Oracle9*i* Application Server

Oracle Wallet Manager User's Guide

Release 1 (v.1.0.2.2)

May 2001

Part No. A90387-01

Oracle9*i* Application Server, Oracle Wallet Manager User’s Guide, Release 1 (v.1.0.2.2)

Part No. A90387-01

Contents

Send Us Your Comments

Oracle9*i* Application Server, Oracle Wallet Manager User’s Guide, Release 1 (v.1.0.2.2)

Part No. A90387-01

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

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

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

- Electronic mail: iasdocs_us@oracle.com
- FAX: (650) 506-7409 Attn: Oracle9*i* Application Server Documentation Manager
- Postal service:
  Oracle Corporation
  Oracle9*i* Application Server Documentation Manager
  500 Oracle Parkway, M/S 2op4
  Redwood Shores, CA 94065
  USA

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

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

Preface

This document contains information about Oracle Wallet Manager and describes how to use it to public-key security credentials on Oracle clients and servers.

This preface contains these topics:

Audience

Oracle Wallet Manager User’s Guide is intended for security administrators who manage public-key security credentials (certificates) on Oracle clients and servers.

To use this document, you need to understand basic security concepts about public-key security credentials.

Organization

This document contains:

Chapter 1, "Using Oracle Wallet Manager"

This chapter provides general concepts related to public-key security credentials management. It also describes how to use Oracle Wallet Manager to manage public-key security certificates.

Glossary

Related Documentation

For more information, see these Oracle resources:

In North America, printed documentation is available for sale in the Oracle Store at

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

Customers in Europe, the Middle East, and Africa (EMEA) can purchase documentation from

`http://www.oraclebookshop.com/`

Other customers can contact their Oracle representative to purchase printed documentation.

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

`http://technet.oracle.com/membership/index.htm`

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

`http://technet.oracle.com/docs/index.htm`

Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- Conventions in Text
- Conventions in Code Examples
- Conventions for Windows Operating Systems

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an **index-organized table**.
Italics	Italic typeface indicates book titles or emphasis.	*Oracle9i Database Concepts* Ensure that the recovery catalog and target database do *not* reside on the same disk.
`UPPERCASE monospace (fixed-width font)`	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles.	You can specify this clause only for a `NUMBER` column. You can back up the database by using the `BACKUP` command. Query the `TABLE_NAME` column in the `USER_TABLES` data dictionary view. Use the `DBMS_STATS.GENERATE_STATS` procedure.

Convention	Meaning	Example
`lowercase monospace (fixed-width font)`	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter `sqlplus` to open SQL*Plus. The password is specified in the `orapwd` file. Back up the datafiles and control files in the `/disk1/oracle/dbs` directory. The `department_id`, `department_name`, and `location_id` columns are in the `hr.departments` table. Set the `QUERY_REWRITE_ENABLED` initialization parameter to `true.` Connect as `oe` user. The `JRepUtil` class implements these methods.
`lowercase monospace (fixed-width font) italic`	Lowercase monospace italic font represents placeholders or variables.	You can specify the *`parallel_clause`*. Run `U`*`old_release`*`.SQL` where *`old_release`* refers to the release you installed prior to upgrading.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[ ]	Brackets enclose one or more optional items. Do not enter the brackets.	`DECIMAL (`*`digits`*` [ , `*`precision`*` ])`
{ }	Braces enclose two or more items, one of which is required. Do not enter the braces.	`{ENABLE | DISABLE}`

|	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	`{ENABLE | DISABLE}`
		
		`[COMPRESS | NOCOMPRESS]`

Convention	Meaning	Example
`. . .`	Horizontal ellipsis points indicate either:	
	■ That we have omitted parts of the code that are not directly related to the example	`CREATE TABLE ... AS` *`subquery`*`;`
	■ That you can repeat a portion of the code	`SELECT` *`col1`*`,` *`col2`*`, ... ,` *`coln`* `FROM employees;`
. . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	

Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	`acctbal NUMBER(11,2);`
		`acct    CONSTANT NUMBER(4) := 3;`

`Italics`	Italicized text indicates placeholders or variables for which you must supply particular values.	`CONNECT SYSTEM/`*`system_password`*
		`DB_NAME =` *`database_name`*

`UPPERCASE`	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.	`SELECT last_name, employee_id FROM employees;`
		`SELECT * FROM USER_TABLES;`
		`DROP TABLE hr.employees;`

`lowercase`	Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files.
	Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.
		`SELECT last_name, employee_id FROM employees;`
		`sqlplus hr/hr`
		`CREATE USER mjones IDENTIFIED BY ty3MU9;`

Conventions for Windows Operating Systems

The following table describes conventions for Windows operating systems and provides examples of their use.

Convention	Meaning	Example
Choose Start >	How to start a program. For example, to start Oracle Database Configuration Assistant, you must click the Start button on the taskbar and then choose Programs > Oracle - *HOME_NAME* > Database Administration > Database Configuration Assistant.	Choose Start > Programs > Oracle - *HOME_NAME* > Database Administration > Database Configuration Assistant
`C:\>`	Represents the Windows command prompt of the current hard disk drive. Your prompt reflects the subdirectory in which you are working. Referred to as the command prompt in this guide.	`C:\oracle\oradata>`
HOME_NAME	Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.	`C:\> net start Oracle`*`HOME_NAME`*`TNSListener`

Convention	Meaning	Example
`ORACLE_HOME` and *`ORACLE_ BASE`*	In releases prior to 8.1, when you installed Oracle components, all subdirectories were located under a top level *`ORACLE_HOME`* directory that by default was: ▪ `C:\orant` for Windows NT ▪ `C:\orawin95` for Windows 95 ▪ `C:\orawin98` for Windows 98 or whatever you called your Oracle home. In this Optimal Flexible Architecture (OFA)-compliant release, all subdirectories are not under a top level *`ORACLE_HOME`* directory. There is a top level directory called *`ORACLE_BASE`* that by default is `C:\oracle`. If you install release 9.0 on a computer with no other Oracle software installed, the default setting for the first Oracle home directory is `C:\oracle\ora90`. The Oracle home directory is located directly under *`ORACLE_BASE`*. All directory path examples in this guide follow OFA conventions. See *Oracle9i Database Getting Starting for Windows* for additional information on OFA compliances and for information on installing Oracle products in non-OFA compliant directories.	Go to the *`ORACLE_BASE`*`\`*`ORACLE_ HOME`*`\rdbms\admin` directory.

1 Using Oracle Wallet Manager

Security administrators use Oracle Wallet Manager to manage public-key security credentials on Oracle clients and servers. The wallets it creates are opened by using either the Oracle Enterprise Login Assistant or the Oracle Wallet Manager.

This chapter describes the Oracle Wallet Manager, in the following sections:

1.1 Overview

Traditional private-key or symmetric-key cryptography requires that entities desiring to establish secure communications possess a single secret key known only to them. *Harriet* and *Dick*, for example, could agree to shift each letter in their private messages by two character positions (A becomes C, B becomes E, and so on) to encrypt the message text. Using this method, a *HELLO* message from Harriet to Dick would read *JGNNP*. The actual encryption methods in current use are much more complex and significantly more secure, but an underlying problem remains—sending messages encrypted with a single key requires prior, *secure* distribution of the key to each participating party. Otherwise, a malicious third party might obtain the key, intercept communications, and compromise security. Public-key cryptography addresses this problem, by providing a secure method for key distribution.

Public-key cryptography requires a party to possess a **public/private key pair**. The **private key** is kept secret and is known only to that party. The **public key**, as the name implies, is freely available. To send a secret message to this party requires that a third party sender encrypt the message with the public key. Such a message can only be decrypted by a party holding the associated private key.

For example, when Dick wants to send a secure message to Harriet, he first asks Harriet for her public key (or obtains it from another, public source). Harriet gives Dick the public key, but Tom, a malicious eavesdropper, also obtains the public key. Nevertheless, when Dick sends Harriet a message encrypted with her public key, Tom cannot decrypt it; the message can only be decrypted with Harriet's private key.

Public-key algorithms thus guarantee the secrecy of a message, but they don't guarantee *secure communications* because they don't verify the identities of the communicating parties. In order to establish secure communications, it is important to verify that the public key used to encrypt a message does in fact belong to the target recipient. Otherwise, a third party can potentially eavesdrop on the communication and intercept public key requests, substituting its public key for a legitimate key.

If Tom, for example, is able to substitute his public key for Harriet's public key and send it to Dick, Dick might then send a message to Harriet encrypted with Tom's public key—believing he was using Harriet's public key. Tom could then decrypt a subsequent intercepted message from Dick using his private key, re-encrypt it with Harriet's public key and re-transmit it to Harriet. Harriet could then decrypt the incoming message using her private key, and never know that it had been intercepted by Tom—the **man-in-the-middle**.

In order to avoid such a man-in-the-middle attack, it is necessary to verify the owner of the public key, a process called **authentication**. Authentication can be accomplished through a **certificate authority** (CA).

A CA is a third party that is trusted by both of the parties attempting secure communication. The CA issues public key certificates that contain an entity's name, public key, and certain other security credentials. Such credentials typically include the CA name, the CA signature, and the certificate effective dates (From Date, To Date).

The CA uses its private key to encrypt a message, while the public key is used to decrypt it, thus verifying that the message was encrypted by the CA. The CA public key is well known, and does not have to be authenticated each time it is accessed. Such CA public keys are stored in an Oracle **wallet**.

1.1.0.1 Wallet Password Management

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

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

1.1.0.2 Strong Wallet Encryption

Oracle Wallet Manager stores private keys associated with X.509 certificates, requiring strong encryption. Accordingly, this release replaces DES encryption with 3-key Triple-DES—a substantially stronger encryption algorithm.

1.1.0.3 Microsoft Windows Registry

Oracle Wallet Manager lets you optionally store multiple Oracle wallets in the user profile area of the Microsoft Windows System Registry (for Windows 95/98/ME/NT 4.0/2000), or in a Windows file management system. Storing your wallets in the registry provides the following benefits:

- **Better Access Control.** Wallets stored in the user profile area of the registry are only accessible by the associated user. User access controls for the system thus become, by extension, access controls for the wallets. In addition, when a user logs out of a system, access to that user's wallets is effectively precluded.
- **Easier Administration.** Since wallets are associated with specific user profiles, no 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, and the wallets are accessible by Oracle Enterprise Login Assistant as well.

- **Improved Security.** Because the wallets are imbedded in the registry, the wallets associated with a particular user profile are transparent to all other users. Viewed in combination with *better access control* and *easier administration*, this amounts to an additional security layer for Oracle wallets.

1.1.0.3.1 Options Supported:

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

See Also:

- *Oracle9i Administrator's Guide for Windows*
- *Oracle9i Network, Directory and Security Guide for Windows*

1.1.0.4 Oracle Wallet Functions

Oracle Wallet Manager is a stand-alone Java application that wallet owners use to manage and edit the security credentials in their Oracle wallets. These tasks include the following:

- Generating a public/private key pair and creating a certificate request for submission to a CA.
- Installing a certificate for the entity.
- Configuring **trusted certificate**s for the entity.
- Opening a wallet to enable access to PKI-based services.
- Creating a wallet that can be accessed by using either Oracle Enterprise Login Assistant or Oracle Wallet Manager.
- Uploading a wallet to an LDAP directory.
- Downloading a wallet from an LDAP directory.

- Importing wallets.
- Exporting wallets.

1.1.0.5 Backward Compatibility

Oracle Wallet Manager is backward-compatible to Oracle Data Server, Release 8.1.5.

1.2 PKCS #12 Support

Oracle Wallet Manager stores X.509 certificates and **private keys** in industry-standard, PKCS #12 format. This makes the Oracle wallet structure interoperable with supported third party PKI applications, and provides wallet portability across operating systems.

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

1.2.1 Importing Third-Party Wallets

Oracle Wallet Manager can import and support the following PKCS #12-format wallets, subject to product-specific procedures and limitations:

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

To import a third-party wallet:

1. Follow the product-specific procedure to export the wallet.
2. Save the exported wallet to a platform-specific file name in a directory expected by your application.

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

   For other platforms, see the platform-specific documentation.

   See Also: Importing a Trusted Certificate on page 1-22.

Notes:

- You must copy the third-party PKCS #12 wallet file name to a directory expected by Oracle Wallet Manager and change the name; the UNIX/NT wallet file name is `ewallet.p12`.
- Since browsers typically do not export **trusted certificate**s under PKCS #12 (other than the signer's own certificate), you may need to add trust points to authenticate the other party in the SSL connection. You can use Oracle Wallet Manager to do this.

1.2.2 Exporting Oracle Wallets

Oracle Wallet Manager can export its own wallets to third party environments. To export a wallet:

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

Note:

- Current browsers typically support import of *only one user certificate per wallet.* Accordingly, you must export an Oracle **single key-pair wallet**—*even though Oracle Wallet Manager supports multiple user certificates per wallet.*

1.3 Multiple Certificate Support

Oracle wallet tools (Oracle Wallet Manager, Enterprise Login Assistant) support multiple **certificate**s for each wallet, supporting the following **Oracle PKI certificate usages**:

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

Oracle Wallet Manager supports multiple certificates for a *single digital entity*, where each certificate can be used for a set of Oracle PKI certificate usages—but the same certificate cannot be used for all such usages (See: Tables 1–2 and 1–3 for legal usage combinations). There must be a one-to-one mapping between certificate requests and certificates. The same certificate request cannot be used to obtain multiple certificates, installed in the same wallet.

Oracle Wallet Manager uses X.509 V3 extension `KeyUsage` to define Oracle PKI certificate usages (Table 1–1):

Table 1–1 KeyUsage Values

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

When installing a certificate (user certificate, **trusted certificate**), Oracle Wallet Manager uses Tables 1–2 and 1–3 to map the KeyUsage extension values to Oracle PKI certificate usages:

Table 1–2 OWM Import of User Certificate to an Oracle Wallet

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

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

Table 1–3 OWM Import of Trusted Certificates to an Oracle Wallet

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

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

You should obtain certificates from the certificate authority with the correct `KeyUsage` value for the required Oracle PKI certificate usage. A single wallet can contain multiple **key pair**s for the same usage. Each certificate can support multiple Oracle PKI certificate usages, as indicated by Tables 1–2 and 1–3. Oracle PKI applications use the first certificate containing the required PKI certificate usage.

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

Note: *SSL Oracle PKI Certificate Usage* is the only usage supported by Oracle PKI applications.

1.4 LDAP Directory Support

Oracle Wallet Manager can upload wallets to—and retrieve them from—an LDAP-compliant directory.

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

Oracle Wallet Manager requires that enterprise users are already defined and configured in the LDAP directory, to be able to upload or download wallets. If a directory contains Oracle8*i* (or prior) users, they are automatically upgraded to use the wallet upload / download feature—upon first use.

See Also: *Oracle Advanced Security Administrator's Guide* for information about creating enterprise users.

Oracle Wallet Manager downloads a user wallet using a simple password-based connection to the LDAP directory. However, for uploads it uses an SSL connection if the open wallet contains a certificate with SSL Oracle PKI certificate usage.

See Also: Multiple Certificate Support on page 1-8, for more information about Oracle PKI certificate user.

If an SSL certificate is not present in the wallet, password-based authentication is used.

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

1.5 Managing Wallets

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

- Starting Oracle Wallet Manager
- Creating a New Wallet
- Opening an Existing Wallet
- Closing a Wallet
- Saving Changes
- Saving the Open Wallet to a New Location
- Saving in System Default
- Deleting the Wallet
- Changing the Password
- Using Auto Login
- LDAP Directory Support

1.5.1 Starting Oracle Wallet Manager

To start Oracle Wallet Manager:

- Microsoft NT: Select `Start—>Programs—>Oracle-OraHome81—>Network Administration—>Wallet Manager`
- UNIX: Enter `owm` at the command line.

1.5.2 Creating a New Wallet

Create a new wallet as follows:

1. Choose `Wallet > New` from the menu bar; the New Wallet dialog box appears.
2. Follow the required guidelines for creating a password and enter a password in the Wallet Password field.

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

Oracle Wallet Manager requires that you choose a password that is at least eight characters long, and contains a mix of alphabetic and numeric or special characters.

It is also a prudent security practice for users to change their passwords periodically, such as once a month or once a quarter.

Example: `gol8fer*`

> **See Also:** Wallet Password Management on page 1-3.

3. Re-enter that password in the Confirm Password field.
4. Choose `OK` to continue.
5. If the entered password does not conform to the required guidelines, the following message appears:

   Password should have a minimum length of 8 characters and should contain both alphabets and either numbers or special characters. Do you want to try again?
6. An Alert is displayed, and informs you that a new empty wallet has been created. It prompts you to decide whether you want to create a certificate request.

   > **See also:** **Adding a Certificate Request** on page 1-18

   If you choose Cancel, you are returned to the Oracle Wallet Manager main window. The new wallet you just created appears in the left window pane. The certificate has a status of `Empty`, and the wallet displays its default trusted certificates.
7. Select `Wallet > Save In System Default` to save the new wallet.

   If you do not have permission to save the wallet in the system default, you can save it to another location.

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

1.5.3 Opening an Existing Wallet

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

1. Choose `Wallet > Open` from the menu bar; the Select Directory dialog box appears.
2. Navigate to the directory location in which the wallet is located, and select the directory.
3. Choose OK; the Open Wallet dialog box appears.
4. Enter the wallet password in the Wallet Password field.
5. Choose OK.
6. The successfully opened message wallet appears at the bottom of the window, and you are returned to the Oracle Wallet Manager main window. The wallet's certificate and its trusted certificates are displayed in the left window pane.
7. The message `Wallet opened successfully` appears at the bottom of the window, and you are returned to the Oracle Wallet Manager main window. The wallet's certificate and its trusted certificates are displayed in the left window pane.

1.5.4 Closing a Wallet

To close an open wallet in the currently selected directory:

- Choose `Wallet > Close`.
- The message `Wallet closed successfully` appears at the bottom of the window, to confirm that the wallet is closed.

1.5.5 Saving Changes

To save your changes to the current open wallet:

- Choose `Wallet > Save`.
- A message at the bottom of the window confirms that the wallet changes were successfully saved to the wallet in the selected directory location.

1.5.6 Saving the Open Wallet to a New Location

Use the `Save As` option to save the current open wallet to a new directory location:

1. Choose `Wallet > Save As;` the select directory dialog box appears.
2. Select a directory location to save the wallet.
3. Choose `OK`.

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

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

   Choose `Yes` to overwrite the existing wallet, or `No` to save the wallet to another directory.

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

1.5.7 Saving in System Default

Use the `Save in System Default` menu option to save the current open wallet to the system default directory location. This makes the current open wallet the wallet that is used by SSL:

- Choose `Wallet > Save in System Default`.
- A message at the bottom of the window confirms that the wallet was successfully saved in the system default wallet location.

1.5.8 Deleting the Wallet

To delete the current open wallet:

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

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

1.5.9 Changing the Password

A password change is effective immediately. The wallet is saved to the currently selected directory, with the new encrypted password.To change the password for the current open wallet:

1. Choose `Wallet > Change Password;` the `Change Wallet Password` dialog box appears.
2. Enter the existing wallet password.
3. Enter the new password.

   See Also: Wallet Password Management on page 1-3, for password policy restrictions.

4. Re-enter the new password.
5. Choose `OK`.

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

1.5.10 Using Auto Login

The Oracle Wallet Manager Auto Login feature opens a copy of the wallet and enables PKI-based access to secure services—as long as the wallet in the specified directory remains open in memory.

You must enable Auto Login if you want single sign-on access to multiple Oracle databases (disabled by default).

1.5.10.1 Enabling Auto Login

To enable Auto Login:

1. Choose `Wallet` from the menu bar.
2. Choose the check box next to the Auto Login menu item; a message at the bottom of the window displays `Autologin enabled`.

1.5.10.2 Disabling Auto Login

To disable Auto Login:

1. Choose `Wallet` from the menu bar.
2. Choose the check box next to the Auto Login menu item; a message at the bottom of the window displays `Autologin disabled`.

1.6 Managing Certificates

Oracle Wallet Manager uses two kinds of certificates: user certificates and trusted certificates. 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.

1.6.1 Managing User Certificates

Managing user certificates involves the following tasks:

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

1.6.1.1 Adding a Certificate Request

You can use this task to add multiple certificate requests. Note that when creating multiple requests, Oracle Wallet Manager automatically populates each subsequent request dialog box with the content of the initial request—which you can then edit.

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

To create a PKCS #10 certificate request:

1. Choose `Operations > Create Certificate Request`; the `Create Certificate Request` dialog box appears.
2. Enter the following information (Table 1–4):

Table 1–4 Certificate Request: Fields and Descriptions

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

3. Choose `OK`. An Oracle Wallet Manager dialog box informs you that a certificate request was successfully created. You can either copy the certificate request text from the body of this dialog panel and paste it into an e-mail message to send to a certificate authority, or you can export the certificate request to a file.
4. Choose `OK`. You are returned to the Oracle Wallet Manager main window; the status of the certificate is changed to `Requested`.

1.6.1.2 Importing the User Certificate into the Wallet

You will receive an e-mail notification from the certificate authority informing you that 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 you receive from the certificate authority, or import the user certificate from a file.

1.6.1.2.1 Pasting the Certificate

To paste the certificate:

1. Copy the certificate text from the e-mail message or file you receive from the certificate authority. Include the lines `Begin Certificate` and `End Certificate.`
2. Choose `Operations > Import User Certificate` from the menu bar; the Import Certificate dialog box appears.
3. Choose the `Paste the Certificate` button, and choose `OK`; an Import Certificate dialog box appears with the following message:

   ```
   Please provide a base64 format certificate and paste it
   below.
   ```
4. Paste the certificate into the dialog box, and choose `OK`. A message at the bottom of the window confirms that the certificate was successfully installed. You are returned to the Oracle Wallet Manager main panel, and the wallet status changes to `Ready`.

1.6.1.2.2 Selecting a File that Contains the Certificate

To select the file:

1. Choose `Operations > Import User Certificate` from the menu bar.
2. Choose the `Select a file...` certificate button, and choose `OK`; the Import Certificate dialog box appears.
3. Enter the path or folder name of the certificate location.
4. Select the name of the certificate file (for example, `cert.txt`).
5. Choose `OK`. A message at the bottom of the window appears, to inform you that the certificate was successfully installed. You are returned to the Oracle Wallet Manager main panel, and the wallet status is changes to `Ready`.

1.6.1.3 Removing a User Certificate from a Wallet

1. Choose `Operations > Remove User Certificate`; a dialog panel appears and prompts you to verify that you want to remove the user certificate from the wallet.
2. Choose `Yes`; you are returned to the Oracle Wallet Manager main panel, and the certificate displays a status of `Requested`.

1.6.1.4 Removing a Certificate Request

To remove a certificate request:

1. Choose `Certificate Request.`
2. Select menu item `Remove Certificate Request.`

> **Note:** You must remove a certificate before removing its associated request.

1.6.1.5 Exporting a User Certificate

Save the certificate in a file system directory when you elect to export a certificate:

1. Choose `Operations > Export User Certificate` from the menu bar; the Export Certificate dialog box appears.
2. Enter the file system directory to save your certificate in, or navigate to the directory structure under Folders.
3. Enter a file name to save your certificate, in the Enter File Name field.
4. Choose `OK`. A message at the bottom of the window confirms that the certificate was successfully exported to the file. You are returned to the Oracle Wallet Manager main window.

1.6.1.6 Exporting a User Certificate Request

Save the certificate request in a file system directory when you elect to export a certificate request:

1. Choose `Operations > Export Certificate Request` from the menu bar; the Export Certificate Request dialog box appears.
2. Enter the file system directory in which you want o save your certificate request, or navigate to the directory structure under Folders.
3. Enter a file name to save your certificate request, in the Enter File Name field.
4. Choose `OK`. A message at the bottom of the window confirms that the certificate request was successfully exported to the file. You are returned to the Oracle Wallet Manager main window.

1.6.2 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
- Exporting a Wallet

1.6.2.1 Importing a Trusted Certificate

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

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

1.6.2.1.1 Pasting the Trusted Certificate To paste the trusted certificate:

1. Choose `Operations > Import Trusted Certificate` from the menu bar; the Import Trusted Certificate dialog panel appears.
2. Choose the `Paste the Certificate` button, and choose `OK`. An Import Trusted Certificate dialog panel appears with the following message:

   ```
   Please provide a base64 format certificate and paste it
   below.
   ```
3. 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.`
4. Paste the certificate into the window, and Choose `OK`. A message at the bottom of the window informs you that the trusted certificate was successfully installed.
5. Choose `OK`; you are returned to the Oracle Wallet Manager main panel, and the trusted certificate appears at the bottom of the Trusted Certificates tree.

1.6.2.1.2 Selecting a File that Contains the Trusted Certificate

To select the file:

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

1.6.2.2 Removing a Trusted Certificate

To remove a trusted certificate from a wallet:

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

   A dialog panel warns you that your user certificate will no longer be verifiable by its recipients if you remove the trusted certificate that was used to sign it.
3. Choose `Yes`; the selected trusted certificate is removed from the Trusted Certificates tree.

Note: A certificate that is signed by a trusted certificate is no longer verifiable when you remove it from your wallet.

Also, you cannot remove a trusted certificate if it has been used to sign a user certificate that is still present in the wallet. To remove such a trusted certificate, you must first remove the certificates that it has signed.

1.6.2.3 Exporting a Trusted Certificate

To export a trusted certificate to another file system location:

1. Select `Operations > Export Trusted Certificate`; the Export Trusted Certificate dialog box appears.
2. Select a file system directory to save your trusted certificate, or choose `Browse` to display the directory structure.
3. Enter a file name to save your trusted certificate.
4. Choose `OK`; you are returned to the Oracle Wallet Manager main window.

1.6.2.4 Exporting All Trusted Certificates

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

1. Choose `Operations > Export All Trusted Certificates`. The Export Trusted Certificate dialog box appears.
2. Select the file system directory to save your trusted certificates, or choose `Browse` to display the directory structure.
3. Enter a file name to save your trusted certificates.
4. Choose `OK`; you are returned to the Oracle Wallet Manager main window.

1.6.2.5 Exporting a Wallet

You can export a wallet to text-based PKI formats. Individual components are formatted according to the following standards (Table 1–5):

Table 1–5 PKI Wallet Encoding Standards

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

Glossary

authentication

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

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.

decryption

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

encryption

The process of disguising a message thereby rendering it unreadable to any but the intended recipient.

key pair

A public key and its associated private key.

man-in-the-middle

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

Oracle PKI certificate usages

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

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**.

private key

In public-key cryptography, this key is the secret key. It is primarily used for decryption but is also used for encryption with digital signatures. See **public/private key pair**.

public key

In public-key cryptography, this key is made public to all. It is primarily used for encryption but can be used for verifying signatures. See **public/private key pair**.

public-key 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.

single key-pair wallet

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

trusted certificate

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

wallet

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

wallet resource locator

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

X.509

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