrator's Guide*

  Description of the Oracle Database 10*g* feature proxy authentication, which allows Oracle Application Server to establish an authenticated session with the database.

- *Oracle Database 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://www.oracle.com/technology/index.html`

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://www.oracle.com/technology/documentation/index.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 Security 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
- Security As a System Issue
- Security Objectives
- Oracle Application Server Middle-Tier Components
- Identity Management Infrastructure
- Configuration Options and Common Topologies
- Security Platform Capabilities in Oracle Application Server 10g

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 Oracle Application Server Infrastructure, 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.

Security As a System Issue

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 all system components, including the following third-party 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, even when using HTTPS.

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.

Overview of SSL Keys and Certificates

The Secure Socket Layer provides secure communications over intranets and the Internet. This section discusses the basic concepts underlying SSL implementations.

In SSL communication between two entities, such as companies or individuals, the server has a *public key* and an associated *private key*. Each key is a number, with the private key of an entity being kept secret by that entity, and the public key of an entity being publicized to any other parties with which secure communication might be necessary. The security of the data exchanged is guaranteed by keeping the private key secret, and by the complex encryption algorithm. This system is known as *asymmetric encryption*, because the key used to encrypt data is not the same as the key used to decrypt data.

Asymmetric encryption has a performance cost due to its complexity. A much faster system is *symmetric encryption*, where the same key is used to encrypt and decrypt data. But the weakness of symmetric encryption is that the same key has to be known by both parties, and if anyone intercepts the exchange of the key, then the communication becomes insecure.

SSL uses both asymmetric and symmetric encryption to communicate. An asymmetric key (*PKI public key*) is used to encode a symmetric encryption key (the *bulk encryption key*); the bulk encryption key is then used to encrypt subsequent communication. After both sides agree on the bulk encryption key, faster communication is possible without losing security and reliability.

When an SSL session is negotiated, the following steps take place:

1. The server sends the client its public key.
2. The client creates a bulk encryption key, often a 128 bit RC4 key, using a specified encryption suite.
3. The client encrypts the bulk key with the server's public key, and sends the encrypted bulk key to the server.
4. The server decrypts the bulk encryption key using the server’s private key.

This set of operations is called *key exchange*. After key exchange has taken place, the client and the server use the bulk encryption key to encrypt all exchanged data.

Note: It is possible, but rare, for the client to have its own private and public keys as well.

In SSL the public key of the server is sent to the client in a data structure known as an X.509 certificate. This certificate, created by a *certificate authority* (CA), contains a public key, information concerning the owner of the certificate, and optionally some digital rights of the owner. Certificates are digitally signed by the CA which created them using that CA's digital certificate public key.

In SSL, the CA's signature is checked by the receiving process to ensure that it is on the *approved list* of CA signatures. This check is sometimes performed by analysis of certificate chains. This occurs if the receiving process does not have the signing CA's public key on the approved list. In that case the receiving process checks to see if the signer of the CA's certificate is on the approved list or the signer of the signer, and so on. This chain of certificate, signer of certificate, signer of signer of certificate, and so on is a *certificate chain*. The highest certificate in the chain (the original signer) is called the *root certificate* of the certificate chain.

The root certificate is often on the approved list of the receiving process. Certificates in the approve list are called *trust points* or trusted certificates. A root certificate can be signed by a CA or can be *self-signed*, meaning that the digital signature that verifies the root certificate is encrypted through the private key that corresponds with the public key that the certificate contains, rather than through the private key of a higher CA.

Functionally, a certificate acts as a container for public keys and associated signatures. A single certificate file can contain one or multiple chained certificates, up to an entire chain. Private keys are normally kept separately to prevent them from being inadvertently revealed, although they can be included in a separate section of the certificate file for convenient portability between applications.

A *keystore* is used to store certificates, including the certificates of all trusted parties, for use by a program. Through its keystore, an entity such as OC4J (for example) can authenticate other parties as well as authenticate itself to other parties. Oracle HTTP Server has what is called a *wallet* for the same purpose. Sun's SSL implementation introduces the notion of a *truststore*, which is a keystore file that includes the trusted certificate authorities that a client will implicitly accept during an SSL handshake.

In Java, a keystore is a `java.security.KeyStore` instance that you can create and manipulate using the `keytool` utility that is provided with the Sun Microsystems JDK. The underlying physical manifestation of this object is a file. Go to `http://java.sun.com/j2se/1.3/docs/tooldocs/win32/keytool.html` for information about `keytool`.

> **Note:** For a full discussion of Oracle Application Server and SSL, see the *Oracle Application Server Administrator's Guide*.

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 multiuser, 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.

  The authentication process deals with the question “Who is trying to access my services?” In any system and application it is paramount to ensure that the identity of the entity or caller trying to access your application is identified in a secure manner. In a multitier application, the entity or caller can be a human user, a business application, a host, or one entity acting on behalf of (or impersonating) another entity.

- **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.

  The authorization (or access control) process deals with the question “Who can access what services offered by which components?” For large-scale enterprises, where the access to various business-critical services and resources by millions of users need to be managed, it is important that a scalable authorization infrastructure be in place to deal with user and application provisioning. Unfortunately, in part due to the complex nature of authorization, this is also an area where confusion reigns and incompatible technologies and standards are prevalent.

- **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.
- **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 in use. 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 3, "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.

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 (JAAS) Provider ensures secure access to and execution of Java applications, as well as integration of Java-based applications with Oracle Application Server Single Sign-On.

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

Applications and Tools

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

- OracleBI Discoverer

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.

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 4, "Oracle Identity Management".

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

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 3, "Recommended Deployment Topologies", and the *Oracle Application Server 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 Oracle Application Server Infrastructure; 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 Oracle Application Server Infrastructure and a OracleAS Portal and Oracle Application Server Wireless middle tier. The Oracle Application Server Infrastructure installation creates a new Oracle Database and Oracle Internet Directory.

- Integration Architect and Process Modeler Topology

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

  - Oracle Application Server Infrastructure
  - J2EE and Oracle Application Server Web Cache middle tier
  - Oracle Application Server Integration BAM

- Departmental Topology

  This topology consists of an Oracle Application Server Infrastructure 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 and Wireless Applications

  This deployment topology supports J2EE applications as well as applications that use components in OracleAS Portal, and OracleAS Wireless. 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.

Security Platform Capabilities in Oracle Application Server 10g

Oracle Identity Management is a security solution for Oracle Application Server 10g. 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 Synchronization Service
- Provisioning Integration Service
- Oracle Delegated Administration Services
- OracleAS Single Sign-On
- OracleAS Certificate Authority

Oracle Identity Management Components

The following 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 provides Windows integration, password policy options, partial replication, and other important security 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 password policy options. In addition, 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 replication models, enabling improved scalability and performance in large network configurations.
- Other Features—Other 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

The features of 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 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
- Oracle Workflow
- Oracle Application Development Framework (Oracle ADF)

Oracle HTTP Server Enhancements

To incorporate the latest optimizations and security features of Apache, the Oracle HTTP Server uses Apache v1.3. 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

Oracle Workflow

With Oracle Application Server 10*g* Release 2 (10.1.2), 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 Application Development Framework (Oracle ADF)

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

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 HTTP Server Security
- J2EE Security and JAAS
- Oracle Application Server Portal Security
- Oracle Application Server Web Cache Security
- Oracle Advanced Security

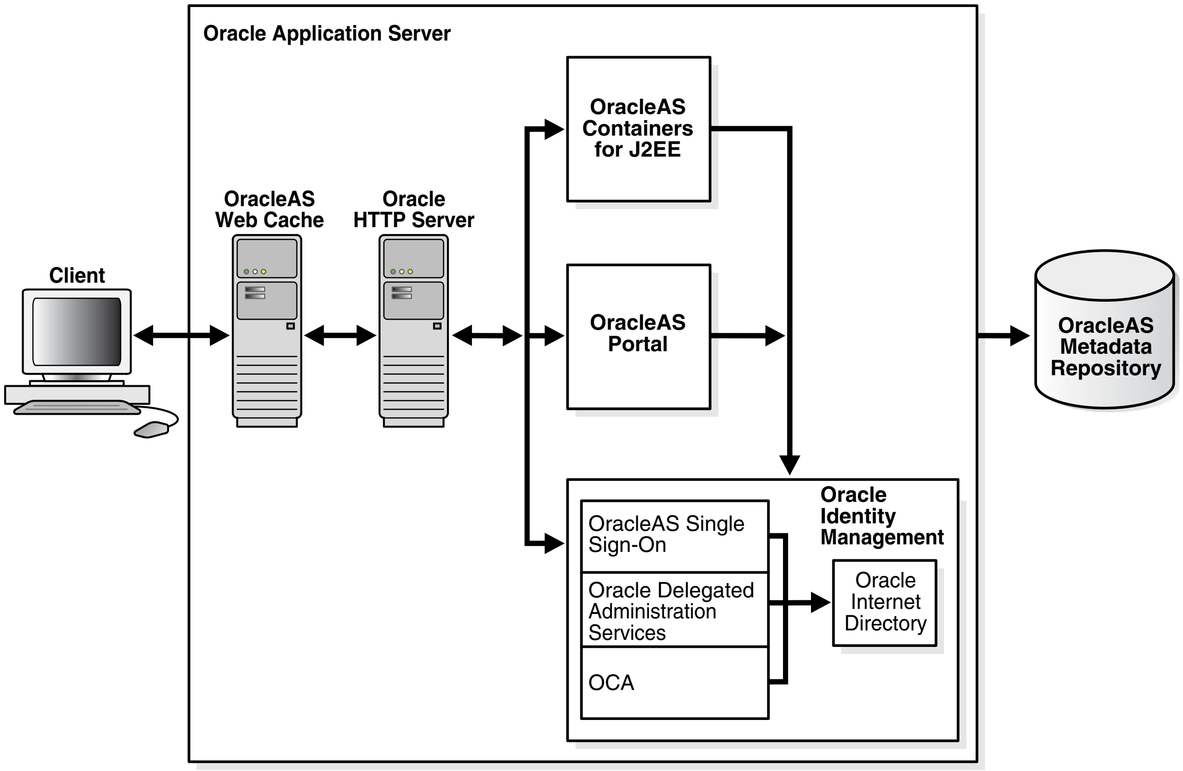
Security Architecture of Oracle Application Server

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 Oracle Application Server Infrastructure. Oracle Application Server Infrastructure 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 Components of Oracle Application Server


Oracle Application Server
Client
OracleAS Web Cache
Oracle HTTP Server
OracleAS Containers for J2EE
OracleAS Portal
Oracle Identity Management
OracleAS Single Sign-On
Oracle Delegated Administration Services
OCA
Oracle Internet Directory
OracleAS Metadata Repository

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 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 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 10g, 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.

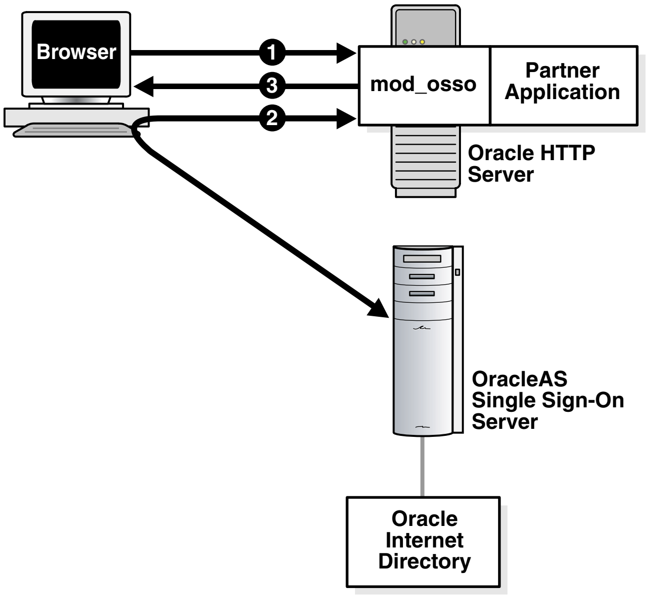
> **Note:** If you protect your J2EE applications with OracleAS Single Sign-On, there is no need to configure `mod_osso` separately.

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 the *Oracle Application Server Administrator's Guide*.

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 provides Oracle Application Server Java Authentication and Authorization Service (JAAS) Provider 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 leverage 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*

Oracle Application Server 10g fully integrates JAAS with the J2EE security model, 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 feature of Oracle Application Server 10g 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.

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*

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 Recommended Deployment Topologies

This chapter describes recommended architectures for deploying the Oracle Application Server 10g 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 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
- General Architecture and Concepts
- Enterprise Data Center Topologies
- OracleAS Single Sign-On and OracleAS Web Cache Considerations

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.

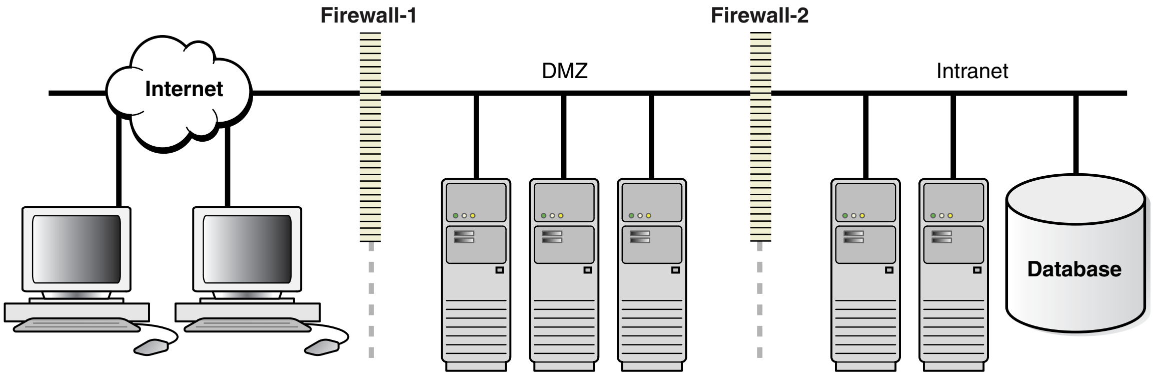
This chapter addresses only application access originating from the Internet. It does not address test, development, or intranet-only applications configurations.

General Architecture and Concepts

The general architectural recommendation is to use the well-known and generally accepted Internet-Firewall-DMZ-Firewall-Intranet architecture shown in Figure 3–1.

> **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 3–1 Traditional DMZ View



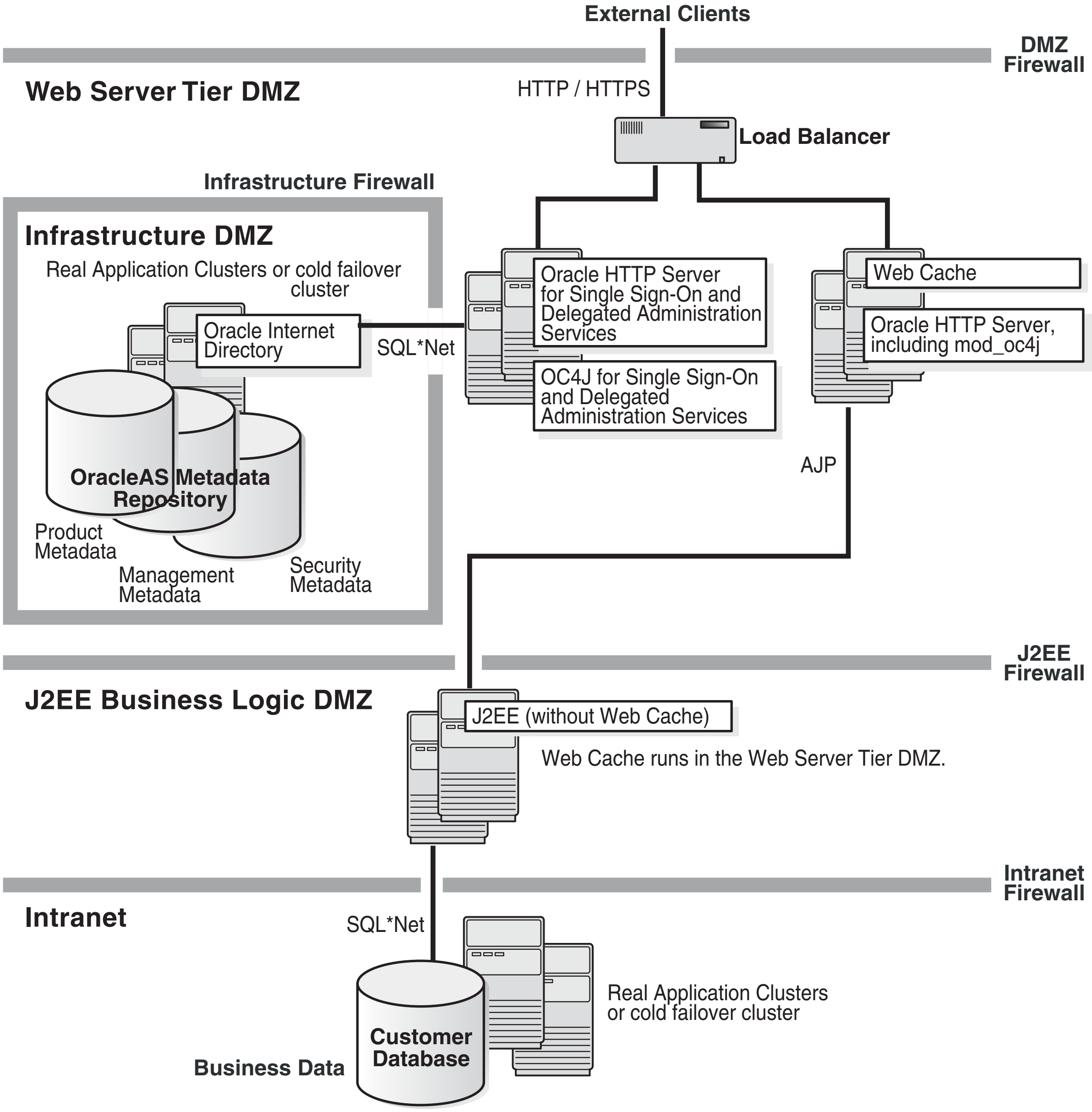
DMZ Zones

In Oracle Application Server 10g, 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 metadata is called the Infrastructure Firewall, and so on. (See Figure 3–2, 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 3–2 DMZ Zones


External Clients
DMZ Firewall
Web Server Tier DMZ
HTTP / HTTPS
Load Balancer
Infrastructure Firewall
Infrastructure DMZ
Real Application Clusters or cold failover cluster
Oracle Internet Directory
SQL*Net
Oracle HTTP Server for Single Sign-On and Delegated Administration Services
OC4J for Single Sign-On and Delegated Administration Services
Web Cache
Oracle HTTP Server, including mod_oc4j
OracleAS Metadata Repository
Product Metadata
Management Metadata
Security Metadata
AJP
J2EE Firewall
J2EE Business Logic DMZ
J2EE (without Web Cache)
Web Cache runs in the Web Server Tier DMZ.
Intranet Firewall
Intranet
SQL*Net
Real Application Clusters or cold failover cluster
Customer Database
Business Data

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 Web server 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 3–3).
- 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.

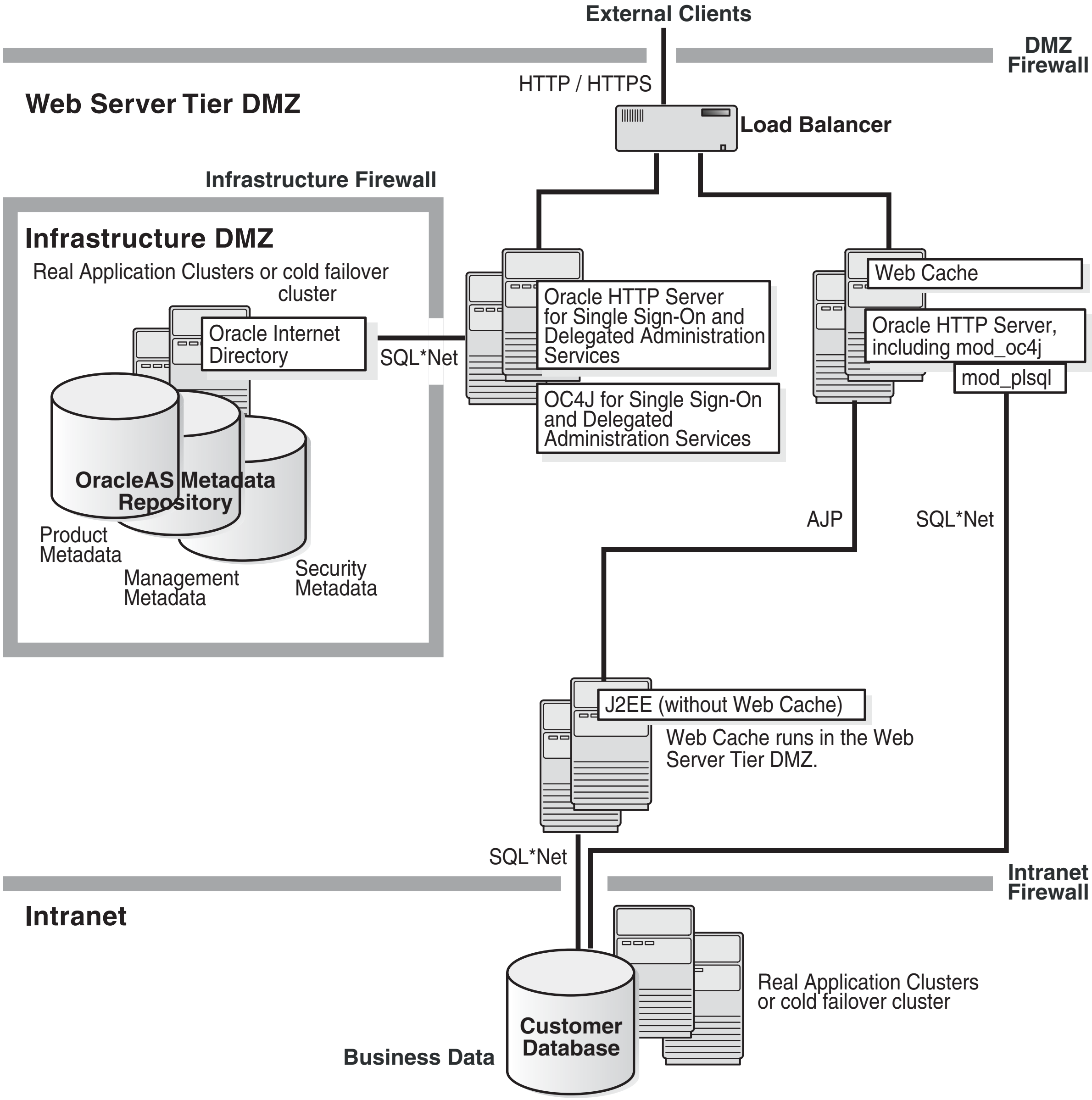
Hardware Load Balancers and HTTPS to HTTP Appliances

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 Web server), and Oracle Internet Directory processes. This is because the Oracle infrastructure provides high scalability and high availability elsewhere, as shown in Figure 3–2 and Figure 3–3.

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 3–3 mod_plsql Access to Business Data


External Clients
DMZ Firewall
HTTP / HTTPS
Web Server Tier DMZ
Load Balancer
Infrastructure Firewall
Infrastructure DMZ
Real Application Clusters or cold failover cluster
Oracle Internet Directory
SQL*Net
Oracle HTTP Server for Single Sign-On and Delegated Administration Services
OC4J for Single Sign-On and Delegated Administration Services
Web Cache
Oracle HTTP Server, including mod_oc4j
mod_plsql
OracleAS Metadata Repository
Product Metadata
Management Metadata
Security Metadata
AJP
SQL*Net
J2EE (without Web Cache)
Web Cache runs in the Web Server Tier DMZ.
SQL*Net
Intranet
Intranet Firewall
Real Application Clusters or cold failover cluster
Customer Database
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 3–2.

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 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 3–3 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 3–2, 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 3–3 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 Installation Guide*.

OracleAS Portal and OracleAS Wireless Applications

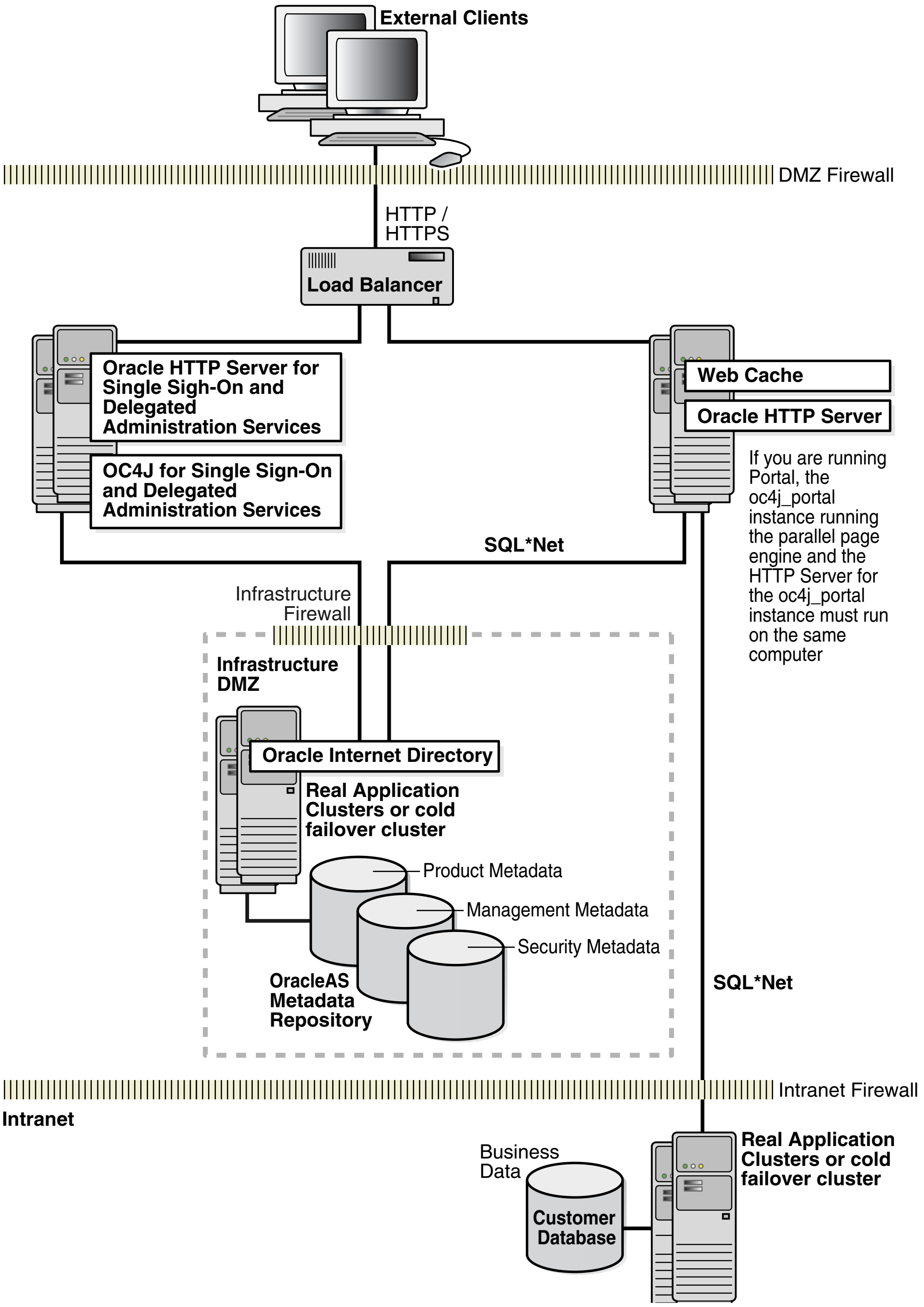
OracleAS Portal has special requirements because its Oracle HTTP Server process must be housed in the same processor box as its OC4J processes; this technical requirement is unique to OracleAS Portal. Figure 3–4 shows a recommended architecture for OracleAS Portal, as well as OracleAS Wireless and Business Intelligence Applications based on OracleAS Portal. Figure 3–4 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 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.

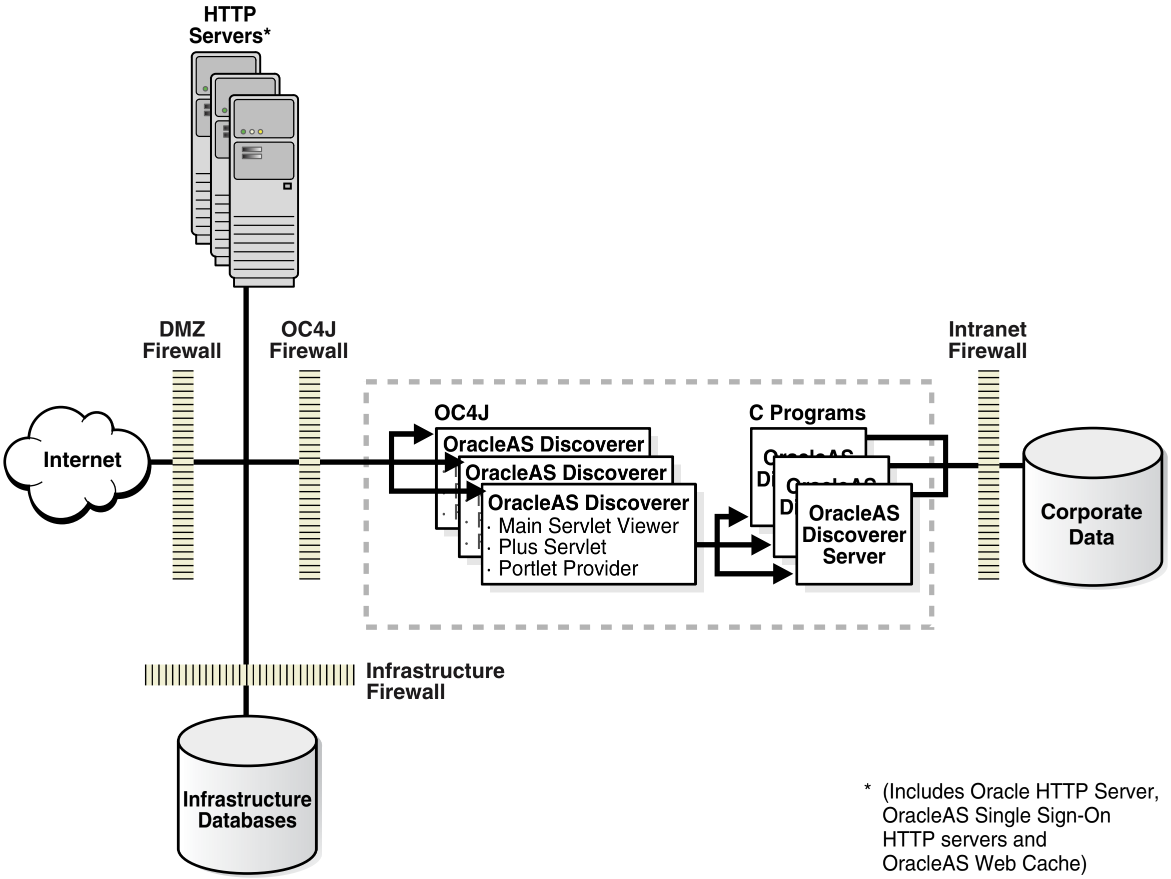
Figure 3–4 Portal and Wireless


External Clients
DMZ Firewall
HTTP / HTTPS
Load Balancer
Oracle HTTP Server for Single Sigh-On and Delegated Administration Services
OC4J for Single Sign-On and Delegated Administration Services
Web Cache
Oracle HTTP Server
If you are running Portal, the oc4j_portal instance running the parallel page engine and the HTTP Server for the oc4j_portal instance must run on the same computer
SQL*Net
Infrastructure Firewall
Infrastructure DMZ
Oracle Internet Directory
Real Application Clusters or cold failover cluster
Product Metadata
Management Metadata
Security Metadata
OracleAS Metadata Repository
SQL*Net
Intranet Firewall
Intranet
Business Data
Customer Database
Real Application Clusters or cold failover cluster

OracleBI Discoverer Recommended Topology

OracleBI Discoverer architecture is complex because its Portlet Provider Servlet is used to provide Portal content. In addition to the Portlet Provider, OracleBI Discoverer also has a Main Servlet Viewer for browser clients and a Plus Servlet to accommodate browser palettes. All these servlets communicate to the OracleBI Discoverer server, a C program that generates the content.

Figure 3–5 Oracle Business Intelligence Discoverer



OracleBI Discoverer allows both OracleAS Single Sign-On authentication and OracleBI Discoverer native authentication. OracleAS Single Sign-On authentication is recommended for OracleBI 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 10g 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.

4

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 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 Sun Java Enterprise System (formerly iPlanet) Directory or Microsoft Active Directory. 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, 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 directory is 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 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 these 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 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 Java Enterprise System 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.

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 5-3 shows, in an Oracle Application Server environment the directory superuser creates:

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

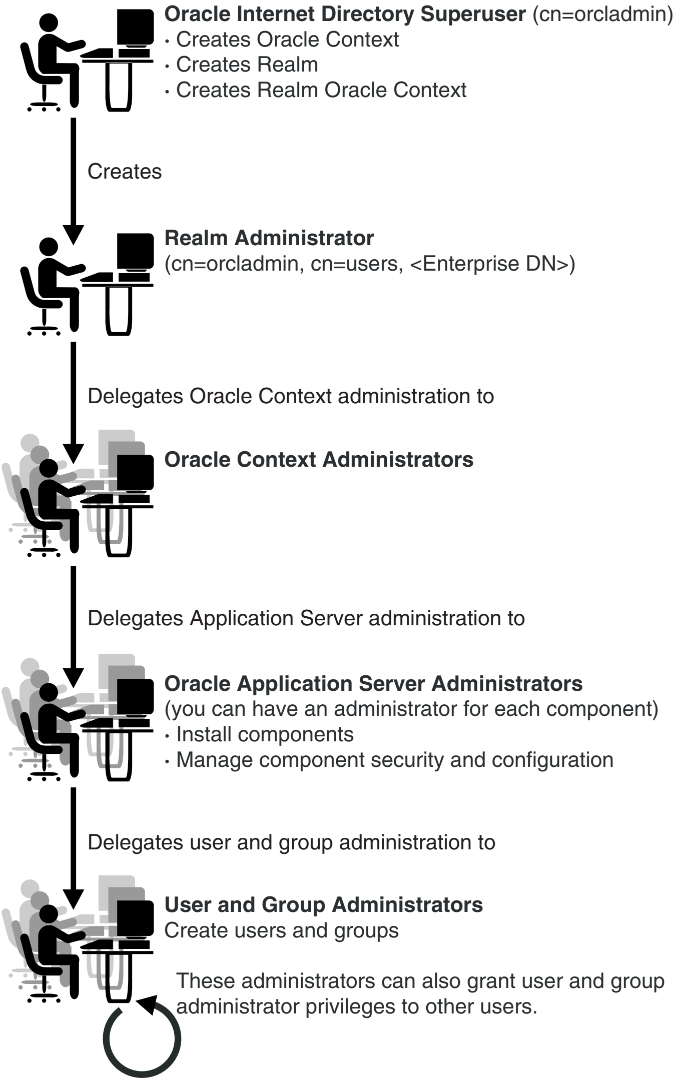
The realm administrator, in turn, delegates administration of the Oracle Context to specific users by assigning those users to the Oracle Context Administrators Group. Oracle Context Administrators then delegate administration of the Oracle Application Server to one or more users by assigning them to 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.

Note: Oracle Internet Directory provides tools, including Oracle Delegated Administration Services, that can be used for privilege delegation. For details see the *Oracle Internet Directory Administrator's Guide*.

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


Oracle Internet Directory Superuser (cn=orcladmin)
· Creates Oracle Context
· Creates Realm
· Creates Realm Oracle Context
Creates
Realm Administrator
(cn=orcladmin, cn=users, <Enterprise DN>)
Delegates Oracle Context administration to
Oracle Context Administrators
Delegates Application Server administration to
Oracle Application Server Administrators
(you can have an administrator for each component)
· Install components
· Manage component security and configuration
Delegates user and group administration to
User and Group Administrators
Create users and groups
These administrators can also grant user and group administrator privileges to other users.

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. Oracle Application Server supports a least privilege model that provides clear separation of duties.

The least privilege model allows developers to:

- Delegate only the privileges necessary for installation and administration
- Grant application administration permissions without making the application administrator an Oracle Internet Directory superuser

Separation of duties allows developers to:

- 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 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.

6 Security Best Practices

This chapter describes security and management best practices for Oracle Application Server. It includes the following topics:

- General Best Practices
- JAAS Best Practices
- J2EE Security Best Practices
- OracleAS Single Sign-On Best Practices
- Oracle Internet Directory Deployment Best Practices

General Best Practices

This section describes general best practices for security and management. It includes the following topics:

- Best Practices for HTTPS Use
- Assign Lowest Level Privileges Adequate for the Task
- Best Practices for Cookie Security
- Best Practices in Systems Setup
- Best Practices for Certificates Use
- Review Code and Content Against Already Known Attack
- Follow Common Sense Firewall Practices
- Leverage Declarative Security
- Use Switched Connections in DMZ
- Place Application Server in the DMZ
- Secure Sockets Layer
- Tune the SSL SessionCacheTimeout Directive
- Plan Out Final Topology Before Installing Security Components

Best Practices for HTTPS Use

The following are recommended for using HTTPS with Oracle Application Server:

- **Configure Oracle Application Server to fail attempts that use weak encryption.** Oracle Application Server can be configured to use only specific encryption ciphers for HTTPS connections. Connections from all old Web browsers that have not upgraded the client-side secure sockets layer (SSL) library to 128-bit can be rejected. This functionality is especially useful for banks and other financial institutions because it provides server-side control of the encryption strength for each connection.
- **Use HTTPS to HTTP appliances for accelerating HTTP over SSL.** Huge performance overhead of HTTPS forces a trade-off in some situations. Use of HTTPS to HTTP appliances can change throughput from 20 to 30 transactions per second on a 500MHz Unix to 6000 transactions per second for a relatively low cost, making this trade-off decision easier. This is a better solution than mathematics/cryptography cards, which can be added to UNIX/NT/Linux computers.
- **Ensure that sequential HTTPS transfers are requested through the same Web server.** Expect 40/50 milliseconds CPU time for initiating SSL sessions on a 500 MHz computer. Most of this CPU time is spent in the key exchange logic, where the bulk encryption key is exchanged. Caching the bulk encryption key will significantly reduce CPU overhead on subsequent access, provided that the access is routed to the same Web server.
- Keep secure pages and pages not requiring security on separate servers. While it may be easier to place all pages for an application on one HTTPS server, the resulting performance cost is very high. Reserve your HTTPS server for pages that require SSL. Put pages that do not require SSL on an HTTP server.

  If secure pages are composed of many `.GIF`, `.JPEG`, or other files that would be displayed on the same screen, it is probably not worth the effort to segregate secure from non-secure static content. The SSL key exchange (a major consumer of CPU cycles) is likely to be called exactly once in any case, and the overhead of bulk encryption is not that high

Assign Lowest Level Privileges Adequate for the Task

When assigning privileges to module(s), use the lowest levels adequate to perform the module(s) function(s). This is essentially "fault containment" which means if security is compromised, it is contained within a small area of the network and cannot invade the entire intranet.

Best Practices for Cookie Security

Use the following as guidelines for cookies:

- **Make sure that cookies have proper expiration dates.** Permanent cookies should have relatively short expiration dates of about three months or less. This will avoid cluttering client Web browsers, which may cause errors if the Web browser cannot transmit all the valid cookies. Non-permanent cookies should be set to expire when the relevant application exits.
- **Make sure that information in cookies contains Method Authentication.** Method Authentication should be used to ensure that cookie data has not been changed since the application set the data. This helps ensure that the cookie cannot be

modified and deceive the application. Also, this helps prevent application failures if the cookie is inadvertently corrupted.

- **Make sure that the size and varieties of cookies are kept low.** There is a finite number and aggregate size of cookies that Web browsers support. If this is exceeded, then the Web browsers will not send all the relevant cookies leading to application failures. Also, very large cookies can result in performance degradation.
- **Carefully use cookie domain name facilities.** Use of cookie domains should ensure that the domain is the smallest possible. Making the domain `oracle.com`, for instance, would mean that any host in `oracle.com` would get the cookie. With hundreds of applications on different parts of `oracle.com`, a domain of `oracle.com` for each of them results in attempts to send hundreds of cookies for each HTTP input operation.

Best Practices in Systems Setup

Use the following as guidelines for system setup:

- **Apply all relevant security patches.** Check MetaLink (`http://metalink.oracle.com`) and Oracle Technology Network (`http://www.oracle.com/technology`) for current security alerts. Many of these patches address publicly announced security issues.
- When deploying software, change all default passwords and close accounts used for samples and examples.
- **Remove unused services from all hosts.** Examples of unused services are FTP, SNMP, NFS, BOOTP, and NEWS. HTTP or WebDAV may be good alternatives.
- **Limit the number of people with root and administrative privileges.**
- **In UNIX, disable the "r" commands if you do not need them.** For example, rhost, rcp.

Best Practices for Certificates Use

Use the following guidelines when using certificates:

- **Ensure that certificate organization unit plus issuer fields uniquely identify the organization across the Internet.** One way to accomplish this would be to include the Dun and Bradstreet or IRS identification as identification for the issuer and the organizational unit within the certificate.
- **Ensure that certificate issuer plus distinguished name uniquely identify the user.** If the combination of issuer and distinguished name is used as identification, there is no duplication risk.
- **Include expiring certificates in tests of applications using certificates.** Expiration is an important consideration for a number of reasons. Unlike most username/password-based systems, certificates expire automatically. With longer duration certificates, fewer re-issues are required, but revocation lists become larger.

  In systems where certificates replace traditional usernames/passwords, expiring certificate situations may result in unexpected bugs. Careful consideration of the effects of expiration is required and new policies will have to be developed because most application and infrastructure developers have not worked in systems where authorization might change during transactions.

- **Use certificate re-issues to update certificate information.** Because certificates expire, infrastructure for updating expired certificates will be required. Take advantage of the re-issue to update organizational unit or other fields. In cases of mergers, acquisitions, or status changes of individual certificate holders, consider re-issuing even when the certificate has not yet expired. But pay attention to key management. If the certificate for a particular person is updated before it expires, for example, put the old certificate on the revocation list.
- **Audit certificate revocations.** Revocation audit trails can help you reconstruct the past when necessary. An important example is replay of a transaction to ensure the same results on the replay as during the original processing. If the certificate of a transaction participant was revoked between the original and the replay, failures may occur which would not have occurred when the original transaction was processed. For these cases, the audit trail should be viewed to simulated authentication at the time when the transaction was initially processed.

Review Code and Content Against Already Known Attack

It is quite common for viruses or known attacks to resurface in slightly altered shape or form. Thus, just because a threat has been apparently eliminated does not mean it will not resurface. Use the following as guidelines to minimize the recurrence of the threat:

- **Ensure that programs are reviewed against double encoding attacks.** There area many cases where special characters, such as <, >, | are encoded to prevent cross-site scripting attacks or for other reasons. For example, "`&lt;`" might be substituted for ">". In a double encoding, the attacker might encode the "&" so that later decoding might involve the inadvertent processing of a >, <, or | character as part of a script. Prevention of this attack, unfortunately, can only be provided by careful program review, although some utilities can be used to filter escape characters that might result in double encoding problems in later processing.
- **Ensure that programs are reviewed against buffer overflow for received data.**
- **Ensure that programs are reviewed against cross-site scripting attacks.** This attack typically tricks HTML and XML processing via input from Web browsers (or processes which act like Web browsers) to invoke scripting engines inappropriately. However, it is not limited to the Web technologies, and all code should be evaluated for this.

Follow Common Sense Firewall Practices

The following are some common recommended practices pertaining to firewalls; while not unique to Oracle Application Server, these are important to overall Oracle Application Server security:

- Place servers providing Internet services behind an exterior firewall of the stateful inspection type. Stateful inspection means that the firewall keeps track of various sessions by protocol and ensures that illegal protocol transitions are disallowed through the firewall. This blocks the types of intrusion that exploit illegal protocol transitions.

- Set exterior firewall rules to allow Internet-initiated traffic only through specific IP and PORT addresses where SMTP, POP3, IMAP, or HTTP services are running. Some protocols (for example, IIOP) leave ports open without receiving processes. PORT and IP combinations that are not assigned to running programs should not be permitted.
- Set interior firewall rules to allow messages through to the intranet only if they originate from servers residing on the perimeter network. All incoming messages must first be processed in the perimeter network.
- Send outgoing messages through proxies on the perimeter network.
- Do not store the information of record on bastion hosts. Bastion hosts are fortified servers on the perimeter network. Information and processing should be segmented such that the bastion hosts provide initial protocol server processing and generally do not contain information of a sensitive nature. The database of record and all sensitive processing should reside on the intranet.
- Disallow all traffic types unless specifically allowed. allow only the traffic required by Oracle Application Server for better security. For example, HTTP, AJP, OCI, LDAP.

Leverage Declarative Security

Oracle HTTP Server has several features that provide security to an application without requiring the application to be modified. These should be leveraged and/or evaluated before programming similar functionality as those features into the application. Specifically:

- **Authentication:** Oracle HTTP Server can authenticate users and pass the authenticated user-id to an application in a standard manner. It also supports single sign-on, thus reusing existing login mechanisms.
- Authorization: Oracle HTTP Server has directives that can allow access to your application only if the end user is authenticated and authorized. Again, no code change is required.
- Encryption: Oracle HTTP Server can provide transparent SSL communication to end customers without any code change on the application.

These three features should be leveraged heavily before designing any application specific security mechanisms.

Use Switched Connections in DMZ

Oracle recommends that all DMZ attached devices be connected by switched, not bussed connections. Furthermore, devices such as the Cisco 11000 series devices, which can provide IP, port, and protocol rules between each pair of connected devices are preferred.

Place Application Server in the DMZ

Application servers should exist in the DMZ. In this architecture Oracle Application Server Web Cache only forwards requests to computers containing Web servers. Web servers only forward requests to application servers or via PL/SQL to database servers. The application servers only forward inward requests to the database or, perhaps, special message processing processors in the intranet. This provides excellent fault containment because a compromised Web server must somehow compromise an application server before the database can be attacked.

Secure Sockets Layer

SSL encryption can be used to secure both LDAP and HTTP traffic that passes between the various components of the Oracle Application Server. To ensure that all LDAP queries being sent to Oracle Internet Directory are SSL-encrypted, you need to configure your Oracle Internet Directory instance to run with a configuration set that supports only SSL-encrypted LDAP connections. The default mode installed with Oracle Application Server allows a given Oracle Internet Directory instance to be configured to listen on both SSL and non-SSL ports. Refer to the Oracle Internet Directory Administrator's Guide for more details on configuring Oracle Internet Directory instances with SSL.

SSL encryption is unrelated to the installation or use of HTTPS, which allows users to access Oracle Application Server components over HTTP while using SSL to encrypt Web client packets."

Tune the SSL SessionCacheTimeout Directive

The Apache server in Oracle Application Server caches a client SSL session information by default. With session caching, only the first connection to the server incurs high latency.

In a simple test to connect and disconnect to an SSL-enabled server, the elapsed time for 5 connections was approximately 11.4 seconds without SSL session caching as opposed to approximately 1.9 seconds when session caching was enabled.

The default SSLSessionCacheTimeout is 300 seconds. Note that the duration of a SSL session is unrelated to the use of HTTP persistent connections. You can change the SSLSessionCacheTimeout directive in httpd.conf file to meet your application needs.

Plan Out Final Topology Before Installing Security Components

Consult the Oracle Application Server Enterprise Deployment Guide and the Oracle Identity Management Concepts and Deployment Planning Guide documents when planning out the final target topology. Identify the steps in installing and configuring the various Oracle Application Server components consistent with the options of the Oracle Universal Installer, rather than approaching the desired topology on an ad-hoc basis.

JAAS Best Practices

Oracle Application Server provides an implementation of Java Authentication and Authorization Service (JAAS) for J2EE applications that is fully integrated with J2EE declarative security. This allows J2EE applications to take advantage of the JAAS constructs such as principal-based security and pluggable login modules. Optionally, the Oracle JAAS implementation allows J2EE applications running on OC4J to leverage the central security services of Oracle Identity Management.

J2EE Security Best Practices

This section describes J2EE security best practices. It includes the following topics:

- Avoid Writing Custom User Managers
- Authentication Mechanism with the JAAS Provider
- Use Fine-Grained Access Control
- Use Oracle Internet Directory as the Central Repository
- Develop Appropriate Logout Functionality for J2EE Applications

Avoid Writing Custom User Managers

The OC4J container continues to provide several methods and levels of extending security providers. The `UserManager` class can be extended to build a custom user manager that allows you to leverage the functionality provided by the JAAS Provider. Both Oracle Application Server Single Sign-On and Oracle Internet Directory provide APIs to integrate with external authentication servers and directories respectively, thus allowing developers more time to focus on actual business logic instead of infrastructure code.

Authentication Mechanism with the JAAS Provider

OC4J allows different authentication options for J2EE applications. Oracle recommends leveraging the OracleAS Single Sign-On server whenever possible for the following reasons:

- It is the default mechanism for most Oracle Application Server components such as OracleAS Portal and OracleAS Wireless.
- It is easy to set up in a declarative fashion and does not require any custom programming.
- It provides a seamless way for PKI integration.

For environments where OracleAS Single Sign-On is not available, and custom authentication is required, one should use JAAS compliant `LoginModules` to extend OC4J authentication. When using `LoginModules`, it is important to only use application relevant principals (roles) associated with the authenticated subject to preserve least privilege.

Use Fine-Grained Access Control

Unlike the coarse-grained J2EE authorization model as it exists today, the JAAS Provider integrated with OC4J allows any protected resource to be modeled using Java permissions. The Java permission model (and associated Permission class) is extensible and allows a flexible way to define fine-grained access control.

For example, a servlet can be written with `Subject.doAs` or `Subject.doPrivileged` to control code that executes sensitive operations.

Use Oracle Internet Directory as the Central Repository

Although the JAAS Provider supports a flat-file XML-based repository useful for development and testing environments, it should be configured to use Oracle Internet Directory for production environments. Oracle Internet Directory provides LDAP standard features for modeling administrative metadata and is built on the Oracle database platform inheriting all of the database properties of scalability, reliability, manageability, and performance. To optimize performance, adjust the caching configurations appropriate for your environment.

Develop Appropriate Logout Functionality for J2EE Applications

Simple J2EE applications using HTTP Basic authentication do not support the concept of logout, relying instead on the user to close the Web browser. When using other forms of authentication, including OracleAS Single Sign-On, it is important to plan out various logout and timeout flows. OC4J has an adjustable HTTP session inactivity parameter that is set to 20 minutes by default. If J2EE applications are leveraging OracleAS Single Sign-On and want to support full logout functionality, they should be written with the appropriate logout dynamic directives.

OracleAS Single Sign-On Best Practices

This section describes OracleAS Single Sign-On best practices. It features the following topics:

- Configure for High Availability
- Leverage Oracle Application Server Single Sign-On
- Use an Enterprise-Wide Directory in Place
- Use OracleAS Single Sign-On Instead of Writing Custom Authentication Logic
- Always Use SSL with Oracle Application Server
- Username and Password Only on Login Screen
- Log Out So Cookies Do Not Remain Active

Configure for High Availability

Single sign-on failure is catastrophic since it means no single sign-on protected application can be accessed. Two recommendations for high availability of OracleAS Single Sign-On are:

- Carefully consider inclusion of any other types of processing on the single sign-on servers since this can make instability more likely.
- Consider deploying multiple single sign-on servers fronted by load balancing hardware to protect against failures in single sign-on listeners. In this case, the address of the load balancer is used as the single sign-on address and the single sign-on listener configuration information is replicated. It is also recommended that the database be a RAC configured for additional improvements in availability. Configuration details for multiple single sign-on servers can be found at Oracle Technology Network (`http://www.oracle.com/technology`).

Leverage Oracle Application Server Single Sign-On

OracleAS Single Sign-On should be used as the primary point of security. This is a benefit administratively and a major convenience to application customers. Also, OracleAS Single Sign-On is well integrated with the rest of Oracle Application Server Infrastructure and can, via Oracle Internet Directory and other means, be integrated with non-Oracle application and infrastructure. Also, as single sign-on becomes a single point for authentication, opportunities to attack the multiple authentication entities of sites today are reduced.

OracleAS Single Sign-On single authenticated user for all applications allows better control for more uniform authorization.

Use an Enterprise-Wide Directory in Place

In order to deploy an effective single sign-on solution, the user population must be centralized in a directory, preferably an LDAP-based directory such as Oracle Internet Directory. Having users represented in multiple systems (for example, in multiple Microsoft Windows NT domains) makes setting up the infrastructure for a common identity more difficult. In addition, clearly defining and automating the user provisioning process makes managing the single sign-on environment much easier.

Use OracleAS Single Sign-On Instead of Writing Custom Authentication Logic

OracleAS Single Sign-On provides the infrastructure to validate credentials and allows for various different authentication mechanisms such as username, password, X.509 certificates. Moreover, since these can be shared across different applications and Web sites, end users do not have to create a new username, password for each different corporate application.

Always Use SSL with Oracle Application Server

The OracleAS Single Sign-On server simplifies user interaction by providing a mechanism to have a single username and password that can be used by multiple partner applications. However, with this ease of use, comes the caution that the single sign-on server should always be accessed in the correct fashion; a breach of the common password can now put all partner applications at risk. Hence, the single sign-on server should always be configured to allow connections in SSL mode only. This protects the end user's credentials going across the wire. Applications where security and data confidentiality is important should also be protected by SSL. From a performance perspective, use of SSL hardware accelerators is recommended.

Username and Password Only on Login Screen

The OracleAS Single Sign-On server provides a standard login screen. This login page is serviced from the single sign-on server, which typically is installed on a different computer from the one the end user is trying to access. Thus, it is critical that before the end user enters their login and password, that a valid single sign-on screen is observed. This prevents users from unknowingly providing their username or password to inappropriate servers.

Log Out So Cookies Do Not Remain Active

Most users do not log out of Internet applications and this creates problems at two levels:

1. A security risk. Another person accessing the work station can now reuse the cookie. Also, since the session remains valid until it times out, a hacker from another machine has a longer time window to guess the session id/cookie value.
2. The system resources on the server associated with the cookie are not released until the session is ended or invalidated.

For application developers and administrators, single sign-on session duration and inactivity timeouts should be configured appropriately (for example, one hour inactivity timeouts for sensitive applications).

For external applications, OracleAS Single Sign-On is unable cannot logout users. Therefore, closing all Web browser windows is important.

Oracle Internet Directory Deployment Best Practices

This section describes Oracle Internet Directory deployment best practices. It includes the following topics:

- Use bulkload.sh Utility
- Replicate for High Availability
- Use SSL Binding
- Use Backup and Restore Utilities
- Monitoring and Auditing Oracle Internet Directory
- Assign Oracle Internet Directory Privileges
- Change Access Control Policies
- Best Practice for Directory Integration Platform
- Recommendations for Migrating Oracle9iAS Applications to an Existing Oracle Internet Directory
- Configuration of the Self-Service Console
- Use opmnctl instead of oidmon and oidctl
- Configure Active Directory Synchronization
- Use User Attributes and Password Hints for Resets

Oracle also recommends the following documentation for deployment of Oracle Internet Directory:

- *Oracle Internet Directory Administrator's Guide*
- *Oracle Application Server Administrator's Guide*
- *Oracle Identity Management Concepts and Deployment Planning Guide*
- *Oracle Process Manager and Notification Server Administrator's Guide*
- *Oracle Application Server Single Sign-On Administrator's Guide*

Use bulkload.sh Utility

The `bulkload.sh` utility checks standard LDIF formatted files for schema violations and duplicates, and generates SQL*Loader intermediate files for fast loading into the database tables underlying Oracle Internet Directory. Use the bulkload.sh utility whenever there is an initial bootstrap required. For example, when setting up synchronization with Microsoft Active Directory or other LDAP directory servers.

Oracle recommends passing the LDIF file output from third-party LDAP directories into bulkload.sh -check mode, which will alert you to any problems with your existing LDAP schema.

Most third-party LDAP directories (including Oracle Internet Directory) support output to LDIF without any operational attributes (which typically cannot be loaded into another vendor's directory). If you are loading data into Oracle Internet Directory from another directory which does not support this, you will have to manually remove any operational attributes prior to sending the LDIF file to bulkload.sh -generate mode.

If your input LDIF file is from another Oracle Internet Directory instance, then you must use the `-restore` option to bulkload.sh to preserve these operational attributes as is during the bulkload.

For more information on the `bulkload.sh` utility refer to the Oracle Internet Directory Administrator’s Guide.

Replicate for High Availability

Oracle Internet Directory supports both multimaster and fan out styles of directory replication. Refer to the *Oracle Identity Management Concepts and Deployment Planning Guide* for guidelines.

For high availability, consider placing an Oracle Internet Directory multimaster replication group behind a network load balancer to provide a single IP address to your LDAP client applications. If a replicated node becomes unavailable, the load balancer can be configured to re-route requests automatically to an available server.

Additionally, each Oracle Internet Directory node can run on Oracle Application Server RAC, further improving availability through increased database uptime and data availability. Other high availability solutions deployable with Oracle Internet Directory are listed in the *Oracle Identity Management Concepts and Deployment Planning Guide*.

Use SSL Binding

SSL is considered the Internet standard protocol for highly secure transportation of data. In addition to the strong PKI authentication using digital certificates, SSL also provides multiple data integrity and data encryption layers to protect your communication channels. SSL provides multiple cipher suites with varieties of encryption algorithms for many security levels.

Oracle Internet Directory supports three SSL authentication modes:

1. Confidentiality mode (no-authentication mode)

   In this mode, SSL cipher suites use the Diffie-Hellman algorithm to generate a session key for client or server at run time. The session key will be used to encrypt the communication channel. No server or user SSL wallet is necessary. In this mode, the channel will be encrypted using a Diffie-Hellman key.

2. Server Authentication only mode

This mode essentially uses certificates for authentication. The client needs to verify the server certificate. This mode is most commonly used in the Internet environment since any client that needs to communicate with aa SSL server does not require a certificate. A client can use their user and password identification to authenticate itself to the server. The username and password are protected by SSL encryption when being transferred on the wire.

3. Server and Client Authentication mode (Mutual authentication)

   In this mode, both client and server use RSA certificates to authenticate each other. First, the client authenticates the server by validating its certificate. In return, the server also requires the client to send its certificate to prove its authenticity.

In addition to choosing an authentication mode, you should choose appropriate security algorithms. For more information refer to the Oracle Internet Directory Administrator's Guide

Use Backup and Restore Utilities

Depending on your Oracle Application Server enterprise topology, you may want to consider backing up Oracle Internet Directory as part of backing up your entire application server environment. For more information, refer to the Oracle Application Server Administrator's Guide and the Oracle Internet Directory Administrator's Guide before deciding on an overall backup and recovery strategy for all of your Oracle Identity Management Infrastructure components.

Monitoring and Auditing Oracle Internet Directory

You can monitor and audit Oracle Internet Directory in one of three ways:

1. The Oracle Enterprise Manager LDAP page provides a very simple way to monitor the LDAP service and determine if it is up and running under its associated load.
2. You can also check the log files of various LDAP processes to ensure there are no errors showing up.
3. LDAP audit log service provides more granular information such as security violation information or sensitive events. The audit log can be further customized to specific directory operations and events.

Oracle recommends that you perform, at the very least, a weekly review of the audit and error logs. System administrators can do a more regular review via Oracle Enterprise Manager to provide better availability.

Assign Oracle Internet Directory Privileges

While it is possible to install Oracle Application Server as an Oracle Internet Directory super user, Oracle recommends that this not be done as it imparts more privileges than required.

To install Oracle Application Server, a user needs to be a member and owner of the Oracle Application Server Administrator's group.

When installing Oracle Application Server, the directory administrator should add the installation user as a member and owner of the Administrator's group. The administrator should then remove the member as the owner once the installation has completed.

Change Access Control Policies

Oracle Internet Directory administrators should change the default access control policies to better control user administration as required.

Oracle Internet Directory administrators should adjust the default access control and password policies using Oracle Directory Manager, in accordance with specific administrative policies for directory access and passwords. This includes both value and state parameters. Refer to the Oracle Internet Directory Administrator's Guide for specific parameters affecting both Access Control and Password Policies."

Best Practice for Directory Integration Platform

This section includes the following topics:

- Use Identity Management Realms
- Configuring DIP Synchronization Service
- Oracle HR Synchronization

Use Identity Management Realms

Directory Integration Platform (DIP) should be used to build connectivity between Oracle Internet Directory and third party directories. This provides seamless integration with other Oracle products. It enables the Oracle products to work in the presence of third party directories in the enterprise and also provides sharing with the same identities in other directories.

The different identities for the same enterprise user from multiple LDAP directories can be joined or unified into a single global identity in Oracle Internet Directory using DIP. This facilitates a true single sign-on environment in an enterprise using Oracle Internet Directory and Oracle Application Server Single Sign-On.

Oracle Internet Directory supports representation of multiple applications and multiple realms or administration Contexts in the Oracle Internet Directory. Various enterprise applications can be provisioned for a single or multiple realms. There are automated tools to create new realms and to provision applications for various realms. These tools setup the various levels of access required by the application to manage the realm.

User definitions from third party identity management systems should be synchronized via the DIP into the appropriate realms to create an enterprise view of all relevant user namespaces and their defined services.

Refer to the Oracle Internet Directory Administrator's Guide for more information.

Configuring DIP Synchronization Service

When configuring DIP, specify only the containers and attributes which are required in the connected directory or in Oracle Internet Directory. LDAP filters can be used as part of mapping configuration profiles to screen out unwanted attribute data and keep synchronization simple.

Each connector and its associated mapping configuration file should be set to an appropriate scheduling interval. No connector needs to fire at the same time or at the same interval as any another, as they are completely independent of one another.

When synchronizing external users and groups into Oracle Internet Directory for use with Oracle Application Server, be sure to establish connectors to the appropriate Identity Management Realm `cn=users` and `cn=groups` container. DIP will then provision all inbound user entries with the Oracle-specific attributes needed to enable users to interact with their deployed Oracle applications.

A synchronization Profile has to be disabled before altering any status attributes through the Oracle Directory Manager. After the change, it needs to be enabled once again.

Refer to the Oracle Internet Directory Administrator's Guide for more information.

Oracle HR Synchronization

Since the `Last Successful Execution Time` connector profile attribute is used to fetch the desired changes from connected directories at a given time, set it initially to some date in the past. Then enable the profile. Note this technique will potentially cause all entries in the connected directory to be synchronized all at once into Oracle Internet Directory. If this is not desirable, use the `bulkload.sh` technique for bootstrapping Oracle Internet Directory and then set the last change number appropriately to begin synchronizing incrementally from the connected directory instead.

It is a good idea to synchronize user data from connected directories to the public `cn=users` container within an Oracle Internet Directory Identity Management realm. This way, all users are immediately accessible to OracleAS Single Sign-On and Oracle Delegated Administration Services such as the Self-Service Console.

The nickname attribute should be synchronized from the connected directory or derived from some attribute which is unique in the connected directory, so that the user can use this identifier with OracleAS Single Sign-On.

Because the `Last Successful Execution Time` connector needs appropriate privilege to read and write to the `cn=users` container under the Identity Management Realm, the profile distinguished name (DN) should be added to the groups `DASCreateUserGroup`, `DASEditUserGroup`, and `DASDeleteUserGroup` for that realm. Refer to the *Oracle Internet Directory Administrator's Guide* for more information.

Recommendations for Migrating Oracle9*i*AS Applications to an Existing Oracle Internet Directory

Oracle Application Server 10g installs an Identity Management infrastructure (including Oracle Internet Directory) as part of each Standard or Enterprise Edition install. However, some organizations may need to delay the upgrade of their Identity Management Infrastructure (Oracle Internet Directory, OracleAS Single Sign-On, DAS, DIP, OCA) and thus need to upgrade mid-tier components from Oracle9*i*AS to Oracle Application Server 10g while maintaining an existing Oracle Internet Directory installation from previous versions of Oracle9*i*AS or Oracle9*i* DB. In this case, the following points need to be considered:

- Before starting the upgrade process, the user keys in the older repository must be made consistent with the keys used to identify users in Oracle Internet Directory. This will enable the upgrade process to correlate the private keys with those present in the production Oracle Internet Directory system.

- The upgrade process will install its own local infrastructure (including a local Oracle Internet Directory instance) against which the Oracle Application Server 10g mid-tier component(s) you are installing can be validated before switching to the previously deployed instance of Oracle Internet Directory.
- Once the upgrade is completed, the upgraded component should be verified for correctness against the newly installed, local Oracle Internet Directory instance.
- Once the verification is complete, simply redirect applications from using the new, local Oracle Internet Directory to using the production Oracle Internet Directory.

As with the upgrade case, the presence of a corporate directory in a deployment influences the process by which the deployment can roll out new services against existing Identity Management infrastructure. In case the existing corporate directory is replicated, some special steps need to be taken by administrators to create a test replica of the production Oracle Internet Directory. Then, install and verify the components against the test replica before switching to the production Oracle Internet Directory service. Refer to the *Oracle Application Server Administrator's Guide* for more information.

Configuration of the Self-Service Console

Rather than creating users and assigning them to groups as separate steps, consider incorporating the group assignment step during user creation. To do this:

1. Log in to the Oracle Internet Directory Self-Service Console as a DAS privileged user (`orcladmin` or designate).
2. Select the Configuration tab.
3. Select **User Entry** > **Add Role**.
4. Search for and select any commonly-subscribed group entries.

Now whenever you or any other DAS privileged user performs a **Create User** sequence, the list of specified groups will appear in the next-to-last step, in a section called **Roles Assignment**. Simply click whichever checkboxes are relevant to the newly-created user, and that user will automatically be made a member of all the groups you specify.

Use opmnctl instead of oidmon and oidctl

In Oracle Application Server, you no longer need to run `oidmon` and `oidctl` to start and stop Oracle Internet Directory processes. OPMN stores the proper sequences and controls these services. Refer to the Oracle Process Manager and Notification Server Administrator's Guide for more information.

Configure Active Directory Synchronization

Prior to configuring Windows Native Authentication, be sure to first configure the Active Directory DIP Connector and bootstrapped the appropriate `cn=users` and `cn=groups` containers within your desired Oracle Identity Management Realm. Do not configure the External Authentication Plug-in for Active Directory if your goal is to enable Windows Native Authentication

See Also:

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

Use User Attributes and Password Hints for Resets

Users who forget their OracleAS Single Sign-On passwords can reset them on their own by using the Oracle Internet Directory Self-Service Console. You must authenticate yourself in one of the following ways:

- If, while previously changing their password, a user specified a password hint question, then the **Confirm Additional Personal Information** window will ask them for the correct answer to the reminder question when attempting a password reset.
- Users who have not previously set a password hint will be presented with the **Confirm Additional Personal Information** window, which prompts them for other personal data as configured the Oracle Delegated Administration Services administrator, via the **Password Reset Validation** field in the Add New Attributes Window of the User Management section of the Self-Service Console.

Refer to the *Oracle Internet Directory Administrator's Guide* for more information.

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