Oracle® Collaboration Suite

Security Guide

10*g* Release 1 (10.1.1)

B14489-02

October 2005

ORACLE®

Oracle Collaboration Suite Security Guide, 10*g* Release 1 (10.1.1)

B14489-02

Primary Author: Meera Viswanathan

Contributors: Laureen Asato, Vimal Chopra, Kiran Choudary, Shruthi Esthuri, Joseph Garcia, Barbara Heninger, Frederic Leblanc, Andrew Mitchell, Pavi Sandhu, Kirill Stoimenov, Richard Strohm, Ginger Tabora

Contents

2 Oracle Collaboration Suite Applications Security

3 Oracle Collaboration Suite Infrastructure Security

Index

List of Figures

List of Tables

Preface

This Preface contains these topics:

Audience

Oracle Collaboration Suite Security Guide is intended for security administrators, application developers, database administrators, system operators, and other Oracle users who perform the following tasks:

- Analyze application security requirements
- Implement security technologies
- Administer Oracle Collaboration Suite Applications tier system security

We assume that the readers have general knowledge of Web server administration, Internet concepts, and networking concepts.

Documentation Accessibility

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

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

Accessibility of Code Examples in Documentation

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

Accessibility of Links to External Web Sites in Documentation

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

TTY Access to Oracle Support Services

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

Related Documents

For more information, refer to the following manuals in the Oracle Collaboration Suite documentation set:

- *Oracle Collaboration Suite Installation Guide for Solaris Operating System*
- *Oracle Collaboration Suite Administrator's Guide*
- *Oracle Collaboration Suite Deployment Guide*
- *Oracle Content Services Administrator's Guide*
- *Oracle Calendar Administrator's Guide*
- *Oracle Real-Time Collaboration Administrator's Guide*
- *Oracle Mail Administrator's Guide*

Printed documentation is available for sale in the Oracle Store at

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

Conventions

The following text conventions are used in this document:

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

Part I

Oracle Collaboration Suite Security

This part contains the following chapters:

1

Overview of Oracle Collaboration Suite Security

Oracle Collaboration Suite is a secure and reliable communications solution that enables individuals and teams to communicate and work together using an integrated suite of applications. This chapter provides an overview of Oracle Collaboration Suite and discusses the security objectives and security architecture of Oracle Collaboration Suite. It contains the following topics:

Overview of Oracle Collaboration Suite

Oracle Collaboration Suite is a messaging and collaboration system, with a standards-based integrated suite built on Oracle Database 10*g*. Oracle Collaboration Suite contains the following tiers:

Oracle Collaboration Suite Infrastructure

Oracle Collaboration Suite Infrastructure provides centralized product metadata, security and management services, configuration information, and data repositories for the **Applications tier**. Oracle Collaboration Suite Infrastructure consists of the following tiers:

Oracle Collaboration Suite Database

Oracle Collaboration Suite Database contains schemas for Oracle Application Server components, such as Oracle Internet Directory, OracleAS Single Sign-On, Oracle

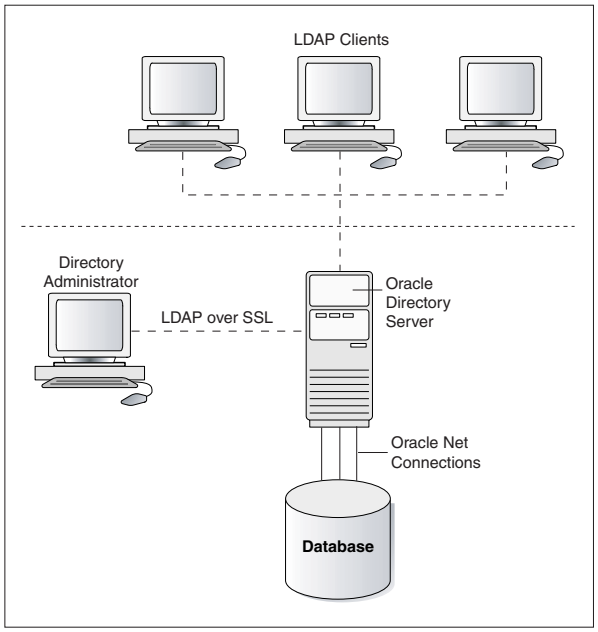
Wireless, and OracleAS Portal, and for Oracle Collaboration Suite components, such as Oracle Calendar, Oracle Mail, and Oracle Content Services. There can be one or several Oracle Collaboration Suite databases, each of which contains one or more sets of schemas, for Oracle Collaboration Suite and Oracle Application Server. This means that each database will contain either all the Oracle Application Server schemas or all the Oracle Collaboration Suite schemas. However, not every instance of each database will be used for each component. For example, if a customer deploys Oracle Mail in database DB1 and Oracle Content Services in database DB2, then each database contains schemas for all Oracle Collaboration Suite components, even though only one schema in each database is actually being used.

Oracle Internet Directory

Oracle Internet Directory is a general-purpose directory service that enables fast retrieval and centralized management of information across distributed network resources. It combines Lightweight Directory Access Protocol (**LDAP**) Version 3 with the high performance, **scalability**, robustness, and availability features of Oracle Database 10*g*.

Oracle Internet Directory runs as an application on an Oracle Database. It communicates with the database by using **Oracle Net** Services, which is an operating system-independent Oracle database connectivity solution. The database may or may not be on the same host. Figure 1–1 illustrates this relationship.

Figure 1–1 Oracle Internet Directory Overview



OracleAS Single Sign-On

OracleAS Single Sign-On enables you to use a single user name, password, and optionally, a realm ID, to access all the features of Oracle Collaboration Suite and other Web applications.

Nonadministrative users first gain access to the Single Sign-On server by entering the URL of a partner application such as OracleAS Portal. Entering such a URL invokes the **single sign-on** login screen. Once you have entered the correct user name and

password, you can gain access to other partner applications and to external applications without having to provide your credentials again.

Administrative users can access the administration home page for single sign-on by typing a URL of the form:

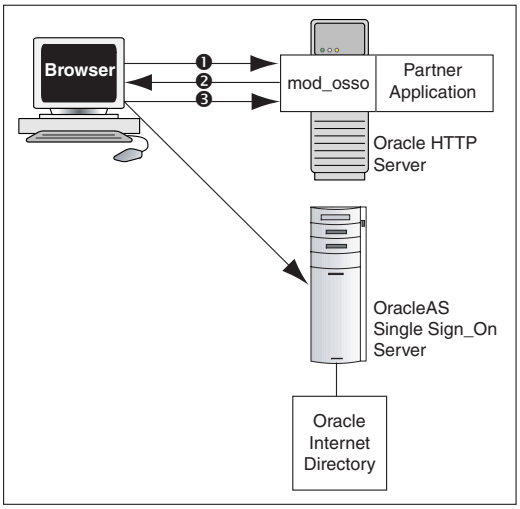
```
http://host:port/pls/orasso
```

where `host` is the computer where the Single Sign-On server is located, `port` is the port number of the server, and `orasso` is the database access descriptor for the single sign-on schema. If the server is enabled for SSL, `https` must be substituted for `http`. If the port number is 80 or 443 (SSL), it may be omitted from the URL. These numbers are the defaults.

Accessing a Partner Application

Figure 1–2 shows what happens when you request the URL of a partner application that is protected by `mod_osso`. `mod_osso` is an Oracle module that examines incoming requests and determines whether the resource requested is protected.

Figure 1–2 Single Sign-On with mod_osso



When you try to access a partner application:

1. You are redirected to the Single Sign-On server. The server prompts you to enter your credentials. After verifying these credentials in Oracle Internet Directory, the server passes these credentials on to the partner application.
2. The application serves the requested content.

Oracle Collaboration Suite Applications

Oracle Collaboration Suite Applications consists of the following components:

- Oracle Calendar
- Oracle Content Services
- Oracle Mail
- Oracle Mobile Collaboration
- Oracle Real-Time Collaboration

- Oracle Voicemail & Fax

Using these integrated application components, you can manage cross-suite business processes. Because Oracle Collaboration Suite is standards-based and not client-dependent, you can access it through any standards-based access method.

> **See Also:** *Oracle Collaboration Suite Concepts Guide* for more information about the various components of Oracle Collaboration Suite

Security Objectives of Oracle Collaboration Suite

The security objectives of Oracle Collaboration Suite are based on the range of operational environments and risk scenarios in which Oracle Collaboration Suite may be deployed. The security objectives are:

- Providing Basic Security Services
- Supporting Standards
- Ensuring Deployment and Configuration Flexibility
- Ensuring Scalability and Predictability

Providing Basic Security Services

Oracle Collaboration Suite provides the following security services required in a multiuser, networked environment:

- **Authentication**

  This service enables a system to verify the identity of users who request access to services or data. Authentication is a prerequisite for other security services, such as access control, authorization, and accountability.

- Authorization and Access Control

  Authorization ensures that a system grants access to resources in compliance with the security policies defined for those resources. Access decisions are based on the authenticated identity and the privileges given to the requesting user.

- Accountability and Intrusion Detection

  Accountability ensures that users who access the system can be held accountable for their usage of the system and system resources. This enables you to monitor system usage to identify unauthorized users. Intrusion detection services are similar to accountability services, but they also detect and react to unauthorized usage, including unauthorized usage by authorized users, in real time.

- Data Protection

  This service prevents unauthorized users from accessing sensitive data. You can protect the confidentiality of data sent through a public network by using **encryption**. You can also use encryption to protect highly sensitive data from users who bypass access control mechanisms of a system.

Supporting Standards

Oracle Collaboration Suite is based on open standards. It complies with the J2EE framework and supports standard protocols, such as HTTP, and markup languages, such as HTML and XML. Oracle Collaboration Suite security services also comply with relevant standards, facilitating interoperability with third-party products. For

example, most Oracle Collaboration Suite applications support browser-based clients. Oracle Collaboration Suite supports security standards, such as SSL and X.509v3, which these browsers implement.

Ensuring Deployment and Configuration Flexibility

Oracle Collaboration Suite security services are designed to support the full range of product deployment options. Security mechanisms in Oracle Collaboration Suite are aimed at ensuring that practical, real-world constraints on deployment can be met. These constraints include the need to deploy certain components of Oracle Collaboration Suite in the Demilitarized Zone (DMZ), others in the corporate intranet, and enable those components to communicate across a **firewall**.

Ensuring Scalability and Predictability

As systems grow in size, there will be a breaking point where an entire, new, expensive server is required because of the overall workload. With Oracle Collaboration Suite, adding resources to the servers will smooth the breaking point because the entire system is designed from the ground up to be deployed on multiple computers. This allows Oracle Collaboration Suite to serve even the largest organizations.

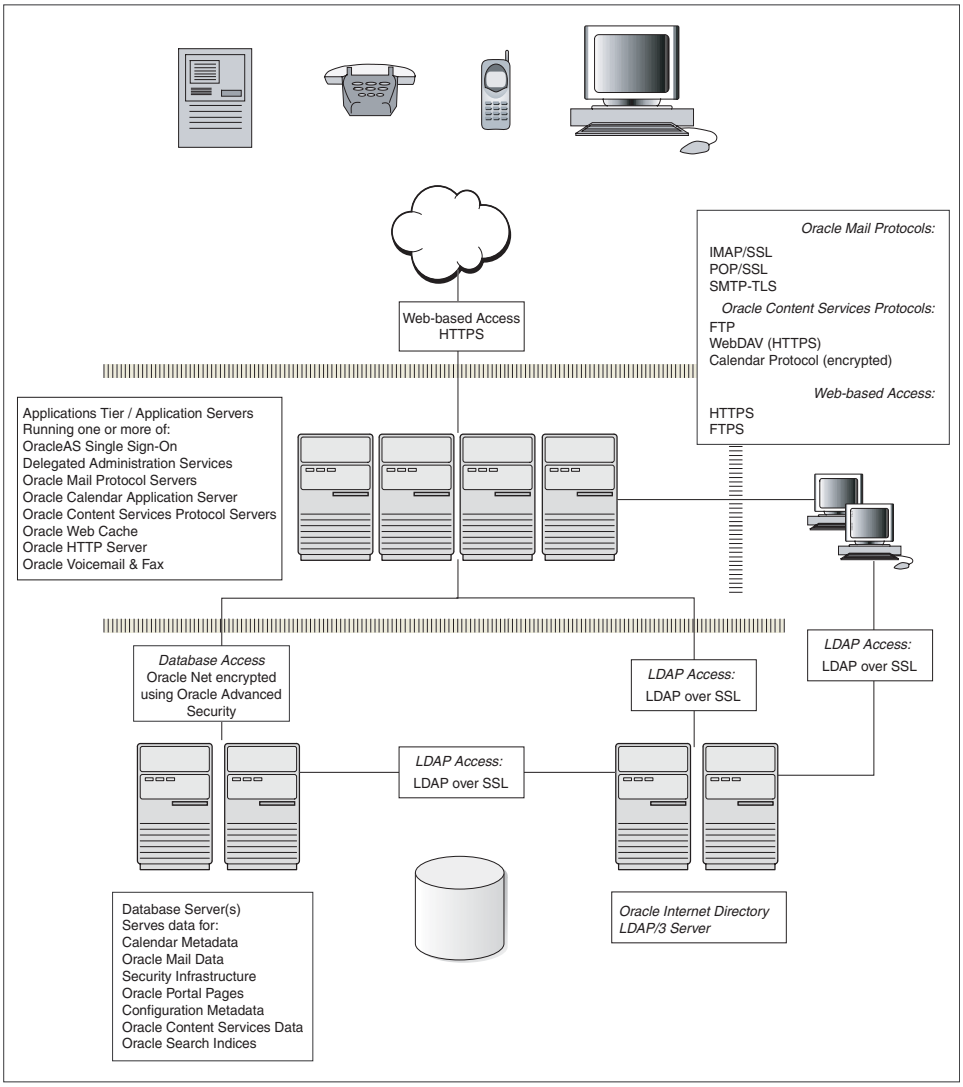
Oracle Collaboration Suite can also scale out to include very large data volumes, with database sizes in excess of 5 terabytes common in implementations, because it is based on Oracle Database 10*g*.

Oracle Collaboration Suite provides rational IT planning by consolidating servers, making it easy to plan for storage, networking, backup, and other key requirements, and reducing costs in the process.

Security Architecture of Oracle Collaboration Suite

Figure 1–3 illustrates the elements of the Oracle Collaboration Suite security architecture.

Figure 1–3 Oracle Collaboration Suite Security Architecture



Oracle Collaboration Suite provides a solid framework for building and deploying Web applications by using the Apache-based Oracle HTTP Server, Oracle Application Server Containers for J2EE (OC4J), and OracleAS Portal, which use the advanced security functionality provided by Oracle Collaboration Suite Infrastructure. Oracle Collaboration Suite Infrastructure consists of Oracle Application Server Metadata Repository and Oracle Identity Management.

Oracle Collaboration Suite security starts from the 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.

Secure Sockets Layer and Public Key Infrastructure Authentication

Secure Sockets Layer (SSL) is an industry-standard protocol for securing network connections. SSL uses **RSA public key cryptography** in conjunction with **symmetric key cryptography** to provide authentication, encryption, and data integrity features. This section contains the following topics:

Overview of SSL and TLS

Although SSL was primarily developed by Netscape Communications Corporation, the Internet Engineering Task Force (IETF) took over development of it and renamed it Transport Layer Security (TLS). Essentially, TLS is an incremental improvement to SSL version 3.0.

SSL Handshake

When you start a network connection over SSL, the client and server perform an SSL handshake. This process includes the following steps:

1. The client and server establish which **cipher suite** to use and which encryption algorithm to use for data transfers.
2. The server sends its certificate to the client, and the client checks whether the certificate of the server is signed by a trusted **certificate authority** (**CA**). A Certificate Authority (CA) is a trusted third party that certifies the identity of entities such as users, administrators, clients, and servers. If client authentication is required, then the client sends its own certificate to the server, and the server verifies whether the certificate of the client is signed by a trusted CA.
3. The client and server exchange information using public key cryptography. Based on this information, each generates a session **key**. All subsequent communications between the client and the server is encrypted and decrypted by using this set of session keys.

The authentication process consists of the following steps:

1. On the client, you initiate an **Oracle Net** connection to the server by using SSL.
2. The client and the server perform the SSL handshake.
3. If the handshake is successful, then you can transfer the data across the network.

Public Key Infrastructure

A Public Key Infrastructure (**PKI**) is a set of network components that provide security based on trust assertions for the entire organization. PKI enables disparate network entities to access security services, which use public key cryptography, on an as-needed basis.

This section presents the following basic concepts of PKI:

- Security Features of PKI
- Benefits of the PKI Approach

Security Features of PKI

Table 1–1 describes the security features of PKI:

Table 1–1 Security Features of PKI

Feature	Explanation
Authentication	The importance of authentication, verifying the identity of users and computers, becomes crucial when an organization opens its doors to the Internet. Strong authentication mechanisms ensure that users and computers are the entities they claim to be.
Encryption	Encryption algorithms are used to secure communications and ensure the privacy of data sent from one computer to another.
Nonrepudiation	PKI can be used to provide nonrepudiation through digital signatures. This proves that a specific user performed certain operations at a given time.

Together, these features can be combined to provide a secure environment for deploying e-commerce. Such an environment can support any type of electronic transaction, from corporate intranets to Web-based e-business applications.

Benefits of the PKI Approach

The following advantages of PKI have led to its emergence as an industry standard for securing e-commerce applications:

- Standards-based technology.
- Enables you to select the trust provider.
- Highly scalable.

  Users maintain their own certificates, and certificate authentication involves exchange of data between client and server only. This means that no third-party authentication server needs to be online.
- Enables delegated trust.

  Users who have obtained certificates from a recognized and trusted certificate authority can authenticate themselves to a server the very first time they connect to that server, without having previously been registered with the server.
- Although PKI is not a single sign-on service, it can be implemented in such a way as to enable single sign-on.

Public Key Infrastructure Components

PKI components include the following:

- Certificate Authority
- Certificates
- Certificate Revocation Lists
- Wallets
- Hardware Security Modules

Certificate Authority

When an entity requests certification, the CA verifies its identity and grants it a certificate, which is signed with the **private key** of the CA.

Different CAs may have different identification criteria. Some CAs may verify a requester's identity with a driver's license or the requester's fingerprints, and some may require that requesters have their certificate request form notarized.

The CA publishes its own certificate, which includes its **public key**. Each network entity has a list of trusted CA certificates. Before communicating with each other, network entities exchange certificates and check whether the other's certificate is signed by one of the CAs on their respective trusted CA certificate lists.

Network entities can obtain their certificates from the same or different CAs. By default, when you create a **wallet**, Oracle Advanced Security automatically installs **trusted certificates** from VeriSign, RSA, Entrust, and GTE CyberTrust. A wallet is a container that is used to store authentication and signing credentials, including private keys, certificates, and trusted certificates needed by SSL.

When a network entity receives a certificate, it verifies that it is a trusted certificate, that is, one that has been issued and signed by a trusted certificate authority. A certificate remains valid until it expires or until it is revoked.

Certificates

A **certificate** is created when an entity's public key is signed by a trusted CA. A certificate ensures that an entity's identification information is correct and that the public key actually belongs to that entity.

A certificate contains the entity's name, public key, and an expiry date, a serial number and certificate chain information. It might also contain information about the privileges associated with the certificate.

Certificate Revocation Lists

A certificate issued by a CA is valid only for a specified period of time. However, certain events, such as user name changes or compromised private keys, can render a certificate invalid before the validity period expires. When this happens, the CA revokes the certificate and adds its serial number to a Certificate Revocation List (CRL). CAs periodically publish CRLs to inform the users about invalid certificates.

In an Oracle environment, when a server or client receives user certificates, it can validate the certificate by checking the expiry date, signature, and revocation status. The certificate revocation status is checked by validating the status against published CRLs. If certificate revocation status checking is enabled, then the server searches for the CRL depending on how this feature has been configured. The server searches for CRLs in the following locations:

- Local file system
- Oracle Internet Directory
- CRL Distribution Point

Wallets

In an Oracle environment, every entity that communicates over SSL must have a wallet containing an X.509v3 certificate, a private key, and a list of trusted certificates.

You can use Oracle Wallet Manager to perform the following:

- Generate a public and private **key pair** and create a certificate request
- Store a user certificate that matches with the private key
- Configure trusted certificates

Security administrators use Oracle Wallet Manager to manage security credentials on the server. Wallet owners use Oracle Wallet Manager to manage security credentials on clients.

Hardware Security Modules

Cryptographic information can be stored on two types of hardware devices:

- Hardware boxes on the server side
- Smart card readers on the client side

An Oracle environment supports hardware devices using Application Programming Interfaces (APIs) that conform to the RSA Security Inc., Public Key Cryptography Standards (PKCS) #11 specification.

Public Key Cryptography and the Public Key and Private Key Pair

Public key cryptography requires that entities that want to communicate in a secure manner should possess certain security credentials. These security credentials are stored in a wallet. Security credentials consist of a public and private key pair, a user certificate, a certificate chain, and trusted certificates.

The secrecy of encrypted data generally depends on the existence of a secret key that is shared between the communicating parties. Providing and distributing such secret keys is one aspect of key management. In a multiuser environment, secure key distribution may be difficult. Public key cryptography was invented to solve this problem.

Public key cryptography is based on a secure key pair. Each key, one half of the pair, can only decrypt information encrypted by its corresponding key. A key pair includes:

- A private key, which is known only to its owner
- A public key, which is associated with its owner but distributed widely

Use of the cryptographic key pair to set up a secure, encrypted channel ensures the privacy of the message passing through the channel and validates the authenticity of the sender of the message. It also enables distributing the public key on a server, or in a central directory, without putting at risk the integrity of the private key component of the key pair. This eliminates the need to transmit the public key to every user in the system.

Each entity in a public key system must have a public and private key pair. The public key for an entity is published by a CA in a user certificate. Entities that want to send secure information to this entity can encrypt the information with the public key of the recipient entity. A public key can also enable the receiving entity to validate the organizational affiliation of the sending entity.

Secure Credentials: Certificate-Based Authentication in PKI

Establishing user identity is of primary concern in distributed environments. Otherwise, there can be little confidence in limiting privileges by user. The use of passwords is the most common authentication method, but to protect highly sensitive data, you need to employ strong authentication services. This section describes the authentication methods that can be used with PKI.

Authentication Methods Used with PKI

The authentication methods that are used with PKI include:

SSL Authentication and X.509v3 Digital Certificates

SSL provides authentication through the exchange of certificates that are verified by trusted certificate authorities. SSL uses **digital certificates** (X.509v3), and a public and private key pair to authenticate users and systems.

PKI relies on **X.509** certificates, also called digital certificates, or public-key certificates, for public-key authentication. The most widely used public-key certificates comply with the X.509 format. The X.509v3 certificate is the current industry standard format.

X.509v3 digital certificates contain the following:

- The certificate owner's **distinguished name (DN)**, which uniquely identifies the owner
- The DN of the certificate issuer, which uniquely identifies the certificate authority
- The certificate owner's public key
- The issuer's signature
- The date up to which the certificate is valid
- The serial number of the certificate

Entrust/PKI Authentication

Entrust Technologies, Inc. provides PKI solutions through their Entrust/PKI software. Entrust/PKI includes products, such as Entrust Profile, which secures user PKI credentials, and Entrust Authority, Entrust's CA product. Oracle Corporation has modified its SSL implementation so that it integrates with Entrust/PKI.

Storing Secure Credentials with PKI

With PKI, secure credentials such as digital certificates can be stored in wallets. A wallet is a transparent database used to manage authentication data such as keys, certificates, and trusted certificates needed by SSL. Wallets can be stored in an LDAP-compliant directory. This implementation enables you to centrally manage users.

Single Sign-On Using PKI

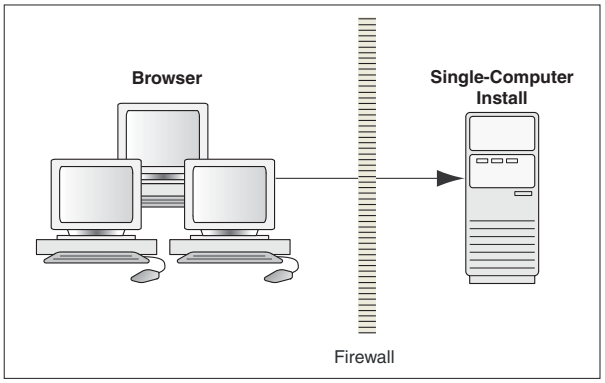
OracleAS Single Sign-On enables users to access multiple accounts and applications with a single password. This feature eliminates the need for multiple passwords for users and simplifies management of user accounts for system administrators.

Recommended Deployment Topologies

This section describes recommended architectures for deploying Oracle Collaboration Suite products to secure Internet access.

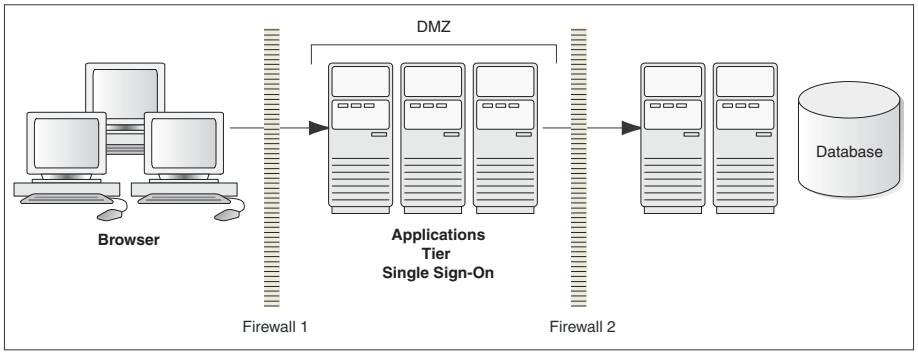
The simplest deployment architecture is the one shown in Figure 1–4. This single-computer deployment may be cost effective for small organizations, however it provides high availability because all components are stored on the same computer.

Figure 1–4 Single-Computer Deployment Architecture



The general architectural recommendation is to use the well-known and generally accepted Internet-Firewall-DMZ-Firewall-Intranet architecture shown in Figure 1–5.

Figure 1–5 Traditional DMZ View



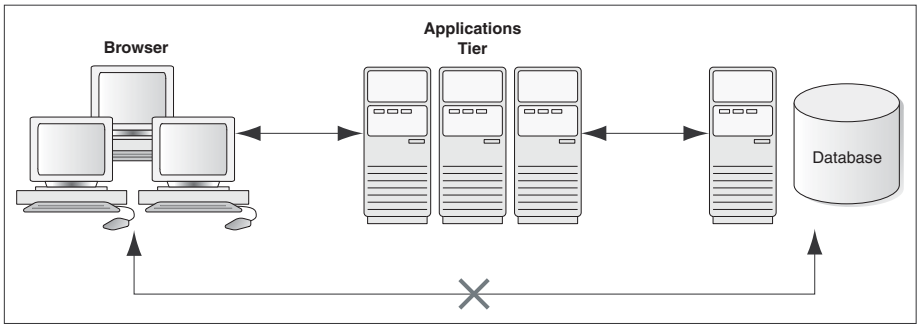
Note: The term **demilitarized zone (DMZ)** refers to a server that is isolated by firewalls from both the Internet and the intranet, thus forming a buffer between the two.

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

Small organizations that cannot invest huge amounts on firewalls can opt for the architecture in Figure 1–6.

Figure 1–6 Deployment Architecture for Small Organizations



See Also: Chapter 3 in *Oracle Collaboration Suite Deployment Guide*

Hardware Load Balancers and HTTPS to HTTP Appliances

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.

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.

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.) Oracle recommends 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.

Compliance Across Oracle Collaboration Suite

Compliance is the process of adhering to a set of established guidelines or rules established by external bodies such as government agencies or by internal corporate bodies. Regulatory compliance has gained increasing attention both due to the number and complexity of regulations that must be addressed and the cost of noncompliance. Through a combination of regulatory changes and increased public awareness of corporate behavior, the cost of noncompliance has dramatically increased. Steep fines, possible jail time for executives, damage to brand reputation, negative impact on stock price or even de-listing from stock exchanges, and risk of shareholder lawsuits are only a few of the possible consequences of compliance failures today.

One of most significant risk management challenges pertains to unstructured content such as electronic documents, e-mail, instant messages, and paper documents. While most businesses keep their financial, employee, or customer records in a secure database, unstructured content that often contains critical information supporting these records is scattered across hundreds or even thousands of minimally secure file and e-mail servers. This section describes how to manage unstructured content in Oracle Collaboration Suite and contains the following topics:

Managing Unstructured Content with Oracle Collaboration Suite

Most organizations store data, for example Personally Identifiable Information (PII), as structured elements in a database. However, unstructured content represents over 80% of all data in an organization, and most companies have a difficult task of getting this data under consistent control. Unstructured content is often the result of collaboration or interactions between employees and customers and may contain information, such as PII. Unstructured content can also consist of scanned paper records such as purchase orders and contracts, which also may contain PII, for example.

Unstructured content is scattered across multiple e-mail and file servers and on desktops and laptop computers. Unlike the records managed in the database, unstructured content is typically not well organized, not easily found, and controlled only under ad hoc security and access control policies. Organizations will need to carefully assess whether the unstructured content repositories contain any information, for example PII, that puts the organization at risk.

Oracle Collaboration Suite provides e-mail, document management, calendar, voicemail and fax, Web conferencing, and wireless access on one integrated platform that allows customers to consolidate unstructured content onto a single database infrastructure.

Gaining Control Over Unstructured Content

When content is scattered across hundreds of servers, meeting the common requirements for compliance and legal risk management becomes a difficult task. This is largely due to a phenomenon known as server drift. In response to the rapid proliferation of information, individual departments and workgroups often set up their own servers. While this is a reasonable way to address immediate localized needs, it also means that IT departments lose control of these servers and, therefore, cannot manage them effectively. Rapid, reliable access to content is complicated by issues, such as not knowing which server the content stored on, what kind of server it is, what version of software the server is running, or even whether the server and its content exist.

In addition, establishment of consistent management policies with regard to such critical activities as folder organization, retention and disposition management, secure access control, action tracking and logging, and even backup planning becomes virtually impossible.

Preventive Measures in Oracle Collaboration Suite

The most important preventive measure that organizations can implement when using Oracle Collaboration Suite is a comprehensive corporate policy and employee education on the use of electronic communication systems and unstructured data containing important information, for example PII. Other preventive measures include

access controls. Oracle recommends using Oracle Internet Directory to manage and provision all users of Oracle Collaboration Suite. The preventive measures for the components of Oracle Collaboration Suite are:

- Oracle Mail
  - Use server-side rules to implement filters that can retain e-mail with certain risk words such as Social Security Number.
  - Educate users on the appropriate use and security implications of using e-mail.
- Oracle Content Services
  - Implement a company policy to ensure that users never store confidential files in public workspaces and only add users to workspaces on a need-to-know basis.
  - Use the administrator search functionality to proactively search for documents containing risk words
- Oracle Real-Time Collaboration
  - Ensure that users notify conference participants when a conference is recorded.
  - Ensure that every conference is protected by an adequate password.
  - Ensure that users never reveal confidential information, PII for example, in Web conferencing chat.
  - Ensure that users are trained on appropriate usage of language in instant messaging.
  - When deploying Oracle Messenger, ensure that the default setting does not turn on instant message archiving on the user desktop. Instead, you may consider creating a server-side message archive, if required by your records retention policy.
  - Ensure that users only turn on Oracle Messenger archiving in accordance with your records retention policy.

Oracle Collaboration Suite contains many additional security and auditing features that will help companies investigate breaches involving information stored in unstructured content. Oracle recommends that organizations carefully assess their collaborative environment to find an adequate balance between useful collaboration and carefully managing the risk from noncompliance.

2

Oracle Collaboration Suite Applications Security

This chapter discusses security in Oracle Collaboration Suite Applications. It contains the following topics:

Controlling Applications Tier Administration and Access

Oracle Collaboration Suite enables administrators to delegate administration responsibilities to other users. For example, the administrator may choose to assign responsibilities to users, as follows:

- Oracle Calendar
  - SYSOP: Start, stop, and manage the Oracle Calendar server
  - Users: Manage groups and holidays on a calendar node

    See Also: *Oracle Calendar Administrator's Guide* and *Oracle Calendar Reference Manual* for more information

- Oracle Content Services
  - Application Administrator: Manage users, quota, libraries, categories, content, and records at the Site and Container level
  - System Administrator: Manage the Oracle Content Services domain by starting and stopping nodes, services, and servers, tuning the system to ensure reliability and performance, creating, modifying, and deleting sites, as well as registering custom workflows

    See Also: *Oracle Content Services Application Administrator's Guide* for more information

- Oracle Mail
  - System Administrator: Change process settings and start and stop servers
  - Domain Administrator: Maintain users and user quota within a specific domain

    See Also: *Oracle Mail Administrator's Guide* for more information

- Oracle Real-Time Collaboration
  - Business Monitor: Monitor or analyze the administrative details of a conference, for example, conference start or end times, attendee join or exit time etc. It also gives you access to a variety of reports, such as Usage, Feedback, Quality of Service and Security.
  - Business Administrator: Start up or shut down Oracle Real-Time Collaboration processes, access log files, configure a new or existing Oracle Real-Time Collaboration host, and manage sites

    Note: Each of these roles could be granted Oracle Real-Time Collaboration system wide or for a specific site.

    See Also: "Oracle Real-Time Collaboration Administration Features" in Chapter 1 of *Oracle Real-Time Collaboration Administrator's Guide* for more information

Using Oracle Collaboration Suite to Access Web Content

Using Oracle Collaboration Suite, you can access your content, such as files, e-mail messages, Web conference, and appointments, through a variety of clients, including browsers, native clients, and wireless devices. Oracle Collaboration Suite uses Oracle Application Server to deliver content to Web and mobile devices. Oracle Application Server uses Secure Sockets Layer (SSL) to secure network traffic between the Web server and the client. SSL, in turn, uses Public Key Infrastructure (PKI) certificates to enable, both network encryption between the client and the server and server authentication. In server authentication, a certificate is provided for a specific server by a Certificate Authority (CA) that the browser trusts.

SSL can be preconfigured for Oracle Collaboration Suite but with a self-signed certificate. When a browser accesses Oracle Collaboration Suite that is using the self-signed certificate, the browser displays a message that the certificate is not trusted. It is important that you obtain a real certificate from a well-known CA such as Verisign.

You need only one certificate for all the Applications tier servers because of the virtual server capabilities of Oracle Application Server. This certificate enables you to secure all Web applications, such as Oracle Calendar, Oracle Mail, Oracle Content Services, and OracleAS Portal.

Using a Wireless Access Protocol (WAP) device to access Oracle Collaboration Suite is a special scenario. A WAP device implements a standard known as Wireless Transport Layer Security (WTLS). A WTLS-compliant WAP gateway must be used to provide the conversion between WTLS and SSL. When this conversion takes place, there is a small period of time for which the content you are reading will exist in the unencrypted format in the memory of the WAP gateway at your service provider. If this is a major

concern, then you can implement your own WAP gateway and dial-up infrastructure. The majority of PDA browsers now implement SSL.

This section contains the following topics:

Client Authentication

Oracle Internet Directory can manage the public keys of users. An advantage of using PKI is the ability to implement client authentication. You authenticate yourself to servers through user names and passwords.

In the case of PKI, clients can use certificates for authentication. Most browsers have wallets to store client SSL certificates. By enabling certificate-based authentication with the OracleAS Single Sign-On server, you will not have to type or remember user names and passwords but will be authenticated to the server and other Web applications using local certificates.

Administration Interfaces

To perform administration tasks for all Applications tier components by using the Application Server Control for Oracle Collaboration Suite or the Grid Control, administrators use an HTML form. Unless Oracle HTTP Server and Oracle Enterprise Manager are configured for SSL, unencrypted passwords are transmitted over the network.

JDBC

By default, Java Database Connectivity (JDBC) does not encrypt network connections between Applications tier components and Oracle Database. Sites can opt to use Oracle Advanced Security to encrypt these connections.

Oracle Internet Directory

You can choose whether or not to use SSL to connect to Oracle Internet Directory. If you do not choose to use SSL, then unencrypted passwords may be sent over network connections between Applications tier processes and Oracle Internet Directory.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for information about configuring SSL for use with Oracle Internet Directory

Securing Oracle Calendar

Security is a primary concern for any application used to manage sensitive, personal information. A number of options are available to an administrator seeking to enhance or customize the security of an Oracle Calendar server installation. In addition to increasing the security of the operating environment and implementing good maintenance and monitoring practices, Oracle Calendar server administrators have access to a configurable, extensible Authentication, Compression, and Encryption (ACE) framework.

This section describes the structure and configuration of the authentication, compression, and encryption methods. Additional security considerations for installations using a directory server are detailed, as well as a number of other measures that may be employed to further protect calendar data. This section contains the following topics:

ACE Framework

The ACE framework was developed by Oracle to allow administrators to ensure the security and integrity of all data passing between Oracle Calendar servers, and between server and client.

Data passes between the server and clients, and between multiple servers if nodes are distributed across more than one host. If an external directory is used, data also passes between the Oracle Calendar server and the directory server. The ACE framework applies to communication between the Oracle Calendar servers, as well as between the Oracle Calendar servers and clients. Refer to Directory Server Security for a separate discussion of the security options for passing data between the Oracle Calendar servers and their supporting directory servers.

Secure connections may involve the use of compression (to reduce the network bandwidth required for communications) and/or encryption (to enhance the security of network communications). Both compression and encryption increase the amount of CPU time required to prepare the communication for transmission. The impact on performance varies with the methods. In general, the better the compression or the more secure the encryption, the greater the impact on performance. This section contains the following topics:

Secure Connections to Clients and Other Calendar Servers

Secure connections to calendar clients and other calendar servers are controlled by a configurable set of authentication, compression, and encryption methods. These methods are determined at the time the connection is requested. The ACE methods are both configurable and extensible. Refer to Available Plug-Ins and Configuration for the relevant configuration parameters, and Extending the ACE Framework for details on extending the available set of methods.

Secure Connections to Clients

Note: Only Oracle Calendar desktop clients 5.0 and higher, Oracle CorporateSync 3.0 and higher, Oracle Calendar Web clients 3.1 and higher, and Oracle Connector for Outlook support the ACE framework. Other clients, including Oracle CorporateSync for Mac 2.1.x, require the use of the `cs-basic` authentication method. If you plan to use Oracle CorporateSync 2.1.x for the Mac, you must add `cs-basic` to the list of supported authentication mechanisms specified by the `[AUTHENTICATION] supported` parameter.

The Oracle Calendar server negotiates with a client as follows:

1. The client starts up and connects to the server.
2. The client queries the server for the supported and default authentication, compression, and encryption methods.
3. The server returns a list of the supported and default authentication, compression, and encryption methods.
4. If the client cannot support one of the default methods, the server and client negotiate using the list of supported methods sent to the client in Step 3 to agree on a method that both support. Note that one of the supported methods for both compression and encryption can be `none`, making both compression and encryption optional.
5. The server authenticates the user using the negotiated authentication method.
6. The client and server communicate using the agreed upon methods for the duration of the user session.

Note: If the client and server cannot agree on authentication, compression, and encryption methods, the negotiation fails and the server does not accept requests from the client.

Secure Connections to Another Calendar Server

The server negotiates with another calendar server as follows:

1. Server A receives a request from Server B.
2. Server A sends Server B a list of the supported and default authentication, compression, and encryption methods.
3. If Server B cannot support one of the default methods, Server A and Server B negotiate using the lists of supported methods sent in Step 2 to agree on a method that both support. Note that one of the supported methods for both compression and encryption can be `none`, making both compression and encryption optional.
4. Server A authenticates Server B using the negotiated authentication method.
5. Servers A and B communicate using the agreed upon methods for the duration of the connection.

   Recall that communication between two calendar servers is through the `uniengd`. In this case, the `uniengd` on Server B asks the `unisncd` on Server B for a connection to a `uniengd` on Server A. The methods are in effect until the requesting `uniengd` on Server B returns the connection to the `unisncd`.

Note: If the two servers cannot agree on authentication, compression and encryption methods, the negotiation fails and Server A does not accept requests from Server B.

Available Plug-Ins

Oracle Calendar provides support for a number of different authentication, compression and encryption methods. This section discusses the plug-ins that are included in a typical Calendar installation. Moreover, they do not require any additional encryption software on the desktop client workstation.

The methods packaged with the Oracle Calendar server are:

Authentication

- **Standard Authentication** (`cs-standard`)

  A secure authentication method where the user's password is encrypted with a secure encryption method using 3DES, and has a unique session key. The transmission between the client and the server is verified for integrity with a cyclic redundancy check to prevent a replay attack.

- **SyncML MD5 1.01 Authentication** (`challenge:SYNCMLMD5_V101`)

  This authentication plug-in is used by SyncML version 1.01 devices when performing challenge-response based authentication.

- **SyncML MD5 1.1 Authentication** (`challenge:SYNCMLMD5_V110`)

  This authentication plug-in is used by SyncML version 1.1 devices when performing challenge-response based authentication.

Compression

- **Run Length Encoding Compression** (`cs-simple`)

  This plug-in will compress all data transferred between the Oracle Calendar client and server using a run length encoding compression algorithm.

Encryption

- **Affine Cipher** (cs-acipher1)

  This plug-in will encrypt all data transferred between the client and server using an affine cipher algorithm.

- **Light Encryption** (cs-light)

  This plug-in will encrypt all data transferred between the client and server using a light and fast encryption algorithm.

- **Oracle AES** (oracle:AES)

  For information about AES Encryption, refer to Advanced Encryption Standard.

- **Oracle DES** (oracle:DES)

  For information about DES Encryption, refer to DES Encryption.

Configuration

To enable the ACE framework and ensure secure server-to-client or server-to-server connections in a node network, set the `[ACE] frameworkenable` parameter in the *`$ORACLE_HOME`*`/ocal/misc/unison.ini` file to `TRUE`.

Table 2–1 lists the parameters used to configure the authentication, compression, and encryption methods used for communication within a calendar network (server to client, server to server). Refer to Extending the ACE Framework for information on extending the sets of supported methods. Refer to Chapter 3 of the *Oracle Calendar Reference Manual*, for details of these parameters.

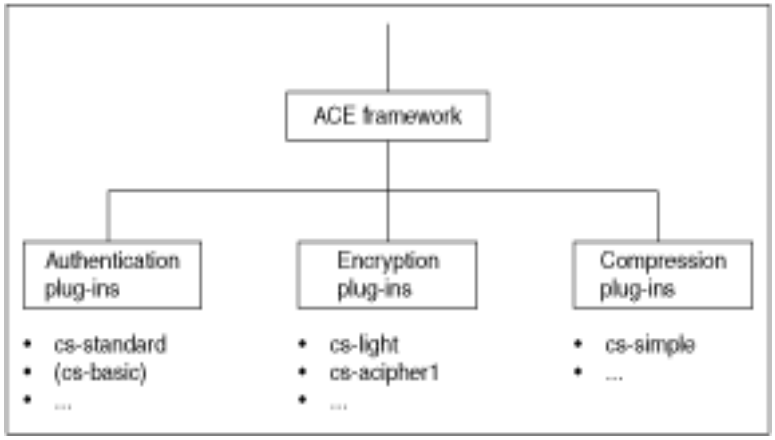
Table 2–1 ACE Configuration Parameters

Section	Parameter	Description
[ACE]	frameworkenable	Enable the ACE framework
[ACE]	minbufsizetocompress	Minimum buffer size for compression
[ACE]	slibcachecount	Maximum number of shared libraries per type
[ACE]	workbufsize	Buffer size for compression and encryption
[ACE_PLUGINS]	gssapi_serviceprincipal	Kerberos 5 principal name
[ACE_PLUGINS_CLIENT]	web_attribute_name	Web authentication - user attribute name
[ACE_PLUGINS_CLIENT]	web_attribute_type	Web authentication - user attribute type
[ACE_PLUGINS_CLIENT]	web_attribute_valuemax	Web authentication - maximum size of user attribute name
[ACE_PLUGINS_CLIENT]	web_cacheexpiresec	Web authentication time-out
[ACE_PLUGINS_CLIENT]	web_cachesize	Web authentication - cache size
[ACE_PLUGINS_CLIENT]	web_CAL_sharedkey	Web authentication - Web:CAL shared key
[ACE_PLUGINS_CLIENT]	web_custom_script	Web authentication - custom user-ID to attribute mapping script
[ACE_PLUGINS_CLIENT]	web_tmppath	Web authentication - path for custom script temporary files
[ACE_PLUGINS_SERVER]	web_CAL_sharedkey	Web authentication — shared key
[ACE_PLUGINS_SERVER]	cs-standard_coexistence	Enable support for cs_standard authentication
[AUTHENTICATION]	admindefault	Default authentication method for administrators
[AUTHENTICATION]	default	Default authentication method for clients
[AUTHENTICATION]	keepresourcepwdincaldb	Location of resource passwords for authentication
[AUTHENTICATION]	servicedefault	Default authentication method for other servers
[AUTHENTICATION]	supported	Supported authentication methods for clients
[COMPRESSION]	admindefault	Default compression method for administrators
[COMPRESSION]	default	Default compression method for clients
[COMPRESSION]	servicedefault	Default compression method for other servers
[COMPRESSION]	supported	Supported compression methods
[ENCRYPTION]	admindefault	Default encryption method for administrators
[ENCRYPTION]	default	Default encryption method for clients
[ENCRYPTION]	needsauthenticate	Encryption methods requiring prior authentication
[ENCRYPTION]	servicedefault	Default encryption method for other servers
[ENCRYPTION]	supported	Supported encryption methods

Extending the ACE Framework

This section describes the use of the ACE plug-ins, and details the mechanism for extending the set of plug-ins available in the server.

Figure 2–1 ACE Framework Architecture



Extending the Set of Plug-Ins

To extend the set of plug-ins available through the ACE framework, first, if the plug-in is not already installed, install the plug-in on your system, and then integrate it into the server. The installation of third-party plug-ins is the responsibility of the system administrator. Refer to the appropriate documentation for details. To integrate the plug-in into the Oracle Calendar server, you will need to add the appropriate keywords to one or more places in the `unison.ini` file.

For all methods except those that support sub-mechanisms, derive the keyword from the name of the plug-in in the following manner. Remove the substring "aut_", "cmp_", or "enc_" and all characters that precede it. Remove the filename extension and the period that precedes it. The remaining string is the keyword to add to the `unison.ini` file.

In the case of plug-ins, which support sub-mechanisms, the keyword has the format `<plug-in_name>:<sub-mechanism_name>`. Derive `<plug-in_name>` as described earlier in this section.

> **See Also:** Table 2–2 for more information on submechanism names and a full list of supported plug-in names and libraries

Plug-In Libraries

Each plug-in is a shared library under UNIX, or a DLL under Windows. The name of the plug-in contains a substring that indicates the type of the plug-in. The following table lists shared library and DLL names for all supported methods of authentication, compression and encryption:

Table 2–2 ACE Configuration Plug-Ins

Substring	Plug-In name	Shared Library or DLL name
aut_	Authentication: cs-standard	libaut_cs-standard.so (Linux/Solaris) libaut_cs-standard.sl (HP-UX) aut_cs-standard.dll (Windows)
aut_	Authentication: challenge:SYNCMLMD5_V101 challenge:SYNCMLMD5_V110	libaut_challenge.so (Linux/Solaris) libaut_challenge.sl (HP-UX) aut_challenge.dll (Windows)
aut_	Authentication: gssapi:kerberos5	libaut_gssapi.so (Linux/Solaris) libaut_gssapi.sl (HP-UX) aut_gssapi.dll (Windows)
aut_	Authentication: web:CAL web:OTMT	libaut_web.so (Linux/Solaris) libaut_web.sl (HP-UX) aut_web.dll (Windows)
cmp_	Compression: cs-simple	libcmp_cs-simple.so (Linux/Solaris) libcmp_cs-simple.sl (HP-UX) cmp_cs-simple.dll (Windows)
enc_	Encryption: cs_acipher1	libenc_cs-acipher1.so (Linux/Solaris) libenc_cs-acipher1.sl (HP-UX) enc_cs-acipher1.dll (Windows)
enc_	Encryption: cs-light	libenc_cs-light.so (Linux/Solaris) libenc_cs-light.sl (HP-UX) enc_cs-light.dll (Windows)
enc_	Encryption: oracle:DES-CFB oracle:DES-CBC oracle:DES-ECB oracle:2DES-CFB oracle:2DES-CBC oracle:2DES-ECB oracle:3DES-CFB oracle:3DES-CBC oracle:3DES-ECB oracle:RC4 oracle:AES-CFB oracle:AES-CBC oracle:AES-ECB	libenc_oracle.so (Linux/Solaris) libenc_oracle.sl (HP-UX) enc_oracle.dll (Windows)

Note: The `cs-basic` authentication plug-in is not a true plug-in in that it is not a shared library. It pre-dates the ACE framework as the authentication method built into the server. The `cs-basic` authentication plug-in must be enabled manually if older clients are to be used, including Oracle CorporateSync 2.1.x for Mac.

Once you have determined the keyword, you must add it to the appropriate list of supported methods in the `unison.ini` file. Add authentication methods to the `[AUTHENTICATION] supported` parameter, compression methods to the `[COMPRESSION] supported` parameter, and encryption methods to the `[ENCRYPTION] supported` parameter. If you want the new method to be the default, also set the appropriate `default` and/or `servicedefault` parameters in the appropriate `[AUTHENTICATION]`, `[COMPRESSION]` or `[ENCRYPTION]` section of the *`$ORACLE_HOME`*`/ocal/misc/unison.ini` file. For more details on these parameters, refer to Chapter 3 of the *Oracle Calendar Reference Manual*.

ACE Configuration Example

Howard administers his organization's installation of Oracle Collaboration Suite. Having done some research, he realizes that some changes must be made to the Oracle Calendar server in order to comply with his organization's security policy.

The only acceptable methods of authentication are either Standard Authentication (`cs-standard`) or SyncML MD5 1.1 Authentication (`challenge:SYNCMLMD5_V110`), and moreover, the default should be the former. In order to enforce this policy, Howard edits the `[AUTHENTICATION]` section of the Oracle Calendar server's *`$ORACLE_HOME`*`/ocal/misc/unison.ini` file with the following parameter modifications:

```
[AUTHENTICATION]
supported = {cs-standard, challenge:SYNCMLMD5_V110}
default = cs-standard
```

In terms of compression, Howard chooses to allow users to decide whether or not they wish to compress the data sent between the Oracle Calendar server and Oracle Calendar clients using `cs-simple`. As a default, compression is not required. In order to enforce this policy, Howard edits the `[COMPRESSION]` section of the Oracle Calendar server's *`$ORACLE_HOME`*`/ocal/misc/unison.ini` file with the following parameter modifications:

```
[COMPRESSION]
supported = {cs-simple, none}
default = none
```

Lastly, Howard would like all data to be encrypted using the `cs-acipher1` plug-in. In order to enforce this policy, Howard edits the `[ENCRYPTION]` section of the Oracle Calendar server's *`$ORACLE_HOME`*`/ocal/misc/unison.ini` file with the following parameter modifications:

```
[ENCRYPTION]
supported = {cs-acipher1}
default = cs-acipher1
```

Setting Different ACE Configuration Settings for Each OCAS Plug-In

Each Oracle Calendar Application System (OCAS) plug-in (Web, Web services, and Sync) has a specific configuration file. Each plug-in configuration file can be used to

set different ACE configurations. The following example demonstrates how to set different ACE configuration settings for each OCAS plug-in.

Consider the following scenario:

- Oracle Calendar Web client needs to use the `web:CAL` authentication scheme.
- Oracle Calendar Web services needs to use the `cs_standard` authentication scheme.
- Oracle Mobile Data Sync needs to use the `enc_cs-light` encryption scheme.

The configuration settings are as follows:

Modify the following parameters in the OCAS configuration file (`ocas.conf`):

```
[ACE_PLUGINS_CLIENT]
web_attribute_type=userid
web_attribute_name=REMOTE_USER
web_CAL_sharedkey=123456789
aut_web_loglevel = 3

[ACE]
Authentication=default
Compression=default
Encryption=default
```

Modify the following parameters in the Oracle Calendar Web client configuration file (`ocwc.conf`):

```
[ACE]
Authentication=web:CAL
Compression=default
Encryption=default
s
```

Modify the following parameters in the Oracle Calendar Web services configuration file (`ocws.conf`):

```
[ACE]
Authentication= cs_simple
Compression=default
Encryption=default
```

Modify the following parameters in the Oracle Mobile Data Sync configuration file (`ocst.conf`):

```
[ACE]
Authentication=default
Compression=default
Encryption=enc_cs-light
```

Integrating the Oracle Calendar Web Client with a Third-Party Authentication Framework

The standalone Oracle Calendar Web client comes with its own login page that allows calendar users to provide their user information and password to access the Oracle Calendar server. For most deployments, this infrastructure is sufficient. However, some customer deployments need to integrate the Oracle Calendar Web client into a solution that includes many other services. When a number of heterogeneous systems are grouped into one solution, it is usually a good practice to offload the security and authentication of users to a service that will harmonize and abstract the various underlying security schemes.

This section describes how to integrate the Oracle Calendar Web client with a third-party authentication framework. It contains the following topics:

- Requirements
- Configuring the Oracle Calendar Server
- Configuring the Oracle Calendar Web Client
- Configuring the Apache Distributed Configuration Files Architecture
- Working of the Components Involved
- Troubleshooting Tips

> **Note:** The example in this section demonstrates how to configure Apache to secure the Oracle Calendar Web client directory using the Apache distributed configuration files architecture (`.htaccess` files). You can follow the same procedure for any third-party authentication framework.

Requirements

The following requirements must be met to configure the Oracle Calendar Web client to integrate with a third-party authentication framework:

- Oracle Calendar Web client 9.0.4.X and above
- Apache 1.3.X and above
- Oracle Calendar server 9.0.4.X and above
- Administrator access to the Apache, Oracle Calendar Web client and Oracle Calendar server directories and applications

> **Caution:** Oracle recommends that both the Oracle Calendar Web client and the Oracle Calendar server be deployed and tested using the Oracle Calendar Web client login page. After you verify that all systems work as expected, you can change the configuration settings.

Configuring the Oracle Calendar Server

To configure the Oracle Calendar server:

1. Edit the *$ORACLE_HOME*`/ocal/misc/unison.ini` file.
2. The list of supported authentication methods is specified in the `[AUTHENTICATION] supported` parameter. To add the Web authentication plug-in, represented by the string `web:CAL` for standalone Calendar server deployments, and by `web:OTMT` for Oracle Collaboration Suite deployments, include the appropriate string within the list of supported methods, as follows:

```
[AUTHENTICATION]
supported = {cs-standard, web:OTMT, challenge:SYNCMLMD5_V101,
challenge:SYNCMLMD5_V110, web:CAL}
default = cs-standard
servicedefault = cs-standard
```

3. Add and configure the following parameters in the `[ACE_PLUGINS_SERVER]` section:

```
[ACE_PLUGINS_SERVER]
cs-standard_coexistence = TRUE
web_CAL_sharedkey=123456789
```

> **Note:** The `web_CAL_sharedkey` parameter is a string composed of ASCII characters.

4. Save the `unison.ini` file.
5. Restart Oracle Calendar server.

> **See Also:** "Configuration Parameters" in Chapter 3 of *Oracle Calendar Reference Manual* for information about the Oracle Calendar server configurable parameters in the `unison.ini` file

Configuring the Oracle Calendar Web Client

To configure the Oracle Calendar Web client:

1. Open the *$ORACLE_HOME*`/ocas/conf/ocwc.conf` file for editing.
2. Enable the Web authentication plug-in by changing the value of the `[ACE] Authentication` parameter to `web:CAL`. Set this parameter as follows:

```
[ACE]
Authentication = web:CAL
```

> **Note:** `web:CAL` authentication can only be specified in Oracle Calendar standalone deployments. If you are using Oracle Calendar with an Oracle Collaboration Suite deployment, set the `Authentication` parameter to `web:OTMT`.

3. Open the *$ORACLE_HOME*`/ocas/conf/ocas.conf` file for editing.
4. Modify the `[ACE_PLUGINS_CLIENT]` section, as follows:

```
#-----------------------------------------------------------
# This section is used to provide ACE configuration settings
# required by the ACE plugins.
#-----------------------------------------------------------
[ACE_PLUGINS_CLIENT]
web_attribute_type=userid
web_attribute_name=REMOTE_USER
web_CAL_sharedkey=123456789
aut_web_loglevel = 3
```

> **Note:** You need to ensure that the value of the `web_CAL_sharedkey` parameter matches that specified in the `unison.ini` file.

> **See Also:** "OCAS.CONF" and "OCWC.CONF" in Chapter 4 of *Oracle Calendar Reference Manual* for information about the `ocas.conf` and `ocwc.conf` configuration parameters, respectively

Configuring the Apache Distributed Configuration Files Architecture

To configure the Apache distributed configuration files architecture, perform the following steps:

1. Modify the following lines in the *$ORACLE_HOME*/ocas/conf/ocal.conf file:

```
<Directory "%ORACLE_HOME%/ocas/bin/">
  SetHandler fastcgi-script
  AllowOverride All
  Options None
  Order allow,deny
  Allow from all
  <IfModule mod_ossl.c>
     SSLOptions +StdEnvVars
  </IfModule>
```

2. Set up your users and the login password they will need to use when accessing the protected directory, as follows:

   a. Change to the Apache/bin directory.

   b. Type the following command to encode the user ID and password:

```
./htpasswd -c $ANY_DIRECTORY/htpasswd $ANY_USERID
```

   Where

   $ANY_DIRECTORY is a directory where the htpasswd file will be written to and $ANY_USERID is the user ID an Oracle Calendar user would use to log in to the Oracle Calendar Web client using the login page.

   Note: The $ANY_USERID *must* be the same as the Calendar user ID for this example to work.

   c. You will be prompted to enter a password for $ANY_USERID. Type any password. This password does not need to be your calendar password.

3. Create a file called .htaccess in the *$ORACLE_HOME*/ocas/bin directory with the following content:

```
AuthUserFile "$ANY_DIRECTORY/htpasswd"
AuthName "My Company custom login"
AuthType Basic
require valid-user
```

   Note: Make sure that the AuthUserFile parameter points to the location where you created your htpasswd file.

4. Restart Apache and the Oracle Calendar Web client.

5. When you type the following URL, you should be prompted for a user ID and password:

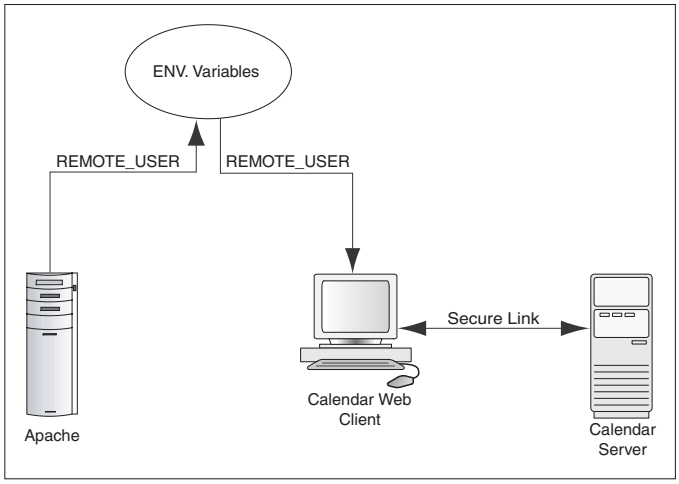
   http://myhost/ocas-bin/ocas.fcgi?sub=web

   If the login is successful, you will automatically be redirected to the Oracle Calendar Web client without having to go through the Calendar Web client login page.

Working of the Components Involved

Figure 2–2 illustrates the components involved while integrating the Oracle Calendar Web client with a third-party authentication framework.

Figure 2–2 Working of the Components Involved



The working of the components is described, as follows:

1. Initially, Apache sets the `REMOTE_USER` environment variable in the FastCgi request environment.
2. The Oracle Calendar Web client uses the configuration file and identifies the `REMOTE_USER` environment variable (`web_attribute_name=REMOTE_USER`).
3. The Oracle Calendar Web client retrieves the `REMOTE_USER` environment variable and passes it to the Oracle Calendar server through the secure link.
4. In addition to the `REMOTE_USER` variable, the Oracle Calendar Web client also informs the Oracle Calendar server about the nature of the value that is sent to it (`web_attribute_type=userid`).
5. The Oracle Calendar server verifies the password (`web_CAL_sharedkey=123456789`) provided by the Oracle Calendar Web client. After the Oracle Calendar server verifies the credentials, the user is allowed to log in.

Using the Custom Solution

To avoid using the Oracle Calendar Web client login page and provide your custom secure solution, you must keep in mind the following:

- The Oracle Calendar Web client will always look for an environment variable to get the user credentials. Your secure solution *must* set that environment variable.
- You can always change the name of the value to be retrieved from the environment variables. For example, `web_attribute_name=SOME_CUSTOM_ENV`.
- You can also instruct the Oracle Calendar Web client to run a shell script that performs some operation to set the environment variable, as follows:

```
[ACE_PLUGINS_CLIENT]
web_attribute_type=custom
web_attribute_name=REMOTE_USER
```

```
web_CAL_sharedkey=123456789
aut_web_loglevel = 3
web_custom_script=/home/custom/script.sh
```

Where `script.sh` could perform some operation to set the `REMOTE_USER` variable. The script can be useful if the user credential used to authenticate against your custom solution is not exactly the same as the one used to log in to the Oracle Calendar server. The script can be used to control the environment variable value to fit what the Oracle Calendar server would expect.

Troubleshooting Tips

In the case of errors, check the following:

- Double-check the `ocas_log` file for errors.
- Refer to the auth_web.log file in the *`$ORACLE_HOME`*`/ocas/bin` directory for information about what could be wrong with the `web:CAL` authentication.
- Refer to the Oracle Calendar server log files or Apache log files for more information.

Kerberos 5 Authentication with Oracle Calendar

Within the ACE framework, Oracle offers the option of configuring the Oracle Calendar server to use Kerberos 5 as an authentication method. This section contains the following topics:

- Background
- Configuring Oracle Calendar with Kerberos 5

Background

Kerberos was developed by MIT as a means of performing secure authentications across an insecure network. In this type of environment, a Kerberos server is used as a central authority for authentications. A user gets a ticket from the Kerberos server that represents their identity, and then uses this ticket to request access from services that also authenticate using Kerberos. When the service verifies that the ticket is valid, the user is granted access to the service.

The Oracle Calendar server uses the `aut_gssapi` ACE plug-in to interface with Kerberos. This interface then allows users to log into the Oracle Calendar server with a valid Kerberos ticket. Because the user's user ID is taken from the Kerberos ticket, there is no need, on the client side, to provide a user name or password.

Configuring Oracle Calendar with Kerberos 5

There are two parts to configuring an Oracle Calendar server to use Kerboros 5:

- Preparing for Kerberos 5 Authentication
- Configuring the Oracle Calendar Server

Preparing for Kerberos 5 Authentication

This section explains the steps that must be carried out on the Kerberos 5 server in order to use Oracle Calendar as a service. Although descriptions of the requirements are provided herein, the detailed technical steps are not. For detailed information relating to Kerboros 5 configuration, refer to MIT's *Kerberos V5 System Administrator's Guide* at the following URL:

http://web.mit.edu/kerberos/www/krb5-1.4/krb5-1.4.1/doc/krb5-admin/index.html

The Kerberos server must be configured to recognize the Oracle Calendar server as a Kerberos-enabled service. This can be achieved in four steps:

1. Define a principal name:

   In order to be recognized by the Kerberos server, the Oracle Calendar server service must have a principal name defined. The principal name can be any string of characters that you wish to use to identify the service. Although the principal name can be any value, Oracle recommends using `uniengd`, as it is the default value that the Oracle Calendar server assumes when the service principal is not specified.

   Note: If multiple instances of Oracle Calendar server exist on the same host, it is recommended to give each instance its own distinct principal name.

2. Define a principal instance:

   The principal instance must be defined using the fully qualified domain name of the machine hosting Oracle Calendar server.

3. Export the service principal's credentials:

   Export the service principal's credentials into a keytab file that the Oracle Calendar server will be able to read. The keytab file must be readable by the owner of the Calendar `uniengd` process, and for best security should not be readable by any other users.

4. Specify the location of the service principal's exported keytab file:

   Specify the full path of the persistent encrypted ticket (defined by the `default_keytab_name` parameter) in the Kerberos 5 workstation's `krb5.conf` file.

Configuring the Oracle Calendar Server

Once the Kerberos 5 setup is complete, the Oracle Calendar server must be configured to allow the use of Kerberos 5 as an authentication method. This is accomplished in five parts, with various steps outlined below:

1. Associate the plug-in with the principal name:

   a. Open the *$ORACLE_HOME*`/ocal/misc/unison.ini` configuration file.

   b. On the last line of the file, add the following section, if it does not already exist, otherwise proceed to the next step:

   ```
   [ACE_PLUGINS]
   ```

   c. Directly under `[ACE_PLUGINS]` section, add the following line:

   ```
   gssapi_serviceprincipal=<kerberos principal name>
   ```

   Where `<kerberos principal name>` represents the principle name defined in Step 1 in Preparing for Kerberos 5 Authentication.

> **Note:** If `uniengd` was used as the Kerberos principal name on the Kerberos server, then this parameter does not need to be added. If the parameter is not specified, the Oracle Calendar server assumes that it is the default: `uniengd`.

d. Save the file.

2. Add Kerberos as a supported authentication mechanism:

a. Open the *$ORACLE_HOME*`/ocal/misc/unison.ini` configuration file.

b. Within the `[AUTHENTICATION]` section of the file, add the following string to the `supported` parameter list:

```
gssapi:kerberos5
```

> **Note:** If you wish to define Kerberos 5 as the default authentication method, the `[AUTHENTICATION] default` parameter must also be set to `gssapi:kerberos5`.

c. Save the file

3. Set your environment variables:

- `OCAL_ADDITIONAL_LIBPATH` should include the path to the Kerberos 5 libraries.

> **Note:** For Oracle Calendar server deployments on Microsoft Windows, the host computer must be restarted so that this environment variable takes effect.

- `KRB5_CONFIG` is the location of the `krb5.conf` file, including the file name. This environment variable does not need to be set if the `krb5.conf` has been installed in the default location: `/etc/krb5.conf`.

> **Note:** These environment variables must always be set prior to starting the Oracle Calendar server.

4. If Calendar has been deployed with Oracle Collaboration Suite, a modification must be made to the `opmn.xml` file in order to apply these configuration changes. If your Calendar server is a standalone deployment, skip this step, and go to Step 5.

a. Open the *$ORACLE_HOME*`/opmn/conf/opmn.xml` file.

b. Locate the `<ias-component id="CalendarServer">` section of the file.

c. Within the `<environment>` sub-section of the `<ias-component id="CalendarServer">` section add the following variable ID:

```
<variable id="KRB5_CONFIG" value="<pathtoKRB5.conf>"/>
```

Where <pathtoKRB5.conf> is the location of the krb5.conf file, including the file name. For example, if your krb5.conf file is located in the /etc directory, the <environment> subsection should have the following entry:

```
<variable id="KRB5_CONFIG" value="/etc/krb5.conf"/>
```

Note: krb5.conf is the Kerberos configuration file on Unix operating systems. On Microsoft Windows, the Kerberos configuration file is krb5.ini.

d. Save the opmn.xml file.

e. Force a reload of the opmn.xml file by running the following command from the $ORACLE_HOME/opmn/bin directory:

```
% opmnctl reload
```

f. Register the changes to Distributed Configuration Management using the following command:

```
dcmctl updateconfig -ct opmn
```

5. Restart the Oracle Calendar server. For information on restarting the Oracle Calendar server, refer to Chapter 4 of the *Oracle Calendar Administrator's Guide*.

Kerberos 5 with Third-Party Directory Servers

When implementing Kerberos with supported third-party directory servers, some additional configuration steps are required. These steps are not required when Calendar is deployed with Oracle Collaboration Suite, or in standalone installations with an internal directory.

1. Open the *$ORACLE_HOME*/ocal/misc/unison.ini configuration file.
2. Add the following parameter to the [DAS] section of the unison.ini file:

```
dir_usewritednforadmin=TRUE
```

3. Add the following parameters with the appropriate values to the [LDAP] section of the unison.ini file:

```
writedn=<writedn>
writednpassword=<encryptedwritednpassword>
```

See Also: Chapter 3 of the *Oracle Calendar Reference Manual* for more information about the writedn and writednpassword parameters

4. Save the file

Directory Server Security

Secure Sockets Layer (SSL) encryption is used by default for all connections to Oracle Internet Directory to protect the data that flows between the Oracle Calendar server and the directory server, and prevent passwords from being sent across the wire in clear text.

Enabling MD5 Authentication

The Oracle Mobile Data Sync server offers direct two-way synchronization with the Oracle Calendar server over any standard Hypertext Transfer Protocol (HTTP) connection, opening up the calendar infrastructure to any SyncML-compliant device or application with Internet access. If secure synchronization of data is a high priority for your organization, then use devices that support SSL synchronization. The list of devices on the Oracle Technology Network site (`http://www.oracle.com/technology/index.html`) specifies which devices support SSL synchronization. Oracle Mobile Data Sync supports SSL connections that are properly configured on the appropriate port during Oracle Collaboration Suite installation and configuration. The combination of using SSL with SyncML basic authentication should be sufficient for most deployments. However, for added security, the Oracle Mobile Data Sync server can be configured to support SyncML MD5 authentication. Note that not all SyncML-compliant mobile devices properly support MD5.

The following section identifies the post-installation steps required to enable MD5 authentication. It contains the following topics:

- Enabling the Dynamic Verifier in Oracle Internet Directory for Passwords
- Enabling the Dynamic Verifier in Oracle Internet Directory for PINs
- Ensuring that the Dynamic Verifier Is Enabled Correctly for Passwords
- Ensuring that the Dynamic Verifier Is Enabled Correctly for PINs
- Steps to be Performed After the Dynamic Verifier Is Enabled
- Enabling MD5 on the Oracle Calendar Server
- Enabling MD5 on the Oracle Mobile Data Sync Server

> **Note:** The instructions provided explain how to enable MD5 authentication for both the single sign-on password and the Wireless PIN. The Oracle Mobile Data Sync server and the Oracle Calendar server can be configured to allow authentication with either the single sign-on password or the Wireless PIN, regardless of whether or not MD5 is being used. In 10*g* Release 1 (10.1.1), the Wireless PIN is the default. If applications are configured for PIN, then only the steps given for PINs need be performed and vice-versa for passwords.

Enabling the Dynamic Verifier in Oracle Internet Directory for Passwords

To enable the dynamic verifier in Oracle Internet Directory for passwords, perform the following steps:

1. Create a file containing the following information:

```
dn:cn=PwdPolicyEntry,cn=Common,cn=Products,cn=OracleContext,<basedn>
changetype: modify
replace: orclpwdencryptionenable
orclpwdencryptionenable: 1
```

   Where

   `<basedn>` is the value provided during install. For example, during test installs, the value is `"dc=ca,dc=oracle,dc=com"`.

2. Run the following command where the port is usually 389:

```
more filename.ldif | ldapmodify -h <OID host> -p <OID port> -D
cn=orcladmin -w <password>
```

Where

`filename.ldif` is the file that was created in Step 1.

Enabling the Dynamic Verifier in Oracle Internet Directory for PINs

To enable the dynamic verifier in Oracle Internet Directory for PINs, perform the following steps:

1. Create a file containing the following information:

```
dn:cn=DefaultSharedPinProfileEntry,cn=Common,cn=Products,cn=OracleContext
changetype: modify
add:orclpwdverifierparams;orclpasswordverifier
orclpwdverifierparams;orclpasswordverifier: crypto:3DES
```

2. Run the following command where the port is usually 389:

```
more filename.ldif | ldapmodify -h <OID host> -p <OID port> -D
cn=orcladmin -w <password>
```

Where

`filename.ldif` is the file that was created in Step 1.

Ensuring that the Dynamic Verifier Is Enabled Correctly for Passwords

To ensure that the dynamic verifier has been enabled correctly for passwords, perform the following steps:

1. Run the following command to check for the password dynamic verifier:

```
ldapsearch -h <OID host> -p <OID port> -D cn=orcladmin -w <password> -b
"cn=PwdPolicyEntry,cn=Common,cn=Products,cn=OracleContext,<basedn>" -s
base "objectclass=*"
```

2. Search for the `orclpwdencryptionenable` attribute and ensure that it is set to 1.

Ensuring that the Dynamic Verifier Is Enabled Correctly for PINs

To ensure that the dynamic verifier has been enabled correctly for PINs, perform the following steps:

1. Run the following command to check for the PIN dynamic verifier:

```
ldapsearch -h <OID host> -p <OID port> -D cn=orcladmin -w <password> -b
"cn=DefaultSharedPinProfileEntry,cn=Common,cn=Products,cn=OracleContext"
-s base "objectclass=*"
```

2. Search for the `orclpwdverifierparams;orclpasswordverifier` attribute and ensure that `3DES` is listed.

Steps to be Performed After the Dynamic Verifier Is Enabled

You *must* reset your single sign-on password from the main preference page and the Wireless PIN from the Mobile Preferences page after the dynamic verifier has been enabled. Failure to do so will result in authentication errors.

Note: You need not change the single sign-on password and Wireless PIN while resetting them.

Enabling MD5 on the Oracle Calendar Server

To enable MD5 on the Oracle Calendar server, perform the following steps:

1. In the *$ORACLE_HOME*/ocal/misc/unison.ini file, add the following line under the [ENG] section:

   ```
   syncml_allowmd5auth  = TRUE
   ```

2. By default, the Oracle Mobile Data Sync server authenticates using the Wireless PIN. To change the option, set the syncml_authcredlabel parameter to PSW. The password to use for basic and MD5 authentication can be further customized by setting the syncml_basicauthcredlabel or syncml_md5authcredlabel to PSW or PIN, respectively, thus overriding the value set in syncml_authcredlabel.

3. Restart the Oracle Calendar server.

Enabling MD5 on the Oracle Mobile Data Sync Server

To enable MD5 on the Oracle Mobile Data Sync server, perform the following steps:

1. In the *$ORACLE_HOME*/ocas/conf/ocst.conf file, change the requiremd5 parameter from False to True under the [ocst] section.

2. Restart the Oracle Mobile Data Sync server.

Other Security Considerations

The following safeguards can be used to enhance the security of calendar data:

- Dedicated Server
- Password Management
- Trust Management
- Networking
- Auditing
- Backup and Recovery
- Defense Against Denial of Service Attacks
- Application Security
- Calendar Administrator
- Oracle Real-Time Collaboration Web Conferencing Server

Dedicated Server

Oracle recommends that the Oracle Calendar server, if financial resources permit, be placed on a dedicated computer. In addition, turn off any TCP and UDP services on the host, which are not critical to the Oracle Calendar server (e.g., FTP, NFS server and client, or X server).

Password Management

Users and administrators should take advantage of the other directory administration tools provided for password management. For Oracle Calendar server `SYSOP` passwords, the following are the policy or procedure recommendations:

- Passwords should never be empty (or blank). This is especially important for the `SYSOP` or node administrator password.
- Passwords should never be words, names, or personal information which would be easy for others to guess.
- Passwords should be at least 8 characters long, and contain a combination of letters and numbers.
- Avoid using the same password to access the Oracle Calendar server and other mission-critical systems (although this may not be possible if these applications all use the same directory server).

Use the `unioidconf` or `unipasswd` utilities in the `$ORACLE_HOME/ocal/bin` directory to change the `SYSOP` password. For more information on changing `SYSOP` passwords, refer to Chapter 4 of the *Oracle Calendar Administrator's Guide*.

The Oracle Calendar server supports passwords of up to 63 characters. Users who set their passwords to more than 15 characters may not be able to sign-in using older Oracle Calendar clients which only support password lengths of 15 characters.

Other configuration parameters to consider are `ssignin` and `ssigninrestrictions` in the `[LIMITS]` section which control whether a user can use the desktop clients' automatic sign on feature. Parameter `invalidlogin_enable` in the `[ENG]` section can be used to enable the invalid sign on counting mechanism. This parameter should only be used when Calendar has been deployed in standalone mode. The `[ENG] allowpasswordchange_user` parameter can be used to stop users from changing their password. For more details on these parameters, refer to Chapter 3 of the *Oracle Calendar Reference Manual*.

Trust Management

Even if the server is dedicated to the Oracle Calendar server, there are still additional security safeguards to consider.

If you have security servers within your organization, consider sending audit trail information from the Oracle Calendar server to your central security server. Turn on auditing for the server and conduct spot audits of the commands issued by the calendar user. The server protects a great deal of aggregate data, so ensure that your backups are protected from theft. Consider separate ownership of the root or administrator (auditing account) and the calendar (server management) accounts. This would allow root or administrator to detect potential abuses by the calendar owner.

Networking

It is more secure to run mission-critical applications within firewall-protected intranets. Make sure that the dial-up connections to your intranet are protected. This can be improved by using one-time password technology (e.g. SecurID). As with many TCP/IP protocols, promiscuous listening (where the attacker monitors network traffic) is a threat in any broadcast network. A number of steps can be performed to reduce the risk of this threat:

- Physically protect hubs and routers. Use switched hubs when possible, especially on the server itself. Some hubs will block unauthorized, or unregistered MAC (or Ethernet addresses) on the LAN.

- Consider router filtering between untrusted internal networks.
- Use commercial firewalls to allow more complex TCP/IP filtering rules.

Auditing

The server generates a number of useful audit trails. It is important to become familiar with these audit trails, and to check them regularly. Many commands will create log files on error conditions. Routinely check for the existence of new log files, and review their contents. Monitor the *$ORACLE_HOME*/ocal/log/act.log for log on attempt abuses. Enable the act.log by setting the [ENG] activity parameter to TRUE in the *$ORACLE_HOME*/ocal/misc/unison.ini file. For more information on this parameter, refer to Chapter 3 of the *Oracle Calendar Reference Manual*. You can detect log on attempt abuses from the originating IP addresses. After the application is initially installed, record the file dates, file sizes, and checksums of all the binaries (the unicksum utility generates a checksum for a file). Periodically check that none of the binaries have been edited by comparing the current file dates, file sizes, and checksums with those recorded. Review <temp> directories for any suspicious files as they can be used as work areas.

Backup and Recovery

Calendar data is very important and should be backed up regularly. Refer to Chapter 14 in *Oracle Calendar Administrator's Guide* for more information.

Defense Against Denial of Service Attacks

Denial of service attacks usually attempt to exhaust or destroy any resource required by a system to deprive users of the services they would normally expect. The most common kind of denial of service attack is simply to send more traffic to the network address of an application than the application can handle.

This section focuses on the class of denial of service attacks that take place while the Oracle Calendar server is running, and for which the attacker has not obtained direct access to the system on which the Oracle Calendar server is running.

The Oracle Calendar server can be configured to warn and protect against such attacks, by configuring parameters in the [ENG] section of the *$ORACLE_HOME*/ocal/misc/unison.ini file. The following is a list of configurable parameters, including brief descriptions, which can be used to warn and protect against denial of service attacks:

- dos_maxsessionsperaddr - Control the number of client connections from a specific IP address.
- dos_maxsessionsperaddrblacklist - Restrict connections to the Oracle Calendar server based on IP address.
- dos_maxsessionsperaddrredline - Specify the maximum number of client connections from one IP address before logging begins.
- dos_maxsessionsperaddrwhitelist - Specify a list of IP addresses that are exempted from being blocked.
- dos_timeoutdatareceived - Timeout value for non-header data.
- dos_timeouthandshake - Timeout value for handshake data.

> **See Also:** Chapter 3 of the *Oracle Calendar Reference Manual* for more information on these parameters

Application Security

The server supports a very rich set of user-controlled access privileges (or rights). It is important to train end users on how these capabilities can be managed, so that users' information is protected from unauthorized access.

Try to limit assigning designate rights. You should only give designate rights to trusted individuals.

The default designate rights should be no designate rights. Set the viewing rights to no privileges, and add privileges as needed.

There are a number of overall limits, set by the server administrator, that can be set for all users.

Disabling attachments (`[LIMITS] allowattachments`) can prevent users from propagating proprietary information improperly. Setting maximum attachment size (`[LIMITS] maxattachmentsize`) can help prevent denial of service attacks caused by very large files that cause a server to run out of disk space. For more details on these parameters and others, refer to Chapter 3 of the *Oracle Calendar Reference Manual*.

Calendar Administrator

It is recommended that you always use Secure Sockets Layer (SSL) encryption to access the Oracle Calendar Administrator, in order to protect sensitive information. Always use the `https://` URL prefix instead of `http://` to ensure secure access.

Oracle Real-Time Collaboration Web Conferencing Server

The Oracle HTTP server (OHS) allows clients and servers to authenticate over SSL using X.509v3 certificates. The certificates are stored in an Oracle Wallet, a container in which certificates and trusted certificates are stored and managed.

When communicating with the Web Conferencing server, the Oracle Calendar server uses a secure HTTP connection (HTTPS) to OHS. To establish an HTTPS connection, a wallet is used with a default certificate which ensures that the information passed between the two servers is encrypted. The location of the wallet is defined by the `[CONFERENCING] walletfile` parameter in the `unison.ini` configuration file. The password of the wallet is defined by the `[CONFERENCING] walletpassword` parameter.

It is possible to replace the default certificates in the wallet with ones from another certificate authority.

The parameter `[CONFERENCING] url` defines the URL used by the Oracle Calendar server to access the Web Conferencing server. By default Secure Sockets Layer (SSL) encryption is used and therefore the "`https://`" URL prefix is used rather than the "`http://`" to ensure secure access. The password and ID used to authenticate to the Web Conferencing server are defined by the parameters `[CONFERENCING] siteid` and `siteauthkey`.

For more information on the Real-Time Collaboration Web Conferencing parameters, refer to Chapter 3 of the *Oracle Calendar Reference Manual*.

Securing Oracle Content Services

Oracle Collaboration Suite 10*g* Content Services (Oracle Content Services) provides the basic security infrastructure required by any shared, network-accessible system, including authentication and authorization. This section describes the architecture and configuration of security in Oracle Content Services. It contains the following topics:

Authentication Using Oracle Internet Directory

Authentication is a process in which a user provides some proof of identity (called a *credential*, which is often constructed from a user's password by means of a hashing or encryption algorithm) before that user can attempt to access objects in the system. Oracle Content Services uses Oracle Internet Directory, Oracle's LDAP-compliant directory service, for authentication.

Users provide their user name and password to the client software. These are passed to the Oracle Content Services protocol servers, which, in turn, pass them to Oracle Content Services for authentication. Then, Oracle Content Services passes the user name and password to Oracle Internet Directory. Oracle Internet Directory determines whether the user name and password are valid for the user.

Security Considerations for Protocol Servers

This section describes the security considerations for protocol servers and contains the following topics:

> **Note:** The defined behavior of some industry-standard protocols is not inherently secure. Oracle has no control over the defined behavior of these protocols, and these security issues do not represent defects in Oracle software.

FTP/FTPS

The File Transfer Protocol (FTP) sends unencrypted user passwords across the network, which means that if one of these passwords is intercepted, then it could provide access to all systems controlled by Oracle Internet Directory for that user. To provide more security, users should create an FTP password (rather than the default Oracle Internet Directory password) to authenticate against FTP.

The FTP password is stored in Oracle Internet Directory and is different from and in addition to the regular Oracle Internet Directory password. Each user can have only one FTP password in one Oracle Content Services domain. FTP requires users to log in with an FTP password rather than an Oracle Internet Directory password.

Users can set their FTP password on the User Preferences page in Oracle Content Services. Users can also use the Oracle Internet Directory Self-Service Console to set their FTP password, by setting the **content** password entry that appears in the Application Passwords section of the Change Password page.

As an alternative, users can use FTPS. FTPS is FTP with the added option of Secure Socket Layer (SSL) security. FTPS does not require an FTP password.

By default, FTP and FTPS are disabled.

> **See Also:** *Oracle Content Services Administrator's Guide* for detailed information about how to enable FTP and FTPS

HTTP/WebDAV

The HTTP/WebDAV protocol allows *digest* (hashed challenge/response), Single Sign-On (from the browser only), and persistent cookie (if the domain and then the user enables the feature) authentication. Whether HTTP/WebDAV uses SSL depends on the configuration of Oracle HTTP Server and on whether Oracle Content Services has been configured for SSL.

Oracle Drive is a desktop client that uses the WebDAV protocol to access Oracle Content Services. After it is installed, Oracle Drive appears as a mapped drive in Windows Explorer. Oracle Drive also provides file synchronization capabilities between your local computer and Oracle Content Services.

> **See Also:** SSL Configuration for Oracle Content Services

Network Channel Encryption

The FTP and HTTP/WebDAV protocols do not encrypt the network channel by default. This means that files transferred using these protocols are susceptible to interception. If you are unwilling to accept this behavior, then you should disable these protocols or configure them to use SSL (HTTP/WebDAV only).

Malicious Uploads

Because user quota is managed asynchronously through the Quota Agent, it is possible for a malicious user to upload a very large file for filling up disk space. To prevent such attacks, you can limit the size of any single file uploaded to Oracle Content Services by setting the `IFS.DOMAIN.MEDIA.CONTENTTRANSFER.ContentLimit` domain property. If you try to upload a file beyond the specified limit, then the upload fails. This limit does not apply to administrators.

When this property is set to `0`, the default value, the content limit is disabled. You will be able to upload any file whose size is within the last calculated available quota, as of the beginning of the upload.

> **See Also:** *Oracle Content Services Administrator's Guide* for more information about the `IFS.DOMAIN.MEDIA.CONTENTTRANSFER.ContentLimit` domain property

Client Session Timeout Period

The *client session timeout period* is the number of minutes of idle time after which a Web user interface session expires. By default, the client session timeout for Oracle Content Services is set to 30 minutes. To change this value, perform the following steps:

1. Access the Oracle Collaboration Suite Control and navigate to the Collaboration Suite Home page.
2. Select **OC4J_Content** and click **Stop**.
3. Click **OC4J_Content** to navigate to the OC4J_Content Home page.
4. Click **Applications**, then click **content** in the Deployed Applications table.
5. On the Applications: content page, click **content** in the Web Modules table.
6. On the Web Module: content page, in the Administration section click **General** under the Properties heading.
7. In the Session Configuration section, change the value for **Session Timeout (minutes)**.
8. Click **Apply**, then click **OK** on the Confirmation page.
9. Return to the Collaboration Suite Home page, select **OC4J_Content**, and click **Start**.

If you have enabled Oracle Records Management, then you can also set the client session timeout period for Oracle Records Management. Repeat these steps for OC4J_RM to change the client session timeout period for Oracle Records Management.

HTTPS Configuration for Oracle Content Services

Perform the following steps only if you configured Oracle Content Services prior to configuring HTTPS:

1. Access the Application Server Control for Collaboration Suite.
2. Sign in as the `ias_admin` user.
3. In the Collaboration Suite Home Page, click **Content Services**.
4. Click **Domain Properties**.
5. Edit `IFS.DOMAIN.APPLICATION.ApplicationPort` to point to the HTTP Server SSL Port.
6. Set `IFS.DOMAIN.APPLICATION.ApplicationUseHttps` to `TRUE`.
7. Restart all OPMN processes:

```
ORACLE_HOME/opmn/bin/opmnctl stopproc ias-component=Content
ORACLE_HOME/opmn/bin/opmnctl starproc ias-component=Content
```

SSL Configuration for Oracle Content Services

You can configure Oracle Content Services to use SSL. Before you do this, you must configure Oracle HTTP Server to use SSL.

You can also use SSL to connect to Oracle Internet Directory. Before you do this, Oracle Internet Directory must be configured for SSL.

> **See Also:** Enabling SSL in Oracle Content Services and Connecting to Oracle Internet Directory Using SSL

SSL Connection to Oracle Internet Directory

You can provide SSL settings after Oracle Content Services has been installed and configured. Before you do this, Oracle Internet Directory must be configured for SSL.

See Also: Connecting to Oracle Internet Directory Using SSL

Oracle Content Services Schema Password

The Oracle Content Services schema password is stored in the following locations:

- Oracle Database
- Oracle Internet Directory
- Any Oracle Content Services Applications tier where you are running repository metrics

See Also: *Oracle Content Services Administrator's Guide* for more information about repository metrics, and *Oracle Collaboration Suite Administrator's Guide* for more information about changing the Oracle Content Services schema password

Oracle Records Management

Oracle Records Management is a records management application that ships with Oracle Content Services.

When you install Oracle Content Services, Oracle Records Management is installed automatically, but the application is disabled by default. You can use the Oracle Collaboration Suite Control to enable Oracle Records Management. You can also configure records management-related metrics.

See Also: *Oracle Content Services Administrator's Guide* for more information about enabling Oracle Records Management

Using a Retention Hardware Solution

Oracle Content Services provides retention hardware capabilities through partnerships with Network Appliance and EMC. You can use the Oracle Collaboration Suite Control to integrate Oracle Content Services with Network Appliance SnapLock or EMC Centera.

To integrate Oracle Content Services with a records management retention device, you must first install the hardware (either EMC Centera or Network Appliance SnapLock). Then, you must specify credential information for the hardware and set retention-related domain properties using the Oracle Collaboration Suite Control.

Once you have created a file plan and defined retention policies in Oracle Records Management, Oracle Content Services will designate appropriate content as records to be stored in a records management retention device.

See Also: *Oracle Content Services Administrator's Guide* for more information

Securing Oracle Mail

This section describes how to secure Oracle Collaboration Suite 10*g* Mail. It contains the following topics:

- Securing Oracle Mail Protocol Servers
- Configuring Oracle Mail Protocol Servers for SSL
- Configuring SSL Between Oracle Collaboration Suite 10g WebMail and Oracle Internet Directory

Securing Oracle Mail Protocol Servers

A thick client that is used to connect to the Oracle Mail message store uses one of the Internet standard e-mail protocols, Post Office Protocol version 3 (POP3) or Internet Mail Access Protocol version 4 (IMAP4). Regardless of the protocol used to connect to the protocol server, you need to provide a user name and password. The server confirms the user name and password against the **Lightweight Directory Access Protocol (LDAP)** server of Oracle, Oracle Internet Directory.

Secure Sockets Layer (SSL) is a protocol for transmitting private documents over the Internet. SSL works by using a public key to encrypt data that is transferred over the SSL connection. Many Web sites use the protocol to obtain confidential user information, such as credit card numbers. By convention, URLs requiring an SSL connection start with `https:` instead of `http:`. You can configure the IMAP and POP protocol servers to use SSL for securely communicating with and authenticating clients. Two separate server instances are required to use both SSL and non-SSL connections because one server instance cannot manage both types of connections. Administrators can choose to configure an existing server instance or create a new instance. By default, server instances are configured to manage only non-SSL connections. The default listening end points for both IMAP and POP protocol servers are created in the `listener.ora` file during installation. It is necessary to ensure that the client connects to the new IMAP/SSL listener. Almost all e-mail clients have an option that enables you to connect securely using SSL.

The primary goal of the Transport Layer Security (TLS) protocol is to provide privacy and data integrity between two communicating applications. The protocol is composed of two layers, the TLS Record Protocol and the TLS Handshake Protocol. At the lowest level, layered on top of some reliable transport protocol, such as TCP, is the TLS Record Protocol. The TLS Record Protocol provides connection security that has two basic properties:

- The connection is private
- The connection is reliable

TLS enables the communication between either client and server, server to server, or both to be secured (more so than traditional SMTP which passes most of its data in clear text over its communication channel).

The security is negotiated between the two sides, so enabling it for a server does not force all other parties to use it, which is important since many mail servers might not support it, or require it. Essentially, TLS allows the user to use the best available security on the server they are using.

Configuring Oracle Mail Protocol Servers for SSL

The IMAP and POP protocol servers can be configured to use SSL for securely communicating with and authenticating clients. To use the SSL client connections, administrators can configure an existing server instance or create a new instance. Two separate server instances are necessary to use both SSL and non-SSL connections. One server instance cannot manage both types of connections. By default, server instances are configured to manage Internet connections only. The default listening end points for both IMAP and POP protocol servers are created in the `listener.ora` file during installation.

To configure an SSL server instance:

1. Log in to Oracle Enterprise Manager 10*g* Application Server Control Console.
2. Select the application server instance where Oracle Mail is installed.
3. Click **Mail Application**.
4. Click **IMAP Server** or **POP Server**.
5. Click the process instance.
6. Select `IMAPSSL`, `POPSSL`, or `Custom` from the **Presentation Name** drop-down list in the **General Parameters** section.

   If you select `Custom`:

   a. Provide a specific presentation name in the corresponding field.

   b. Change the **SSL Enabled** parameter to `True` and verify that there is a description entry in the `listener.ora` file for the presentation name you specified.

   c. Verify in the `listener.ora` file on the Oracle Collaboration Suite Applications Tier host that the SSL presentation specific `PROTOCOL` is set to `TCPS` and that the `PORT` is set to the default SSL port number of the protocol server. The default SSL port number for IMAP is 993 and for POP it is 995.
7. Obtain an SSL certificate and configure the network listener for SSL as described in "Certificates and Oracle Wallets" on page 5-5.

> **See Also:** "Managing Oracle Mail Servers and Instances" in Chapter 3 of *Oracle Mail Administrator's Guide* for more information about configuring Oracle Mail protocol servers

Configuring SSL Between Oracle Collaboration Suite 10*g* WebMail and Oracle Internet Directory

Oracle Collaboration Suite 10*g* WebMail (Oracle WebMail) relies on Oracle Internet Directory for authentication, through Oracle Application Server Single Sign-On, to look up users in the directory, and to access the user's address book. The connection between Oracle WebMail and Oracle Internet Directory can be secured by configuring SSL.

To configure SSL for the connection between Oracle WebMail and Oracle Internet Directory, add the following properties to the `oc4j.properties` file:

```
oracle.mail.ldap.connectssl=true
oracle.mail.ldap.sslport=4031
```

See Also: "Oracle WebMail Client Properties" in Chapter 4 of *Oracle Mail Administrator's Guide* for a complete list of Oracle WebMail client properties

Configuring Oracle Mail Protocol Servers for TLS

TLS is used to provide a layer of security for the protocol servers that do not have a dedicated port for handling SSL connections, such as the SMTP Inbound server.

For those protocol servers that must handle both SSL and non-SSL traffic on the same port, the ability to switch over to SSL mode at the will of the client, is necessary. With TLS configured, clients can connect in non-SSL mode, check the server capabilities, and, if SSL is supported, issue a `STARTTLS` command and switch over to SSL mode.

To configure a TLS server instance:

1. Log in to Oracle Enterprise Manager 10*g* Application Server Control Console.
2. Select the application server instance where Oracle Mail is installed.
3. Click **Mail Application**.
4. Click **IMAP Server** or **SMTP Inbound Server**.
5. Click the process instance.
6. Specify the wallet location in the **Wallet Location for TLS Support** parameter in the **General Parameters** section. Enter `file:` followed by the absolute path to the directory in which the SSL wallet is located.
7. Set the **Support STARTTLS Command** parameter to `True`.
8. Obtain an SSL certificate and configure the network listener for SSL as described in "Certificates and Oracle Wallets" on page 5-5.

Configuring SASL for Oracle Mail

Simple Authentication and Security Layer (SASL) is a method for adding authentication support to connection-based protocols. To use SASL, a protocol includes a command for identifying and authenticating a user to a server and for optionally negotiating protection of subsequent protocol interactions. If the use of SASL is successfully negotiated, then a security layer is inserted between the protocol and the connection.

To configure SASL for Oracle Mail:

1. Configure Oracle Internet Directory to generate dynamic password verifiers.

   Oracle Internet Directory generates the attribute `orclrevpwd` when you provision a user if the attribute `orclpwdencryptionenable` in the realm password policy entry is set to `1`. Therefore, you must set `orclpwdencryptionenable` to `1` before you provision users. Alternatively, if users were provisioned before you set `orclpwdencryptionenable`, all users must reset their user passwords to trigger the generation of the encrypted value.
2. Use Oracle Enterprise Manager to edit the following protocol server parameters at the default level:
   - **Allow Clear Text Login**: `False`
   - **SASL Authentication**: `Enabled`
   - **SASL Protection**: `Confidentiality`

See Also: "Managing Oracle Mail Servers and Instances" in Chapter 3 of *Oracle Mail Administrator's Guide* for more information about configuring protocol server parameters

Providing Virus Protection

E-mail viruses have long been a concern in any IT-enabled industry. Viruses cause lost productivity and consequent losses in revenue. As a result, it is important to eliminate viruses wherever and whenever possible. According to the Radicati Group, the economic damage caused by computer viruses is likely to rise from almost $30 billion in 2003 to over $70 billion in 2007.

Viruses can enter a system through infected e-mail messages. The first and best place to detect and eliminate e-mail viruses is where they enter the system, which is at the Message Transfer Agent (MTA)/Simple Mail Transfer Protocol (SMTP) level. Oracle currently provides tight integration at the MTA level with Symantec's antivirus offering, but any SMTP-based antivirus scanner can be used to relay to the `SMTP_IN` MTA of Oracle.

These third-party virus-scanning software offerings scan each message that passes through the SMTP server. The virus-scanning software can be set to reject the infected message on arrival or fix the virus (if possible), preventing the virus e-mail from entering the e-mail system. If there is a virus outbreak before the third-party software has a chance to upgrade itself, then some virus e-mail messages may already be present in the e-mail message store.

There will always be a lag between the time that a new virus spreads and the time that antivirus vendors fingerprint the virus, update their signature database, and you apply the update to your antivirus scanner. So, there is always a possibility that a virus could enter the message store. The Oracle Mail virus scrubber process, described in Prescanning Using the Virus Scrubber, can be used to scan the entire message store, repair, or remove virus e-mail messages.

See Also: "Symantec AntiVirus Scan Engine" in Chapter 6 of *Oracle Mail Administrator's Guide* for more information about Oracle Mail virus protection

Prescanning Using the Virus Scrubber

The Oracle Mail virus scrubber is a server process that scans for and cleans up virus-infected e-mail messages already in the message store. When rapid measures are required to immediately clean virus-infected messages, the virus scrubber prescans a message store to isolate suspect messages from a system, based on headers, subject, or attachment names. Prescanning isolates suspect messages so that users are not able to access them and possibly cause damage. Prescanning never deletes a message. After prescanning, the virus scrubber uses the external scanner to individually scan the isolated messages. A message that is deemed clean or repaired by the virus detection software is restored to its original folder.

If a message is identified as infected and not repairable, then the administrator can either delete the message immediately or quarantine it to a special folder for later processing. For example, an infected message can be quarantined to wait for a future release of virus definitions that may be able to repair the message. Oracle Mail can be configured to send a message to either the mail recipient or sender notifying them that it was identified as infected. Such notifications are useful to explain to users why some of their messages seem to have disappeared.

The e-mail message store of Oracle Collaboration Suite is Oracle Collaboration Suite Database, which has been tuned to excel at sifting, sorting, and operating on large amounts of data. Using the `MAIL_AV` antivirus packages within the message store database, administrators can scrub the entire message store very quickly and efficiently with no downtime for their users. Oracle Global IT, for example, was able to scrub 2.5 terabytes of e-mail to quarantine over one million infected e-mails in well under half an hour, while users experienced no downtime. Compare this with a distributed message store, where each store that has been scrubbed is often reinfected by an infected e-mail from a to-be-scrubbed store, or bringing offline all the message stores while they are individually scrubbed.

The Oracle Mail virus scrubber is different from the SMTP-based virus scanner that filters out virus-infected messages before they enter the system. The Oracle Mail virus scrubber is a necessary complement to the SMTP-based virus scanner because new types of viruses continue to appear before virus detection software can be updated to detect and repair them. There is always a possibility that by the time the virus software is updated, some infected messages have already entered the system. The virus scanner can be used to retroactively rid the system of such viruses. This message store-based scanner can also be used to scan viruses coming in through a non-SMTP route such as IMAP append.

Rejecting Spam

Another widespread issue related to e-mail is that of unsolicited commercial and bulk e-mail, which is commonly referred to as spam. A study released by Ferris Research has estimated the cost of spam to U.S. corporations at $8.9 billion a year.

As in the case of virus control, Oracle Collaboration Suite defers to vendors who specialize in complex spam management, but provides routing control that can prevent some spam from entering the Oracle Mail system.

When routing control is utilized, every e-mail passing through the Oracle Mail MTA is subject to checks for both trusted and nontrusted e-mail features in the:

- Sender's address
- Sender's domain
- Recipient's address
- Recipient's domain
- IP address of sending computer
- IP address of receiving computer
- Sender/recipient pairs
- E-mail headers with specified values
- Attachment names with specified values

Oracle Collaboration Suite also provides Denial of Service prevention. If more than a specified number of connections or messages are received from a host within a given time interval, then all further messages from that host are blocked. The time period and count are both configurable. If the MTA cannot perform reverse DNS lookups for e-mail sending domains due to deployment reasons, then you could use `SMTP-AUTH`, but security is still required.

> **See Also:** "Oracle Mail Routing Control" in Chapter 8 of *Oracle Mail Administrator's Guide* for more information about configuring Oracle Mail to reject spam

Preventing Mailing List Abuse

Although server-based mailing lists or distribution lists are useful, they can be abused if they are not effectively controlled and managed. The mailing list server in Oracle Collaboration Suite provides the ability to manage subscriptions to lists and also control who can or cannot send e-mail to the list.

Mailing lists can be of the following types:

- Announcement: Users can send messages to the list but replies to the entire list are not allowed.
- Discussion: Users can send messages and also reply to the list.
- Edited: Only designated editors can post messages to the list. Messages from other users are rejected. The list owner specifies editors for the list.
- Moderated: A moderator chosen by the list owner screens all messages. Only approved messages are posted to the list.

Mailing lists may also have one of the following subscription criteria:

- Unrestricted or open lists: Open to subscription by all users.
- Restricted lists: New subscribers need to be approved by the list owner.
- Closed lists: A user is invited to join the list to subscribe.

These subscription criteria ensure that content for mailing lists is sent and received by users.

Implementing Secure Multipurpose Internet Mail Extension (S/MIME)

Although it is possible to secure network traffic between the client and server when you authenticate and retrieve your e-mail, any e-mail that is sent is subject to transmission through several computers on the Internet. The Internet standard for relaying e-mail between servers is SMTP, and although this protocol has an SSL-like extension called TLS, there is no guarantee that every SMTP server relaying your message will use TLS.

Many e-mail clients support the use of S/MIME, and Oracle Collaboration Suite also supports the use of this feature. S/MIME allows a certificate, similar to those used for enabling SSL on servers, to be imported into a client wallet within the browser or e-mail client. Therefore, any S/MIME compatible client can be used with Oracle Mail.

When you compose an e-mail, it can be electronically signed using the certificate in your local wallet. The e-mail is sent in the clear, but the recipients can detect whether the e-mail content has changed in transit and also verify that the senders are indeed who they say they are. You can also choose to encrypt the e-mail for the recipients, so that only the recipients can read that e-mail by using their local certificate to decrypt the e-mail.

For example, if Alice wants to send Bob an encrypted e-mail, then Alice needs the public part of Bob's key. Similarly, for Charlie to verify an e-mail signed by Alice, Charlie needs the public part of Alice's key. It is easy to see that for a large organization, expecting every user to keep the public part of every other user's key is unreasonable. Oracle Internet Directory can be used to store a central repository of public keys of users so that S/MIME implementation becomes easy.

> **See Also:** Chapter 6 of the *Oracle Mail Administrator's Guide* for information about Oracle Mail security

Securing Oracle Mobile Collaboration

Security in Oracle Collaboration Suite 10*g* Mobile Collaboration is provided by the underlying technology in Oracle Application Server Wireless.

Oracle Application Server Wireless combines advanced content transformation, device adaptation, and network adaptation services with end-user customization, providing enterprises, mobile operators, content providers, or wireless ISPs with a platform to create and deploy mobile applications.

> **See Also:** *Oracle Application Server Wireless Administrator's Guide* for information about OracleAS Wireless security

Additionally, security for Oracle Mobile Data Sync is provided by the underlying technology in Oracle Calendar.

> **See Also:** Securing Oracle Calendar for more information

This section contains the following topics:

Introducing Push Mail Security

Mobile Push Mail provides an always-available e-mail experience on supported mobile devices. Mobile Push Mail is implemented as a client-server system. The client side code is referred to as Push Mail Client and the server side code is referred to as Push Mail Server.

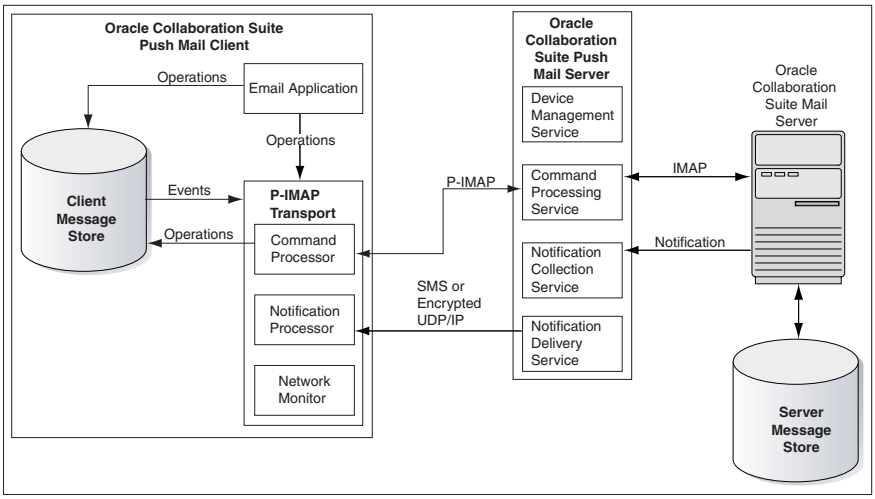
The Push Mail Client contains three major components: an e-mail application, a client message store, and an e-mail transport. The e-mail application (such as the Inbox application on Microsoft Windows mobile devices) provides a user interface for manipulating messages in the client message store. The e-mail transport synchronizes the client message store with the e-mail server. The e-mail transport for the Push Mail Client is called the P-IMAP Transport because it uses the P-IMAP protocol to communicate with Push Mail Server.

Push Mail Server supports the P-IMAP protocol over several bindings. In 10*g* Release 1 (10.1.1), it supports P-IMAP bindings over the HTTP and HTTPS protocols. In addition, Push Mail Server also supports notification delivery to mobile devices using connectionless, Short Messaging Service (SMS) channel or UDP/IP channel. It also has a device management service that, among other things, installs, updates, uninstalls, and wipes Push Mail Client on the mobile device over the air (OTA). Push Mail Server communicates with the Oracle Collaboration Suite Mail server, which is where all the e-mail messages are stored. Each client message store has a corresponding server message store in the e-mail server.

Push Mail System Architecture

The Mobile Push Mail System Architecture is shown in Figure 2–3.

Figure 2–3 Mobile Push Mail System Architecture



Push Mail Client runs on a mobile device and has three major components: e-mail application, client message store, and P-IMAP Transport.

The e-mail application (such as the Inbox application on Microsoft Windows mobile devices), provides the user interface for mobile users. The e-mail application may support multiple client e-mail accounts.

In response to the user's actions, the e-mail application invokes the methods provided by P-IMAP Transport, or performs operations on the client message store. For example, if a user clicks Connect in the e-mail application, the e-mail application calls the connect method of P-IMAP Transport. If the user reads or otherwise manipulates a message, the e-mail application sets the appropriate flag associated with the message in the client message store. In response to this operation, the client message store generates events. P-IMAP Transport receives these events through a call back function it registers with the client message store. The call back function processes the events and inserts appropriate P-IMAP client events into the event queue.

In addition to providing a method for the e-mail application and responding to events generated by the client message store, the Command Processor component of P-IMAP Transport communicates with the Command Processing service of Push Mail Server to send and receive updates and apply these updates to the client message store. This communication is done using the P-IMAP protocol.

The Command Processing service of Push Mail Server establishes a session (server session) with Oracle Collaboration Suite Mail Server on behalf of the user. The communication protocol used in this session is IMAP. Push Mail Server uses this session to update the server message store and get changes for the client message store. This session may use the IDLE command of IMAP protocol to get unsolicited server updates. Alternatively, it may occasionally issue the FETCH command to poll the server for changes on the client. The server session is kept alive for an optimal period of time.

P-IMAP defines two ways in which a client can receive data pushed (notification) from a server. The first, called In-Band, uses the IDLE command of P-IMAP so that the client can listen to the server on the same channel it uses to communicate with the server. The second, called Out-Band, requires the server to use a separate channel to push the data. In Oracle Collaboration Suite 10*g* Release 1 (10.1.1), the separate channel can be a Short Message Service channel or UDP/IP channel. If a client chooses to use Out-Band

notification with the UDP channel, the P-IMAP Transport establishes a UDP/IP session with the server.

The Notification Processor component of P-IMAP Transport waits for the arrival of a notification from the Notification Delivery Service of Push Mail Server either by intercepting and inspecting the incoming SMS messages, or by listening for notifications from the server through a UDP/IP channel. When a notification is received, Notification Processor inserts an appropriate event into the event queue, depending on the content of the notification.

Oracle Collaboration Suite Mail Server publishes notification messages for users who have registered to receive notifications for certain events. The Notification Collection service of Oracle Collaboration Suite Push Mail Server registers the P-IMAP notification filters for each user and retrieves the notifications from the notification message queue. The Notification Delivery Service of Oracle Collaboration Suite Push Mail Server then delivers the messages to the mobile devices through the selected Out-Band notification channel, the encrypted UDP/IP channel or the Short Message Service channel.

The Network Monitor component of the P-IMAP Transport monitors network connectivity of mobile devices. It calls back registered observers whenever there is a change in network connectivity. Network Monitor gathers information about available networks, make a decision as to which network to use, and passes relevant information about the network to the callback functions. The information passed by Network Monitor enables callers to determine the type of network (cradle, WiFi, GPRS, and others). Observers can decide what to do based on the network status.

The Device Management service is responsible for installing, updating, and removing P-IMAP Transport and other client programs, and setting/resetting some registry entries for users.

Mobile Push Mail Security

To better understand the security features of Mobile Push Mail, it is important to know the steps a mobile user takes to install and use the Mobile Push Mail:

- Register and download Push Mail Client

  The user must first use the Mobile Preferences web page through a desktop browser in order to identify himself and register his device. Based on the device capability, the user is shown a menu of client programs available for installation. If the device can run Push Mail Client, and the binary code for the client for the device is available in Push Mail Server, then the user is presented with the option to install Push Mail Client. If the user chooses to install the client, the Device Management service sends an SMS message to the device. The SMS message contains a URL constructed for the device. When the user goes to the URL, the Device Management service installs the necessary binaries on the device and sets the registry entries.

- Normal use with In-Band notification

  The user uses the Mobile Push Mail feature with a client that uses In-Band notification mechanism. This applies to all users using the Windows Mobile client supplied by Oracle Corporation or the BlackBerry client supplied by Consilient Corporation.

- Normal use with Out-Band notification

The user uses the Mobile Push Mail feature with a client that uses Out-Band notification mechanism. This applies to many Nokia and Sony-Ericcsson clients supplied by Consilient Corporation.

- Upgrading the Push Mail Client

  Whenever a new version of Push Mail Client is available (that is, uploaded to Oracle Collaboration Suite Push Mail Server), the Device Management service of Push Mail Server notifies the user with an SMS message. The user can upgrade the client program by going to the URL contained in the message.

- Loss of device

  When the user loses a device, the device must be locked down and wiped cleaned of all programs and data. The user or the administrator can do this by using the remote lock down feature of Push Mail Server. The mobile user can also do it by accessing the Mobile Preferences page and deleting the device from your list of devices.

- Preventing malicious actions against the client or the server

  In these days of increasing sophistication in attack against computing system on the Internet, Mobile Push Mail has been designed to prevent and withstand malicious use of both the client and the server. This requires the server to be deployed properly, as described in the guidelines in *Oracle Collaboration Suite Deployment Guide*.

The security features in each step are described in the following topics:

- Downloading and Registering Push Mail Client
- Normal Use with In-Band Notification
- Normal Use with Out-Band Notification
- Loss of Device
- Preventing Malicious Actions Against the Client and the Server
- Deployment Options for Push Mail Server

Downloading and Registering Push Mail Client

To use the Push Mail feature, users must first be created as valid users, and be provisioned to access e-mail. The user then must register each of his mobile devices to be able to download the Push Mail client. To register a device, the user must first log in to the Oracle Collaboration Suite portal using the single sign-on password. This interaction is done through HTTPS, which provides mutual authentication of client and server, privacy, and message integrity.

From the portal page, the user chooses the Mobile Preferences page on which the user registers the mobile device. For each device, the user is presented with the mobile client programs available for the device. If the user's device is one of the supported devices, then the user can select the appropriate choice, and click Configure. The Push Mail Server performs the following tasks to install the push mail client program on the device.

1. The Push Mail Server generates a unique key over a very large key space that is practically impossible to guess.
2. The server associates the key with the user and his device. It then constructs a URL to the server with the key as a parameter, and sends the URL through an SMS message to the user's device.

3. The user can go to the URL to download a bootstrap program to set up the Push Mail Client. For those devices that are not capable of receiving SMS messages or for additional security, the URL with the key is also displayed to the user so that the user can enter it in the browser's address bar on the device. In any case, the key is good until the user completes the installation or returns to the Mobile Preferences page to download the client for the same device. This is to prevent replay attack on the server.

Depending on the device type, the bootstrap program will differ, and the user may be required to take slightly different actions. Regardless of the differences, the bootstrap program will use HTTPS protocol to download the Push Mail Client. Again, the use of HTTPS guarantees authentication of client and server, privacy of communication, and integrity of the messages being exchanged between the client and the server.

The bootstrap program will also set appropriate registry or environment variables for the Push Mail Client.

Normal Use with In-Band Notification

The security features designed for normal use with In-Band notification are: mobile device security, security of communication between the P-IMAP Transport and the Push Mail Server, and security of Push Mail Server. These features are discussed in the following sections:

- Mobile Device Security
- Communication Security
- Push Mail Server Security

Mobile Device Security

If not managed carefully, a mobile device can become a weak link in enterprise security. In 10*g* Release 1 (10.1.1), some aspects of mobile device security are built into the system while others are provided as recommended guidelines the mobile users should adhere to. It is imperative to recognize that the mobile user is the first line of defense against attacks on the Push Mail system.

When the Oracle Collaboration Suite Push Mail account is used on a mobile device, the e-mail application prompts the user for single sign-on password. The e-mail application passes the password to P-IMAP Transport, which uses it to connect the user to the Push Mail Server. The e-mail application may provide an option to store the password. The mobile user should carefully consider the convenience of not reentering the password frequently against the risk to enterprise security before storing the password.

Many e-mail applications also permit the user to specify the idle time period after which the application is locked requiring the user to reenter the password after the application has been idle for that time period. Although security is enhanced by a short idle time period, it also increases the inconvenience associated with using the application. Users must choose an idle time period that strikes a good balance between the conflicting goals of security and convenience. It should be noted that even if the e-mail application is locked, the P-IMAP Transport continues to receive pushed messages from the server.

The application lock out feature is necessary, and prevents a casual thief from obtaining the information stored on the device, but it is not sufficient to prevent a determined thief from getting the information. To further strengthen mobile device security, enterprises should install file encryption software on the device. These encryption software work transparently with any applications and they encrypt all

data stored in the device using well-known encryption algorithms with varying key lengths.

To further minimize the risk of a mobile device being used for malicious purposes, the user should wipe the device as soon as it is lost or stolen. There are two ways to wipe the device. One way is for the user to log in to Oracle Collaboration Suite server, go to the Mobile Preferences page, and remove the device from the list of communication devices. As a result of this action, the Mobile Push Mail server sends a lock down message to the device. The device removes the message store associated with the account, removes the P-IMAP Transport, and resets the appropriate registry entries.

Alternately, the user can inform the administrator who can use Enterprise Manager to remotely wipe the device using the lock down command.

Other defensive measures against malicious use include not installing any software on the device that is not provisioned by the enterprise administrator and not browsing any sites whose URL is not provisioned by the enterprise administrator.

Communication Security

Push Mail Client uses P-IMAP to communicate with Oracle Collaboration Suite Push Mail Server. In 10*g* Release 1 (10.1.1), P-IMAP is tunneled through the HTTPS protocol, which provides authentication of client and server using digital certificates, privacy of message using encryption, and integrity of message (to make sure that the message is not tampered with in transit) using message digest.

To enable server authentication by the client, a public key certificate issued by a Certificate Authority (CA) known to the client, must be installed on the server. The HTTP software on the client checks the certificate to authenticate the server. For example, Microsoft Windows mobile devices have built-in root certificates from several well-known CAs, and can validate certificate issued by these CAs.

The server may authenticate the client by simply requiring the client to enter a valid single sign-on password. This password is used by the Command Processing Service of the Push Mail Server to log in to the Oracle Collaboration Suite Mail server. The Oracle Collaboration Suite Mail server uses Oracle Application Server Identity Management service to authenticate the user.

To further strengthen the client authentication, a public key certificate can also be installed on the client. Oracle Application Server includes a Certificate Authority that can be used to generate client certificate that Oracle servers can use to authenticate the client.

For Microsoft Windows mobile clients, Microsoft has added support for the installation of local digital certificates. Local digital certificates can be used to allow specific devices such as the Pocket PC to access a Web site while preventing other devices that do not have the certificate installed. Network administrators can install their own root and local certificates, as well as manage the certificates on the Pocket PC. Because the certificate manager user interface is available to users of the Pocket PC, network administrators should tell users *not* to delete certificates or they may not be able to access some SSL Web sites.

The SSL client on the device negotiates with the SSL program on the server to agree on the encryption algorithm and session key to be use for encrypting messages.

Once the secure HTTPS channel is established, Push Mail Server sends an encryption key for the device management commands to the client. The client stores the encryption key in a registry.

Push Mail Server Security

Oracle Collaboration Suite Deployment Guide includes various deployment configurations for Oracle Collaboration Suite Push Mail Server. Because Push Mail Server must communicate with Push Mail Client over a wireless network, Push Mail Server should be placed behind a firewall that can filter the incoming traffic from the Internet and reject malicious use of the server.

For Oracle Collaboration Suite Push Mail Client using In-Band notification, Oracle Collaboration Suite Push Mail Server must be set up to listen to HTTPS port (port 443) for Internet traffic. No other port needs to be opened for the Internet traffic.

The Notification Delivery Service of Push Mail Server is used for device management purposes only, and device management messages are sent to the device using the normal SMS channel. The messages are encrypted using the key sent to the client during the bootstrap process.

Push Mail Server communicates with the Oracle Collaboration Suite Mail server using IMAP, with the SMTP server using SMTP, and with Oracle Database using the SQL*NET protocol. These communication ports must be opened for Push Mail Server but must not be opened for general Internet traffic.

Normal Use with Out-Band Notification

With Oracle Collaboration Suite Push Mail clients using the Out-Band notification mechanism, the client establishes a UDP/IP channel with Oracle Collaboration Suite Push Mail Server in addition to the HTTPS channel. The server sends notifications regarding the events relevant for the client through this UDP channel. This section describes how this channel is established and used securely. The Mobile device security is the same as in normal use with In-Band notification. This section contains the following topics:

- Communication Security
- Push Mail Server Security
- Upgrading the Push Mail Client

Communication Security

The Command Processor of the client that uses Out-Band Notification first establishes an HTTPS session with the Command Processing service of the Oracle Collaboration Suite Push e-mail server. Once authentication is done and a secure HTTPS channel is established, the Command Processor sends a P-IMAP command to select either SMS or UDP/IP as the desired mechanism for receiving notification. Push Mail Server then sends relevant information (such as the host IP address and port number) of the Notification Delivery Service along with an encryption key and a single use key (called a limited entry visa). The visa is a secure random number generated by the Oracle Collaboration Suite Push Mail Server over a very large range of numbers and almost never reused. The encryption key depends on the client. For Microsoft Windows 2003 mobile devices, the encryption algorithm used is Triple-DES and the combined key length is 128 bits. For BlackBerry devices, the encryption algorithm is AES and the key length is 128 bits. The client then uses the visa to communicate with the Notification Delivery Service of Oracle Collaboration Suite Push Mail Server to set up the UDP/IP channel. The message payload is encrypted with the encryption key provisioned by the server.

Push Mail Server can be configured to expire the encryption key used for Out-Band notification every so often. Whenever the encryption key has expired, the Notification Deliver Service issues a command to the Notification Processor that informs the client that the server wants to change the encryption key. The Notification Processor then requests the Command Processor to go through the process of obtaining a new key. It

then resumes the process of communicating with the Notification Delivery Service using the new encryption key.

Push Mail Server Security

Server security for Oracle Collaboration Suite Push Mail Server that support Out-Band notification is identical to the servers that support In-Band notification except that the server that supports Out-Band Notification has the Notification Delivery Service that listens to UDP/IP messages on a certain port. This requires the firewall in front of the Oracle Collaboration Suite Push Mail Server to open the port and filter the messages to screen out malicious use.

The limited-entry visa used to establish the UDP/IP session with Push Mail Server can prevent Denial of Service attacks as those UDP/IP messages that do not carry a valid visa will be rejected right away with very little resource utilization.

Upgrading the Push Mail Client

Whenever a new version of Push Mail Client is available, the administrator can use Enterprise Manager to upload the client to Push Mail Server. When the upload is complete, Push Mail Server sends out an SMS message to each user affected by the new client. The SMS message contains the URL to download the new client.

Before downloading the new client, mobile users must ensure that the SMS came from the right phone number or short code and that the URL uses the HTTP scheme and refers to the enterprise Push Mail Server.

Loss of Device

Oracle Collaboration Suite Push Mail Server can remotely wipe a mobile device clean of data and code. This feature is designed to clean up a lost device.

When a user loses his device, the user can do one of two things to wipe the device clean of enterprise e-mail content and the Oracle Collaboration Suite Push Mail Client program. The user can visit the Oracle Collaboration Server Portal page, go to the Mobile Preferences page and to the Advanced page to delete the device from the My Communication Devices section. Malicious use of this feature is prevented by user authentication required to access the Portal page.

The user can also inform the administrator who can then use Enterprise Manager to wipe the device. Malicious use is prevented by requiring the administrator with the right privilege to take the action.

Preventing Malicious Actions Against the Client and the Server

Although malicious use of mobile devices has not been a concern up to now, this may change soon as mobile device usage increases. The mobile device user is the first line of defense against intruders attempting to enter enterprise information systems.

The Web browser on the device is the most vulnerable point in the defense of the device. As mobile device browsers become more and more powerful, they become susceptible to the same kind of attacks that desktop browsers suffer from. Enterprises should have a mobile device usage policy that includes permissible Web sites for browsing from mobile devices.

Other points of attack are the various listeners listening on various ports on mobile devices. For example, the User Agent Server component of SIP agent may be listening on a port for a SIP command and hackers may try to enter the mobile device using this port.

The Push Mail Client may open up to two ports on the mobile device. The first is an HTTPS port, which is always required. The HTTPS channel is not a general listener. It is a client-initiated point-to point channel to Push Mail Server that is securely established and used. Therefore, this port is secure.

The second is the UDP/IP port used only by Push Mail Client that uses the Out-Band notification mechanism. The UDP/IP channel is always initiated by the Notification Processor of the P-IMAP Transport running on the client. Thus, the client can be protected by a proxy server such as the Access Point of a cellular wireless network.

The client uses two mechanisms to ensure security of the channel: the visa and the encryption key. The client obtains the visa and the encryption key along with host address and port number of the Notification Delivery Service from Push Mail Server using the secure HTTPS channel. Thus, the identity of the client and the server are already verified by the HTTPS protocol.

The client will reject all the messages it receives on the UDP port that it cannot decrypt using the encryption key. As a result, the client can reject malicious UDP communication without using up too much of its resources.

The Notification Delivery Service of Push Mail Server will also reject all attempts to establish a UDP/IP session with it if the first UDP message does not contain a valid visa or the message payload cannot be decrypted. After that, the client and server negotiate a UDP time-out value. Once the time-out is agreed upon, the server will invalidate the visa and any attempts to establish a UDP channel with the server using the visa will be rejected. This prevents a replay attack on the server.

The Notification Delivery Service will change the encryption key from time to time. When the Notification Delivery Service wants to change the encryption key, it sends a message to the client. The client then goes through the entire process again to obtain the new key and establish a new channel to the server.

Push Mail Server is also designed to mitigate the effects of Denial of Service attacks. For example, several P-IMAP commands, such as `LOGIN` and `CAPABILITY` can be issued to the server without the user being authenticated. If the client does not carry a public key certificate, the server could be swamped with such requests sent from an attacker. To mitigate the effects from such attacks, Push Mail Server is designed to use as few resources as possible to respond to these requests.

Deployment Options for Push Mail Server

In the simplest case, you can deploy Oracle Collaboration Suite on a single server that has a Network Address Translator (NAT) firewall with a public IP address in front of it. You can forward port 443 to Oracle HTTP Server port 443 for HTTPS and another, say 9300, for UDP if you want to support Oracle Collaboration Suite Push Mail Clients using Out-Band notification. You can then register your domain name with the IP address of the NAT firewall. For example, your domain name could be `MyCompany.com`. You can configure Oracle HTTP Server to route the incoming traffic on port 443 with the URL, `MyCompany.com/pushMail` to the port the Push Mail Server is listening on. You can also route the URL, `MyCompany.com/nds` on port 9300 to the port the Notification Delivery Service is listening on.

Although this scheme will work, in reality, you may want to protect your server with more advanced firewalls such as those that perform packet filtering or stateful inspection to defend against Denial of Service attacks. Depending on your IT budget and your security needs, you may also consider setting up a demilitarized zones. Refer to *Oracle Collaboration Suite Deployment Guide* for more information on this topic.

Conclusions

Mobile devices are inherent weak links in enterprise information security. The Mobile Push Mail feature of Oracle Collaboration Suite is designed to minimize the security risks associated with using mobile devices to access enterprise e-mail.

With a combination of guidelines of mobile device usage and the built-in features of Oracle Mobile Collaboration Server, enterprises can safely deploy Mobile Push Mail features.

Oracle has had the security of Mobile Push Mail assessed by independent security consultants. All issues identified by them have been addressed in 10*g* Release 1 (10.1.1).

Securing Oracle Real-Time Collaboration

This section describes the features of Oracle Real-Time Collaboration that keep your data secure and prevent unwanted access to your system. It covers the following topics:

- Oracle Real-Time Collaboration Architecture and Security
- Secure Access for Oracle Real-Time Collaboration Clients
- Secure Connections for Oracle Real-Time Collaboration
- Oracle Real-Time Collaboration User Management and Authentication
- Oracle Real-Time Collaboration User Roles and User Privileges
- Secure Archives for Oracle Real-Time Collaboration
- Security Report for Oracle Real-Time Collaboration

Oracle Real-Time Collaboration Architecture and Security

The components of the Oracle Real-Time Collaboration Server are designed to provide appropriate levels of security for the entire enterprise. Figure 2–4 shows the basic components of the system.

Figure 2–4 Oracle Real-Time Collaboration High-Level Architecture

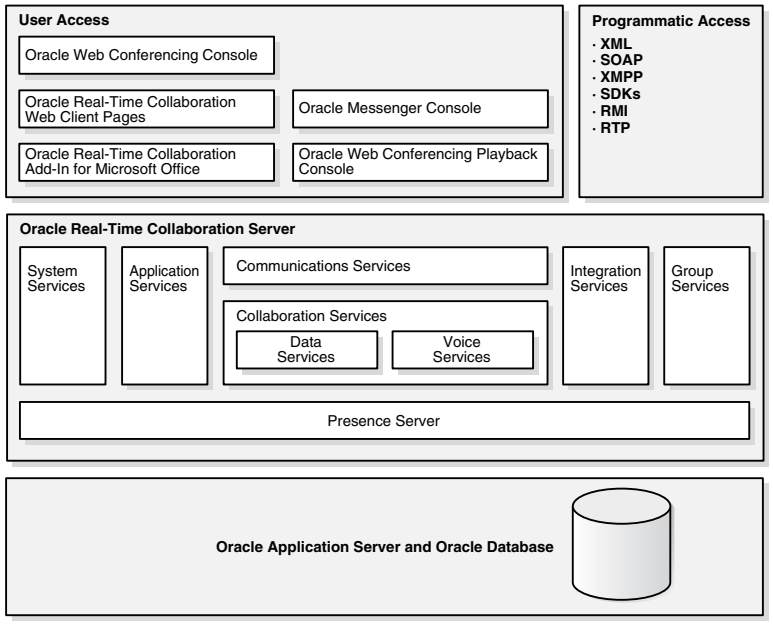


Table 2–3 provides a brief overview of security features for the components in the Oracle Real-Time Collaboration architecture.

Table 2–3 Security Features in Oracle Real-Time Collaboration Architecture

Component	Security Features
Oracle Real-Time Collaboration Consoles (Clients)	Use the Oracle Identity Management infrastructure to authenticate users and provide single sign-on. Clients can connect directly within your company firewall, over the Internet, and over the Internet from behind a firewall.
Oracle Real-Time Collaboration Application Pages	Use the Oracle Identity Management infrastructure to authenticate users and provide single sign-on, supporting centralized management of user accounts. The application pages can be deployed with any type of standard security and network devices, such as hardware load balancers and SSL accelerators.
Programmatic Access through Integration Services	Use standard SOAP and XML over HTTP/HTTPS protocols. All services utilize `AuthenticationService` Web Service for authentication. `AuthenticationService` works with Oracle Real-Time Collaboration `SiteID/AuthToken` and the S2S authentication framework of Oracle Internet Directory.
Communication Services	Client connections can be direct or through standard ports in a firewall (Port 80 and Port 443) with no additional configuration. The client connections can be either SSL or non-SSL.
Oracle Real-Time Collaboration Services (data, voice, and presence)	Protected behind the Communications Services.
System Services and Application Services	Oracle Real-Time Collaboration application administrative pages and the rtcctl utility allow administrators to monitor the system and status of instances and components. However, administrators do not have privileges to join a conference to which they have not been invited, or participate in a message that was not sent to them.

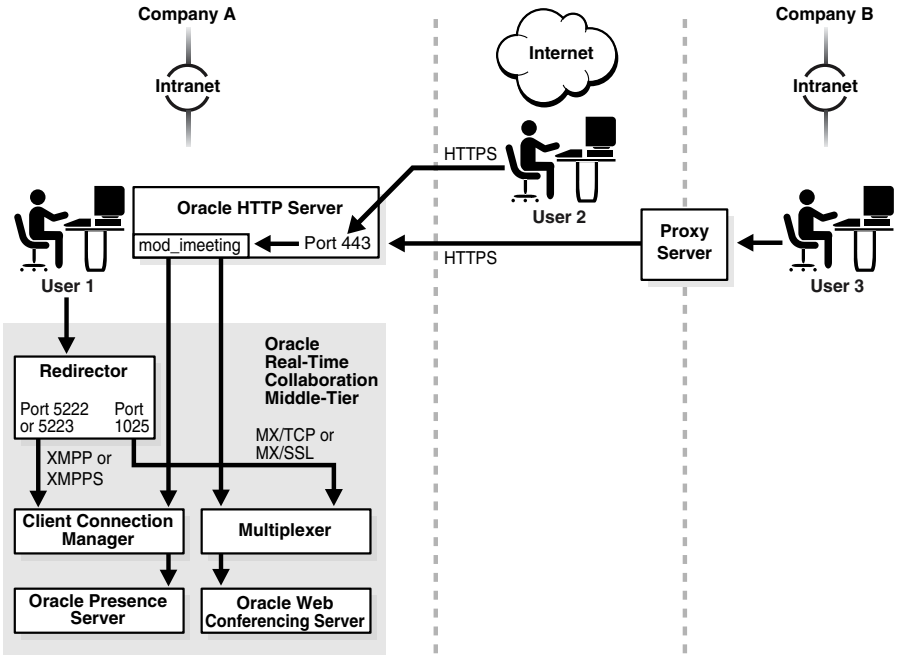
Table 2–3 (Cont.) Security Features in Oracle Real-Time Collaboration Architecture

Component	Security Features
Oracle Real-Time Collaboration Repository	Stores user roles, user documents, conference details, and conference and message archives is never directly accessible to users. Administrators access it through a secure database account. The full set of Oracle database features protect the data.

Secure Access for Oracle Real-Time Collaboration Clients

The Oracle Real-Time Collaboration system provides secure access for clients from within the intranet, from the open Internet or across transparent proxies, and from behind a firewall. Figure 2–5 shows these connections.

Figure 2–5 Client Connections to Oracle Real-Time Collaboration



The Oracle Web Conferencing and Oracle Messenger console clients attempt to connect to the Oracle Real-Time Collaboration system using one of the following methods, attempting them in the following order until successful:

1. **Direct connection:** Clients within a corporate intranet connect directly to the RTC Redirector, which hands off connections to the Client Connection Manager (for instant messages) or Multiplexer (for Web conferences), using Jabber XMPP/XMPPS protocols for messaging, or proprietary protocols (MX) with TCP/IP or SSL for Web conferences.
2. **HTTPS direct:** Clients in the open Internet or across transparent proxies connect through HTTPS. The `mod_imeeting` plug-in uses the Oracle HTTP Server as the single listening point over port 443, then hands the socket off to the Connection Manager or Multiplexer.
3. **HTTPS tunnel:** Clients in a different intranet coming through their own internal proxy provide the console with proxy information from the browser settings. The console establishes a connection to the Oracle HTTP Server, which hands the

connection off to the Connection Manager or Multiplexer over an HTTPS tunnel through the remote proxy. Again, the listening port is 443.

Note that for scenarios 2 and 3 to succeed, port 443 must be open to the Internet. In addition, the Oracle Real-Time Collaboration Applications tier must have an Internet-routable address, or you can use a Network Address Translator (NAT) to map the internal IP address to an external IP address.

Third-party XMPP clients for instant messaging, such as Gaim, can connect only using the direct TCP/IP method, assuming the client's user is provisioned in the Oracle Internet Directory. To allow third-party client access to instant messaging over the Internet, you must open the XMPPS port (by default, port 5223). Alternately, you may use a Network Address Translator (NAT) to map the internal IP address to an external IP address, and map port 5223 to the externally accessible port 443.

Secure Connections for Oracle Real-Time Collaboration

An administrator can set an entire Oracle Real-Time Collaboration deployment, or just an Oracle Real-Time Collaboration site, to use Secure Sockets-Layer encryption connections during Web conferences or when sending instant messages.

Administrators set these options using Oracle Real-Time Collaboration *properties*. Refer to "Configuring SSL Security" in Chapter 3 of *Oracle Real-Time Collaboration Administrator's Guide* for details about SSL-related properties.

Note: Using controls provided in Oracle Collaboration Suite, an administrator can mandate use of SSL for some or all URLs in the Oracle Real-Time Collaboration Web application pages. Therefore, even if SSL encryption settings are not enabled as outlined in this section, users could be directed to a secure URL for the Web pages.

The administrator can use Oracle Real-Time Collaboration properties to do any of the following:

- Enable SSL-encrypted conferences and messaging, so that conference hosts can choose the SSL option when scheduling a conference, or the servlet that supports Oracle Messenger connections can choose XMPPS or HTTPS to make the connection.
- Force SSL-encrypted conferences and SSL-encrypted messages on this system. Conference hosts and messaging users cannot choose non-SSL connections.

If SSL-encryption has been enabled, but not forced, then Web conference hosts can specify that a conference should use SSL encryption in either of the following ways:

- To set a specific conference to use encryption, users select **Oracle Web Conferencing Console** on the Schedule tab and set **Secure Communications** on.
- To set all conferences they host to use encryption by default, users select **Oracle Web Conferencing Console** from the Preference page, and set **Secure Communications** on.

If SSL-encryption has been enabled, but not forced, the servlet that chooses connections for Oracle Messenger will try to make HTTPS or XMPPS connections before attempting HTTP or XMPP connections. This happens automatically, provided Oracle Messenger users have set their **Connection** option to "Automatic Configuration for RTC Connection," in the **Options** command under **Tools**. (Automatic Configuration is the default option for all Oracle Messenger clients.)

Voice Chat Encryption in Oracle Messenger

Voice chat encryption is done using symmetric keys. The keys are generated for each voice session and the key length and algorithm are negotiated between the clients starting from the most secure AES 256 bit to RC40 bit. Clients could be running on different operating system versions. Therefore, the key length and algorithm varies across operating systems. The encryption is implemented using Microsoft Windows Crypto API. The following algorithms are supported by Oracle in the current implementation:

- AES 256 (Windows XP, Windows Server 2003)
- 3DES 168 (Windows 2000, Windows NT)
- RC2 40 (All Microsoft Windows platforms)

After the encryption mechanism is negotiated, the keys and the algorithm are exchanged over the secure SSL channel between the Oracle Real-Time Collaboration messengers through the Oracle Presence Server. Because the keys are exchanged over the primary connection to the Oracle Presence Server, if this is connection is not secure, that is, SSL is not used, then voice chat encryption is turned off. In essence, voice chat is encrypted only when the clients connect to the Oracle Presence Server over SSL.

Note: The lock icon on the upper-right corner of the Oracle Messenger screen indicates whether or not the voice chat session is encrypted.

Oracle Real-Time Collaboration User Management and Authentication

The Oracle Real-Time Collaboration system, like the rest of the components in the Oracle Collaboration Suite, uses the Oracle Internet Directory to manage user data and the Oracle Internet Directory store, which uses LDAP (Lightweight Directory Access Protocol), to authenticate users. The Oracle Internet Directory host is specified at installation. All users of this Oracle Internet Directory can be provisioned to use Oracle Real-Time Collaboration. This supports centralized management of user accounts and privileges.

Users outside of the Oracle Internet Directory can be allowed limited access to Oracle Real-Time Collaboration services, or can be explicitly prevented from using those services. In addition, users within the Oracle Internet Directory can be assigned additional privileges for monitoring or managing the Oracle Real-Time Collaboration system. Refer to "Oracle Real-Time Collaboration User Roles and User Privileges" for more details about guest user and administrative user roles.

Users log in to Oracle Real-Time Collaboration Web pages and clients using their Single Sign-On user name and password. Only users who log in to the Web Client may download the Oracle Messenger client or the Oracle RTC Add-In to Microsoft Office, schedule or host Web conferences, or view archived records of conferences and chat sessions. Only users who log in to the Oracle Messenger client can send or receive instant messages, hold chat conferences, or publish their presence to other clients.

An administrator should inform users that it is important to log out and close browser and client windows whenever they have completed their tasks with Oracle Real-Time Collaboration. This prevents unauthorized access to system features and to archived records, reports or other data that should remain confidential.

Authenticating Oracle Real-Time Collaboration Integration Services

The Oracle Real-Time Collaboration Integration Services are SOAP and XML services that provide access to Oracle Real-Time Collaboration features and functions. Integrated Services are supported by creation of a *site*, which is assigned a unique ID and associated authentication token when the site is created. When external clients access the services, those services pass the site ID and token to authenticate the client during the session for the provided service.

An administrator can choose to enable or disable access to some or all integration services using Oracle Real-Time Collaboration properties. Refer to "Enabling Integration Services" in Chapter 3 of *Oracle Real-Time Collaboration Administrator's Guide* for details.

For more information about Oracle Real-Time Collaboration integration services, refer to *Oracle Real-Time Collaboration Application Developer's Guide*.

Accounts for Automated Tests of Oracle Messenger

An end-to-end test of Oracle Messenger checks whether a user can successfully pass a message to another user through the Oracle Presence Server. The test is run automatically on the Oracle Real-Time Collaboration system at periods defined by the administrator. Administrators can also run this test manually by using the `imtest` option with the `rtcctl runTests` command.

Two test user accounts are automatically provisioned for use by this test. The passwords for each account are different for each instance of the Oracle Real-Time Collaboration components and these passwords are generated by the system based on a base64 encoding of part of the instance key. The account names are test.user@im-agents.*your_domain*.com and im.test.user@*your_domain*.com.

Oracle Real-Time Collaboration User Roles and User Privileges

There are two basic categories of users who may access the Oracle Real-Time Collaboration system:

- **nonregistered user**: Also called a **guest user**. A user who is not provisioned in the Oracle Internet Directory. Users who are not provisioned through the Oracle Internet Directory can be invited to public conferences, or allowed to participate in instant messages in a "live help"-type session supported through Integration Services. These users are allowed only limited ability to use Oracle Real-Time Collaboration features.
- **registered user**: A user provisioned in Oracle Internet Directory. This user is assigned an end-user role by default. Registered users can upload documents, host conferences, record and publish conferences, and can participate in conferences that are restricted to registered users only. Registered users can also download the Oracle Messenger client and use all of its features to send and receive messages, publish their presence, hold chat conferences, and so forth.

Administrators can choose to prevent nonregistered user access. By setting the Oracle Real-Time Collaboration property `GuestUserAccessEnabled` to false, administrators limit access to Oracle Real-Time Collaboration features:

- The Archive tab, the Conferences in Progress, Scheduled Conferences, and New User sub-tabs, and the Join Conference area are removed from the prelogin Home page.
- Instant conferences default to the "Registered Users" audience setting. (The normal default is for Instant conferences to be available to "All Users.")

- Registered users scheduling conferences cannot choose the "All Users" audience setting to allow guest users.
- When conference hosts publish URLs to let users join a conference, play back a conference, download the conference recording, or view conference summary and details, users who access those URLs must log in before they can do any of these tasks.
- Oracle Real-Time Collaboration Integration Services prevent other scheduling applications from scheduling a conference that allows guest users. Such applications include both Oracle Collaboration Suite components such as Oracle Calendar, and any integrated applications that call Oracle Real-Time Collaboration APIs.
- Oracle Real-Time Collaboration Integration Services will not support nonregistered users in chats. Therefore, integrations to support a "live help" scenario, allowing guest users to chat with users of Oracle Messenger, will not work.

Note: If `GuestUserAccessEnabled` is set to false, and you are using Oracle Calendar to schedule Web conferences, then you must set the `allowguestusers` parameter in Oracle Calendar to false as well. If the `allowguestusers` parameter is true, then when you try to create Web conferences that include nonregistered users through Oracle Calendar, the conference will not be created.

Creating Administrative Users

In addition to the basic role allowed for registered users, Oracle Real-Time Collaboration administrators can assign additional roles to registered users that allow different levels of privileged access to Oracle Real-Time Collaboration management features.

- **business monitor**: A registered user who has been assigned the ability to monitor the system and access system reports.
- **business administrator**: A registered user who has been assigned the ability to start, stop, configure, and administer the system deployment. This user has the greatest number of privileges on the system.
- **site business monitor** or **site business administrator**: A registered user who has been assigned the business monitor or business administrator privileges described previously, but only for a specific site.

Controlling User Privileges with Properties

Users with the business administrator role can set a number of Oracle Real-Time Collaboration properties that control some of the options available to users as they participate in Web conferences or instant messages. Other properties control what features are available in the Oracle Real-Time Collaboration Web application pages. Chapter 3 of *Oracle Real-Time Collaboration Administrator's Guide* has a complete list of available properties.

Users also have the option to set defaults for Web conferences using the Preferences command from the Web application, or defaults for instant messaging using the Options command under the Tools menu. Any settings by the administrator that affect these options can supersede these preferences. For example, if the administrator has disabled the desktop sharing interface inside all Web conferences, then that feature will not be available, even if the user has chosen it as a default.

Administrators can set properties at a system, site, instance, or Oracle Real-Time Collaboration component level. Typically, user privileges are set at the system or the site level. If a property is set on a system, then the site will inherit that value. But a separate value can be also be set on the site. So, for example, an administrator could allow access to desktop sharing on a site, even if it is disabled on the system.

An administrator can use the `-force` option of the rtcctl utility to force property settings on an entire system. Refer to "Using Oracle Real-Time Collaboration Properties" in Chapter 3 of *Oracle Real-Time Collaboration Administrator's Guide* for more details about forcing property settings.

Using Conference Keys to Protect Conference Access

Hosts can create a conference key to protect unauthorized entry to both Web conferences and chat conferences. The host specifies a key when creating a chat or instant Web conference, or when scheduling a future Web conference. The host can then send the key string to invitees using whatever method desired. For scheduled conferences, an encoded string for the key is included in the link to join the conference.

Only attendees to whom the host gives the key may enter the conference. This prevents others from entering, even if the conference is published in the public listing.

If a conference included a key, then users must enter that key in order to view the conference archives. This protects the archive from unauthorized access as well.

Privileges Within Web and Chat Conferences

Within Web conferences or Oracle Messenger chat conferences, the person who hosts the Web conference or starts the chat conference has more privileges than other users. For example, Web conference hosts can prevent any other attendees from interacting with the desktop. Chat conference mediators can expel members from a chat session. This ensures that the person who started the conference can control the information being exchanged. Only the conference host or mediator can choose to allow other users to expand their roles in the conference.

For complete details about the various roles within Web or chat conferences, refer to the Oracle Real-Time Collaboration online Help.

Restricting Access to Web Conferences by User Role

Only registered users can schedule Web conferences. During scheduling, a host can choose what type of user can access the conference and whether the conference is visible to others from the Oracle Real-Time Collaboration Web pages. Table 2–4 shows what settings can be applied to control access to and visibility of scheduled conferences. All of these settings can be applied when using the Schedule tab.

Table 2–4 assumes that nonregistered users (guest users) are allowed access to Oracle Real-Time Collaboration. If guest users are not allowed, then the "All Users" option is not available, and conferences listed on a public Web page can be seen only by registered users.

Table 2–4 Audience and Visibility Settings to Control Conference Access

Audience Setting	Visibility Setting	Accessibility
All Users	List conference on public Web page	Both nonregistered and registered users may attend. Anyone can see the conference in the Scheduled Conferences table, without logging in.

Table 2–4 (Cont.) Audience and Visibility Settings to Control Conference Access

Audience Setting	Visibility Setting	Accessibility
All Users	Off (not listed)	Both nonregistered and registered users may attend, if they are given the conference ID and conference key. Only registered users who are invited can see the conference in their "My Conferences" list from the home page.
Registered Users	List conference on public Web page	Only registered users may attend. Only registered users can see the conference in the Scheduled Conferences table post login.
Registered Users	Off (not listed)	Only registered users may attend if they know the conference ID and conference key. Only registered users who are invited can see the conference in a listing.
Registered Users by Invitation Only	Off (not listed). This is the only possible setting for this audience.	Only registered users who have received an invitation from the host may attend, and only those users can see the conference in a listing.

Note that if a registered user is invited to a conference, that user will see a list of conferences to which he or she is invited when logging in. If a user is not invited to a conference, then he or she cannot see information about an upcoming conference unless the host has chosen to list it publicly. This way, a host can keep a conference confidential by choosing not to list it.

Privileges for an Acting Conference Host

If the person who schedules a conference is not going to lead the conference, then that user can create an acting host key that another user can enter when joining the conference. The user with the acting host key becomes the conference's effective host.

Only a registered user can use the acting host key. Nonregistered users cannot use it. In addition, the acting host key does not override any conference permissions. If a conference is restricted to registered users by invitation, then only such a user can join with the acting host key.

There can be only one host for a conference, so only the *first* person to enter the conference with the acting host key is the host. After the conference is started by an acting host, the original host and any other user who joins with the acting host key join as presenters. If a conference is started by an acting host, then the conference ends when the acting host leaves, unless the acting host assigns another host before leaving.

Finally, by default, the conference archive is visible only to the original host who scheduled the conference, not the acting host. The scheduling host can choose to publish the archive as discussed in Web Conference Archives.

Secure Archives for Oracle Real-Time Collaboration

The details of every Oracle Web Conferencing session and each one-to-one instant message conducted on the system can be archived and stored in the Oracle Real-Time Collaboration repository.

Web Conference Archives

All Web conferences are archived and stored in the Oracle Real-Time Collaboration repository. A conference archive consists of the following information:

- Conference title, conference ID, and conference keyword, if any
- Host name
- Conference start date, time, and duration
- Voice start time and duration (if voice streaming was used)
- Number of attendees, including anonymous attendees
- Conference recording (if any)
- Details about each attendee: name, e-mail address, company, postal address, telephone number, times each attendee entered and exited the conference and voice streaming, and any additional information gathered through customized properties
- Documents uploaded for the conference
- URLs visited during the conference
- Transcripts of chat sessions held within the conference
- Results of any polls held during the conference

Conference archives can be viewed under the Archive tab. By default, the security restrictions of the actual conference (as shown in Table 2–4, " Audience and Visibility Settings to Control Conference Access") are applied to the archive. For example, if only registered, invited users could attend the conference, then only those users can view the archive.

The default information published contains only the conference basics (the title, start date and time, and number of attendees). The host can control what other information is published, and whether additional users can view it. The playback of the recording can also be secured through SSL encryption.

Oracle Messenger Archives

By default, archives are *not* enabled for Oracle Messenger. However, archives can be saved both on the server side, and on each client's side. Message archives contain the text of each message exchanged with another user, stored by user and sorted by message date. Message archives can be enabled in either or both of the following ways:

- Administrators can set an Oracle Real-Time Collaboration property, IMArchiveEnabled, to save message archives to the Oracle Real-Time Collaboration repository. Refer to "Controlling Archives" in Chapter 3 of *Oracle Real-Time Collaboration Administrator's Guide* for details about setting this property.
- Users can choose to save their messages locally, by selecting **Tools** and then **Options** from the Oracle Messenger client window, then under **Instant Messages** selecting **Save Messages**. Users can also set the size of the archive.

  If users choose to save their messages to the client computer, they can view archives by right-clicking a contact name and choosing **View Message Archive**.

If archives are enabled for Oracle Messenger, then the information stored includes the following:

- Name of the contact
- Date the message was exchanged
- Entire text of the message

Message archives are available to only those users who participated in the message. That is, if an administrator chooses to save archives to the repository, each Oracle

Real-Time Collaboration user will be able to view only the messages that he or she participated in, using the **Archive** tab in the Web application.

> **Note:** Text chat conferences (a chat window with three or more users) are not archived, and participants cannot view archives of those chat sessions.

Creating a Privacy or Acceptable Use Policy

Administrators may want to create a privacy statement or statement of acceptable use to present to employees, informing them that details of instant messages and Web conferences are recorded and stored in archives. To do so, administrators set the following Oracle Real-Time Collaboration properties:

- `PrivacyLink` sets the URL to the Web page with your privacy statement.
- `ShowPrivacyLink` sets whether the link to the Web page appears on Oracle Real-Time Collaboration Web pages.

Refer to "Configuring the Oracle Real-Time Collaboration Web Client Pages" in Chapter 3 of *Oracle Real-Time Collaboration Administrator's Guide* for more details about how to set these properties.

Security Report for Oracle Real-Time Collaboration

The Security report is available to business monitor and business administrator users under the **Reports** tab of the Oracle Real-Time Collaboration Web application pages. Administrators can review this report to identify how many security features are being used to protect conferences on the system.

Refer to "Oracle Real-Time Collaboration Security Report" in Chapter 6 of *Oracle Real-Time Collaboration Administrator's Guide* for more details about this report.

Securing Oracle Voicemail & Fax

Oracle Voicemail & Fax provides basic security features including authentication and securing connections to Oracle Collaboration Suite Database. This section covers the following topics:

- Authenticating Using Oracle Internet Directory
- Securing Oracle Voicemail & Fax Connections
- Changing Passwords

Authenticating Using Oracle Internet Directory

Oracle Voicemail & Fax uses Oracle Internet Directory for authentication. You can set preferred credentials for yourself, for selected users, and for all users who connect to Oracle Internet Directory from Enterprise Manager.

> **See Also:** *Oracle Voicemail & Fax Administrator's Guide* for information about setting preferred credentials

Securing Oracle Voicemail & Fax Connections

You can secure your Oracle Voicemail & Fax connections by encrypting connections to the Oracle Collaboration Suite Database and by using SSL connections. From the

moment the Oracle Voicemail & Fax user starts interacting with Oracle Voicemail & Fax, any communication Oracle Voicemail & Fax performs with other Oracle Collaboration Suite components can be secured. There are two major connection points, Oracle Collaboration Suite Database connectivity and Oracle Internet Directory connectivity.

Encrypting Connections to the Oracle Collaboration Suite Database

You can encrypt all communications with the Oracle Collaboration Suite Database by configuring the Oracle Net connection between Oracle Voicemail & Fax and the Oracle Collaboration Suite Database. This ensures that any communication between Oracle Voicemail & Fax and the Oracle Collaboration Suite Database is secure. However, this requires configuration of the Oracle Collaboration Suite Database to allow encrypted or secure connections from Oracle Database clients.

See Also: *Oracle Advanced Security Administrator's Guide* for more information about encrypting your Oracle Net connections

SSL Connections

SSL connections are used in various places, including when Enterprise Manager connects to Oracle Internet Directory, when the Oracle Voicemail & Fax Applications connect to Oracle Internet Directory, and between the Oracle Collaboration Suite 10*g* WebMail client and Oracle Internet Directory when authenticating users and accessing a user's address book.

See Also:

- "Configuring SSL Between Oracle Collaboration Suite 10g WebMail and Oracle Internet Directory" on page 2-31 for more information
- "Enabling SSL in Oracle Voicemail & Fax" on page 7-5 for more information on how to use SSL when connecting to Oracle Internet Directory

Changing Passwords

When you install Oracle Voicemail & Fax, you are prompted to provide passwords for the `um` and `ovfmetrics` user names, which are used, respectively, to connect the Voicemail & Fax Application and the Message Delivery Service to the Oracle Collaboration Suite Database. If, for some reason, you need to change a password for these users in the database, then you must also update the client applications that connect to the database with the new password.

See Also: *Oracle Voicemail & Fax Administrator's Guide* for information on how to change the passwords in Oracle Voicemail & Fax

3

Oracle Collaboration Suite Infrastructure Security

This chapter discusses Oracle Collaboration Suite Infrastructure security. It contains the following sections:

- Security in Oracle Collaboration Suite Infrastructure
- Oracle Identity Management
- SSL Configuration in Oracle Internet Directory
- Privilege Delegation

Security in Oracle Collaboration Suite Infrastructure

This section describes the security features in Oracle Collaboration Suite Infrastructure. It contains the following sections:

- Oracle HTTP Server Security
- Directory Security Concepts
- Physical Hardware Security
- Network Security
- Operating System Security
- Database Security
- Application Server Security
- Third-Party Software Security
- User Security
- Password Security

Oracle HTTP Server Security

Oracle HTTP Server controls access to resources based on user identity. Identity is established through several authentication mechanisms. These include standard Apache authentication mechanisms, such as basic authentication or SSL with a client certificate and OracleAS Single Sign-On by using `mod_osso`. All Oracle Collaboration Suite applications can obtain an OracleAS Single Sign-On user identity from Oracle HTTP Server by using the Apache header created by `mod_osso`.

In Oracle Collaboration Suite, `mod_osso` authenticates users by specifying whether they should have access to server resources (URLs, directories) or not. Applications

accessible through Oracle HTTP Server enable you to access resources by using the user identity authenticated by OracleAS Single Sign-On.

Oracle HTTP Server can be configured to protect data exchanged between the server and Web clients by 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 the use of an Oracle Wallet.

Directory Security Concepts

You can deploy multiple Oracle components to work with a shared instance of Oracle Internet Directory and the associated infrastructure. This sharing lets an enterprise simplify security management across all applications. In addition to the role it plays in the Oracle Identity Management Infrastructure, Oracle Internet Directory provides the following features to protect information:

- Data Integrity
- Data Privacy
- Authorization
- Authentication
- Protection of User Passwords for Directory Authentication
- Password Policies

Data Integrity

Oracle Internet Directory uses SSL to ensure that data has not been modified, deleted, or retransmitted during transmission. SSL generates a cryptographically secure **message digest**, through cryptographic checksums, using either the **MD5** algorithm or **Secure Hash Algorithm** (**SHA**), and includes the digest with each packet sent across the network.

Data Privacy

Oracle Internet Directory ensures that data is not disclosed during transmission by using **public key encryption** available with SSL. In public key encryption, the sender encrypts the message by using the public key of the recipient. When the message is delivered, the recipient decrypts the message by using the recipient's private key.

Authorization

Authorization is the permission given to a user, program, or process to access an object or a set of objects. When you perform directory operations within a directory session, the directory server ensures that you have the permissions to do so. The directory server prevents unauthorized operations on data by using access control information.

Access control information is the directory metadata that constitutes the administrative policies relating to access control. This information is stored in Oracle Internet Directory as user-modifiable operational attributes, each of which is called an Access Control Item (ACI).

A list of ACI attribute values is called Access Control List (ACL). The ACL is associated with directory objects. The attribute values in the ACL represent the permissions that various directory user entities have on a given object.

Authentication

Authentication enables a directory server to establish the true identity of the user connecting to the directory. Authentication is performed whenever a Lightweight Directory Access Protocol (LDAP) session is established. As a result, every session has an associated user identity.

Oracle Internet Directory provides Simple Authentication and Security Layer (SASL) authentication over an SSL connection. During SASL authentication, both the client and server authenticate themselves to each other by providing certificates.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for more information about SASL

Protection of User Passwords for Directory Authentication

Oracle Internet Directory protects directory passwords by storing them as one-way hashed values. Storing passwords as one-way hashed values, rather than as encrypted values, secures the passwords because a malicious user can neither read nor decrypt them.

Password Policies

A password policy is the rule that governs how passwords are used. When you attempt to access a directory, the directory server ensures that the password you provide conforms to the password policy.

When you establish a password policy, you can set the following types of rules:

- Minimum password length
- Expiration of passwords
- Alphanumeric requirements, such as ensuring at least one numeric character in the password
- Whether a new password can be the same as an old password

Physical Hardware Security

You cannot secure any system where the physical access to the systems is also not secured. Systems storing or serving sensitive information must be physically secured because it is easy to damage or steal the hardware.

Network Security

At the lowest levels of the network, techniques to enforce security include the use of firewalls to create a demilitarized zone (DMZ). This helps ensure that the network traffic follows specified rules. For example, traffic from the Internet may not pass directly to the database but must go through both the DMZ and the application server tier. Good network security practices employ firewalls from different vendors to form the DMZ. You need to ensure that firewall rules are correctly configured and that the firewall and router passwords are secure. Switched networks, rather than bus topology-based networks, provide an intruder with less opportunity for packet sniffing.

Operating System Security

In addition to securing the firewalls and the network, you need to secure the operating system that runs Oracle Collaboration Suite. Vendors, Oracle Consulting, and

third-party security consultants often have well-defined procedures and practices to secure the operating system. These include:

- Closing unused ports, such as telnet, ftp, and finger, using `xinetd or inetd` with TCP wrappers
- Removing unused or default user accounts
- Applying the latest patches
- Removing unnecessary software
- Checking for `setuid/setgid` software
- Installing and configuring intrusion detection software
- Removing host banners

Database Security

You need at least two databases to run Oracle Collaboration Suite and at times, three databases, one each for:

- Oracle Application Server Infrastructure and Oracle Internet Directory
- Oracle Mail
- Oracle Content Services

All these databases must be secured against attack.

Security Checklist for Oracle Database 10*g*

For databases, establishing a secure configuration is the first line of defense against attacks.

Implementing the following recommendations provides the basis for a secure database configuration:

- Install only what is required.

  Do a custom installation. Choose to install only additional products and options that you require, in addition to the database server.

- **Lock and expire default user accounts.**

  Oracle Collaboration Suite Database installs with many default database server user accounts. After you create a database server instance, the Database Configuration Assistant automatically locks and expires most default database user accounts. After the database is installed, lock the `SYS` and `SYSTEM` user accounts and use the `AS SYSDBA` account for administrator access. Specify the administrative passwords individually.

  Note: If you use Oracle Universal Installer or Database Configuration Assistant, then you will be prompted for new `SYS` and `SYSTEM` passwords. The defaults values, `change_on_install` or `manager`, will not be accepted.

  The `AS SYSDBA` account tracks the operating system user name, maintaining accountability. If you only need access for database startup and shutdown, then use `AS SYSOPER`, which has fewer administrative privileges than `SYS`, but sufficient to perform basic operations such as startup, shutdown, mount, backup, archive, and recover.

Database Configuration Assistant is not used during a manual installation. So, all the default database users remain unlocked and gain unauthorized access to data. In addition, the default database users can disrupt database operations. After a manual installation, use SQL to lock and expire all default database user accounts except `SYS`, `SYSTEM`, `SCOTT`, and `DBSNMP`. If a locked account is needed later, then a database administrator can unlock and activate that account with a new password.

Table 3–1 lists the database users that are available after a typical Oracle Database installation using Database Configuration Assistant.

Table 3–1 Default Accounts and Their Status (Standard Installation)

USER NAME	ACCOUNT_STATUS
ANONYMOUS	EXPIRED & LOCKED
CTXSYS	EXPIRED & LOCKED
DBSNMP	EXPIRED & LOCKED
DIP	EXPIRED & LOCKED
DMSYS	EXPIRED & LOCKED
EXFSYS	EXPIRED & LOCKED
HR	EXPIRED & LOCKED
MDDATA	EXPIRED & LOCKED
MDSYS	EXPIRED & LOCKED
MGMT_VIEW	EXPIRED & LOCKED
ODM	EXPIRED & LOCKED
ODM_MTR	EXPIRED & LOCKED
OE	EXPIRED & LOCKED
OLAPSYS	EXPIRED & LOCKED
ORDPLUGINS	EXPIRED & LOCKED
ORDSYS	EXPIRED & LOCKED
OUTLN	EXPIRED & LOCKED
PM	EXPIRED & LOCKED
QS	EXPIRED & LOCKED
QS_ADM	EXPIRED & LOCKED
QS_CB	EXPIRED & LOCKED
QS_CBADM	EXPIRED & LOCKED
QS_CS	EXPIRED & LOCKED
QS_ES	EXPIRED & LOCKED
QS_OS	EXPIRED & LOCKED
QS_WS	EXPIRED & LOCKED
RMAN	EXPIRED & LOCKED
SCOTT	EXPIRED & LOCKED
SH	EXPIRED & LOCKED
SI_INFORMTN_SCHEMA	EXPIRED & LOCKED

Table 3–1 (Cont.) Default Accounts and Their Status (Standard Installation)

USER NAME	ACCOUNT_STATUS
SYS	OPEN
SYSMAN	EXPIRED & LOCKED
SYSTEM	OPEN
WK_TEST	EXPIRED & LOCKED
WKPROXY	EXPIRED & LOCKED
WKPROXY	EXPIRED & LOCKED
WMSYS	EXPIRED & LOCKED
XDB	EXPIRED & LOCKED

- Change default user passwords.

  Security is breached when the default database server user account has the default password even after installation. To fix this:

  - Change the default passwords of administrative users immediately after installing the database server.

    In any Oracle environment (production or test), assign strong, meaningful passwords to the `SYS` and `SYSTEM` user accounts immediately after successful installation of the database server. The passwords for `SYS` and `SYSTEM` should never remain in their default states. Similarly, for production environments, do not use default passwords for any administrative accounts, including `SYSMAN` and `DBSNMP`.

  - Change the default passwords of all users immediately after installation.

    Lock all accounts and expire all default passwords after installation. If any such account is later activated, then change its default password to a new meaningful password.

  - Enforce password management.

    Apply basic password management rules, such as password length, history, and complexity, to all user passwords.

    Request all users to change their passwords regularly, such as every eight weeks.

    If possible, use Oracle Advanced Security with network authentication services (such as Kerberos), token cards, smart cards, or X.509 certificates. These services provide strong user authentication and enable better protection against unauthorized access.

- Enable data dictionary protection.

  Implement data dictionary protection to prevent users having the `ANY` system privilege from using it on the data dictionary. Oracle Collaboration Suite Database sets the `O7_DICTIONARY_ACCESSIBILITY` to `FALSE`. This setting prevents using the `ANY` system privilege on the data dictionary, except for authorized users making DBA-privileged connections (for example, `CONNECT/AS SYSDBA`).

- Practice the principle of least privilege.

  The following practices implement this principle:

  - Grant necessary privileges only.

Do not provide database users more privileges than necessary. Enable only those privileges that are required to efficiently perform necessary jobs:

* Restrict the number of system and object privileges granted to database users.
* Restrict the number of SYS-privileged connections to the database. For example, there is no need to grant CREATE ANY TABLE to any non-DBA-privileged user.

– Revoke unnecessary privileges and roles from the PUBLIC database server user group.

The default role, granted to every user in an Oracle database, enables unrestricted use of its privileges, such as EXECUTE on various PL/SQL packages. If unnecessary privileges and roles are not revoked from the PUBLIC user group, then a minimally privileged user can access and run packages otherwise inaccessible to the user.

– Grant users roles only if they need all of the role's privileges.

Roles (groups of privileges) are useful for granting permissions to users quickly and easily. If your application users do not need all the privileges encompassed by an existing role, then create your own roles containing only the necessary privileges. Similarly, ensure that roles contain only the privileges that reflect job responsibility.

For example, grant the CREATE SESSION privilege to users to authorize them to log in to the database, rather than granting them the CONNECT role, which has many additional privileges. Unless users require all the extra privileges contained in the CONNECT role or any other role, individually assign to each user only the minimum set of individual privileges needed. Alternatively, create your own roles and assign only the required privileges.

For example, it is imperative to strictly limit the privileges of SCOTT. Drop the CREATE DBLINK privilege for SCOTT. You need to drop the entire role for SCOTT because privileges acquired by means of a role cannot be dropped individually. Recreate your own role with only the privileges needed, and grant that new role to the user. Similarly, to enhance security, drop the CREATE DBLINK privilege from all users who do not require it.

– Restrict permissions on run-time facilities.

Do not assign all permissions to any database server run-time facility, such as the Oracle Java Virtual Machine (OJVM).

Instead, grant specific permissions to the explicit document root file paths for such facilities that may run files and packages outside the database server.

Here is an example of a vulnerable run-time call:

```
call dbms_java.grant_permission('SCOTT',
'SYS:java.io.FilePermission','<<ALL FILES>>','read');
```

Here is an example of a secure run-time call:

```
call dbms_java.grant_permission('SCOTT',
'SYS:java.io.FilePermission','<<actual directory path>>','read');
```

■ Enforce Access Controls Effectively.

Although remote authentication can be enabled (TRUE), your installation is more secure with it off (FALSE, which is the default). With remote authentication turned

on, the database implicitly trusts every client, because it assumes every client was authenticated by the remote authenticating system. However, clients such as remote computers cannot be trusted to perform proper operating system authentication. So, enabling this feature is a very poor security practice. To enforce proper server-based authentication of clients connecting to an Oracle database, leave or disable this feature (`remote_os_authentication=FALSE`, which is the default).

- Restrict Operating System Access.

  The following practices implement the required restrictions on operating system access:

  - Limit the number of operating system users.
  - Limit the privileges of the operating system accounts (administrative, root-privileged or DBA) on the Oracle Database host (physical computer) to the fewest and least powerful privileges required for each user.
  - Disallow modifying the default permissions for the Oracle Database home (installation) directory or its contents, even by privileged operating system users or the Oracle owner.
  - Restrict symbolic links. Ensure that when any path or file to the database is provided, neither that file nor any part of that path is modifiable by an untrusted user. The file and all components of the path should be owned by the DBA or some trusted account, such as `root`. This recommendation applies to all types of files, such as data files, log files, trace files, external tables, and bfiles.

- Restrict Network Access.

  The following practices implement the required restrictions on network access

  - Use a firewall.

    Keep the database server behind a firewall. Oracle Net, which is the network infrastructure of the Oracle database, supports various firewalls from different vendors. Supported proxy-enabled firewalls include Network Associates' Gauntlet and Axent's Raptor. Supported packet-filtered firewalls include Cisco's PIX Firewall and supported stateful inspection firewalls include CheckPoint's Firewall-1.

  - Do not leave any port open in the firewall.

    For example, if the Oracle database is behind a firewall, then do not leave the 1521 port of the Oracle listener open to make a connection to the Internet.

    Leaving the port open introduces several security vulnerabilities, including more port openings through the firewall, multithreaded operating system server issues, and revelation of crucial information about databases behind the firewall. Moreover, an Oracle listener running without an established password may be probed for critical details about the databases on which it is listening, such as trace and logging information, banner information, and database descriptors and service names.

    The availability of this information and an ill-configured firewall will enable an attacker to launch malicious attacks on target databases.

  - Authenticate resources accessing your systems.

    Authenticating client computers over the Internet is problematic. Instead, authenticate users. This helps avoid client system issues that include falsified

IP addresses, hacked operating systems or applications, and falsified or stolen client system identities. The following steps improve client computer security:

* Configure the connection to use SSL. Using SSL communication does not allow eavesdropping and enables the use of certificates for user and server authentication.
* Set up certificate authentication for clients and servers such that:

  i. The organization is identified by unit and certificate issuer and the user is identified by distinguished name and certificate issuer.

  ii. Expired certificates are tested by applications.

  iii. Certificate revocation lists are audited.

– Check network IP addresses.

  Use the Oracle Net valid node checking security feature to allow or deny access to Oracle server processes from network clients with specified IP addresses. To use this feature, set the following `protocol.ora` (Oracle Net configuration file) parameters:

```
tcp.validnode_checking = YES
tcp.excluded_nodes = {list of IP addresses}
tcp.invited_nodes = {list of IP addresses}
```

  The first parameter activates the feature, whereas the next two parameters deny or allow specific client IP addresses from making connections to the Oracle Listener. This can prevent potential Denial of Service attacks.

– Encrypt network traffic.

  If possible, use Oracle Advanced Security to encrypt network traffic between clients, databases, and application servers.

  Note: Oracle Advanced Security can be configured, after licensing, with the Oracle Net Manager tool or by manually setting six `sqlnet.ora` parameters to enable network encryption.

– Secure the operating system.

  Secure the host operating system by disabling all unnecessary operating system services. Both UNIX and Microsoft Windows platforms provide several operating system services, most of which are not necessary for most deployments. Such services include FTP, TFTP, and TELNET. Close both the UDP and TCP ports for each service that is being disabled. Disabling one type of port and enabling the other does not secure the operating system.

■ Apply All Security Patches and Workarounds.

  Always apply all the relevant and current security patches for the operating system on which Oracle Database resides, for the Oracle Database itself, and for all installed Oracle Database options and components thereof.

  Periodically check the security site on Oracle Technology Network for information about security alerts released by Oracle.

  http://www.oracle.com/technology/deploy/security/alerts.htm

  Also, check the Web site of Oracle Worldwide Support Service, Metalink, for information about available and upcoming security-related patches.

http://metalink.oracle.com

- Contact Oracle Security Products.

  If you find a security vulnerability in the Oracle database, then raise a service request to Oracle Worldwide Support Services by using Oracle*Metalink* or send an e-mail with a complete description of the problem, including product version and platform, together with any scripts and examples to the following address:

  secalert_us@oracle.com

Application Server Security

Oracle Collaboration Suite applications run within Oracle Application Server. This section describes the security checklist for Oracle Application Server.

Security Checklist for Oracle Application Server

The security checklist for Oracle Application Server is discussed under the following headings:

- Operating System Security
- General Oracle Collaboration Suite Security
- PL/SQL Security

Operating System Security

To secure the operating system:

- Apply operating system vendor-recommended security patches
- Disable unused or unnecessary services
- Remove any operating system samples or example code that is not in use
- Remove any user accounts that are not in use
- Consider moving to SSH and disable Telnet/FTP

General Oracle Collaboration Suite Security

To secure Oracle Collaboration Suite:

- Check the Oracle Technology Network (OTN) site for relevant Oracle security alerts
- Check Oracle*Metalink* for patch set releases
- Remove unused services
- Utilize RedirectMatch and Rewrite rules
- When using Allow or Deny rules, use IP addresses
- Reduce the `TimeOut` setting
- Protect administrative URIs
- Remove any samples and examples from the Oracle Collaboration Suite code tree
- Delete all users and code not in use
- Configure only the ports that are necessary for the application
- Change user of `httpd` daemon to unprivileged user

- Replace standard error pages with custom error pages
- Remove any unnecessary directory references from `httpd.conf`
- Remove actual hosting server name from `httpd.conf`
- Disable the Oracle Application Server banner from the server
- Place all files that are not necessary for the application in a location inaccessible by a Web user
- Remove any unnecessary directives, such as documentation directives
- Turn off indexing for directories

PL/SQL Security

Securing PL/SQL Data Access Descriptors (DADs) is critical for securing applications that use the PL/SQL toolkit. To secure PL/SQL DADs:

- Remove or restrict administration access
- Protect PL/SQL Toolkit
- Remove DAD configurations that are not required
- Restrict DAD access
- Limit privileges of DAD owners and users
- Encrypt DAD passwords
- Apply necessary toolkit/PLSQL patches

Third-Party Software Security

Oracle Collaboration Suite implementation may include third-party products such as SMTP relay servers, antispam, and antivirus software. These products must also be secured against potential attacks and each vendor should be consulted for best practices.

User Security

Oracle Collaboration Suite uses OracleAS Single Sign-On to provide a single password for all Oracle Collaboration Suite applications and many other Oracle and third-party systems. When users are required to use multiple passwords for different systems, they tend to do one of the following:

- Use the same password for many systems. If one password is broken, then all systems with the same password are compromised.
- Store passwords insecurely, such as writing passwords on sticky notes.
- Use easily remembered passwords.

When using OracleAS Single Sign-On, you are required to only log in once for all the application components (Oracle Content Services, Oracle Calendar, Oracle Search etc.).

Password Security

User passwords are stored not as clear text but as one-way hash values. Oracle Internet Directory enables administrators to manage users from a centralized location for account enabling or disabling and provisioning. In addition, Oracle Internet Directory helps administrators implement password policies to ensure that users are not using

easily guessed or easily broken passwords. The password policies enforced by organizations are configurable.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for more information about password policies

Oracle Identity Management

Oracle products use Oracle Identity Management, an integrated infrastructure, to secure users and applications across an enterprise. This section describes the Oracle Identity Management infrastructure and components. It contains the following sections:

- Overview of Identity Management
- Infrastructure of Oracle Identity Management
- Oracle Identity Management and Third-Party Applications
- Benefits of Oracle Identity Management

Overview of Identity Management

An identity is a set of attributes that uniquely identifies a network entity. Identity management is the process by which application user identities are defined and managed in the enterprise environment.

Identity management enables you to perform the following tasks:

- Provision and coordinate user identities
- Automate user account provisioning
- Manage user roles, privileges, and credentials
- Delegate responsibility
- Deploy applications easily and securely
- Manage your preferences and passwords
- Provide users with single sign-on access

By using an identity management system, an enterprise can:

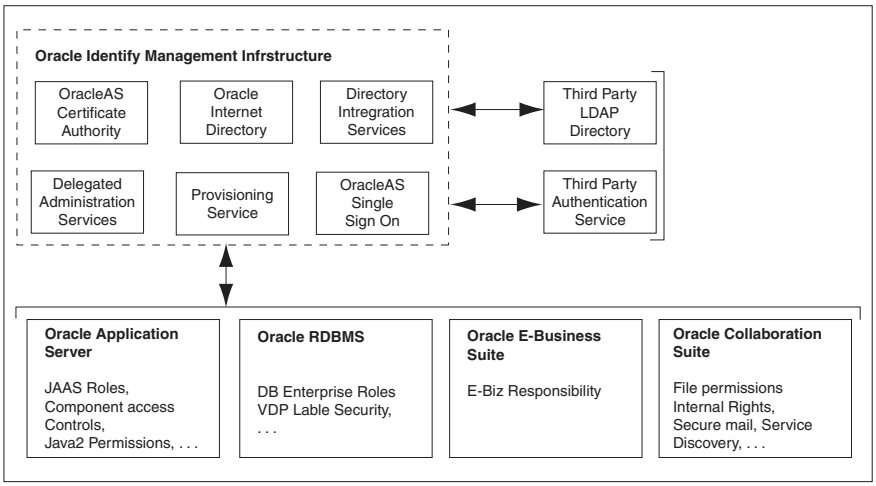
- Reduce administration costs through centralized account management and automated tasks.
- Accelerate application deployment by enabling new applications to leverage the existing infrastructure to provision user accounts and privileges.
- Improve security and usability by centrally managing user passwords and security credentials and customizing applications to leverage centralized authorization and policy information.

Oracle Identity Management enables rapid deployment of Oracle products in the enterprise, without the cost and complexity associated with integrating disparate systems. Oracle Identity Management also functions as a single point of integration between the Oracle environment and any third-party identity management environments.

Infrastructure of Oracle Identity Management

Figure 3–1 illustrates the components of the Oracle Identity Management infrastructure and displays how various Oracle products and third-party products rely on it.

Figure 3–1 Oracle Identity Management Infrastructure



The Oracle Identity Management infrastructure includes the following components:

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

Oracle Application Server Single Sign-On

Using OracleAS Single Sign-On, you need to sign on to the enterprise network only once instead of being prompted for sign-on credentials each time you access other Web applications. When you log on to an application for the first time using OracleAS Single Sign-On, your identity is validated and you do not need to provide your credentials to invoke different applications during a session.

Provisioning Service

Provisioning Service 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 and by enterprise deployment policies. Provisioning Service facilitates both integration and automation of provisioning-related tasks.

Delegated Administration Services

Centralized management of user identities and other security information has its benefits. However, the process of administration could become unscalable if administrative functions are not delegated to different sets of administrators. To support this delegation, the Oracle Delegated Administration Services component

defines a delegation model based on Role-Based Access Control (RBAC). The Oracle Identity Management infrastructure also supports the interfaces required to implement this delegation model and for applications that rely on Oracle Identity Management.

Oracle Delegated Administration Services consists of the following:

- Interfaces to enable end-user self services, such as:
  - Updating, resetting, and recovering user password
  - Managing user preferences and profiles
  - Looking up white pages in the directory
- Interfaces to enable directory administrator services, 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
  - Configuring Oracle Identity Management service-related administration roles

Oracle Internet Directory

Oracle Internet Directory includes:

- Oracle directory server, which responds to client requests for information about resources and to updates of that information by using a multitiered architecture directly over TCP/IP
- Oracle directory replication server, which replicates LDAP data between Oracle directory servers
- Directory administration tools, which include:
  - Oracle Directory Manager, which simplifies directory administration by using a Java-based graphical user interface
  - Command-line administration and data management tools invoked from LDAP clients
  - Directory server management tools within Oracle Enterprise Manager Application Server Control. These tools enable you to:
    * Monitor real-time events and statistics from a browser
    * Integrate the data into a new repository
- Oracle Internet Directory Software Developer's Kit

Oracle Application Server Certificate Authority

Oracle Application Server Certificate Authority provides a simple self-service interface for OracleAS Single Sign-On users to acquire their own X.509 certificates. If you want to deploy PKI to enable high levels of security by using OracleAS Certificate Authority, then you can do so without incurring significant overheads.

Oracle Identity Management and Third-Party Applications

Although Oracle Identity Management is designed to provide an enterprise infrastructure for Oracle products, it can also function as a general-purpose identity management solution for user-written and third-party enterprise applications. It provides a robust and scalable enterprise-wide identity management platform for third-party applications, hardware, and network operating systems. Custom applications can leverage Oracle Identity Management by using the following services and APIs:

- Oracle Internet Directory provides LDAP APIs for C, Java, and PL/SQL, and is compatible with other LDAP SDKs.
- Oracle Delegated Administration Services provide a core self-service console that may be customized to support third-party applications. In addition, it provides a number of services for building customized administration interfaces that manipulate directory data.
- Oracle Provisioning and Directory Integration Services enable you to provision third-party applications and integrate the Oracle environment with other provisioning systems.
- OracleAS Single Sign-On provides APIs to develop and deploy partner applications that share a single sign-on session with other Oracle Web applications.
- Java AuthoriZatioN (JAZN), the Oracle implementation of the Java Authentication and Authorization Service (JAAS) standard, enables applications developed for the Web using Oracle J2EE environment to leverage the Oracle Identity Management infrastructure for authentication and authorization.

Benefits of Oracle Identity Management

This section describes the following benefits provided by Oracle Identity Management to enterprise applications deployed on Oracle Application Server:

Centralized User Management

Oracle Internet Directory facilitates centralized user management for the Oracle environment and 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, but 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 enable administrators to establish rules that you must follow while setting and using passwords to authenticate yourself to the applications on the network. Oracle Identity Management password policies can be customized during deployment.

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

- Value-based password policies make it difficult to guess passwords. These policies force the password values to be arbitrarily complex, such as minimum lengths and presence of minimum number of special characters.
- 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 authentication environments. Password expiration policies and lockout policies based on maximum number of retries are examples of state-based password policies.

You can use the Oracle Internet Directory plug-in capability to implement custom password policies.

SSL Configuration in Oracle Internet Directory

Oracle Internet Directory clients can use SSL 2.0 or SSL 3.0. This section describes how to configure SSL for Oracle Internet Directory. It contains the following topics:

- Configuring SSL Parameters
- Starting a Directory Server Instance with SSL Enabled
- Limitations of the Use of SSL in Oracle Internet Directory

Configuring SSL Parameters

During start-up of a directory server instance, the directory reads a set of configuration parameters, including the parameters for the SSL profile. If you are going to run the directory with SSL enabled, you need to examine—and possibly reconfigure—the SSL parameters in the configuration set entry.

To run a server instance in secure mode, set the `SSL Enable` parameter in the configuration settings to `1`: the default secure port is `3031`. To allow the same instance to run non-secure connections concurrently, set SSL Enable to `2`: the default non-secure port is `3060`.

You can create and modify multiple sets of configuration parameters with differing values, using a different configuration set entry for each instance of Oracle Internet Directory. This is a useful way to accommodate clients with different security needs..

Note: Oracle recommends that you create separate configuration sets and modify their SSL values, rather than modify SSL values in the default configuration set. The default set may be required by Oracle Support Services in the diagnosis of certain technical issues..

See Also: *Oracle Internet Directory Administrator's Guide* for information about configuring SSL parameters by using Oracle Directory Manager and command-line tools

Starting a Directory Server Instance with SSL Enabled

In Oracle Internet Directory, only wallets in encrypted format (`cwallet.sso`) are supported. Before you can start an SSL instance, you must use Oracle Wallet Manager, which is enabled to `Auto Login`, to open the wallet.

On Microsoft Windows, before starting a directory server instance with SSL enabled, you must change the Logon Account of the Oracle Directory Service from Local

System Account to the user who owns the wallet. This user should be member of the Administrator Group.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for more information about configuring SSL parameters

Limitations of the Use of SSL in Oracle Internet Directory

If you want to support both SSL and non-SSL clients on the same host, then you need to configure two distinct server instances.

In Oracle Internet Directory, the Oracle directory replication server cannot communicate directly with SSL-enabled Oracle directory server instances.

Privilege Delegation

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

- Security Goals for the Privilege Delegation Model
- Understanding the Delegation Model
- Understanding Roles and Responsibilities
- Delegating Privileges
- Granting Privileges to Manage User and Group Data
- Delegating Privileges for Component Runtime

Security Goals for the Privilege Delegation Model

Oracle Collaboration Suite provides fine-grained control over system administration and management privileges. This enables 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 the permission to deploy one component does not grant the right to deploy or administer other components

Understanding the Delegation Model

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

To delegate the necessary privileges, you assign the user to the administrative group. If you store the data for both enterprise users and the e-mail service in the directory, then you need to specify a unique administrator for each set of data. For example, to

specify a user as the administrator of enterprise users, you assign that user to the Enterprise User Administrators Group. Similarly, to specify a user as the administrator of the e-mail services, you assign that user to the E-mail Service Administrators Group.

Understanding Roles and Responsibilities

The new privilege model supports the following user roles:

- Oracle Application Server Installation Administrator

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

- Oracle Application Server Application Administrator

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

- Oracle Identity Management Infrastructure Administrator

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

- Oracle Application Server Application User

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

Note: The same user may perform multiple roles.

Delegating Privileges

In an Oracle Collaboration Suite environment, the directory superuser creates the following:

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

If you are working in an existing Oracle Internet Directory, then 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).

Granting Privileges to Manage 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, such as 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, such as 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, such as 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.

Delegating 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, the server:
  - Connects to Oracle Internet Directory using its own identity
  - Verifies whether the password entered by the user matches the password stored in the directory

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

In general, Oracle components can require these privileges:

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

> **See Also:** *Oracle Internet Directory Administrator's Guide* for a comprehensive discussion of privilege delegation

4

Oracle Collaboration Suite Database Security

Database security requirements arise from the need to protect data from accidental loss and corruption as well as from deliberate unauthorized attempts to access or alter that data. Security concerns also include protecting against undue delays in accessing or using data, interference, and in extreme cases, denial of service.

This chapter presents the fundamental concepts of database security requirements and contains the following sections:

Introduction to Database Security Concepts

Confidentiality, integrity, and **availability** are the cornerstones of database security. Who should have the right to access data? What portion of all the data should a particular user be able to access? What operations should an authorized user be able to perform on the data? Can authorized users access valid data when necessary? Authorization is the permission given to a user, program, or process to access an object or set of objects. The type of data access granted to a user can be read-only or read/write. Privileges specify the type of Data Manipulation Language (DML) operations that the user can perform on data.

Database integrity ensures that data in the database is correct and consistent. Database integrity mechanisms can be divided into those that support system integrity and those that enforce relational database integrity properties such as entity integrity, referential integrity, transaction integrity, and business rules.

Traditional system integrity involves ensuring that data retrieved from the system is the same as the data when it was originally inserted. In addition, data must not be altered or deleted by a user who is not authorized to do so. A database must ensure that data adheres to certain business rules, as determined by the database administrator or application developer. For example, assume that a business rule says that no employee in the emp table can receive a raise greater than 20 percent of the value in the salary column. If an insert or update statement attempts to violate this

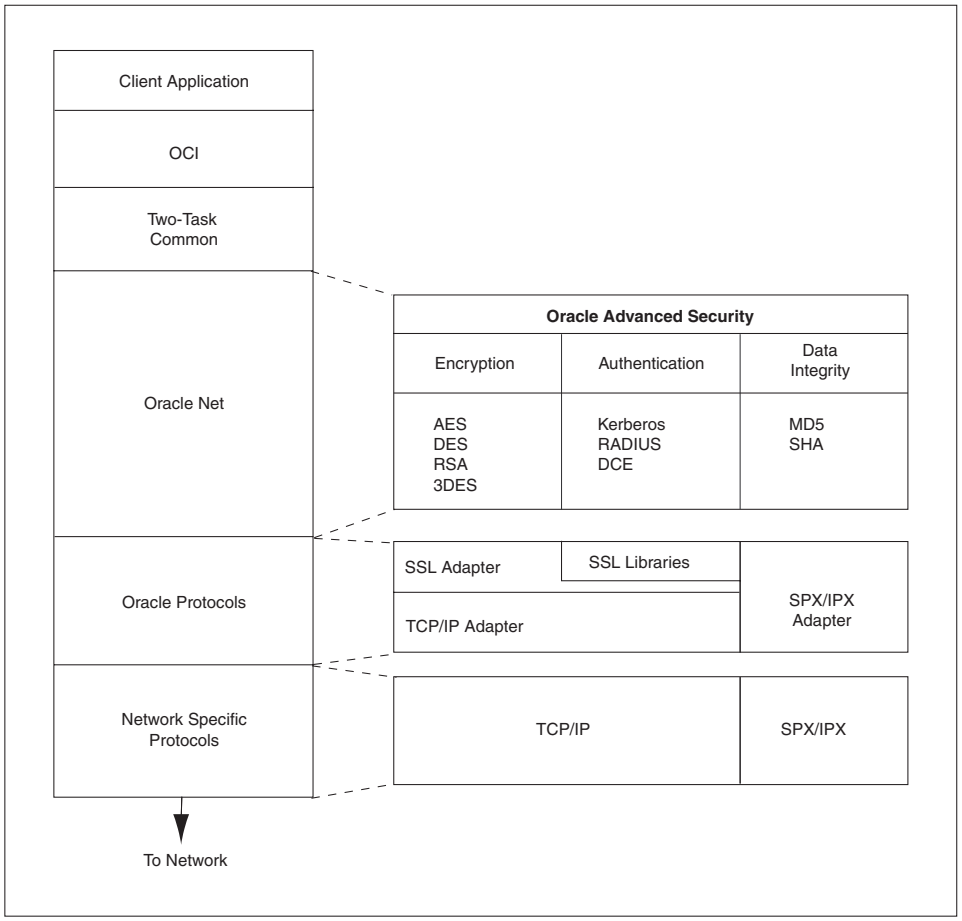
integrity rule, then the statement must fail. Integrity constraints and database triggers can be used to enforce data integrity rules within a database.

Referential integrity rules form the basis of data integrity in relational databases. Referential integrity ensures that data entered in one table, which references another table in a relational database, and the rules that govern this data are in conformance with the referenced values. Referential integrity includes rules that dictate the types of data manipulation allowed on referenced values, and these rules are used to determine the effect of data manipulation on dependent values.

Oracle Advanced Security Architecture

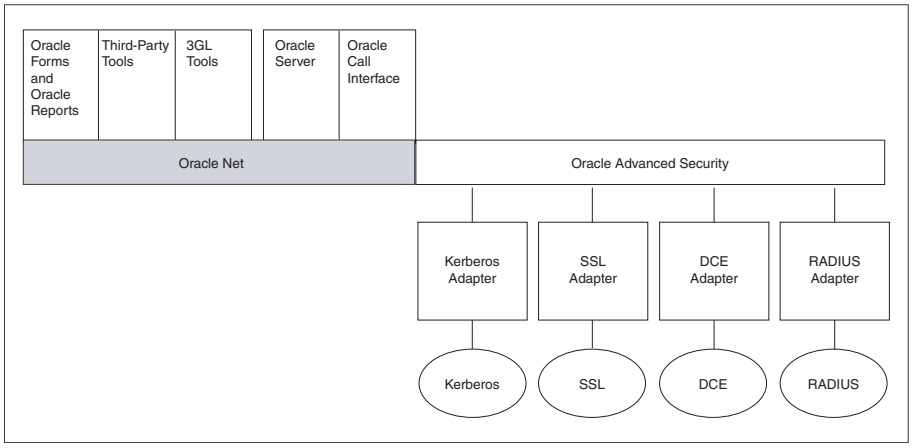
Oracle Advanced Security provides a comprehensive suite of security features to protect enterprise networks and securely extend corporate networks to the Internet. It provides a single source of integration with network encryption and authentication solutions, single sign-on services, and security protocols. By integrating industry standards, it delivers unparalleled security to an Oracle networking environment. Oracle Advanced Security complements an Oracle server or client installation with advanced security features. Figure 4–1 shows the Oracle Advanced Security architecture within an Oracle networking environment.

Figure 4–1 Oracle Advanced Security in an Oracle Networking Environment



Oracle Advanced Security supports authentication through adapters that are similar to the existing Oracle protocol adapters. As shown in Figure 4–2, authentication adapters integrate under the Oracle Net interface and allow existing applications to take advantage of new authentication systems transparently, without any changes to the application.

Figure 4–2 Oracle Net with Authentication Adapters



Solving Security Challenges with Oracle Advanced Security

To solve enterprise computing security problems, Oracle Advanced Security provides various options to ensure industry standards-based data privacy, integrity, authentication, single sign-on, and access authorization. For example, you can configure either Oracle Net native encryption or Secure Sockets Layer (SSL) for data privacy. Oracle Advanced Security also provides the choice of several strong authentication methods, including Kerberos, smart cards, and digital certificates.

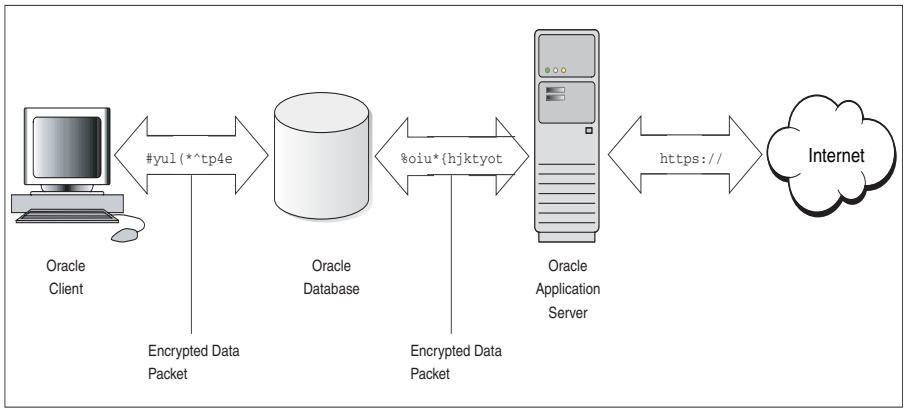
Oracle Advanced Security provides the following security features:

- Data Encryption
- Strong Authentication

Data Encryption

Sensitive information that travels over enterprise networks and the Internet can be protected by encryption algorithms. An encryption algorithm transforms information into a form that can be deciphered only with a **decryption** key. Figure 4–3 shows how encryption works to ensure the security of a transaction. For example, if a manager approves a bonus, this data should be encrypted when sent over the network to avoid eavesdropping. If all communication between a client, database, and an application server is encrypted, then when the manager sends the bonus amount to the database, the information is protected.

Figure 4–3 Encryption



This section discusses the following topics:

- Supported Encryption Algorithms
- Data Integrity
- FIPS

Supported Encryption Algorithms

Oracle Advanced Security provides the following encryption algorithms to protect the privacy of network data transmissions:

- RC4 Encryption
- DES Encryption
- Triple-DES (3DES) Encryption
- Advanced Encryption Standard

Selecting the network encryption algorithm is a user configuration option that provides varying levels of security and performance for different types of data transfer. Prior versions of Oracle Advanced Security provided three editions: Domestic, Upgrade, and Export--each with different key lengths. Oracle Advanced Security 10g Release 1 (10.1) contains a complete complement of the available encryption algorithms and key lengths, previously only available in the Domestic edition. Users deploying prior versions of the product can obtain the Domestic edition for a specific product release.

RC4 Encryption

The RC4 encryption module uses the RC4 encryption algorithm created by RSA Security, Inc. Using a secret, randomly-generated key that is unique to each session, all network traffic is fully safeguarded, including all data values, SQL statements, and stored procedure calls and results. The client, server, or both, can require or request the use of the encryption module to guarantee data protection. The optimized implementation supported by Oracle provides a high degree of security for a minimal performance penalty. For the RC4 algorithm, Oracle provides encryption key lengths of 40-bits, 56-bits, 128-bits, and 256-bits.

DES Encryption

Oracle Advanced Security implements the U.S. Data Encryption Standard (DES) algorithm with a standard, optimized, 56-bit key encryption, and also provides DES40, a 40-bit version, for backward compatibility.

Triple-DES (3DES) Encryption

Oracle Advanced Security also supports Triple-DES encryption (3DES), which encrypts message data with three passes of the DES algorithm. 3DES provides a high degree of message security, but with considerable performance penalty. The magnitude of penalty depends on the speed of the processor performing the encryption. 3DES typically takes three times as long to encrypt a data block as compared to the standard DES algorithm. 3DES is available in two-key and three-key versions with effective key lengths of 112 bits and 168 bits, respectively. Both versions operate in outer Cipher Block Chaining (CBC) mode.

Advanced Encryption Standard

Approved by the National Institute of Standards and Technology (NIST) in Federal Information Processing Standards (FIPS) Publication 197, Advanced Encryption Standard (AES) is a new cryptographic algorithm standard developed to replace DES. AES is a symmetric block cipher that can process data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256 bits, which are referred to as AES-128, AES-192, and AES-256, respectively. All three versions operate in outer CBC mode.

Data Integrity

To ensure the integrity of data packets during transmission, Oracle Advanced Security can generate a cryptographically secure message digest, using MD5 or SHA-1 hashing algorithms, and include it with each message sent across a network.

Data integrity algorithms add little overhead, and protect against the following attacks:

- Data modification
- Deleted packets
- Replay attacks

FIPS

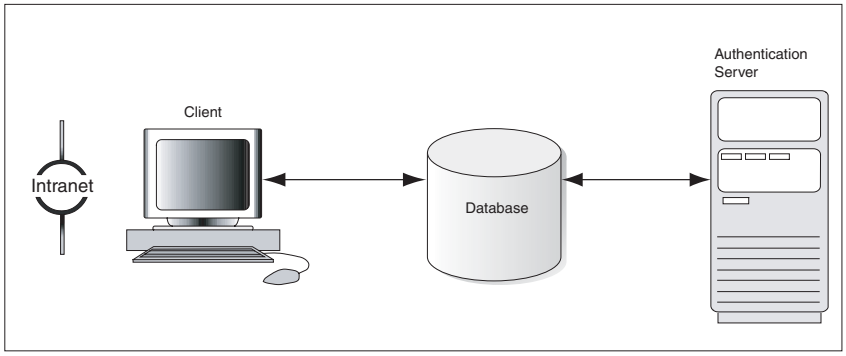
Oracle Advanced Security Release 8.1.6 has been validated under U.S. FIPS 140-1 at the Level 2 security level. This provides independent confirmation that Oracle Advanced Security conforms to federal government standards.

Strong Authentication

Authentication is used to prove the identity of a user. Authentication, without which there can be little confidence in network security, is imperative in distributed environments. Passwords are the most common means of authentication. Oracle Advanced Security provides strong authentication with Oracle authentication adapters that support various third-party authentication services, including SSL with digital certificates.

Figure 4–4 shows user authentication with an Oracle database configured to use a third-party authentication server. Having a central facility to authenticate all the members of a network (clients to servers, servers to servers, and users to both clients and servers) is one effective way to address the threat of network nodes falsifying their identities.

Figure 4–4 Strong Authentication with Oracle Authentication Adapters



See Also: *Oracle Advanced Security Administrator's Guide* for more information about the authentication methods supported by Oracle Advanced Security

SSL Combined with Other Authentication Methods

You can configure Oracle Advanced Security to use SSL concurrently with database user names and passwords, RADIUS, and Kerberos, which are discussed in the following sections:

- Oracle Advanced Security and SSL
- How SSL Works with Other Authentication Methods
- SSL and Firewalls
- SSL Usage Issues

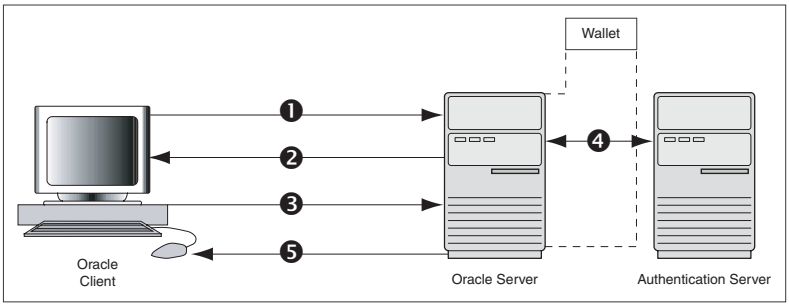
Oracle Advanced Security and SSL

Figure 4–1, which displays the Oracle Advanced Security implementation architecture, shows that Oracle Advanced Security operates at the session layer on top of SSL and uses TCP/IP at the transport layer. This separation of functionality lets you employ SSL concurrently with other supported protocols.

How SSL Works with Other Authentication Methods

Figure 4–5 illustrates a configuration in which SSL is used in combination with another authentication method supported by Oracle Advanced Security. In this example, SSL is used to establish the initial handshake (server authentication), and a different authentication method is used to authenticate the client.

Figure 4–5 SSL Combined with Other Authentication Methods



1. The client seeks to connect to the Oracle database server.
2. SSL performs a handshake during which the server authenticates itself to the client and both the client and server establish which cipher suite to use.
3. Once the SSL handshake is successfully completed, the user seeks access to the database.
4. The Oracle database server authenticates the user with the authentication server using a non-SSL authentication method such as Kerberos or RADIUS.
5. Upon validation by the authentication server, the Oracle database server grants access and authorization to the user, and then the user can access the database securely by using SSL

SSL and Firewalls

Oracle Advanced Security supports two types of firewalls:

- Application proxy-based firewalls, such as Network Associates Gauntlet or Axent Raptor.
- Stateful packet inspection firewalls, such as Check Point Firewall-1 or Cisco PIX Firewall.

When you enable SSL, stateful inspection firewalls behave similar to application proxy firewalls because they do not decrypt encrypted packets.

Firewalls do not inspect encrypted traffic. When a firewall encounters data addressed to an SSL port on an intranet server, it checks the target IP address against its access rules. It then allows the SSL packet to pass through to permitted SSL ports, rejecting all other packets.

With the Oracle Net Firewall Proxy kit, a product offered by some firewall vendors, firewall applications can provide specific support for database network traffic. If the proxy kit is implemented in the firewall, the following processing takes place:

- The Net Proxy (a component of the Oracle Net Firewall Proxy kit) determines where to route the traffic it handles.
- The database listener requires access to a certificate to participate in the SSL handshake. The listener inspects the SSL packet and identifies the target database, returning the port on which the target database listens to the client. This port must be designated as an SSL port.
- The client communicates on this server-designated port in all subsequent connections.

- The number of ports that are open in the firewall increase as a function of the number of database connections requested for different databases. However, this approach prevents the database server from using randomly chosen SSL ports. This is because the SSL ports chosen by the database must match the SSL ports on the firewall. You can avoid this condition by deploying Oracle Connection Manager, an application included with Oracle Database Enterprise Edition.

Oracle Connection Manager enables you to route client connections over multiple Oracle Net protocols. Each client connection request establishes an SSL connection between the client and Oracle Connection Manager, which in turn establishes a TCP/IP connection with the target database. Multiple clients can thus connect to multiple databases behind the firewall, using a single SSL port on the firewall.

SSL Usage Issues

Although the use of SSL enables secure communication with other Oracle products, such as Oracle Internet Directory, there are some issues that you need to bear in mind:

- Because SSL supports both authentication and encryption, the client/server connection is somewhat slower than the standard Oracle Net TCP/IP transport (using native encryption).
- Each SSL authentication mode requires configuration settings.

Secure Configuration Practices

Any techniques that you employ to assure data security can be rendered useless unless the database administrator follows good security practices. For example, you should always:

- Revoke privileges from public accounts.
- Lock unused accounts.
- Change any default passwords after installation.
- Set the proper permissions for users accessing databases. You could also implement roles to manage privileges. Check whether permissions are correctly set in operating system. If these are not correctly set, there may be a security loophole.

Database Security Policies

This section briefly introduces security policies. It covers:

- Security Threats and Countermeasures
- What Information can Security Policies Cover

Security Threats and Countermeasures

An organization should create a well-documented security policy. This policy should enumerate the security threats the organization is trying to guard against and the specific measures the organization must take when it encounters these threats. Security threats can be addressed by the following measures:

- Procedural, such as requiring data center employees to display security badges
- Physical, such as securing computers containing critical information in restricted-access facilities

- Technical, such as implementing strong authentication requirements for critical business systems
- Personnel-related, such as performing background checks or vetting key personnel

The organization should consider whether the response to a threat is procedural, physical, technical, personnel-related, or a combination of these. For example, one possible security threat is the disruption of critical business systems caused by a malicious person damaging a computer. A physical response to this threat is to secure key business computers in a restricted-access facility. A procedural response is to create system backups at regular intervals. Personnel measures could include background checks on employees who access or manage key business systems.

What Information can Security Policies Cover

In addition to addressing requirements unique to the organization environment, the organization should also design and implement technical measures in its information security policies to address generic issues, as listed in Table 4–1:

Table 4–1 Issues and Actions Governing Security Policies

Security Concern/Practice	Recommended Actions
Establish and maintain application-level security	Attach privileges and roles to each application. Ensure that users cannot misuse these roles and privileges when they are not using the application. Base the use of roles on user-specific criteria, such as a user connecting only from a particular IP address, or only through a particular application tier.
Manage privileges and attributes (system/object/user)	Permit only a limited set of users to access, process, or alter data or to run a particular type of SQL statement or to access another user's object. Apply the required limitations on user access to objects or actions on objects, such as schemas, tables, or rows, or resources, such as system usage time (CPU, connect, or idle times).
Create, manage, and control roles (database, enterprise)	Create named groups of privileges called roles to facilitate granting them to users, including previously named groups.
Establish the granularity of access control desired	Set up session-based attributes in a secure manner. For example, store user attributes (user name, employee number, and so on) to be retrieved later in the session, enabling fine-grained access control. Create security policy functions and attach them to critical or sensitive tables, views, or synonyms used by an application. DML statements on such objects are then modified dynamically, and transparently to the user, to disable inappropriate access.
Establish and manage the use of encryption	Use Secure Socket Layer (SSL) connections, well-established encryption suites, or PKI certificates for critical/sensitive transmissions/applications.
Establish and maintain security in 3-tier applications	Preserve user identity through a application tier to the database. Avoid the overhead of separate database connections by proxying user identities (and credentials such as a password or certificate) through the application tier to the database.
Control query access, data misuse, and intrusions	Monitor query access based on specific content or row to detect data misuse or intrusions. Use proxy authentication to support auditing of proxied user connections. Use Regular Auditing and Fine-Grained Auditing to detect unauthorized or inappropriate access or actions.

Authentication by the Oracle Database

Oracle can authenticate users attempting to connect to a database, by using information stored in that database.

To set up Oracle to use database authentication, you create each user with an associated password that must be supplied when the user attempts to establish a connection. This process prevents unauthorized use of the database because the connection will be denied if the user provides an incorrect password. Oracle stores a user's password in the data dictionary in an encrypted format to prevent unauthorized alteration, but a user can change his own password at any time.

To establish which authentication protocols are allowed by the client or database, a DBA can explicitly set the `SQLNET.ALLOWED_LOGON_VERSION` parameter in the `sqlnet.ora` server file. Then each connection attempt is tested, and if the client or server does not meet the minimum version specified by its partner, authentication fails with an `ORA-28040` error. The parameter can take the values `10`, `9`, or `8` (default), which represent the database server versions. Oracle recommends the value `10`.

Database authentication includes the following facilities:

- Password Encryption While Connecting
- Account Locking
- Password Lifetime and Expiration
- Password History
- Password Complexity Verification

Password Encryption While Connecting

Passwords are always automatically and transparently encrypted during network (client/server and server/server) connections. This is done by using a modified DES (Data Encryption Standard) or 3DES algorithm before sending them across the network.

Account Locking

Oracle can lock a user's account after a specified number of consecutive failed log-in attempts. You can configure the account to unlock automatically after a specified time interval or by database administrator intervention.

Use the `CREATE PROFILE` statement to specify the number of failed login attempts allowed before the account locks and the duration for which the account remains locked before it unlocks automatically.

The DBA can also lock accounts manually. In these cases, the accounts cannot unlock automatically and must be unlocked explicitly by the DBA.

Password Lifetime and Expiration

The database administrator can specify a lifetime for passwords, after which they expire and must be changed before account login is permitted. A grace period can be specified, during which each attempt to log in to the database account receives a warning message to change the password. If it is not changed by the end of that period, the account is locked. No further login to that account is allowed without assistance by the DBA.

The DBA can also manually set the password state to expired, causing the user's account status to change to expired. The user or the DBA must then change the password before the user can log in to the database.

Password History

The password history option checks each newly specified password to ensure that a password is not reused for a specified amount of time or for a specified number of password changes. The database administrator can configure the rules for password reuse with `CREATE PROFILE` statements.

Password Complexity Verification

Password complexity verification checks that each password is complex enough to provide reasonable protection against intruders who try to break into the system by guessing passwords.

The sample Oracle password complexity verification routine (PL/SQL script `UTLPWDMG.SQL`, which sets the default profile parameters) specifies that a password must:

- Be a minimum of four characters in length.
- Not be the same as the user ID.
- Include at least one alphabet character, one numeric character, and one punctuation mark.
- Not match any word on an internal list of simple words, such as welcome, account, database, and user.
- Differ from the old password by at least three characters.

Part II

Secure Sockets Layer Configuration

This part contains the following chapters:

5

Overview of SSL Configuration in Oracle Collaboration Suite

In Oracle Collaboration Suite, components send requests to and receive responses from other components. These components can be Oracle Application Server components, such as OracleAS Single Sign-On, OracleAS Web Cache, or Oracle HTTP Server, or external clients such as browsers. To secure these communications, you can configure Oracle Collaboration Suite to use Secure Sockets Layer (SSL), which is an industry standard for securing communications.

This chapter contains the following sections:

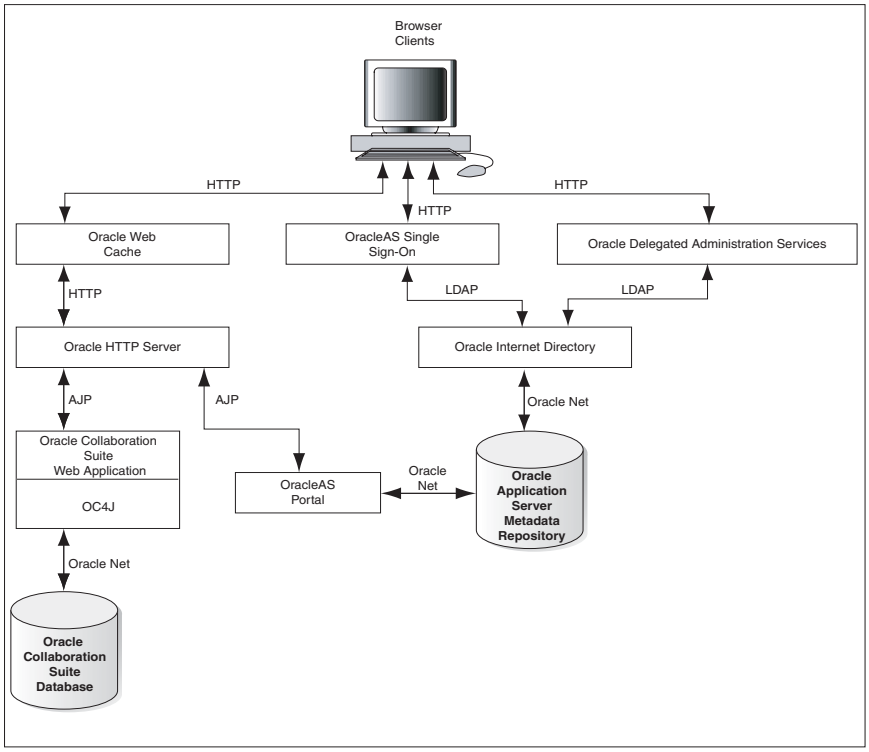
SSL Configuration Overview

SSL secures communication between two parties: a client and a server. If three or more parties are involved, for example, client browser, Web Cache, Oracle HTTP Server, and OC4J, then you may have to configure all components to use SSL.

Figure 5–1 shows typical communication paths between Oracle Collaboration Suite components and the protocols that they use. For example, browsers use HTTP to communicate with OracleAS Web Cache, and Oracle HTTP Server uses Apache Java Protocol (AJP) to communicate with OC4J. All these protocols can work with SSL.

When you enable SSL for Oracle Collaboration Suite, the HTTP communications depicted in Figure 5–1 will then use the HTTPS protocol for secure connections. URLs that require an SSL connection begin with `https` rather that `http`.

Figure 5–1 Communication Paths Between Components in Oracle Collaboration Suite



Default SSL Configuration

If you select the default options in the Oracle Collaboration Suite Infrastructure installation, none of the components are configured for SSL.

Partial SSL Configuration

Oracle Collaboration Suite enables you to selectively configure SSL for the paths that you want to secure. There are many paths used by components, as illustrated in Figure 5–1. You might not want to secure all the paths for the following reasons:

- SSL is resource-intensive. If you have heavy SSL traffic, then you should consider offloading SSL processing to an SSL accelerator.
- If your computers are behind firewalls, then you might want to secure only those paths that are accessed by the public. For example, you might choose to secure only OracleAS Web Cache and Oracle HTTP Server if the public can access only these components. This chapter only provides instructions for this scenario.

High-Level Tasks to Enable SSL in Oracle Collaboration Suite

To enable SSL communications in Oracle Collaboration Suite, follow these high-level steps:

1. Ensure that you have a working installation of Oracle Collaboration Suite. To verify that you have a valid Oracle Collaboration Suite installation, check the list of prerequisites for Oracle Collaboration Suite SSL. Refer to Validating Your Installation.

2. Obtain certificates for your Oracle Collaboration Suite components, and create a wallet on each computer in your Oracle Collaboration Suite installation that will accept SSL connections. Refer to Creating a Complete Wallet: Process Overview.
3. Configure the Infrastructure for SSL. Refer to SSL Configuration in Oracle Collaboration Suite Infrastructure.
4. Configure SSL for the Oracle Collaboration Suite Applications tier components. Refer to SSL Configuration in Oracle Collaboration Suite Applications.
5. Make configuration changes for Oracle Real-Time Collaboration to work with SSL. Refer to Enabling SSL in Oracle Real-Time Collaboration.
6. Troubleshoot your Oracle Collaboration Suite SSL set up. Refer to Appendix B, "Troubleshooting SSL Configuration" for solutions to some common problems that you may encounter.

Validating Your Installation

Before you begin configuring your Oracle Collaboration Suite installation for SSL, you must first ensure that you have a valid, working installation. It is important that you complete the original installation process without errors. Otherwise, you may not be able to complete the reregistration process necessary for SSL. This section provides some instructions for testing your current Oracle Collaboration Suite installation before beginning the SSL configurations.

Note: If you are not sure about the ports that were assigned at installation time, then you will find the port assignments for your initial installation in:

`$ORACLE_HOME/install/portlist.ini`

To validate your Oracle Collaboration Suite installation, run the following tests:

1. Verify that you can log in to OracleAS Single Sign-On Server Administration as the `orcladmin` user at the following URL:

   `http://infratierhostname:port/pls/orasso`

2. Verify that you can log in to Oracle Internet Directory Delegated Administration Services as the `orcladmin` user at the following URL:

   `http://infratierhostname:port/oiddas`

   Ensure that you are able to add a new user and log out.

3. Verify that you can access all configured Oracle Collaboration Suite applications that require OracleAS Single Sign-On at the following URLs:

   - User URLs

     Welcome Page: `http://appstierhostname:port/welcome/index.jsp`

     Oracle Mail URL = `http://appstierhostname:port/um`

     OracleAS Portal URL = `http://appstierhostname:port/pls/portal`

     Oracle Search URL = `http://appstierhostname:port/search`

     Oracle Calendar URL = `http://appstierhostname:port/ocas-bin/ocas.fcgi?sub=web`

Oracle Discussions URL=
`http://appstierhostname:port/discussions/app`

Oracle Workspaces URL =
`http://appstierhostname:port/workspaces`

Oracle Content Services URL =
`http://appstierhostname:port/content`

Web Access = `http://appstierhostname:port/ocsclient`

Client Download Page =
`http://appstierhostname:port/welcome/download.jsp`

Oracle Real-Time Collaboration URL =
`http://appstierhostname:port/imtapp/app/prelogin.uix`

- Administrator URLs

  Enterprise Manager

  Infrastructure tier = `http://infratierhostname:1156`

  Applications tier = `http://infratierhostname:1156`

  Oracle Mail = `http://appstierhostname:port/um/`

  Oracle Voicemail & Fax = `http://appstierhostname:port/um/`

  Oracle Calendar:
  `http://appstierhostname:port/ocad-bin/ocad.cgi?object=node adm`

  Oracle Mobile Collaboration:
  `http://appstierhostname:port/webtool/login.uix`

  Oracle Search: `http://appstierhostname:port/search`

  Oracle Real-Time Collaboration:
  `http://appstierhostname:port/imtapp/app/prelogin.uix`

4. The Distributed Configuration Management utility, `dcmctl`, provides a command-line alternative to using Oracle Enterprise Manager. Ensure that you can run the `dcmctl updateConfig` command. Before running `dcmctl`, you must first shut down Oracle Enterprise Manager (`emctl`). Run the following commands:

   ```
   $ORACLE_HOME/bin/emctl stop
   $ORACLE_HOME/dcm/bin/dcmctl updateConfig -v -d
   ```

5. Correct any problems with your installation before proceeding. After you have confirmed that there are no errors, shut down everything and perform a cold backup.

System Requirements for Using SSL in Oracle Collaboration Suite

The requirements described in this section assume that your Oracle Collaboration Suite installation meets the following system requirements:

- Oracle Collaboration Suite 10*g* Release 1 (10.1.1) is installed.
- ORACLE_HOME is set to the location that was specified during installation.

- PATH environment variable includes the *$ORACLE_HOME*/bin directory and *$ORACLE_HOME*/opmn/bin directory. You should update your PATH on both the Infrastructure and Applications tier.

Certificates and Oracle Wallets

You need a certificate for your site. When clients connect to your site requesting SSL communication, you usually have to send your certificate to them so that they can authenticate you.

Oracle Application Server supports X.509 V3 certificates, and certificates that conform to the PKIX standard (RFC 3280).

Obtaining an SSL Certificate

You get certificates from certificate authorities (CAs). CAs are trusted entities who sign the certificates that they issue with their private key. Clients can verify the issuer of a certificate (by using the public key of the CA). Examples of CAs include Verisign (http://www.verisign.com) and Thawte (http://www.thawte.com).

Oracle Application Server also has a certificate authority, called OracleAS Certificate Authority (OCA). You can use it to set up your own certificate authority. Refer to the *Oracle Application Server Certificate Authority Administrator's Guide* for details.

To get a certificate, you submit a certificate request to a CA. The certificate request contains your information including your public key. You can use tools to generate a certificate request. These tools can generate private and public key pairs for you. Examples of tools that can generate certificate requests include Oracle Wallet Manager and Sun's keytool (for OC4J only). For information on Oracle Wallet Manager, see Chapter 6, "Managing Wallets and Certificates".

Among other items, a certificate includes the following pieces of data:

- Certificate owner's name
- Certificate owner's public key
- CA's name
- Certificate expiration date
- Certificate serial number

Certificates are valid until they expire or until they are revoked.

Note that if you use OracleAS Certificate Authority (OCA) to create certificates for your server, most browsers will not accept these certificates without input from the browser user. This is because most browsers are preconfigured to accept certificates from certain CAs, and OCA is not one of them. The browser will reject the certificate from the server unless the user chooses to accept certificates from the server or import the CA's certificate.

This problem exists for all CAs until the CA's certificate is imported into the browsers. For more information, refer to *Oracle Application Server Certificate Authority Administrator's Guide*.

For servers to communicate securely with clients, you must obtain an SSL server certificate for each computer and configure its network listener to use that certificate.

In the Oracle environment, use the Oracle Wallet Manager for secure creation and storage of certificates and the corresponding private keys. To obtain a certificate, use

the Oracle Wallet Manager as described in the *Oracle Advanced Security Administrator's Guide*.

The general steps are as follows:

1. Create a new wallet, if one does not already exist. The same wallet can be used by all servers running on that computer.
2. Generate a certificate request, entering the host name along with the domain name as the Common Name. Requesting a certificate request generates the corresponding private key and stores it in the wallet.
3. Send the certificate request to a Certificate Authority, such as VeriSign, for signing.
4. Store the signed certificate in the wallet with the **Auto Login** option enabled. The certificate status should be set to **Ready**.
5. Remember to store the wallet with the **Auto Login** option enabled. The option is under the **Wallet** menu option in the Oracle Wallet Manager. This creates a `cwallet.sso` file in addition to the `wallet.p12` that is the actual wallet. The files are stored in the `$ORACLE_HOME/etc/ORACLE/WALLETS/`*`userID`* directory.

Configuring the Network Listener for SSL

During installation, the `listener.ora` file is updated with the required SSL and non-SSL listening end points for both the IMAP and POP servers. Administrators only need to set the wallet location in the `listener.ora` and the `sqlnet.ora` files, along with any optional SSL parameters, for the listener to receive SSL connections. These settings can be done manually or by using the Oracle Network Manager.

Manually Setting Wallet Location and Client Authentication

Add the following `WALLET_LOCATION` and `SSL_CLIENT_AUTHENTICATION` entries to the beginning of the `$TNS_ADMIN/listener.ora` and `$TNS_ADMIN/sqlnet.ora` files located on the Oracle Collaboration Suite Applications Tier:

```
WALLET_LOCATION = (SOURCE = (METHOD = FILE)
(METHOD_DATA =
(DIRECTORY = Directory_path_containing_the_cwallet.sso_file)
)

SSL_CLIENT_AUTHENTICATION = FALSE
```

A typical directory parameter value looks like the following line:

```
/etc/ORACLE/WALLETS/userID
```

If the `SSL_CLIENT_AUTHENTICATION` parameter is not set, the default setting is `TRUE` and clients are required to present a certificate during the SSL handshake. If the intent is only to secure the communication, not to authenticate the client using the certificate, set this parameter to `FALSE`.

Oracle Wallet

An Oracle wallet is a container that stores your credentials, such as certificates, certificate requests, and private keys. You can store Oracle wallets on the file system or in LDAP directories such as Oracle Internet Directory. Oracle wallets are password-protected.

You manage Oracle wallets using Oracle Wallet Manager. Use Oracle Wallet Manager to perform tasks such as creating Oracle wallets, creating certificate requests, importing certificates into the wallet, and uploading wallets to an LDAP directory.

Oracle Wallet Manager supports PKCS #11 and **PKCS #12** wallets:

- Scenario 1: You generate a certificate request using Oracle Wallet Manager and decide to store the private key on the file system. When you get your certificate from the CA, you can import it into an Oracle wallet. This wallet uses the PKCS #12 format. Refer to Creating a Standard Wallet in Chapter 6, "Managing Wallets and Certificates" for details.
- Scenario 2: You generate a certificate request using Oracle Wallet Manager and decide to store the private key on a hardware security module. When you get your certificate from the CA, you can import it into an Oracle wallet. This wallet uses the PKCS #11 format. Refer to Creating a Wallet to Store Hardware Security Module Credentials in Chapter 6, "Managing Wallets and Certificates" for details.
- Scenario 3: You already have a certificate in a wallet that uses the PKCS #12 format, and you want to use it with Oracle Application Server. The wallet was created using a third-party tool. In this case, use the tool that was used to create the wallet and export the wallet to a file on the file system. You can then import the wallet. Refer to Importing Certificates Created with a Third-Party Tool in Chapter 6, "Managing Wallets and Certificates" for details.

Components that Use Oracle Wallets

Oracle Application Server components that act as SSL servers need Oracle wallets (the wallet already contains the certificate that you want the server to use). Examples of these components include Oracle HTTP Server, OracleAS Web Cache, OPMN, Oracle Internet Directory, and the Port Tunneling daemon (`iaspt`).

You configure the component with the location of the Oracle wallet. For example, to configure Oracle HTTP Server for SSL, you specify the location of the wallet using the `SSLWallet` directive. Refer to the component guide for specific instructions on how to specify the wallet location for the component.

Note: The OC4J component uses a *keystore* instead of an Oracle wallet to store its certificate. You use a tool called `keytool` to import certificates into keystores. Refer to the *Oracle Application Server Containers for J2EE Security Guide* for details on keystores and `keytool`.

Client Certificates

If you need to authenticate your clients, you can configure the Oracle HTTP Server to require clients to send their certificates. Clients can also get their certificates from CAs.

If the clients are Oracle components, for example, OracleAS Web Cache can act as a client when communicating with Oracle HTTP Server, the client component can store its certificate in an Oracle wallet. OPMN also acts as a client when configured for SSL.

If the client is a browser, the client does not need an Oracle wallet. You can just import the certificate into the browser.

Other types of clients, such as SOAP or Web Services clients, have their own ways of configuring certificates and certificate stores.

Integration with Hardware Security Modules

When clients connect to your site using SSL, the extra processing required for SSL strains your servers, and your site as a whole (SSL as well as non-SSL connections) will experience slower performance and throughput. You should consider using SSL accelerator hardware to offload SSL computations and improve performance.

The various SSL accelerators are:

- Protocol Converters
- Mathematics Accelerators (PKCS #11 Integration)

Protocol Converters

Protocol converters convert HTTPS traffic to HTTP. Protocol converters are standalone hardware computers. Oracle Application Server supports protocol converters from companies such as:

- F5 (http://www.f5.com)
- Cisco (http://www.cisco.com)
- SonicWall (http://www.sonicwall.com)

Note: SSL connections to protocol converters terminate at the protocol converter. When the converters forward the requests to Oracle Application Server, most of them do so in an unencrypted fashion.

For the protocol converters that forward the requests to Oracle Application Server using SSL, this is still faster than not using a protocol converter because using a protocol converter eliminates most SSL key exchanges (which is the expensive operation).

Mathematics Accelerators (PKCS #11 Integration)

Mathematics accelerators improve the speed of math operations used by SSL. Such devices are usually plugged into a server (often through TCP/IP) and have additional features such as key management and secure key stores.

Oracle Collaboration Suite supports mathematics accelerators that follow the PKCS #11 standard. For a list of certified accelerators, you can check the Oracle Metalink site, http://metalink.oracle.com.

6

Managing Wallets and Certificates

This appendix explains how to obtain and manage security credentials for Oracle Application Server resources. Security administrators can use Oracle Wallet Manager and its command-line utility, `orapki`, to manage Public Key Infrastructure (PKI) credentials on Oracle clients and servers. These tools create credentials that can be read by Oracle Database, Oracle Application Server, and the Oracle Identity Management infrastructure.

This appendix contains the following topics:

- Using Oracle Wallet Manager
- Performing Certificate Validation and CRL Management With the orapki Utility
- Interoperability With X.509 Certificates

> **Note:** If you already have certificates provisioned, then the following sections may provide all the information you need:
>
> Starting Oracle Wallet Manager
>
> Interoperability With X.509 Certificates

Using Oracle Wallet Manager

This section describes Oracle Wallet Manager, a graphical user interface (GUI) tool used to manage PKI certificates. It contains the following topics:

- Overview of Oracle Wallet Manager
- Starting Oracle Wallet Manager
- Creating a Complete Wallet: Process Overview
- Managing Wallets
- Managing Certificates

Overview of Oracle Wallet Manager

Oracle Wallet Manager is an application used to manage and edit security credentials in Oracle wallets. A wallet is a password-protected container that stores authentication and signing credentials, including private keys, certificates, and trusted certificates, all of which are used by SSL for strong authentication. You can use Oracle Wallet Manager to perform the following tasks:

- Create wallets

- Generate certificate requests
- Open wallets to access PKI-based services
- Save credentials to hardware security modules by using APIs, which comply to Public Key Cryptography Standard (PKCS#11) specification
- Upload wallets to and download them from an LDAP directory
- Import third-party PKCS #12-format wallets to use in an Oracle environment
- Export Oracle wallets to third-party environments

The following sections describe the features of Oracle Wallet Manager:

- Wallet Password Management
- Strong Wallet Encryption
- Microsoft Windows Registry Wallet Storage
- Third-Party Wallet Support
- LDAP Directory Support

Wallet Password Management

Oracle wallets are password-protected. Oracle Wallet Manager includes an enhanced wallet password management module that enforces the following password management policy guidelines:

- Minimum password length: 8 characters
- Maximum password length: Unlimited
- Alphanumeric character mix: Required

Strong Wallet Encryption

Oracle Wallet Manager stores private keys associated with X.509 certificates and uses Triple-DES encryption.

Microsoft Windows Registry Wallet Storage

As an option, Oracle Wallet Manager enables you to store multiple Oracle wallets in the user profile area of the Microsoft Windows system registry or in a Microsoft Windows file management system. Storing your wallets in the registry provides the following benefits:

- Better Access Control: Wallets stored in the user profile area of the registry are accessible only by the associated user. User access controls for the system become, by extension, access controls for the wallets. In addition, when you log out of a system, access to your wallet is effectively prohibited.
- Easier Administration: Because wallets are associated with specific user profiles, no file permissions need to be managed, and the wallets stored in the profile are automatically deleted when the user profile is deleted. Oracle Wallet Manager can be used to create and manage the wallets in the registry.

Oracle Wallet Manager lets you perform the following tasks:

- Open the wallet from the registry
- Save the wallet to the registry
- Save the wallet to a different registry location

- Delete the wallet from the registry
- Open the wallet from the file system and save it to the registry
- Open the wallet from the registry and save it to the file system

Third-Party Wallet Support

Oracle Wallet Manager can use PKI credentials from the following third-party applications:

- Microsoft Internet Explorer 5.0 and later
- Netscape Communicator 4.7.2 and later
- OpenSSL

Browser PKI credential stores (those from Microsoft Internet Explorer and Netscape) hold user certificates, which contain the subject's public key and identifying information, and their associated trusted certificates. To use these credentials, you must export them from the third-party environment and save them in PKCS #12 format. Then, you can use Oracle Wallet Manager to open them for use with SSL.

See Also: Importing Wallets Created with a Third-Party Tool

LDAP Directory Support

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

Directory user entries must be defined and configured in the LDAP directory before Oracle Wallet Manager can be used to upload or download wallets for a user. Oracle Wallet Manager downloads a user wallet by using a simple password-based connection to the LDAP directory. However, for uploads, Oracle Wallet Manager uses an SSL connection, if the open wallet contains a certificate with SSL Oracle PKI certificate usage. If an SSL certificate is not present in the wallet, then password-based authentication is used.

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

See Also:

- Uploading a Wallet to an LDAP Directory
- Downloading a Wallet from an LDAP Directory
- Multiple Certificate Support

Starting Oracle Wallet Manager

To start Oracle Wallet Manager:

- On Microsoft Windows, click **Start**, **Programs**, ***ORACLE_HOME***, **Network Administration**, and then **Wallet Manager**.

- On UNIX, type `owm` at the command prompt.

Creating a Complete Wallet: Process Overview

A wallet is a necessary repository in which you can securely store user certificates and the trust points needed to validate the certificates of peers.

The following steps provide an overview of the complete wallet creation process:

1. Use Oracle Wallet Manager to create a new wallet:
   - Refer to Guidelines for Creating Wallet Passwords for information about creating a wallet password
   - Refer to Creating a Wallet for information about creating standard wallets (store credentials on your file system) and hardware security module wallets.
2. Generate a certificate request. When you create a wallet with Oracle Wallet Manager, the tool automatically prompts you to create a certificate request. Refer to Adding a Certificate Request for information about creating a certificate request.
3. Send the certificate request to the Certificate Authority (CA) you want to use. You can copy and paste the certificate request text into an e-mail message, or you can export the certificate request to a file. Refer to Exporting a User Certificate Request. The certificate request becomes part of the wallet and must remain there until you remove its associated certificate.
4. When the CA sends your signed user certificate and its associated trusted certificate, you can import these certificates in the following order:
   a. Import the trusted certificate of the CA into the wallet. Refer to Importing a Trusted Certificate. This step is optional if the new user certificate has been issued by one of the CAs whose trusted certificate is already present in Oracle Wallet Manager by default.
   b. After you have successfully imported the trusted certificate, import the user certificate that the CA sent to you into your wallet. Refer to Importing the User Certificate Into the Wallet.

   Note: User certificates and trusted certificates in the PKCS #7 format can be imported at the same time.
5. Set the Auto Login feature for the wallet. Refer to Using Auto Login. This step is optional.

   Typically, this feature, which enables PKI-based access to services without a password, is required for most wallets. It is required for database server and client wallets. It is only optional for products that take the wallet password at the time of startup.

After completing the preceding steps, you have a wallet that contains a user certificate and its associated trust points.

Managing Wallets

This section describes how to create a wallet and perform associated wallet management tasks in the following subsections:

- Guidelines for Creating Wallet Passwords

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

Guidelines for Creating Wallet Passwords

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

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

Caution: It is strongly recommended that you avoid choosing easily guessed passwords based on user names, phone numbers, or government identification numbers, such as `admin0`, `oracle1`, or `2135551212A`. This prevents a potential attacker from using personal information to deduce user passwords. It is also a prudent security practice for you to change your passwords periodically, such as once in a month or once in three months.

When you change passwords, you must regenerate Auto Login wallets.

See Also:

- Wallet Password Management
- Using Auto Login

Creating a Wallet

You can use Oracle Wallet Manager to create PKCS #12 wallets (the standard default wallet type) that store credentials in a directory on your file system. It can also be used to create PKCS #11 wallets that store credentials on a hardware security module for servers, or private keys on tokens for clients. The following sections explain how to create both types of wallets by using Oracle Wallet Manager:

- Creating a Standard Wallet

- Creating a Wallet to Store Hardware Security Module Credentials

Creating a Standard Wallet

Unless you have a hardware security module (a PKCS #11 device), you should use a standard wallet that stores credentials in a directory on your file system.

To create a standard wallet:

1. From the **Wallet** menu, select **New**. The New Wallet dialog box is displayed.
2. Follow the guidelines specified in Guidelines for Creating Wallet Passwords and enter a password in the Wallet Password field. This password protects unauthorized use of your credentials.
3. Reenter that password in the **Confirm Password** field.
4. Select **Standard** from the Wallet Type list.
5. Click **OK** to continue. If the entered password does not conform to the required guidelines, then the following message is displayed:

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

6. An alert that informs you that a new empty wallet has been created appears. It prompts you to choose whether you want to add a certificate request. Refer to Adding a Certificate Request.

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

7. From the **Wallet** menu, select **Save In System Default** to save the new wallet.

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

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

Creating a Wallet to Store Hardware Security Module Credentials

To create a wallet to store credentials on a hardware security module that complies with PKCS #11:

1. From the **Wallet** menu, select **New**. The New Wallet dialog box is displayed.
2. Follow the guidelines specified in Guidelines for Creating Wallet Passwords and enter a password in the Wallet Password field.
3. Reenter that password in the **Confirm Password** field.
4. Choose **PKCS11** from the Wallet Type list, and click **OK** to continue. The New PKCS11 Wallet dialog box is displayed.
5. Choose a vendor name from the Select Hardware Vendor list.

Note: In the current release of Oracle Wallet Manager, only nCipher hardware has been certified to interoperate with Oracle wallets.

---

6. In the PKCS11 library filename field, enter the path to the directory in which the PKCS11 library is stored, or click **Browse** to find it by searching the file system.
7. Enter the SmartCard password, and click **OK**.

   The SmartCard password, which is different from the wallet password, is stored in the wallet.
8. An alert that informs you that a new empty wallet has been created appears. It prompts you to decide whether you want to add a certificate request. Refer to Adding a Certificate Request.

   If you click **No**, then you are redirected to the Oracle Wallet Manager main window. The new wallet you just created is displayed in the left pane. The certificate has a status of [Empty], and the wallet displays its default trusted certificates.
9. From the **Wallet** menu, select **Save In System Default** to save the new wallet.

   If you do not have permission to save the wallet in the system default, then you can save it to another location.

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

Note: If you change the SmartCard password or move the PKCS #11 library, then an error message appears when you try to open the wallet. Then, you are prompted to enter the new SmartCard password or the new path to the library.

Opening an Existing Wallet

To open a wallet that already exists in the file system directory:

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

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

Closing a Wallet

To close an open wallet in the currently selected directory, select **Close** from the Wallet menu.

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

Exporting Oracle Wallets to Third-Party Environments

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

To export a wallet to third-party environments:

1. Use Oracle Wallet Manager to save the wallet file.

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

Note:

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

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

You can export a wallet to text-based PKI format if you want to put a wallet into a tool that does not support PKCS #12. Individual components are formatted according to the standards listed in Table 6–1. Within the wallet, only those certificates with SSL key usage are exported with the wallet.

To export a wallet to text-based PKI format:

1. From the **Operations** menu, select **Export Wallet...**. The Export Wallet dialog box is displayed.
2. Enter the destination file system directory for the wallet, or navigate to the directory structure under Folders.
3. Enter the destination file name for the wallet.
4. Click **OK** to return to the main window.

Table 6–1 PKI Wallet Encoding Standards

Component	Encoding Standard
Certificate chains	X.509v3
Trusted certificates	X.509v3
Private keys	PKCS #8

Uploading a Wallet to an LDAP Directory

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

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

To upload a wallet:

1. From the **Wallet** menu, select **Upload Into The Directory Service...**. If the currently open wallet has not been saved, then the following message is displayed:

   Wallet needs to be saved before uploading.

   Click **Yes** to proceed.

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

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

     If the connection fails, then you are prompted for the directory password of the specified DN. Oracle Wallet Manager attempts to connect to the LDAP directory server using this password and displays a warning message if the attempt fails. Otherwise, Oracle Wallet Manager displays a status message at the bottom of the window indicating that the upload was successful.

Downloading a Wallet from an LDAP Directory

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

See Also:

- Saving Changes
- Saving an Open Wallet to a New Location
- Saving a Wallet in System Default

To download a wallet from an LDAP directory:

1. From the **Wallet** menu, select **Download From The Directory Service...**.
2. A dialog box prompts for your DN, the LDAP directory password, host name, and port information. Oracle Wallet Manager uses simple password authentication to connect to the LDAP directory.

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

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

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

Saving Changes

To save your changes to the current open wallet, select **Save** from the Wallet menu.

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

Saving an Open Wallet to a New Location

To save open wallets to a new location:

1. From the **Wallet** menu, select **Save As...**. The Select Directory dialog box is displayed.
2. Select a directory location to save the wallet.
3. Click **OK**.

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

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

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

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

Saving a Wallet in System Default

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

From the **Wallet** menu, select **Save In System Default**.

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

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

Note:

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

Deleting a Wallet

To delete an open wallet:

1. From the **Wallet** menu, select **Delete**. The Delete Wallet dialog box is displayed.
2. Review the displayed wallet location to confirm that you are deleting the correct wallet.
3. Enter the wallet password.

4. Click **OK**. A dialog box is displayed that informs you that the wallet was successfully deleted.

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

Changing the Password

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

> **Note:** If you are using a wallet with Auto Login enabled, then you must regenerate the Auto Login wallet after changing the password. Refer to Section , "Using Auto Login".

To change the password for the current open wallet:

1. From the **Wallet** menu, select **Change Password**. The Change Wallet Password dialog box is displayed.
2. Enter the existing wallet password.
3. Enter the new password.
4. Reenter the new password.
5. Click **OK**.

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

See Also:

- Guidelines for Creating Wallet Passwords
- Wallet Password Management

Using Auto Login

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

You must enable Auto Login if you want single sign-on access to multiple Oracle databases, which is disabled by default. Sometimes these are called SSO wallets because they provide single sign-on capability.

Enabling Auto Login

To enable Auto Login select **Auto Login** from the **Wallet** menu. A message at the bottom of the window indicates that Auto Login is enabled.

Disabling Auto Login

To disable Auto Login, clear Auto Login from the **Wallet** menu. A message at the bottom of the window indicates that Auto Login is disabled.

Managing Certificates

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

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

- Managing User Certificates
- Managing Trusted Certificates

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

Managing User Certificates

User certificates can be used by end users, smart cards, or applications, such as Web servers. Server certificates are a type of user certificate. For example, if a CA issues a certificate for a Web server by placing its DN in the Subject field, then the Web server is the certificate owner or the user for this user certificate. User certificates do not validate other user certificates, except when they are used as a trusted certificate in a user-centric trust model.

> **See Also:** *Understanding Public-Key Infrastructure: Concepts, Standards, and Deployment Considerations* by Carlisle Adams and Steve Lloyd, a third-party publication, for a discussion of user-centric and other trust models.

Managing user certificates involves the following tasks:

- Adding a Certificate Request
- Importing the User Certificate Into the Wallet
- Importing Certificates Created with a Third-Party Tool
- Removing a User Certificate from a Wallet
- Removing a Certificate Request
- Exporting a User Certificate
- Exporting a User Certificate

Adding a Certificate Request

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

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

To create a PKCS #10 certificate request:

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

See Also: Exporting a User Certificate Request

Table 6–2 Certificate Request: Fields and Descriptions

Field Name	Description
Common Name	Mandatory. Enter the name of the identity of the user or service identity. Enter a user name in `firstname.lastname` format. Example: Eileen.Sanger
Organizational Unit	Optional. Enter the name of the organizational unit of the identity. Example: Finance.
Organization	Optional. Enter the name of the organization of the identity. Example: XYZ Corp.
Locality/City	Optional. Enter the name of the locality or city in which the identity resides.
State/Province	Optional. Enter the full name of the state or province in which the identity resides. Enter the full state name, because some certificate authorities do not accept two-letter abbreviations.
Country	Mandatory. Select to view a list of country abbreviations. Choose the country in which the organization is located.
Key Size	Mandatory. Select to view a list of key sizes to use when creating the **public/private key pair**. Refer to Table 6–3 to evaluate key sizes.
Advanced	Optional. Select **Advanced** to view the Advanced Certificate Request dialog box. Use this dialog box to edit or customize the identity's DN. For example, you can edit the full state name and locality.

Table 6–3 lists the available key sizes and the relative security each size provides. Typically, CAs use key sizes of 1024 or 2048 bits. When certificate owners wish to keep their keys for a longer duration, they choose 3072 or 4096 bit keys.

Table 6–3 Available Key Sizes

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

Importing the User Certificate Into the Wallet

The CA sends you an e-mail notification when your certificate request has been fulfilled. Import the certificate into a wallet in either of two ways, copy and paste the certificate from the e-mail of the CA, or import the user certificate from a file. CAs may send your certificate in a PKCS #7 certificate chain file, or as an individual X.509 certificate. Oracle Wallet Manager can import both types. PKCS #7 certificate chains are a collection of certificates, including the user's certificate and all of the supporting CA and subCA certificates. In contrast, an X.509 certificate file contains an individual certificate without the supporting certificate chain.

To copy and paste the text only (BASE64) user certificate from the e-mail of the CA:

1. Copy the certificate text from the e-mail message or file you receive from the CA. Include the lines `Begin Certificate` and `End Certificate.`
2. From the **Operations** menu, select **Import User Certificate...**. The Import Certificate dialog box is displayed.
3. Select **Paste the certificate**, and then click **OK**. Another Import Certificate dialog box is displayed with the following message:

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

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

To import a file that contains the user certificate:

1. From the **Operations** menu, select **Import User Certificate...**. The Import Certificate dialog box is displayed.
2. Choose **Select a file that contains the certificate**, and click **OK**. Another Import Certificate dialog box is displayed.
3. Enter the path or folder name of the certificate file location.
4. Select the name of the certificate file (for example, `cert.txt`).
5. Click **OK**. A message at the bottom of the window confirms that the certificate was successfully installed. You are redirected to the Oracle Wallet Manager main window, and the status of the corresponding entry in the left panel subtree changes to [Ready].

> **Note:** The file containing the user certificate should have been saved in either text (BASE64) or binary (`der`) format.

Importing Certificates Created with a Third-Party Tool

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

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

To import a certificate created with a third-party tool, you must first export it from the application you are using, and then save it as a wallet file that can be read by Oracle Wallet Manager. Refer to Importing Wallets Created with a Third-Party Tool for information about importing certificates that are created with third-party tools.

Removing a User Certificate from a Wallet

To remove a user certificate from a wallet:

1. In the left panel, select the certificate that you want to remove.
2. From the **Operations** menu, select **Remove User Certificate...**. A dialog box is displayed and it prompts you to verify that you want to remove the user certificate from the wallet.
3. Click **Yes** to return to the Oracle Wallet Manager main panel. The certificate displays a status of [Requested].

Removing a Certificate Request

You must remove a certificate before removing its associated request.

To remove a certificate request:

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

Exporting a User Certificate

To save the certificate in a file system directory, export the certificate, as follows:

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

> **See Also:** Exporting Oracle Wallets to Third-Party Environments for information about exporting wallets. Note that Oracle Wallet Manager supports storing multiple certificates in a single wallet, yet current browsers typically support only single-certificate wallets. For these browsers, you must export an Oracle wallet that contains a single key-pair.

Exporting a User Certificate Request

To save the certificate request in a file system directory, export the certificate request, as follows:

1. In the left panel, select the certificate request that you want to export.
2. From the **Operations** menu, select **Export Certificate Request...**. The Export Certificate Request dialog box is displayed.

3. Enter the file system directory location in which you want to save your certificate request, or navigate to the directory structure under Folders.
4. Enter a file name for your certificate request in the **Enter File Name** field.
5. Click **OK**. A message at the bottom of the window confirms that the certificate request was successfully exported to the file. You are redirected to the Oracle Wallet Manager main window.

Managing Trusted Certificates

Managing trusted certificates includes the following tasks:

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

Importing a Trusted Certificate

You can import a trusted certificate into a wallet either by pasting the trusted certificate from an e-mail that you receive from the CA, or importing the trusted certificate from a file.

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

To copy and paste the text-only (BASE64) trusted certificate:

1. Copy the trusted certificate from the body of the e-mail message you received that contained the user certificate. Include the lines `Begin Certificate` and `End Certificate.`
2. From the **Operations** menu, select **Import Trusted Certificate...**. The Import Trusted Certificate dialog box is displayed.
3. Select **Paste the Certificate** and click **OK**. Another Import Trusted Certificate dialog box is displayed with the following message:

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

4. Paste the certificate into the window and click **OK**. A message at the bottom of the window informs you that the trusted certificate was successfully installed.
5. Click **OK**. You are redirected to the Oracle Wallet Manager main panel, and the trusted certificate is displayed at the bottom of the Trusted Certificates list.

To import a file that contains the trusted certificate:

1. From the **Operations** menu, select **Import Trusted Certificate...**. The Import Trusted Certificate dialog box is displayed.
2. Enter the path or folder name of the trusted certificate location.
3. Select the name of the trusted certificate file (for example, `cert.txt)`.
4. Click **OK**. A message at the bottom of the window informs you that the trusted certificate was successfully imported into the wallet.
5. Click **OK** to exit the dialog box. You are redirected to the Oracle Wallet Manager main window, and the trusted certificate is displayed at the bottom of the Trusted Certificates list.

Note: The file containing the trusted certificate should have been saved in either text (BASE64) or binary (`der`) format.

Removing a Trusted Certificate

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

To remove a trusted certificate from a wallet:

1. Select the trusted certificate listed in the Trusted Certificates list.
2. From the **Operations** menu, select **Remove Trusted Certificate...**.

   A dialog box warns you that your user certificate will no longer be verifiable by its recipients if you remove the trusted certificate that was used to sign it.
3. Click **Yes**. The selected trusted certificate is removed from the Trusted Certificates list.

Exporting a Trusted Certificate

To export a trusted certificate to another file system location:

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

Exporting All Trusted Certificates

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

1. From the **Operations** menu, select **Export All Trusted Certificates...**. The Export Trusted Certificate dialog box is displayed.
2. Enter a file system directory location in which you want to save your trusted certificates, or navigate to the directory structure under Folders.
3. Enter a file name to save your trusted certificates.
4. Click **OK**. You are redirected to the Oracle Wallet Manager main window.

Performing Certificate Validation and CRL Management With the orapki Utility

The `orapki` utility is a command-line tool that you can use to manage certificate revocation lists (CRLs), create and manage Oracle wallets, and create signed certificates for testing purposes.

The following sections describe this tool and how to use it:

- Overview of orapki
- Displaying orapki Help

Overview of orapki

The `orapki` utility is provided to manage PKI elements, such as wallets and certificate revocation lists, on the command line so that the tasks it performs can be incorporated into scripts. This enables you to automate many of the routine tasks of maintaining a PKI.

This command-line utility can be used to perform the following tasks:

- Create signed certificates for testing purposes
- Manage Oracle wallets:
  - Create and display Oracle wallets
  - Add and remove certificate requests
  - Add and remove certificates
  - Add and remove trusted certificates
- Manage CRLs:
  - Rename CRLs with a hash value for certificate validation
  - Upload, list, view, and delete CRLs in Oracle Internet Directory

orapki Utility Syntax

The basic syntax of the `orapki` command-line utility is:

```
orapki module command -parameter value
```

In the preceding command, *`module`* can be `wallet` (Oracle wallet), `crl` (certificate revocation list), or `cert` (PKI digital certificate). The available commands depend on the `module` you are using. For example, if you are working with a `wallet`, then you can add a certificate or a key to the wallet by using the `add` command. The following example adds the user certificate located at `/private/lhale/cert.txt` to the wallet located at `$ORACLE_HOME/wallet/ewallet.p12`:

```
orapki wallet add -wallet $ORACLE_HOME/wallet/ewallet.p12 -user_cert -cert
/private/lhale/cert.txt
```

Displaying orapki Help

You can display all the `orapki` commands that are available for a specific mode by entering the following at the command line:

```
orapki mode help
```

For example, to display all available commands for managing CRLs, enter the following at the command line:

```
orapki crl help
```

Note: Using the `-summary`, `-complete`, or `-wallet` command options is optional. A command will still run if these command options are not specified.

Creating Signed Certificates for Testing Purposes

This command-line utility provides a convenient, lightweight way to create signed certificates for testing purposes. The following syntax can be used to create signed certificates and to view certificates:

To create a signed certificate for testing purposes, use the following command:

```
orapki cert create [-wallet wallet_location] -request certificate_request_location
-cert certificate_location -validity number_of_days [-summary]
```

This command creates a signed certificate from the certificate request. The `-wallet` parameter specifies the wallet containing the user certificate and private key that will be used to sign the certificate request. The `-validity` parameter specifies the number of days, starting from the current date, that this certificate will be valid. Specifying a certificate and certificate request is mandatory for this command.

To view a certificate, use the following command:

```
orapki cert display -cert certificate_location [-summary | -complete]
```

This command enables you to view a test certificate that you have created with `orapki`. You can choose either `-summary` or `-complete`, which determines how much detail the command will display. If you choose `-summary`, then the command will display the certificate and its expiration date. If you choose `-complete`, then it will display additional certificate information, including the serial number and public key.

Managing Oracle Wallets with the orapki Utility

The following sections describe the syntax used to create and manage Oracle wallets with the `orapki` command-line utility. You can use these `orapki` utility `wallet` module commands in scripts to automate the wallet creation process. This section contains the following topics:

- Creating and Viewing Oracle Wallets with the orapki Utility
- Adding Certificates and Certificate Requests to Oracle Wallets with orapki
- Exporting Certificates and Certificate Requests from Oracle Wallets with the orapki Utility

Note: The `-wallet` parameter is mandatory for all `wallet` module commands.

Creating and Viewing Oracle Wallets with the orapki Utility

To create an Oracle wallet, use the following command:

```
orapki wallet create -wallet wallet_location
```

The preceding command prompts you to enter and reenter a wallet password. It creates a wallet in the location specified for `-wallet`.

To create an Oracle wallet with Auto Login enabled, use the following command:

```
orapki wallet create -wallet wallet_location -auto_login
```

The preceding command creates a wallet with Auto Login enabled. This command can also be used to enable Auto Login on an existing wallet. If `wallet_location` already contains a wallet, then Auto Login will be enabled for it. To turn the Auto Login feature off, use Oracle Wallet Manager. Refer to Using Auto Login for details.

> **Note:** For wallets with the Auto Login feature enabled, you are prompted for a password only for operations that modify the wallet, such as `add`.

To view an Oracle wallet, use the following command:

```
orapki wallet display -wallet wallet_location
```

The preceding command displays the certificate requests, user certificates, and trusted certificates contained in the wallet.

Adding Certificates and Certificate Requests to Oracle Wallets with orapki

To add a certificate request to an Oracle wallet, use the following command:

```
orapki wallet add -wallet wallet_location -dn user_dn -keySize 512|1024|2048
```

The preceding command adds a certificate request to a wallet for the user with the specified DN, `user_dn`. You can also specify the key size of the requested certificate: 512, 1024, or 2048 bits. To sign the request, export it with the export option. Refer to Exporting Certificates and Certificate Requests from Oracle Wallets with the orapki Utility.

To add a trusted certificate to an Oracle wallet, use the following command:

```
orapki wallet add -wallet wallet_location -trusted_cert -cert
certificate_location
```

This command adds a trusted certificate, at `-cert certificate_location`, to a wallet. You must add all trusted certificates in the certificate chain of a user certificate before adding a user certificate, or the command to add the user certificate will fail.

To add a root certificate to an Oracle wallet, use the following command:

```
orapki wallet add -wallet wallet_location -dn certificate_dn -keySize
512|1024|2048 -self_signed -validity number_of_days
```

The preceding command creates a new self-signed root certificate and adds it to the wallet. The `-validity` parameter, which is mandatory, specifies the number of days, starting from the current date, that this certificate will be valid. You can specify a key size for this root certificate: 512, 1024, or 2048 bits.

To add a user certificate to an Oracle wallet, use the following command:

```
orapki wallet add -wallet wallet_location -user_cert -cert certificate_location
```

The preceding command adds the user certificate at the location specified with the `-cert` parameter to the Oracle wallet at `wallet_location`. Before you add a user certificate to a wallet, you must add all the trusted certificates that make up the certificate chain. If all trusted certificates are not installed in the wallet before you add the user certificate, then adding the user certificate fails.

Exporting Certificates and Certificate Requests from Oracle Wallets with the orapki Utility

To export a certificate from an Oracle wallet, use the following command:

```
orapki wallet export -wallet wallet_location -dn certificate_dn -cert certificate_
filename
```

The preceding command exports a certificate with the subject's DN from a wallet to a file that is specified by `-cert`.

To export a certificate request from an Oracle wallet, use the following command:

```
orapki wallet export -wallet wallet_location -dn certificate_request_dn -request
certificate_request_filename
```

The preceding command exports a certificate request with the subject's DN from a wallet to a file that is specified by `-request`.

Managing Certificate Revocation Lists (CRLs) with the orapki Utility

CRLs must be managed with the `orapki utility`. This utility creates a hashed value of the CRL issuer's name to identify the location of the CRL in your system. If you do not use `orapki`, then your Oracle server cannot locate CRLs to validate PKI digital certificates. The following sections describe CRLs, how you use them, and how to use the `orapki` utility to manage them:

- Certificate Validation with Certificate Revocation Lists
- Certificate Revocation List Management

Certificate Validation with Certificate Revocation Lists

The process of determining whether a given certificate can be used in a given context is referred to as certificate validation. Certificate validation includes determining that:

- A trusted CA has digitally signed the certificate.
- The certificate's digital signature corresponds to the independently calculated hash value of the certificate itself and the public key of the CA.
- The certificate has not expired.
- The certificate has not been revoked.

The SSL network layer automatically performs the first three validation checks, but you must configure CRL checking to ensure that certificates have not been revoked. CRLs are signed data structures that contain a list of revoked certificates. They are usually issued and signed by the same entity who issued the original certificate.

What CRLs Should You Use?

You should have CRLs for all of the trust points that you honor. The trust points are the trusted certificates from a third-party identity that is qualified with a level of trust. Typically, the certificate authorities you trust are called trust points.

How does CRL Checking Work?

Certificate revocation status is checked against CRLs which are located in file system directories, Oracle Internet Directory, or downloaded from the location specified in the CRL Distribution Point (CRL DP) extension on the certificate. If you store your CRLs on the local file system or in the directory, then you must update them regularly. If you

use CRL DPs, then CRLs are downloaded each time a certificate is used so there is no need to regularly refresh the CRLs.

The server searches for CRLs in the following locations, in the order listed subsequently. When the system finds a CRL that matches the DN of the certificate CA, it stops searching.

1. Local file system

   The system checks the `sqlnet.ora` file for the `SSL_CRL_FILE` parameter first, followed by the `SSL_CRL_PATH` parameter. If these two parameters are not specified, then the system checks the wallet location for any CRLs.

   Note: If you store CRLs on your local file system, then you must use the `orapki` utility to periodically update them. Refer to Renaming CRLs with a Hash Value for Certificate Validation.

2. Oracle Internet Directory

   If the server cannot locate the CRL on the local file system and directory connection information has been configured in the `ORACLE_HOME/ldap/admin/ldap.ora` file, then the server searches in the directory. It searches the CRL subtree by using the DN) of the CA and the DN of the CRL subtree.

   The server must have a properly configured `ldap.ora` file to search for CRLs in the directory. It cannot use the Domain Name System (DNS) discovery feature of Oracle Internet Directory. In addition, if you store CRLs in the directory, then you must use the `orapki` utility to periodically update them. Refer to Uploading CRLs to Oracle Internet Directory.

3. CRL DP

   If the CA specifies a location in the CRL DP X.509, version 3, certificate extension when the certificate is issued, then the CRL that contains revocation information for that certificate is downloaded. Currently, Oracle Advanced Security supports downloading CRLs over HTTP and LDAP.

   Notes:

   - For performance reasons, only user certificates are checked.
   - Oracle recommends that you store CRLs in the directory rather than the local file system.

Certificate Revocation List Management

Before you enable certificate revocation status checking, you must ensure that the CRLs you receive from the CAs you use are in a form (renamed with a hash value) or in a location (uploaded to the directory) in which your system can use them. Oracle Advanced Security provides a command-line utility, `orapki`, that you can use to perform the following tasks:

- Renaming CRLs with a Hash Value for Certificate Validation
- Uploading CRLs to Oracle Internet Directory
- Listing CRLs Stored in Oracle Internet Directory
- Viewing CRLs in Oracle Internet Directory

- Deleting CRLs from Oracle Internet Directory

> **Note:** CRLs must be updated at regular intervals (before they expire) for successful validation. You can automate this task by using `orapki` commands in a script.

You can also use LDAP command-line tools to manage CRLs in Oracle Internet Directory.

Renaming CRLs with a Hash Value for Certificate Validation

When the system validates a certificate, it must locate the CRL issued by the CA who created the certificate. The system locates the CRL by matching the issuer name in the certificate with the issuer name in the CRL.

When you specify a CRL storage location for the Certificate Revocation Lists Path field in Oracle Net Manager (sets the `SSL_CRL_PATH` parameter in the `sqlnet.ora` file), use the `orapki` utility to rename CRLs with a hash value that represents the issuer's name. Creating the hash value enables the server to load the CRLs.

On UNIX operating systems, `orapki` creates a symbolic link to the CRL. On Microsoft Windows operating systems, it creates a copy of the CRL file. In either case, the symbolic link or the copy created by `orapki` are named with a hash value of the issuer's name. Then, when the system validates a certificate, the same hash function is used to calculate the link (or copy) name so that the CRL can be loaded.

Depending on your operating system, enter one of the following commands to rename CRLs stored in the file system.

To rename CRLs stored in UNIX file systems:

```
orapki crl hash -crl crl_filename [-wallet wallet_location] -symlink crl_directory
[-summary]
```

To rename CRLs stored in Microsoft Windows file systems:

```
orapki crl hash -crl crl_filename [-wallet wallet_location] -copy crl_directory
[-summary]
```

In the preceding commands, `crl_filename` is the name of the CRL file, `wallet_location` is the location of a wallet that contains the certificate of the CA that issued the CRL, and `crl_directory` is the directory in which the CRL is located.

Using `-wallet` and `-summary` are optional. Specifying `-wallet` causes the tool to verify the validity of the CRL against the certificate of the CA prior to renaming the CRL. Specifying the `-summary` option causes the tool to display the CRL issuer's name.

Uploading CRLs to Oracle Internet Directory

Publishing CRLs in the directory enables CRL validation throughout your enterprise, eliminating the need for individual applications to configure their own CRLs. All applications can use the CRLs stored in the directory in which they can be centrally managed, reducing the administrative overhead of CRL management and use.

You must be a member of the directory group `CRLAdmins` (`cn=CRLAdmins,cn=groups,%s_OracleContextDN%`) to upload CRLs to the directory by using `orapki`. This is a privileged operation because these CRLs are accessible to the entire enterprise. Contact your directory administrator to be added to this administrative directory group.

To upload CRLs to the directory, enter the following at the command line:

```
orapki crl upload -crl crl_location -ldap hostname:ssl_port -user username
[-wallet wallet_location] [-summary]
```

In the preceding command, *crl_location* is the file name or URL in which the CRL is located, *hostname* and *ssl_port* (SSL port with no authentication) are the host name and SSL port of the system on which your directory is installed, *username* is the directory user who has permission to add CRLs to the CRL subtree, and `wallet_location` is the location of a wallet that contains the certificate of the CA that issued the CRL.

Using `-wallet` and `-summary` are optional. Specifying `-wallet` causes the tool to verify the validity of the CRL against the certificate of the CA prior to uploading it to the directory. Specifying the `-summary` option causes the tool to print the CRL issuer's name and the LDAP entry in which the CRL is stored in the directory.

Note:

- The `orapki` utility prompts you for the directory password when you perform this operation.
- Ensure that you specify the directory SSL port on which the Diffie-Hellman–based SSL server is running. This is the SSL port that does not perform authentication. Neither the server authentication nor the mutual authentication SSL ports are supported by the `orapki` utility.

Listing CRLs Stored in Oracle Internet Directory

You can display a list of all CRLs stored in the directory with `orapki`, which lets you browse to locate a particular CRL to view or download to your local system. This command displays the CA who issued the CRL (Issuer) and its location (DN) in the CRL subtree of your directory.

To list CRLs in Oracle Internet Directory, enter the following at the command line:

```
orapki crl list -ldap hostname:ssl_port
```

In the preceding command, *hostname* and *ssl_port* are the host name and SSL port of the system on which your directory is installed. Note that this is the directory SSL port with no authentication, as described in the preceding section.

Viewing CRLs in Oracle Internet Directory

You can view specific CRLs that are stored in Oracle Internet Directory in a summarized format, or you can request a complete listing of revoked certificates for the specified CRL. A summary listing provides the CRL issuer's name and its validity period. A complete listing provides a list of all revoked certificates contained in the CRL.

To view a summary listing of a CRL in Oracle Internet Directory, enter the following at the command line:

```
orapki crl display -crl crl_location [-wallet wallet_location] -summary
```

In the preceding command, *crl_location* is the location of the CRL in the directory. It is convenient to paste the CRL location from the list that is displayed when you use the `orapki crl list` command. Refer to Listing CRLs Stored in Oracle Internet Directory.

To view a list of all revoked certificates contained in a specified CRL, which is stored in Oracle Internet Directory, enter the following at the command line:

```
orapki crl display -crl crl_location [-wallet wallet_location] -complete
```

For example, the following `orapki` command:

```
orapki crl display -crl $T_WORK/pki/wlt_crl/nzcrl.txt -wallet $T_WORK/pki/wlt_crl
-complete
```

produces the following output, which lists the CRL issuer's DN, its publication date, date of its next update, and the revoked certificates it contains:

```
issuer = CN=root,C=us, thisUpdate = Sun Nov 16 10:56:58 PST 2003, nextUpdate = Mon
Sep 30 11:56:58 PDT 2013, revokedCertificates = {(serialNo =
153328337133459399575438325845117876415, revocationDate - Sun Nov 16 10:56:58 PST
2003)}
CRL is valid
```

Using the `-wallet` option causes the `orapki crl display` command to validate the CRL against the certificate of the CA.

Depending on the size of your CRL, choosing the `-complete` option may take a long time to display.

You can also use Oracle Directory Manager, a GUI tool that is provided with Oracle Internet Directory, to view CRLs in the directory. CRLs are stored in the following directory location:

```
cn=CRLValidation,cn=Validation,cn=PKI,cn=Products,cn=OracleContext
```

Deleting CRLs from Oracle Internet Directory

To delete CRLs from the directory by using `orapki`, you must be a member of the directory group `CRLAdmins`. Refer to Uploading CRLs to Oracle Internet Directory for information about this directory administrative group.

To delete CRLs from the directory, enter the following at the command line:

```
orapki crl delete -issuer issuer_name -ldap host:ssl_port -user username
[-summary]
```

In the preceding command, *`issuer_name`* is the name of the CA who issued the CRL, `hostname` and *`ssl_port`* are the host name and SSL port of the system on which your directory is installed, and `username` is the directory user who has permission to delete CRLs from the CRL subtree. Note that this must be a directory SSL port with no authentication. Refer to Uploading CRLs to Oracle Internet Directory for more information about this port.

Using the `-summary` option causes the tool to print the CRL LDAP entry that was deleted.

For example, the following `orapki` command:

```
orapki crl delete -issuer "CN=root,C=us" -ldap machine1:3500 -user cn=orcladmin
-summary
```

produces the following output, which lists the location of the deleted CRL in the directory:

```
Deleted CRL at cn=root
cd45860c.rN,cn=CRLValidation,cn=Validation,cn=PKI,cn=Products,cn=OracleContext
```

orapki Utility Commands

This section lists and describes the following `orapki` commands:

- orapki cert create
- orapki cert display
- orapki crl delete
- orapki crl display
- orapki crl hash
- orapki crl list
- orapki crl upload
- orapki wallet add
- orapki wallet create
- orapki wallet display
- orapki wallet export

orapki cert create

The following sections describe this command.

Purpose

Use this command to create a signed certificate for testing purposes.

Syntax

```
orapki cert create [-wallet wallet_location] -request certificate_request_location
-cert certificate_location -validity number_of_days [-summary]
```

- The `-wallet` parameter specifies the wallet containing the user certificate and private key that will be used to sign the certificate request.
- The `-request` parameter (mandatory) specifies the location of the certificate request for the certificate you are creating.
- The `-cert` parameter (mandatory) specifies the directory location in which the tool places the new signed certificate.
- The `-validity` parameter (mandatory) specifies the number of days, starting from the current date, that this certificate will be valid.

orapki cert display

The following sections describe this command.

Purpose

Use this command to display details of a specific certificate.

Syntax

```
orapki cert display -cert certificate_location [-summary|-complete]
```

- The `-cert` parameter specifies the location of the certificate you want to display.

- You can use either the `-summary` or the `-complete` parameter to display the following information:
  - `-summary` displays the certificate and its expiration date.
  - `-complete` displays additional certificate information, including the serial number and public key.

orapki crl delete

The following sections describe this command.

Purpose

Use this command to delete CRLs from Oracle Internet Directory. Note that the user who deletes CRLs from the directory by using `orapki` must be a member of the `CRLAdmins` (`cn=CRLAdmins,cn=groups,%s_OracleContextDN%`) directory group.

Syntax

```
orapki crl delete -issuer issuer_name -ldap hostname:ssl_port -user username
[-wallet wallet_location] [-summary]
```

- The `-issuer` parameter specifies the name of the CA who issued the CRL.
- The `-ldap` parameter specifies the host name and SSL port for the directory in which the CRLs are to be deleted. Note that this must be a directory SSL port with no authentication. Refer to Uploading CRLs to Oracle Internet Directory for more information about this port.
- The `-user` parameter specifies the user name of the directory user who has permission to delete CRLs from the CRL subtree in the directory.
- The `-wallet` parameter (optional) specifies the location of the wallet that contains the certificate of the certificate authority (CA) who issued the CRL. Using it causes the tool to verify the validity of the CRL against the CA's certificate prior to deleting it from the directory.
- The `-summary` parameter is optional. Using it causes the tool to print the CRL LDAP entry that was deleted.

orapki crl display

The following sections describe this command.

Purpose

Use this command to display specific CRLs that are stored in Oracle Internet Directory.

Syntax

```
orapki crl display -crl crl_location [-wallet wallet_location]
[-summary|-complete]
```

- The `-crl` parameter specifies the location of the CRL in the directory. It is convenient to paste the CRL location from the list that displays when you use the `orapki crl list` command. Refer to orapki crl list.
- The `-wallet` parameter (optional) specifies the location of the wallet that contains the certificate of the certificate authority (CA) who issued the CRL. Using

it causes the tool to verify the validity of the CRL against the CA's certificate prior to displaying it.

- Selecting either the -summary or the -complete parameters displays the following information:
  - `-summary` provides a listing that contains the CRL issuer's name and the CRL's validity period.
  - `-complete` provides a list of all revoked certificates that the CRL contains. Note that this option may take a long time to display, depending on the size of the CRL.

orapki crl hash

The following sections describe this command.

Purpose

Use this command to generate a hash value of the CRL issuer to identify the location of the CRL in your file system for certificate validation.

Syntax

```
orapki crl hash -crl crl_filename|URL [-wallet wallet_location] [-symlink|-copy]
crl_directory [-summary]
```

- The `-crl` parameter specifies the filename that contains the CRL or the URL in which it can be found.
- The `-wallet` parameter (optional) specifies the location of the wallet that contains the certificate of the certificate authority (CA) who issued the CRL. Using it causes the tool to verify the validity of the CRL against the CA's certificate prior to uploading it to the directory.
- Depending on your operating system, use either the `-symlink` or the `-copy` parameter:
  - (UNIX) Use `-symlink` to create a symbolic link to the CRL at the *`crl_directory`* location.
  - (Microsoft Windows) Use `-copy` to create a copy of the CRL at the *`crl_directory`* location.
- The `-summary` parameter (optional) causes the tool to display the CRL issuer's name.

orapki crl list

The following sections describe this command.

Purpose

Use this command to display a list of CRLs stored in Oracle Internet Directory. This is useful for browsing to locate a particular CRL to view or download to your local file system.

Syntax

```
orapki crl list -ldap hostname:ssl_port
```

The `-ldap` parameter specifies the host name and SSL port for the directory server from which you want to list CRLs. Note that this must be a directory SSL port with no

authentication. Refer to Uploading CRLs to Oracle Internet Directory for more information about this port.

orapki crl upload

The following sections describe this command.

Purpose

Use this command to upload CRLs to the CRL subtree in Oracle Internet Directory. Note that you must be a member of the directory administrative group `CRLAdmins` (`cn=CRLAdmins,cn=groups,%s_OracleContextDN%`) to upload CRLs to the directory.

Syntax

```
orapki crl upload -crl crl_location -ldap hostname:ssl_port -user username
[-wallet wallet_location] [-summary]
```

- The `-crl` parameter specifies the directory location or the URL of the CRL that you are uploading to the directory.
- The `-ldap` parameter specifies the host name and SSL port for the directory to which you are uploading the CRLs. Note that this must be a directory SSL port with no authentication. Refer to Uploading CRLs to Oracle Internet Directory for more information about this port.
- The `-user` parameter specifies the user name of the directory user who has permission to add CRLs to the CRL subtree in the directory.
- The `-wallet` parameter specifies the location of the wallet that contains the certificate of the CA who issued the CRL. This is an optional parameter. Using it causes the tool to verify the validity of the CRL against the certificate of the CA certificate prior to uploading it to the directory.
- The `-summary` parameter is also optional. Using it causes the tool to display the CRL issuer's name and the LDAP entry in which the CRL is stored in the directory.

orapki wallet add

The following sections describe this command.

Purpose

Use this command to add certificate requests and certificates to an Oracle wallet.

Syntax

To add certificate requests:

```
orapki wallet add -wallet wallet_location -dn user_dn -keySize 512|1024|2048
```

- The `-wallet` parameter specifies the location of the wallet to which you want to add a certificate request.
- The `-dn` parameter specifies the distinguished name of the certificate owner.
- The `-keySize` parameter specifies the key size for the certificate.
- To sign the request, export it with the export option. Refer to orapki wallet export.

To add trusted certificates, use the following command:

```
orapki wallet add -wallet wallet_location -trusted_cert -cert certificate_location
```

- The `-trusted_cert` parameter causes the tool to add the trusted certificate, at the location specified with `-cert`, to the wallet.

To add root certificates, use the following command:

```
orapki wallet add -wallet wallet_location -dn certificate_dn -keySize
512|1024|2048 -self_signed -validity number_of_days
```

- The `-self_signed` parameter causes the tool to create a root certificate.
- The `-validity` parameter is mandatory. Use this parameter to specify the number of days, starting from the current date, that this root certificate will be valid.

To add user certificates:

```
orapki wallet add -wallet wallet_location -user_cert -cert certificate_location
```

- The `-user_cert` parameter causes the tool to add the user certificate at the location specified with the `-cert` parameter to the wallet. Before you add a user certificate to a wallet, you must add all the trusted certificates that make up the certificate chain. If all trusted certificates are not installed in the wallet before you add the user certificate, then adding the user certificate will fail.

orapki wallet create

The following sections describe this command.

Purpose

Use this command to create an Oracle wallet or to set Auto Login on for an Oracle wallet.

Syntax

```
orapki wallet create -wallet wallet_location [-auto_login]
```

- The `-wallet` parameter specifies a location for the new wallet or the location of the wallet for which you want to turn on Auto Login.
- The `-auto_login` parameter creates an Auto Login wallet, or it turns on automatic login for the wallet specified with the `-wallet` option. Refer to Using Auto Login on page 6-11 for details about Auto Login wallets.

orapki wallet display

The following sections describe this command.

Purpose

Use this command to view the certificate requests, user certificates, and trusted certificates in an Oracle wallet.

Syntax

```
orapki wallet display -wallet wallet_location
```

The `-wallet` parameter specifies a location for the wallet you want to open if it is not located in the current working directory.

orapki wallet export

The following sections describe this command.

Purpose

Use this command to export certificate requests and certificates from an Oracle wallet.

Syntax

To export a certificate from an Oracle wallet, use the following command:

```
orapki wallet export -wallet wallet_location -dn certificate_dn -cert certificate_
filename
```

- The `-wallet` parameter specifies the location of the wallet from which you want to export the certificate.
- The `-dn` parameter specifies the distinguished name of the certificate.
- The `-cert` parameter specifies the name of the file that contains the exported certificate.

To export a certificate request from an Oracle wallet:

```
orapki wallet export -wallet wallet_location -dn certificate_request_dn -request
certificate_request_filename
```

The `-request` parameter specifies the name of the file that contains the exported certificate request.

Interoperability With X.509 Certificates

Oracle Wallet Manager functionality supports users who already have certificates provisioned. If you do not use Oracle Wallet Manager to create certificates, then you can use it to manage and store certificates created previously.

Public Key Cryptography Standards (PKCS) Support

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

Oracle Wallet Manager wallets can be enabled to store credentials on hardware security modules using APIs that conform to the PKCS #11 specification. When `PKCS11` wallet type is chosen at the time of wallet creation, then all keys stored in that wallet are saved to a hardware security module or token, such as smart cards, PCMCIA cards, smart diskettes, or other types of portable hardware devices that store private keys, perform cryptographic operations, or both.

See Also:

- Importing Certificates Created with a Third-Party Tool
- Exporting Oracle Wallets to Third-Party Environments
- Creating a Wallet to Store Hardware Security Module Credentials
- To view PKCS standards documents, navigate to the following URL:

  `http://www.rsasecurity.com/rsalabs/`

Multiple Certificate Support

Oracle Wallet Manager enables you to store multiple certificates for each wallet, supporting the following Oracle PKI certificate usages:

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

Oracle Wallet Manager supports multiple certificates for a single digital entity, where each certificate can be used for a set of Oracle PKI certificate usages, but the same certificate cannot be used for all such usages (Refer to Table 6–4 and Table 6–5 for legal usage combinations). There must be a one-to-one mapping between certificate requests and certificates. The same certificate request can be used to obtain multiple certificates. However, more than one certificate for each certificate request cannot be installed in the same wallet at the same time.

Oracle Wallet Manager uses the X.509 Version 3 `KeyUsage` extension types to define Oracle PKI certificate usages. The key usage extension types are optional bits that can be set in certificates. Setting these bits defines what purpose the key of the certificate can be used for. When certificates are issued, the certificate authority sets these bits according to the type of certificate that you have requested. Table 6–4 lists and describes these key usage types.

Table 6–4 X.509 Version 3 KeyUsage Extension Types, Values, and Descriptions

KeyUsage Extension Type	Value	Description
digitalSignature	0	Used for entity authentication and to authenticate data origin integrity.
nonRepudiation	1	Used to protect against the signing entity falsely denying some action.
keyEncipherment	2	Used when the subject public key is used for key transport.
dataEncipherment	3	Used when the subject public key is used for enciphering data, other than cryptographic keys.
keyAgreement	4	Used when the subject public key is used for key agreement during SSL connection negotiation.
keyCertSign	5	Used when the subject public key is used for verifying a signature on certificates. May only be used in CA certificates.

Table 6–4 (Cont.) X.509 Version 3 KeyUsage Extension Types, Values, and Descriptions

KeyUsage Extension Type	Value	Description
cRLSign	6	Used when the subject public key is used for verifying a signature on certificate revocation lists.
encipherOnly	7	When the encipherOnly bit is asserted, the keyAgreement bit must also be set. When these two bits are set the subject public key may be used only for enciphering data while performing key agreement.
decipherOnly	8	As with the encipherOnly bit, the keyAgreement bit must also be set when decipherOnly is set. When these two bits (decipherOnly and keyAgreement) are set the subject public key may be used only for deciphering data while performing key agreement.

See Also: The Internet Engineering Task Force RFC #2459, *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, for a complete description of the `KeyUsage` extension types at the following URL:

`http://www.ietf.org/rfc/`

When installing a certificate (user certificate or trusted certificate), Oracle Wallet Manager maps the `KeyUsage` extension values to Oracle PKI certificate usages as specified in Table 6–4 and Table 6–5.

Table 6–5 Oracle Wallet Manager Import of Trusted Certificates to an Oracle Wallet

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

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

You should obtain certificates from the certificate authority with the correct `KeyUsage` value for the required Oracle PKI certificate usage. A single wallet can contain multiple key pairs for the same usage. Each certificate can support multiple Oracle PKI certificate usages, as indicated by Table 6–4 and Table 6–5. Oracle PKI applications use the first certificate containing the required PKI certificate usage.

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

If you do not have a certificate with SSL usage, then an `ORA-28885` error (`No certificate with required key usage found`) is returned.

Importing Wallets Created with a Third-Party Tool

Oracle has developed wallets to function as secure containers to keep PKI credentials together. Oracle wallets hold user certificates, which contain the subject's public key and identifying information, and their associated trusted certificates. Third-party applications whose PKI credentials interoperate with Oracle Wallet Manager use various ways to organize credentials, but some are not as tightly organized as Oracle wallets are. Some provide the option to export the associated trusted certificates when you export a user certificate, but some do not provide this option. When you import certificates from third-party tools that do not provide the option to include the associated trusted certificates, you must manually add the trusted certificates by using Oracle Wallet Manager.

Oracle Wallet Manager can import and support PKCS #12-format certificates from the following applications, subject to procedures and limitations specific to the program you use:

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

To import a certificate created with a third-party tool, perform the following steps:

1. Follow the procedures for your particular product to export the certificate and its associated trusted certificates.

   If your third-party product does not provide the option to include the trusted certificates, then you must export them separately and save them in either text (BASE64) or binary (`der`) format.

2. Save the exported certificate to a file name according to your operating system in a directory expected by Oracle Wallet Manager.

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

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

3. Use Oracle Wallet Manager to navigate to the directory in which you saved the `ewallet.p12` file and open it to use the PKI credentials it contains.

   If you exported the trusted certificate separately, then you must import the trusted certificate first before you open the `ewallet.p12` file that contains the imported third-party user certificate.

See Also: Importing a Trusted Certificate

7

Enabling SSL in Oracle Collaboration Suite

This chapter provides instructions for enabling and configuring SSL in Oracle Collaboration Suite.

It contains these topics:

Recommended SSL Configurations

The *Oracle Collaboration Suite Deployment Guide* discusses various deployment topologies. It presents sample architectures for Oracle Collaboration Suite installation types. After you have identified the components on which you need to enable SSL, use the instructions in this chapter to configure the components.

SSL Configuration in Oracle Collaboration Suite Infrastructure

To enable SSL on Infrastructure, run the following script on the Infrastructure instance:

```
$ORACLE_HOME/bin/SSLConfigTool -config_w_default -opwd <orcladmin user password>
```

SSL Configuration in Oracle Collaboration Suite Applications

This section contains the following topics:

Running the SSL Script on the Applications Tier

Run the following script on the Applications tier instance:

```
$ORACLE_HOME/bin/midtierSSLConfigTool.<sh|bat> <oid hostname> <oid port> <oid
admin dn> <oid admin password> <http server SSL port> <https> <hostname of the
computer> <True | False>
```

Where:

- `oid hostname` is the host name of the Oracle Internet Directory computer the Applications tier is associated with
- `oid port` is the port that the Oracle Internet Directory server is listening on
- `oid admin dn` is the administrative DN for the Oracle Internet Directory admin user (`cn=orcladmin`)
- `oid admin password` is the password for the Oracle Internet Directory admin user
- `http server SSL port` is the SSL port that was configured during the install. This value can be found in the *`$ORACLE_HOME`*`/install/portlist.ini` file as the `Oracle HTTP Server SSL port` value.
- `https` is the schema that the customer wants to update the Oracle Collaboration Suite service registry entries with
- `hostname of the computer` is the host name that the computer is using with reference to its Oracle Collaboration Suite configuration. This may be the local host name or a load balancer virtual host name, as the case may be.
- `True | False` is the boolean flag that determines the communication to OID
  - `True`: Communication to the Oracle Internet Directory using SSL only
  - `False`: Normal communication to Oracle Internet Directory

Start and Stop OC4J_OCSClient by using the following commands:

```
./opmnctl stopproc process-type=OC4J_OCSClient
./opmnctl startproc process-type=OC4J_OCSClient
```

Enabling SSL in Oracle Mobile Collaboration

Run the `reRegisterSSO.sh` script located at *`$ORACLE_HOME`*`/wireless/bin` to register Oracle Mobile Collaboration. The syntax is

```
./reRegisterSSO.sh host_URL ORACLE_HOME Admin_dn
```

For example:

```
./reRegisterSSO.sh https://host:4443 $ORACLE_HOME cn=orcladmin
```

To update the Oracle Mobile Push Mail configuration to use SSL, run the `mcsutil` script located at *`$ORACLE_HOME`*`/wireless/install`:

```
mcsutil.<sh|bat> -U -H <oldhost> -P <oldport> -N <new host> -W <new port>
```

> **Note:** You need to verify the host and ports before the update is done.

Enabling SSL in Oracle Content Services

You can provide SSL settings after Oracle Content Services has been installed and configured. To do this, perform the steps outlined in Setting Parameters in the Application Server Control for Collaboration Suite and Setting Additional SSL Information.

This section contains the following topics:

Setting Parameters in the Application Server Control for Collaboration Suite

Use the Application Server Control for Collaboration Suite to set server configuration properties, as follows:

1. From the Collaboration Suite Home page, click the name of the Oracle Content Services domain. The Content Services Home page appears.
2. Click **Domain Properties** (under the Administration heading).
3. In the Properties section, select **IFS.DOMAIN.APPLICATION.Application UseHttps** and click **Edit**.
4. Set **Value** to `True` and click **OK**.
5. Select **IFS.DOMAIN.APPLICATION.ApplicationPort** and click **Edit**.
6. Set the **Value** to be one of the following:
   - If you are using OracleAS Web Cache, then enter the Web Cache SSL port.
   - If you are not using OracleAS Web Cache, then enter the non-Web Cache SSL port.
7. Click **OK** on the Edit Property page.
8. Click **OK** on the Edit Server Configuration page.
9. Restart the Oracle Content Services domain.

See Also: *Oracle Content Services Administrator's Guide*

Setting Additional SSL Information

In addition to setting Oracle Content Services server configuration parameters, you may need to set URLs in OracleAS Portal. Wherever the Oracle Content Services Portlet has been registered in OracleAS Portal, you should update the Oracle Content Services Portlet URLs.

Connecting to Oracle Internet Directory Using SSL

If you did not provide Oracle Internet Directory SSL information during Oracle Content Services configuration but still want to connect to Oracle Internet Directory using SSL, then perform the following steps using the Application Server Control for Collaboration Suite:

1. From the Collaboration Suite Home page, click the name of the Oracle Content Services domain. The Content Services Home page appears.
2. Click **Service Configurations** (under the Configuration heading).
3. Click the name of the service configuration you are using (for example, **SmallServiceConfiguration**). The Edit page appears.
4. In the Properties section, click **IFS.SERVICE.CREDENTIALMANAGER.Oid.OidSsl**. You may need to move to the second or subsequent page to see this property.

5. Set **Value** to `true` and click **OK**.
6. Select **IFS.SERVICE.CREDENTIALMANAGER.Oid.OidUrl** and click **Edit**.
7. Change the port number listed in the URL to be the SSL-enabled Oracle Internet Directory port, typically `636` or `4031`.
8. Click **OK** on the Edit Property page.
9. Click **OK** on the Edit Server Configuration page.
10. Restart the Oracle Content Services domain.

> **See Also:** *Oracle Content Services Administrator's Guide*for more information

Enabling SSL in Oracle Real-Time Collaboration

You can set Oracle Real-Time Collaboration so that all conferences and messages use secure HTTPS connections. To do so, you set the following properties for the Oracle Real-Time Collaboration system.

Perform the following steps on one of your Oracle Real-Time Collaboration core component instances. You need to perform these steps only once, because you will set all other instances by using the `-system true` option.

1. On an Oracle Real-Time Collaboration core components instance, start the `rtcctl` configuration utility:

   $ORACLE_HOME`/imeeting/bin/rtcctl`

2. Set the following property to indicate that the Oracle HTTP Server uses SSL connections:

   ```
   rtcctl> setProperty -system true -pname ApacheProtocolSecure -pvalue true
   ```

   > **Note:** The Oracle Real-Time Collaboration property for the HTTPS port is set at installation. If you change the HTTPS port after installation, then you must also change the `ApacheWebSecurePort` value to match the HTTPS port value. The syntax is:
   >
   > ```
   > rtcctl> setProperty -system true -pname ApacheWebSecurePort -pvalue
   > port_value
   > ```

3. Set the following property to enable SSL and force all instant messaging sessions and Web conferences to use SSL:

   ```
   rtcctl> setProperty -system true -pname RTCSSLSupportEnabled -pvalue true
   ```

   > **Note:** If you do not want to force SSL but just allow it as an option, you can set the `SSLRequiredForMeetings` and `IMSSLRequiredForXMPP` properties to `True` and use the `-force false` option. Refer to the *Oracle Real-Time Collaboration Administrator's Guide* for more details about these properties.

4. Exit `rtcctl` using the following command:

   ```
   rtcctl> exit
   ```

The multiplexer used for Web conferences and the connection manager used for instant messaging on the Oracle Real-Time Collaboration core components system use the same wallet that the Oracle HTTP Server uses. You must set the WalletLocation property on *each* Oracle Real-Time Collaboration core components instance , to identify where the wallet file is located.

1. On the first of your Oracle Real-Time Collaboration core components instances, enter:

```
$ORACLE_HOME/imeeting/bin/rtcctl
rtcctl> setProperty -pname WalletLocation -pvalue "$ORACLE_
HOME/Apache/Apache/conf/ssl.wlt/default"
rtcctl> exit
```

   The default path to the wallet file is shown in the example.

2. Repeat Step 1 on each of your Oracle Real-Time Collaboration core component instances.

For more information about all the properties discussed here, see Chapter 3 in *Oracle Real-Time Collaboration Administrator's Guide*. For more information about using the `rtcctl` utility, see Chapter 4 in the same manual.

Enabling SSL in Oracle Voicemail & Fax

SSL connections are used in two places, when Enterprise Manager connects to Oracle Internet Directory and when the Oracle Voicemail & Fax Applications connect to Oracle Internet Directory.

> **See Also:** *Oracle Internet Directory Administrator's Guide* Administrator's Guide for more information on setting up Oracle Internet Directory for SSL connections

Enterprise Manager Connections

By default, when Enterprise Manager connects to Oracle Internet Directory, SSL is enabled for a target. Because SSL slows down the connection speed, you may want to disable SSL if you are behind a firewall and your network is secure.

To enable or disable SSL for Enterprise Manager connections to Oracle Internet Directory:

1. Log in to Enterprise Manager Grid Control, and navigate to the home page.
2. In the Target Search section, select Oracle Voicemail & Fax in the Search list and click **Go.**
3. Select the Voicemail & Fax target from the All Targets list and click **Configure.**
4. In the Configure Voicemail & Fax: *name_of_Voicemail_&_Fax_target* page, select **Always require SSL for connections** to enable SSL for all connections from Enterprise Manager to Oracle Internet Directory for this target.

   To disable SSL connections for this target, deselect **Always require SSL connections.**
5. Click **Finish.**

Voicemail & Fax Application Connections

You can use an SSL connection when Voicemail & Fax Applications connect to Oracle Internet Directory. This is set in the `%ORACLE_HOME%\config\ias.properties`

file with the SSLONLY setting. By default, this is set as follows: `SSLONLY=false`. Turn SSL on if your network is not very secure or if you want all of your data encrypted.

> **Note:** The SSLONLY setting affects how all applications configured under this Oracle home connect to the Oracle Collaboration Suite Database. Therefore, if other Oracle applications, in addition to Oracle Voicemail & Fax, are installed in this directory, they will be affected by changes to the SSLONLY setting.

Securing Enterprise Manager

Perform the following steps to access the Enterprise Manager using SSL:

1. Shut down Application Server Control for Collaboration Suite by using the following command:

   ```
   ./emctl stop iasconsole
   ```

2. Run the following command:

   ```
   ./emctl secure em
   ```

3. Start the Application Server Control for Collaboration Suite by using the following command:

   ```
   ./emctl start iasconsole
   ```

Enterprise Manager can now be accessed only by using HTTPS.

In case Enterprise Manager is to be accessed again using http, then perform the following steps:

1. Shut down the Application Server Control for Collaboration Suite by using the following command:

   ```
   ./emctl stop iasconsole
   ```

2. Edit *$ORACLE_HOME*`/sysman/config/emd.properties`:

   In `EMD_URL`, change `https` to `http`.

3. Edit *$ORACLE_HOME*`/sysman/j2ee/config/emd-web-site.xml`:

   In the line starting with `<web-site...>`, change `secure=true` to `secure=false`

4. Edit *$ORACLE_HOME*`/sysman/emd/targets.xml`:

   Change the `StandaloneconsoleURL` property for the `oracle_ias` target from `https` to `http`.

5. Start the Application Server Control for Collaboration Suite by using the following command:

   ```
   ./emctl start iasconsole
   ```

Part III

Appendixes

This part contains the following appendixes:

- Appendix A, "System Security and Non-Oracle Components"
- Appendix B, "Troubleshooting SSL Configuration"

A

System Security and Non-Oracle Components

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

- Web Browsers
- Firewalls
- Load Balancers
- Virtual Private Networks

Web Browsers

In the overall system security picture, the Web browser is the component over which e-business sites have least control. For example, when you run a Web storefront, you may not be able to control the browser that customers use. The customer's browser 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.509v3 certificates, and Java.

Most commercially available Web browsers support several of these security-related features. However, you 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 (SSL) protocol to encrypt HTTP transmissions. 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 the server to the client. In this case, 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 the data can be intercepted by a third party. For this reason, the application should encrypt or obfuscate information that is stored in cookies.

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

Firewalls

Firewalls control access between the Internet and the internal network of a corporation. A firewall defines which type 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. So, firewalls cannot protect your application against misuse of permitted communications channels. For example, 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 servers fail. Load balancing has a significant impact on security design, especially on encryption issues. For example, 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

A Virtual Private Network (VPN) allows applications to use the 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. Similar to other network-based security solutions, VPNs cannot prevent the transmission of viruses, nor can they control the content of the information being transmitted.

B

Troubleshooting SSL Configuration

This appendix provides solutions to some problems and errors that you may encounter with your Secure Sockets Layer (SSL) configuration on Oracle Collaboration Suite. This appendix contains the following sections:

Troubleshooting SSL Configuration in OracleAS Portal

This section lists the common errors that you might encounter when accessing OracleAS Portal using the SSL port.

Problem1

WWC-41439 error when trying to access Portal using the 443 SSL port with URL: `https://`*`appstierhostname`*`/pls/portal`. This is a known issue with the `ptlasst.sh` script inserting SSL port 443 into ENABLER tables.

Solution1

Update the tables manually.

For example, log on to SQL*Plus to the `portal` schema:

```
SQL> SELECT LSNR_TOKEN,site_id FROM wwsec_enabler_config_info$;
LSNR_TOKEN SIT E_ID xyz.us.oracle.com:7777 1324 xyz.us.oracle.com:443 1329
SQL> UPDATE wwsec_enabler_config_info$ SET LSNR_TOKEN = 'appstierhost.domain.com'
WHERE site_id = 1329;
SQL> commit;
```

You should now be able to log on to the Applications tier by using the following URL:

`https://`*`appstierhostname`*

Problem 2

The OracleAS portlet could not be contacted. SSL Handshake Failed NZERROR=28858.

> **Note:** You may encounter this error while attempting to access `https://`*`appstierhostname`*`/pls/portal`

Solution 2

Check `$ORACLE_HOME/Webcache/logs/event_log` on the Applications tier. Then, check the accuracy of the entry that you created in the `web.xml` file.

Problem 3

The OracleAS portlet could not be contacted. SSL Handshake Failed NZERROR=28874.

> **Note:** You may still receive NZERROR=28874 while attempting to access `https://`*`appstierhostname`*`/pls/portal`

Solution 3

Check `$$ORACLE_HOME/Webcache/logs/event_log` on the Applications tier. This is a known issue with the `http_client.jar` file.

You can fix this by downloading and applying the patch specified in Metalink Note: 225502.1.

Problem 4

The address from which this authentication request was made does not match your IP address. Notify your administrator if you believe this message to be in error. (WWC-41452)

> **Note:** You might encounter this error when attempting to log in to OracleAS Portal only.

Solution 4

You can disable the IP Check feature by running the following SQL commands:

```
sqlplus portal schema/portal password
SQL>SELECT url_cookie_ip_check FROM wwsec_enabler_config_info$;
```

See if `url_cookie_ip_check` is set to N. Else, run the following commands:

```
SQL>UPDATE wwsec_enabler_config_info$ SET url_cookie_ip_check='N';
SQL>commit;
SQL>exit
```

Problem 5

The style sheet was the only component on the OracleAS Portal page that could not be secured.

Solution 5

This style sheet issue is corrected by the 9.0.4.2.0 patch set.

Problem 6

Incorrect rendering while connecting to `http://`*`appstierhostname.domain`*`.com:443`

The header link in the e-mail portlet will open the following URL:

http://*hostname.domain*.com:443

The URL does not render properly. The first time you access the URL, it may display properly, but subsequent accesses will not display properly.

Solution 6

Modify the `$ORACLE_HOME/j2ee/OC4J_UM/config/oc4j.properties` file on the Applications tier. Set `oracle.mail.client.portlet.HTTPStowebmail=TRUE`. Then, stop and restart the Applications tier by running the following commands:

```
opmnctl stopall
opmnctl startall
```

Troubleshooting SSL Configuration in Oracle Mail

This section lists the common errors that you might encounter when accessing Oracle Mail using the SSL port.

Problem 7

In `Traffic_cop,` if you log in as `orcladmin` or as an administrative user, then you can create new users in Oracle Mail. If the user is not already created in Oracle Internet Directory, then the Oracle WebMail client will prompt you to create the user first and will provide you a link to Oracle Internet Directory Delegated Administration Services. This link is incorrect.

Solution 7

Ensure that the URL for Delegated Administration Services is configured correctly. Clear the OracleAS Portal cache and Oracle Internet Directory cache, as follows:.

To clear the OracleAS Portal cache:

1. Shut down the Applications tier, as follows:

   ```
   opmnctl stopall
   ```

2. Delete the `plsql` and `session` directories in `$ORACLE_HOME/Apache/modplsql/`cache on the Applications tier.
3. Restart the Applications tier, as follows:

   ```
   opmnctl startall
   ```

To clear the Oracle Internet Directory cache:

1. Log in to OracleAS Portal as a portal user.
2. Click **Builder, Admin, Global Settings,** and then the **SSO/OID** tab.
3. Select **Refresh Cache** for Oracle Internet Directory parameters.
4. Click **Apply**.

Problem 8

Browse buttons have OracleAS Single Sign-On warnings.

Solution 8

Refer to Solution 7.

Troubleshooting SSL Configuration in Oracle Real-Time Collaboration

This section lists the common errors that you might encounter when accessing Oracle Real-Time Collaboration using the SSL port.

Problem 9

The Oracle Real-Time Collaboration portlet does not work properly.

When you attempt to access a conference from the main portal site, you are directed to an HTTP error page that displays the message, "Page cannot be found".

Solution 9

This can be fixed by editing the *$ORACLE_HOME*/j2ee/OC4J_UM/config/oc4j.properties file on the Applications tier. Change the oracle.mail.Portlet.httpsToWebmail parameter to TRUE to correct the problem. You may need to restart OC4J_IMEETING.

Troubleshooting SSL Configuration in Oracle Calendar

This section lists the common errors that you might encounter when accessing Oracle Calendar using the SSL port.

Problem 10

Referencing the old OracleAS Single Sign-On site ID.

Solution 10

The Calendar issue may be corrected by changing the httpd.conf file on the Applications tier. Comment out the include line, as shown:

```
# General setup for the virtual host
# include "appstier_install_path/.../Apache/Apache/conf/modosso_https.conf"
```

Glossary

Applications tier

The tier of Oracle Collaboration Suite that runs the server applications that provide specific functionality to end users. The term "Applications tier" replaces the term "middle tier" that was used in previous releases. Each Applications tier corresponds to an instance of Oracle Application Server. See also **Oracle Collaboration Suite Applications**.

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.

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

Unencrypted data in ASCII format.

Client tier

The tier of Oracle Collaboration Suite that consists of the end-user applications that reside on client devices, such as desktops, laptops, wireless phones, and PDAs. See also **Oracle Collaboration Suite Applications**.

decryption

The process of converting the contents of an encrypted message back into its original readable format.

DES

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

demilitarized zone (DMZ)

A set of computers that is 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 computers is viewed as semi-secure. They are protected from the open Internet, but are not completely trusted like computers 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.

digital certificates

See **certificate**.

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**.

firewall

A computer that acts as an intermediary to protect a set of computers or networks from outside attack. It regulates access to computers on a local area network from outside, and regulates access to outside computers from within the local area network. A firewall can work either by acting as a proxy server that forwards requests so that

the requests behave as though they were issued by the firewall computer, or by examining requests and attempting to eliminate suspect calls.

Infrastructure tier

The tier of Oracle Collaboration Suite that consists of the components that provide services, such as identity management and metadata storage, for the **Applications tier**. Components of the Infrastructure tier include **Oracle Collaboration Suite Database** and **Oracle Identity Management**. See also **Oracle Collaboration Suite Infrastructure**.

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)**

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.

MD5

A hashing algorithm intended for use on 32-bit computers 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.

Oracle Collaboration Suite

An integrated suite of software applications to enable communication, messaging, and content sharing in an enterprise environment. At an architectural level, it includes three tiers: an **Applications tier**, which consists of server applications that provide the basic functionality, a **Client tier**, which consists of applications on desktops, laptops, and wireless devices, and an **Infrastructure tier**, which provides centralized services, such as identity management and metadata storage, for the applications.

Oracle Collaboration Suite Applications

The applications that make up Oracle Collaboration Suite, namely:

- Oracle Calendar
- Oracle Collaboration Suite Search
- Oracle Content Services
- Oracle Discussions
- Oracle Mail
- Oracle Mobile Collaboration
- Oracle Real-Time Collaboration

- Oracle Voicemail & Fax
- Oracle Workspaces

Each of the preceding applications is a component of Oracle Collaboration Suite Applications. These applications rely on the services provided by the **Infrastructure tier**. See also **Applications tier**.

Oracle Collaboration Suite Database

The default database included with Oracle Collaboration Suite to hold application data and metadata. The Oracle Collaboration Suite Database is part of the **Oracle Collaboration Suite Infrastructure**.

Oracle Collaboration Suite Infrastructure

The underlying components that support Oracle Collaboration Suite and provide centralized product metadata and security services, configuration information, and data repositories for **Oracle Collaboration Suite Applications**. Oracle Collaboration Suite Infrastructure uses and builds on OracleAS Infrastructure. It includes the **Oracle Collaboration Suite Database** and **Oracle Identity Management**. See also **Infrastructure tier**.

Oracle Identity Management

An integrated set of components that provide distributed security to Oracle products and make it possible to centrally and securely manage enterprise identities and their access to applications in the enterprise. It includes the following components: Oracle Internet Directory, Oracle Directory Integration and Provisioning, Oracle Delegated Administration Services, OracleAS Single Sign-On, and Oracle Application Server Certificate Authority.

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.

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**.

private key

In **public key cryptography**, this key is the secret key. It is primarily used for decryption but is also used for encryption with digital signatures. See **public/private key pair**.

proxy server

A server that typically sits on a network firewall and allows clients behind the firewall to access Web resources. All requests from clients go to the proxy server rather than directly to the destination server. The proxy server forwards the request to the destination server and passes the received information back to the client. The proxy

server channels all Web traffic at a site through a single, secure port; this allows an organization to create a secure firewall by preventing Internet access to internal computers, while allowing Web access.

public key

In **public key cryptography**, this key is made public to all. It is primarily used for encryption but can be used for verifying signatures. See **public/private key pair**.

public key cryptography

Encryption method that uses two different random numbers (**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.

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**.

SSL

See **Secure Sockets Layer (SSL)**

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 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 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 certificates

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.

WebDAV

Web-based Distributed Authoring and Versioning. A protocol extension to HTTP 1.1 that supports distributed authoring and versioning. With WebDAV, the Internet becomes a transparent read and write medium, where content can be checked out, edited, and checked into a URL address.

X.509

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

Index

J

L

M

N

O

P

R

S

T

U

W