Oracle® Database

Advanced Security Administrator's Guide

10*g* Release 1 (10.1)

Part No. B10772-01

December 2003

ORACLE®

Oracle Database Advanced Security Administrator's Guide, 10*g* Release 1 (10.1)

Part No. B10772-01

Primary Author: Laurel P. Hale

Contributors: Rajbir Chahal, Min-Hank Ho, Michael Hwa, Sudha Iyer, Adam Lindsey Jacobs, Supriya Kalyanasundaram, Lakshmi Kethana, Andrew Koyfman, Van Le, Nina Lewis, Stella Li, Janaki Narasinghanallur, Vikram Pesati, Andy Philips, Richard Smith, Deborah Steiner, Philip Thornton, Ramana Turlapati

Graphic Designer: Valarie Moore

responsible for the performance of the Kerberos software, does not provide technical support for the software, and shall not be liable for any damages arising out of any use of the Kerberos software.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

---

derivative works of the Source Code, whether created by OpenVision or by a third party. The OpenVision copyright notice must be preserved if derivative works are made based on the donated Source Code.

OpenVision Technologies, Inc., has donated this Kerberos Administration system to M.I.T. for inclusion in the standard Kerberos 5 distribution. This donation underscores our commitment to continuing Kerberos technology development and our gratitude for the valuable work which has been performed by M.I.T. and the Kerberos community.

Portions contributed by Matt Crawford <crawdad@fnal.gov> were work performed at Fermi National Accelerator Laboratory, which is operated by Universities Research Association, Inc., under contract DE-AC02-76CHO3000 with the U. S. Department of Energy.

Contents

2 Configuration and Administration Tools Overview

Part II Network Data Encryption and Integrity

3 Configuring Network Data Encryption and Integrity for Oracle Servers and Clients

4 Configuring Network Data Encryption and Integrity for Thin JDBC Clients

Part III Oracle Advanced Security Strong Authentication

5 Configuring RADIUS Authentication

6 Configuring Kerberos Authentication

7 Configuring Secure Sockets Layer Authentication

8 Using Oracle Wallet Manager

12 Enterprise User Security Configuration Tasks and Troubleshooting

13 Administering Enterprise User Security

C Integrating Authentication Devices Using RADIUS

D Oracle Advanced Security FIPS 140-1 Settings

E orapki Utility

F Entrust-Enabled SSL Authentication

G Using the User Migration Utility

List of Figures

List of Tables

Send Us Your Comments

Oracle Database Advanced Security Administrator's Guide, 10*g* Release 1 (10.1)

Part No. B10772-01

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: infodev_us@oracle.com
- FAX: (650) 506-7227 Attn: Server Technologies Documentation Manager
- Postal service:
  Oracle Corporation
  Server Technologies Documentation
  500 Oracle Parkway, Mailstop 4op11
  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

Welcome to the Oracle Database Advanced Security Administrator's Guide for the 10*g* Release 1 (10.1) of Oracle Advanced Security.

Oracle Advanced Security contains a comprehensive suite of security features that protect enterprise networks and securely extend them to the Internet. It provides a single source of integration with multiple network encryption and authentication solutions, single sign-on services, and security protocols.

The Oracle Database Advanced Security Administrator's Guide describes how to implement, configure and administer Oracle Advanced Security.

This preface contains these topics:

Audience

The Oracle Database Advanced Security Administrator's Guide is intended for users and systems professionals involved with the implementation, configuration, and administration of Oracle Advanced Security including:

- Implementation consultants
- System administrators
- Security administrators
- Database administrators (DBAs)

Organization

This document contains the following chapters:

Part I, "Getting Started with Oracle Advanced Security"

Chapter 1, "Introduction to Oracle Advanced Security"

This chapter provides an overview of Oracle Advanced Security features provided with this release.

Chapter 2, "Configuration and Administration Tools Overview"

This chapter provides an introduction and overview of Oracle Advanced Security GUI and command-line tools.

Part II, "Network Data Encryption and Integrity"

Chapter 3, "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients"

This chapter describes how to configure data encryption and integrity within an existing Oracle Net Services 10*g* Release 1 (10.1) network.

Chapter 4, "Configuring Network Data Encryption and Integrity for Thin JDBC Clients"

This chapter provides an overview of the Java implementation of Oracle Advanced Security, which lets Thin Java Database Connectivity (JDBC) clients securely connect to Oracle Database databases.

Part III, "Oracle Advanced Security Strong Authentication"

Chapter 5, "Configuring RADIUS Authentication"

This chapter describes how to configure Oracle for use with RADIUS (Remote Authentication Dial-In User Service). It provides an overview of how RADIUS works within an Oracle environment, and describes how to enable RADIUS authentication and accounting. It also introduces the challenge-response user interface that third party vendors can customize to integrate with third party authentication devices.

Chapter 6, "Configuring Kerberos Authentication"

This chapter describes how to configure Oracle for use with MIT Kerberos and provides a brief overview of steps to configure Kerberos to authenticate Oracle users. It also includes a brief section that discusses interoperability between the Oracle Advanced Security Kerberos adapter and a Microsoft KDC.

Chapter 7, "Configuring Secure Sockets Layer Authentication"

This chapter describes how Oracle Advanced Security supports a public key infrastructure (PKI). It includes a discussion of configuring and using the Secure Sockets Layer (SSL), certificate validation, and hardware security module support features of Oracle Advanced Security.

Chapter 8, "Using Oracle Wallet Manager"

This chapter describes how to use Oracle Wallet Manager to manage Oracle wallets and PKI credentials.

Chapter 9, "Configuring Multiple Authentication Methods and Disabling Oracle Advanced Security"

This chapter describes the authentication methods that can be used with Oracle Advanced Security, and how to use conventional user name and password authentication. It also describes how to configure the network so that Oracle clients can use a specific authentication method, and Oracle servers can accept any method specified.

Chapter 10, "Configuring Oracle DCE Integration"

This chapter provides a brief discussion of Open Software Foundation (OSF) DCE and Oracle DCE Integration, including what you need to do to configure DCE to use Oracle DCE Integration, how to configure the DCE CDS naming adapter, DCE

parameters, and how clients outside of DCE can access Oracle databases using another protocol such as TCP/IP.

Part IV, "Enterprise User Security"

Chapter 11, "Getting Started with Enterprise User Security"

This chapter describes the Oracle LDAP directory and database integration that enables you to store and manage users' authentication information in Oracle Internet Directory. This feature makes identity management services available to Oracle databases, which provides single sign-on to users (users can authenticate themselves to the database once and subsequent authentications occur transparently). It describes the components and provides an overview of how Enterprise User Security works.

Chapter 12, "Enterprise User Security Configuration Tasks and Troubleshooting"

This chapter explains how to configure Enterprise User Security, providing a configuration steps roadmap and the tasks required to configure password-, SSL-, and Kerberos-based Enterprise User Security authentication.

Chapter 13, "Administering Enterprise User Security"

This chapter describes how to use the Enterprise Security Manager to define directory identity management realm properties and to manage enterprise users, enterprise domains, and enterprise roles.

Part V, "Appendixes"

Appendix A, "Data Encryption and Integrity Parameters"

This appendix describes Oracle Advanced Security data encryption and integrity configuration parameters.

Appendix B, "Authentication Parameters"

This appendix describes Oracle Advanced Security authentication configuration file parameters.

Appendix C, "Integrating Authentication Devices Using RADIUS"

This appendix explains how third party authentication device vendors can integrate their devices and customize the graphical user interface used in RADIUS challenge-response authentication.

Appendix D, "Oracle Advanced Security FIPS 140-1 Settings"

This appendix describes the `sqlnet.ora` configuration parameters required to comply with the FIPS 140-1 Level 2 evaluated configuration.

Appendix E, "orapki Utility"

This appendix provides the syntax for the `orapki` command line utility. This utility must be used to manage certificate revocation lists (CRLs). You can also use this utility to create and manage Oracle wallets; create certificate requests, signed certificates, and user certificates for testing purposes; and to export certificates and certificate requests from Oracle wallets.

Appendix F, "Entrust-Enabled SSL Authentication"

This appendix describes how to configure and use Entrust-enabled Oracle Advanced Security for Secure Sockets Layer (SSL) authentication.

Appendix G, "Using the User Migration Utility"

This appendix describes the User Migration Utility, which can be used to perform bulk migrations of database users to an LDAP directory where they are stored and managed as enterprise users. It provides utility syntax, prerequisites, and usage examples.

Glossary

Related Documentation

For more information, see these Oracle resources:

- *Oracle Net Services Administrator's Guide*
- *Oracle Database Heterogeneous Connectivity Administrator's Guide*
- *Oracle Database JDBC Developer's Guide and Reference*
- *Oracle Internet Directory Administrator's Guide*
- *Oracle Database Administrator's Guide*
- *Oracle Database Security Guide*

Many books in the documentation set use the sample schemas of the seed database, which is installed by default when you install Oracle. Refer to *Oracle Database Sample Schemas* for information on how these schemas were created and how you can use them yourself.

Printed documentation is available for sale in the Oracle Store at

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

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

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

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

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

For information from third-party vendors, see:

- *ACE/Server Administration Manual, from Security Dynamics*
- *ACE/Server Client for UNIX, from Security Dynamics*
- *ACE/Server Installation Manual, from Security Dynamics*
- *RADIUS Administrator's Guide*
- Notes about building and installing Kerberos from Kerberos version 5 source distribution
- *Entrust/PKI for Oracle*
- *Administering Entrust/PKI on UNIX*
- *Transarc DCE User's Guide and Reference*
- *Transarc DCE Application Development Guide*
- *Transarc DCE Application Development Reference*
- *Transarc DCE Administration Guide*
- *Transarc DCE Administration Reference*
- *Transarc DCE Porting and Testing Guide*
- *Application Environment Specification/Distributed Computing*
- *Transarc DCE Technical Supplement*

For conceptual information about the network security technologies supported by Oracle Advanced Security, you can refer to the following third-party publications:

- *Applied Cryptography, Second Edition: Protocols, Algorithms, and Source Code in C* by Bruce Schneier. New York: John Wiley & Sons, 1996.
- *SSL & TLS Essentials: Securing the Web* by Stephen A. Thomas. New York: John Wiley & Sons, 2000.
- *Understanding and Deploying LDAP Directory Services* by Timothy A. Howes, Ph.D., Mark C. Smith, and Gordon S. Good . Indianapolis: New Riders Publishing, 1999.
- *Understanding Public-Key Infrastructure: Concepts, Standards, and Deployment Considerations* by Carlisle Adams and Steve Lloyd. Indianapolis: New Riders Publishing, 1999.

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.	*Oracle Database Concepts* Ensure that the recovery catalog and target database do *not* reside on the same disk.

Convention	Meaning	Example
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.
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 italic monospace (fixed-width) font	Lowercase italic monospace 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]
...	Horizontal ellipsis points indicate either: ■ That we have omitted parts of the code that are not directly related to the example ■ That you can repeat a portion of the code	CREATE TABLE ... AS *subquery*; SELECT *col1*, *col2*, ... , *coln* FROM employees;
. . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	SQL> SELECT NAME FROM V$DATAFILE; NAME ---------- /fsl/dbs/tbs_01.dbf /fs1/dbs/tbs_02.dbf . . . /fsl/dbs/tbs_09.dbf 9 rows selected.
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;

Convention	Meaning	Example

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.	To start the Database Configuration Assistant, choose Start > Programs > Oracle - *HOME_NAME* > Configuration and Migration Tools > Database Configuration Assistant.
File and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe (|), and dash (-). The special character backslash (\) is treated as an element separator, even when it appears in quotes. If the file name begins with \\, then Windows assumes it uses the Universal Naming Convention.	`c:\winnt"\"system32 is the same as C:\WINNT\SYSTEM32`
`C:\>`	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the *command prompt* in this manual.	`C:\oracle\oradata>`

Convention	Meaning	Example
Special characters	The backslash (\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.	`C:\>exp scott/tiger TABLES=emp QUERY=\"WHERE job='SALESMAN' and sal<1600\"` `C:\>imp SYSTEM/`*`password`* `FROMUSER=scott TABLES=(emp, dept)`
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`
ORACLE_HOME and *ORACLE_BASE*	In releases prior to Oracle8*i* release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level *ORACLE_HOME* directory. For Windows NT, the default location was `C:\orant`. This release complies with Optimal Flexible Architecture (OFA) guidelines. 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 the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is `C:\oracle\ora`*nn*, where *nn* is the latest release number. The Oracle home directory is located directly under *ORACLE_BASE*. All directory path examples in this guide follow OFA conventions. Refer to *Oracle Database Platform Guide for Windows* for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.	Go to the *ORACLE_BASE*`\`*ORACLE_HOME*`\rdbms\admin` directory.

Documentation Accessibility

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

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

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

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

What's New in Oracle Advanced Security?

This section describes new features of Oracle Advanced Security 10*g* Release 1 (10.1) and provides pointers to additional information. New features information from the previous release is also retained to help those users migrating to the current release.

The following sections describe the new features in Oracle Advanced Security:

- Oracle Database 10g Release 1 (10.1) New Features in Oracle Advanced Security
- Oracle9i Release 2 (9.2) New Features in Oracle Advanced Security

Oracle Database 10*g* Release 1 (10.1) New Features in Oracle Advanced Security

Oracle Advanced Security 10*g* Release 1 (10.1) includes new features in the following areas:

- New Features in Strong Authentication
- New Features in Enterprise User Security

New Features in Strong Authentication

Oracle Advanced Security provides several strong authentication options, including support for RADIUS, Kerberos, and PKI (public key infrastructure). This release provides the following new features for strong authentication:

- Support for TLS (Transport Layer Security), version 1.0

  TLS is an industry-standard protocol which provides effective security for transactions conducted on the Web. It has been developed by the Internet

Engineering Task Force (IETF) to be the successor to SSL version 3.0. TLS is a configurable option provided in Oracle Net Manager.

See Also: Chapter 7, "Configuring Secure Sockets Layer Authentication" for configuration details

- Support for Hardware Security Modules, including Oracle Wallet Manager Integration

  In this release, Oracle Advanced Security supports hardware security modules which use APIs that conform to the RSA Security, Inc., Public-Key Cryptography Standards (PKCS) #11. In addition, it is now possible to create Oracle Wallets that can store credentials on a hardware security module for servers, or private keys on tokens for clients. This provides roaming authentication to the database.

  Hardware security modules can be used for the following functions:

  – Store cryptographic information, such as private keys, which provides stronger security

  – Perform cryptographic operations to off load RSA operations from the server, freeing the CPU to respond to other transactions

  See Also:

  - "Configuring Your System to Use Hardware Security Modules" on page 7-48 for configuration details
  - "Creating a Wallet to Store Hardware Security Module Credentials" on page 8-11

- CRL (Certificate Revocation Lists) and CRLDP (CRL Distribution Point) Support for Certificate Validation

  In the current release, you now have the option to configure certificate revocation status checking for both the client and the server. 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. The `orapki` utility has also been added for CRL management and for managing Oracle wallets and certificates.

See Also:

- "Certificate Validation with Certificate Revocation Lists" on page 7-35 for details
- Appendix E, "orapki Utility" for details about `orapki` command line utility

New Features in Enterprise User Security

- Kerberos Authenticated Enterprise Users

  Kerberos-based authentication to the database is available for users managed in an LDAP directory. This includes Oracle Internet Directory or any other third-party directory that is synchronized to work with Oracle Internet Directory by using the Directory Integration Platform. To use this feature, all directory users, including those synchronized from third-party directories, must include the Kerberos principal name attribute (`krbPrincipalName` attribute).

  See Also: "Configuring Enterprise User Security for Kerberos Authentication" on page 12-18 for configuration details

- Public Key Infrastructure (PKI) Credentials No Longer Required for Database-to-Oracle Internet Directory Connections

  In this release, a database can bind to Oracle Internet Directory by using password / SASL-based authentication, eliminating the overhead of setting up PKI credentials for the directory and multiple databases. SASL (Simple Authentication and Security Layer) is a standard defined in the Internet Engineering Task Force RFC 2222. It is a method for adding authentication support to connection-based protocols such as LDAP.

  See Also: "Configuring Enterprise User Security for Password Authentication" on page 12-16 for configuration details

- Support for User Management in Third-Party LDAP Directories

  In the current release of Enterprise User Security, you can store and manage your users and their passwords in third-party LDAP directories. This feature is made possible with

  – Directory Integration Platform, which automatically synchronizes third-party directories with Oracle Internet Directory, and

- Oracle Database recognition of standard password verifiers, which is also new in this release.

- Tool Changes
  - New Tool: Enterprise Security Manager Console

    The Enterprise Security Manager Console, which is based on the Oracle Internet Directory Delegated Administration Service (DAS), is new in this release. Administrators can use this tool to create enterprise users, enterprise user security groups, and to configure identity management realm attributes in the directory that relate to Enterprise User Security.
  - In this release, Oracle Enterprise Login Assistant functionality has been migrated to the new Enterprise Security Manager Console and Oracle Wallet Manager. The following table lists which tool you should now use to perform tasks that you previously performed by using Oracle Enterprise Login Assistant:

If you used Oracle Enterprise Login Assistant to...	Then now you should use...
Change the directory-to-database password	Enterprise Security Manager Console
Change an Oracle wallet password	Oracle Wallet Manager
Enable auto login for an Oracle wallet	Oracle Wallet Manager

See Also: The following sections for information about Enterprise Security Manager Console and how to use it:

- "Enterprise Security Manager Console Overview" on page 2-22, which provides a brief introduction to the tool.
- Chapter 13, "Administering Enterprise User Security", which provides procedural information for using the tool to manage enterprise users.

Oracle9*i* Release 2 (9.2) New Features in Oracle Advanced Security

The new features for Oracle Advanced Security in release 2 (9.2) include the following:

- Support for Advanced Encryption Standard (AES)

  AES is a new cryptographic algorithm standard developed to replace Data Encryption Standard (DES).

  See Also:

  - "Advanced Encryption Standard" on page 1-6 for a brief overview of this encryption algorithm
  - Chapter 3, "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients" for configuration details

- SSL Hardware Accelerator Support

  In release 2 (9.2), complex public key cryptographic operations can be off loaded to hardware accelerators to improve the performance of SSL transactions.

  See Also: "Configuring Your System to Use Hardware Security Modules" on page 7-48 for configuration details

- New Enterprise User Security Tool: User Migration Utility

  This utility enables administrators to perform bulk migrations of database users to Oracle Internet Directory for centralized user storage and management.

  See Also: Appendix G, "Using the User Migration Utility" for information about this tool and how to use it.

Part I

Getting Started with Oracle Advanced Security

This part introduces Oracle Advanced Security, describing the security solutions it provides, its features, and its tools. It contains the following chapters:

- Chapter 1, "Introduction to Oracle Advanced Security"
- Chapter 2, "Configuration and Administration Tools Overview"

1

Introduction to Oracle Advanced Security

This chapter introduces Oracle Advanced Security, summarizing the security risks it addresses, and describing its features. These features are available to database and related products that interface with Oracle Net Services, including Oracle Database, Oracle Application Server, and Oracle Identity Management infrastructure.

This chapter contains the following topics:

Security Challenges in an Enterprise Environment

To increase efficiency and lower costs, companies adopt strategies to automate business processes. One such strategy is to conduct more business on the Web, but that requires greater computing power, translating to higher IT costs. In response to rising IT costs, more and more businesses are considering enterprise **grid computing** architectures where inexpensive computers act as one powerful machine. While such strategies improve the bottom line, they introduce risks, which are associated with securing data in motion and managing an ever increasing number of user identities.

This section examines the security challenges of today's enterprise computing environments in the following topics:

Security in Enterprise Grid Computing Environments

Grid computing is a computing architecture that coordinates large numbers of servers and storage to act as a single large computer. It provides flexibility, lower costs, and IT investment protection because inexpensive, off-the-shelf components can be added to the grid as business needs change. While providing significant benefits, grid computing environments present unique security requirements because their computing resources are distributed and often heterogeneous. The following sections discuss these requirements.

Distributed Environment Security Requirements

Enterprise grid computing pools distributed business computing resources to cost effectively harness the power of clustered servers and storage. A distributed environment requires secure network connections. Even more critical in grid environments, it is necessary to have a uniform definition of "who is a user" and "what are they allowed to do." Without such uniform definitions, administrators frequently must assign, manage, and revoke authorizations for every user on different software applications to protect employee, customer, and partner information. This is expensive because it takes time, which drives up costs. Consequently, the cost savings gained with grid computing are lost.

Heterogeneous Environment Security Requirements

Because grid computing environments often grow as business needs change, computing resources are added over time, resulting in diverse collections of hardware and software. Such heterogeneous environments require support for different types of authentication mechanisms which adhere to industry standards. Without strict adherence to industry standards, integrating heterogeneous components becomes costly and time consuming. Once again the benefits of grid computing are squandered when the appropriate infrastructure is not present.

Security in an Intranet or Internet Environment

Oracle databases power the largest and most popular Web sites on the Internet. In record numbers, organizations throughout the world are deploying distributed databases and client/server applications based on Oracle Database and Oracle Net Services. This proliferation of distributed computing is matched by an increase in

the amount of information that organizations place on computers. Employee and financial records, customer orders, product information, and other sensitive data have moved from filing cabinets to file structures. The volume of sensitive information on the Web has thus increased the value of data that can be compromised.

Common Security Threats

The increased volume of data in distributed, heterogeneous environments exposes users to a variety of security threats, including the following:

- Eavesdropping and Data Theft
- Data Tampering
- Falsifying User Identities
- Password-Related Threats

Eavesdropping and Data Theft

Over the Internet and in wide area network environments, both public carriers and private networks route portions of their network through insecure land lines, vulnerable microwave and satellite links, or a number of servers— exposing valuable data to interested third parties. In local area network environments within a building or campus, the potential exists for insiders with access to the physical wiring to view data not intended for them, and network **sniffer**s can be installed to eavesdrop on network traffic.

Data Tampering

Distributed environments bring with them the possibility that a malicious third party can compromise integrity by tampering with data as it moves between sites.

Falsifying User Identities

In a distributed environment, it is more feasible for a user to falsify an identity to gain access to sensitive information. How can you be sure that user *Pat* connecting to Server A from Client B really is user `Pat`?

Moreover, in distributed environments, malefactors can hijack connections. How can you be sure that Client B and Server A are what they claim to be? A transaction that should go from the Personnel system on Server A to the Payroll system on Server B could be intercepted in transit and re-routed to a terminal masquerading as Server B.

Password-Related Threats

In large systems, users typically must remember multiple passwords for the different applications and services that they use. For example, a developer can have access to a development application on a workstation, a PC for sending e-mail, and several computers or intranet sites for testing, reporting bugs, and managing configurations.

Users typically respond to the problem of managing multiple passwords in several ways:

- They may select easy-to-guess passwords—such as a name, fictional character, or a word found in a dictionary. All of these passwords are vulnerable to **dictionary attack**s.
- They may also choose to standardize passwords so that they are the same on all machines or web sites. This results in a potentially large exposure in the event of a compromised password. They can also use passwords with slight variations that can be easily derived from known passwords.
- Users with complex passwords may write them down where an attacker can easily find them, or they may just forget them—requiring costly administration and support efforts.

All of these strategies compromise password secrecy and service availability. Moreover, administration of multiple user accounts and passwords is complex, time-consuming, and expensive.

Solving Security Challenges with Oracle Advanced Security

To solve enterprise computing security problems, Oracle Advanced Security provides industry standards-based data privacy, integrity, authentication, single sign-on, and access authorization in a variety of ways. 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:

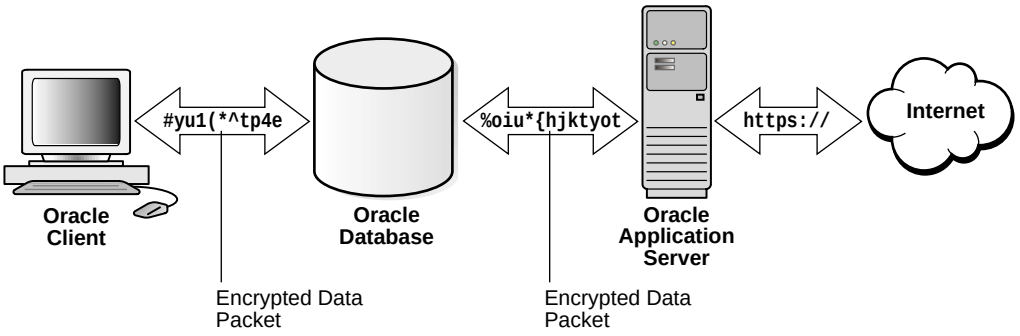
- Data Encryption
- Strong Authentication
- Enterprise User Management

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 with a decryption key.

Figure 1–1 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 the client, the database, and the application server is encrypted, then when the manager sends the bonus amount to the database, it is protected.

Figure 1–1 Encryption



This section discusses the following topics:

- Supported Encryption Algorithms
- Data Integrity
- Federal Information Processing Standard

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 Encryption
- Advanced Encryption Standard

Selecting the network encryption algorithm is a user configuration option, providing varying levels of security and performance for different types of data transfers.

Prior versions of Oracle Advanced Security provided three editions: Domestic, Upgrade, and Export—each with different key lengths. 10*g* 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.

Note: The U.S. government has relaxed its export guidelines for encryption products. Accordingly, Oracle can ship Oracle Advanced Security with its strongest encryption features to all of its customers.

RC4 Encryption The RC4 encryption module uses the RSA Security, Inc., RC4 encryption algorithm. Using a secret, randomly-generated key 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 request or require the use of the encryption module to guarantee that data is protected. Oracle's optimized implementation 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 algorithm (DES) with a standard, optimized 56-bit key encryption algorithm, and also provides DES40, a 40-bit version, for backward compatibility.

Triple-DES 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 a 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 with 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.

See Also:

- Chapter 3, "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients"
- Appendix A, "Data Encryption and Integrity Parameters"

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

Note: SHA-1 is slightly slower than MD5, but produces a larger message digest, making it more secure against brute-force collision and inversion attacks.

See Also: Chapter 3, "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients", for information about MD5 and SHA-1.

Federal Information Processing Standard

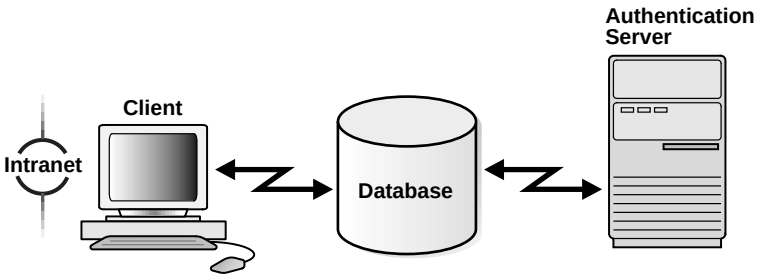
Oracle Advanced Security Release 8.1.6 has been validated under U.S. Federal Information Processing Standard 140-1 (FIPS) at the Level 2 security level. This provides independent confirmation that Oracle Advanced Security conforms to federal government standards. FIPS configuration settings are described by Appendix D, "Oracle Advanced Security FIPS 140-1 Settings".

Strong Authentication

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

Figure 1–2 shows user authentication with an Oracle database configured to use a third-party authentication server. Having a central facility to authenticate all members of the network (clients to servers, servers to servers, users to both clients and servers) is one effective way to address the threat of network nodes falsifying their identities.

Figure 1–2 Strong Authentication with Oracle Authentication Adapters



This section contains the following topics:

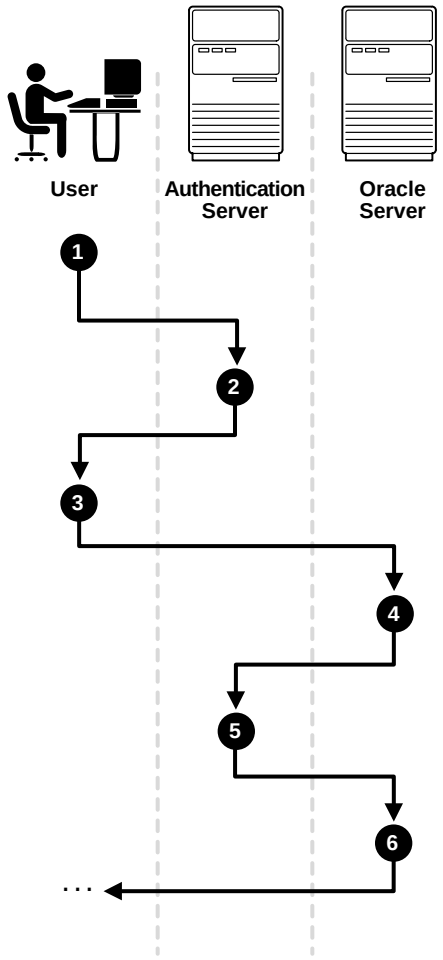
- Centralized Authentication and Single Sign-On
- Supported Authentication Methods

Centralized Authentication and Single Sign-On

Centralized authentication also provides the benefit of **single sign-on (SSO)** for users. Single sign-on enables users to access multiple accounts and applications with a single password. A user only needs to log on once and can then automatically connect to any other service without having to give a username and password again. Single sign-on eliminates the need for the user to remember and administer multiple passwords, reducing the time spent logging into multiple services.

How Centralized Network Authentication Works Figure 1–3 shows how a centralized network authentication service typically operates:

Figure 1–3 How a Network Authentication Service Authenticates a User



1. A user (client) requests authentication services and provides identifying information, such as a token or password.
2. The authentication server validates the user's identity and passes a ticket or credentials back to the client, which may include an expiration time.

3. The client passes these credentials to the Oracle server concurrent with a service request, such as connection to a database.
4. The server sends the credentials back to the authentication server for authentication.
5. If the authentication server accepts the credentials, then it notifies the Oracle Server, and the user is authenticated.
6. If the authentication server does not accept the credentials, then authentication fails, and the service request is denied.

Supported Authentication Methods

Oracle Advanced Security supports the following industry-standard authentication methods:

- Kerberos
- RADIUS (Remote Authentication Dial-In User Service)
- DCE (Distributed Computing Environment)
- Secure Sockets Layer (with digital certificates)
- Entrust/PKI

Kerberos Oracle Advanced Security support for Kerberos provides the benefits of single sign-on and centralized authentication of Oracle users. Kerberos is a trusted third-party authentication system that relies on shared secrets. It presumes that the third party is secure, and provides single sign-on capabilities, centralized password storage, database link authentication, and enhanced PC security. It does this through a Kerberos authentication server. See Chapter 6, "Configuring Kerberos Authentication" for information about configuring and using this adapter.

Note: Oracle authentication for Kerberos provides database link authentication (also called proxy authentication). Kerberos is also an authentication method that is supported with Enterprise User Security.

RADIUS (Remote Authentication Dial-In User Service) RADIUS is a client/server security protocol that is most widely known for enabling remote authentication and access. Oracle Advanced Security uses this standard in a client/server network environment to enable use of any authentication method that supports the RADIUS

protocol. RADIUS can be used with a variety of authentication mechanisms, including token cards and smart cards. See Chapter 5, "Configuring RADIUS Authentication" for information about configuring and using this adapter.

- **Smart Cards**

  A RADIUS-compliant smart card is a credit card-like hardware device. It has memory and a processor and is read by a smart card reader located at the client workstation.

- **Token Cards**

  Token cards (SecurID or RADIUS-compliant) can improve ease of use through several different mechanisms. Some token cards dynamically display one-time passwords that are synchronized with an authentication service. The server can verify the password provided by the token card at any given time by contacting the authentication service. Other token cards have a keypad and operate on a challenge-response basis. In this case, the server offers a challenge (a number) that the user enters into a token card. The token card provides a response (another number cryptographically derived from the challenge) that the user enters and sends to the server.

  You can use SecurID tokens through the RADIUS adapter.

DCE (Distributed Computing Environment) DCE is a set of integrated network services that works across multiple systems to provide a distributed environment. Oracle DCE Integration consists of the following two components:

- DCE Communication / Security
- DCE Cell Directory services Native Naming

Oracle DCE Integration provides applications the flexibility to have different levels of integration with DCE services. Depending on the need, applications can choose to integrate very tightly with the DCE services or choose to plug in the other security authentication services provided by Oracle Advanced Security. See Chapter 10, "Configuring Oracle DCE Integration" for information about configuring and using this adapter.

Secure Sockets Layer Secure Sockets Layer (SSL) is an industry standard protocol for securing network connections. SSL provides **authentication**, data **encryption**, and data **integrity**.

The SSL protocol is the foundation of a **public key infrastructure (PKI)**. For authentication, SSL uses digital certificates that comply with the X.509v3 standard, and a **public and private key pair**.

Oracle Advanced Security SSL can be used to secure communications between any client and any server. You can configure SSL to provide authentication for the server only, the client only, or both client and server. You can also configure SSL features in combination with other authentication methods supported by Oracle Advanced Security (database usernames and passwords, RADIUS, and Kerberos).

To support your PKI implementation, Oracle Advanced Security includes the following features in addition to SSL:

- Oracle wallets, where you can store PKI credentials
- Oracle Wallet Manager, which you can use to manage your Oracle wallets
- Certificate validation with certificate revocation lists (CRLs)
- Hardware security module support

See Also:

- Chapter 7, "Configuring Secure Sockets Layer Authentication" for conceptual, configuration, and usage information about SSL, certificate validation, and hardware security modules.
- Chapter 8, "Using Oracle Wallet Manager" for information about using this tool to manage Oracle wallets.
- Chapter 9, "Configuring Multiple Authentication Methods and Disabling Oracle Advanced Security" for information about configuring SSL in combination with other authentication methods.

Entrust/PKI Oracle Advanced Security supports the public key infrastructure provided by the Entrust/PKI software from Entrust Technologies, Inc. Entrust-enabled Oracle Advanced Security lets Entrust users incorporate Entrust single sign-on into their Oracle applications, and it lets Oracle users incorporate Entrust-based single sign-on into Oracle applications. See Appendix F, "Entrust-Enabled SSL Authentication" for more information about this feature.

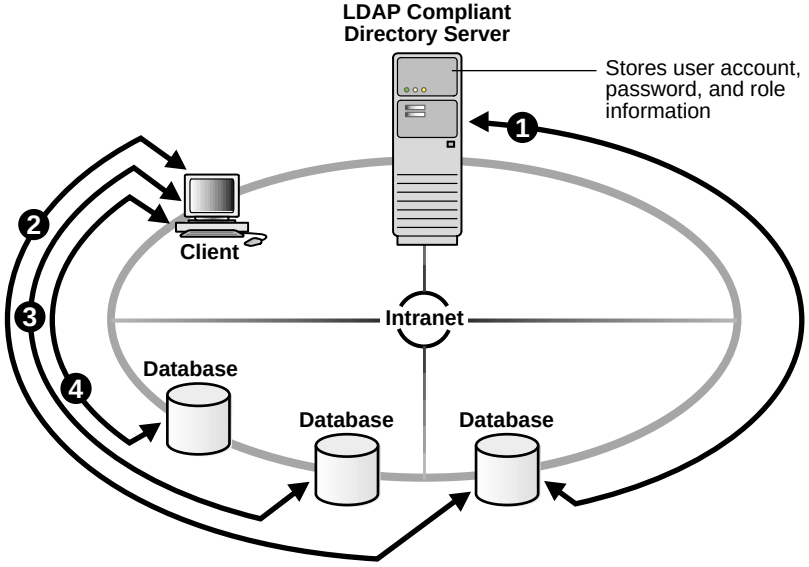
Enterprise User Management

Enterprise user management is provided by the Enterprise User Security feature of Oracle Advanced Security. Enterprise User Security enables storing database users and their corresponding administrative and security information in a centralized directory server.

Figure 1–4 shows how a directory server can be used to provide centralized storage and management of user account, user role, and authentication information.

1. A database server authenticates a user by accessing information stored in the directory.
2. **- 4.** Once authenticated, a user can access the databases, which are configured for enterprise user security.

Figure 1–4 Centralized User Management with Enterprise User Security



This centralized configuration enables the administrator to modify information in one location, the directory. It also lowers the cost of administration and makes the enterprise more secure because there is only one set of user information to manage and track.

Enterprise User Security supports the following authentication methods:

- Passwords
- Kerberos
- Secure Sockets Layer (SSL) with digital certificates

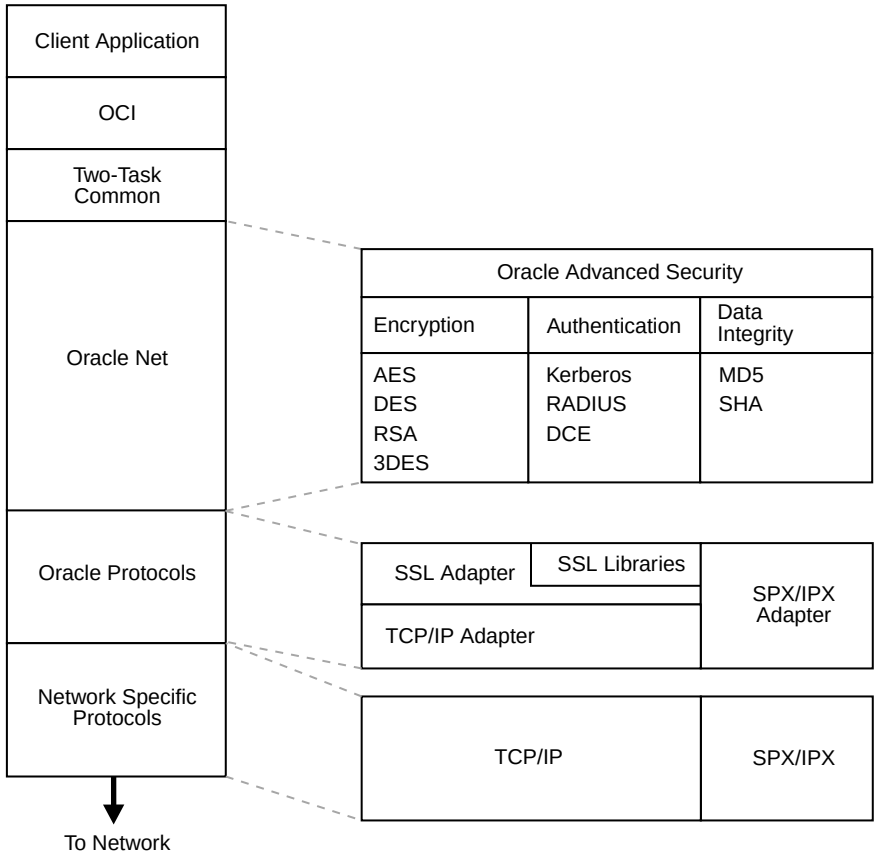
See Also: For detailed discussions of Enterprise User Security concepts, configuration, and management, refer to the following chapters in this manual:

- Chapter 11, "Getting Started with Enterprise User Security"
- Chapter 12, "Enterprise User Security Configuration Tasks and Troubleshooting"
- Chapter 13, "Administering Enterprise User Security"

Oracle Advanced Security Architecture

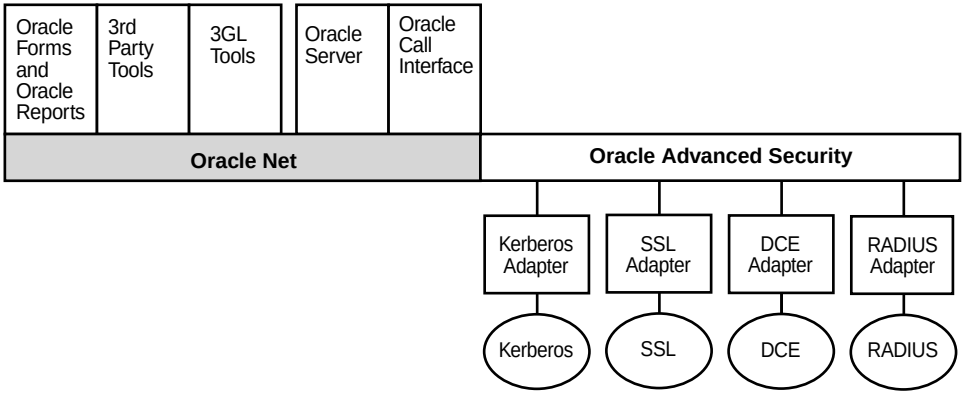
Oracle Advanced Security complements an Oracle server or client installation with advanced security features. Figure 1–5 shows the Oracle Advanced Security architecture within an Oracle networking environment.

Figure 1–5 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 1–6, authentication adapters integrate below the Oracle Net interface and let existing applications take advantage of new authentication systems transparently, without any changes to the application.

Figure 1–6 Oracle Net with Authentication Adapters



See Also: *Oracle Net Services Administrator's Guide*, for more information about stack communications in an Oracle networking environment

Secure Data Transfer Across Network Protocol Boundaries

Oracle Advanced Security is fully supported by Oracle Connection Manager, making secure data transfer a reality across network protocol boundaries. Clients using LAN protocols such as NetWare (SPX/IPX), for example, can securely share data with large servers using different network protocols such as LU6.2, TCP/IP, or DECnet. To eliminate potential weak points in the network infrastructure and to maximize performance, Connection Manager passes encrypted data from protocol to protocol without the cost and exposure of decryption and re-encryption.

System Requirements

Oracle Advanced Security is an add-on product bundled with the Oracle Net Server or Oracle Net Client. It must be purchased and installed on both the client and the server.

Oracle Advanced Security 10*g* Release 1 (10.1) requires Oracle Net 10*g* Release 1 (10.1) and supports Oracle Database Enterprise Edition. Table 1–1 lists additional system requirements.

Note: Oracle Advanced Security is not available with Oracle Database Standard Edition.

Table 1–1 Authentication Methods and System Requirements

Authentication Method	System Requirements
Kerberos	■ MIT Kerberos Version 5, release 1.1 ■ The Kerberos authentication server must be installed on a physically secure machine.
RADIUS	■ A RADIUS server that is compliant with the standards in the Internet Engineering Task Force (IETF) RFC #2138, *Remote Authentication Dial In User Service (RADIUS)* and RFC #2139 *RADIUS Accounting*. ■ To enable challenge-response authentication, you must run RADIUS on an operating system that supports the Java Native Interface as specified in release 1.1 of the Java Development Kit from JavaSoft.
SSL	■ A wallet that is compatible with the Oracle Wallet Manager version 10*g*. Wallets created in earlier releases of the Oracle Wallet Manager are not forward compatible.
Entrust / PKI	■ Entrust IPSEC Negotiator Toolkit Release 6.0 ■ Entrust / PKI 6.0

Oracle Advanced Security Restrictions

Oracle Applications support Oracle Advanced Security encryption and data integrity. However, because Oracle Advanced Security requires Oracle Net Services to transmit data securely, Oracle Advanced Security external authentication features are not supported by some parts of Oracle Financial, Human Resource, and Manufacturing Applications when they are running on Microsoft Windows. The portions of these products that use Oracle Display Manager (ODM) do not take advantage of Oracle Advanced Security, since ODM does not use Oracle Net Services.

2

Configuration and Administration Tools Overview

Configuring advanced security features for an Oracle database includes configuring encryption, integrity (checksumming), and strong authentication methods for Oracle Net Services. Strong authentication method configuration can include third-party software, as is the case for Kerberos or RADIUS, or it may entail configuring and managing a public key infrastructure, as is required for Secure Sockets Layer (SSL). In addition, an Oracle database can be configured to interoperate with an LDAP directory, such as Oracle Internet Directory, to enable Enterprise User Security, a feature that enables you to store and manage database users in a centralized directory.

Such diverse advanced security features require a diverse set of tools with which to configure and administer them. This chapter introduces the tools used to configure and administer advanced security features for an Oracle database in the following topics:

- Network Encryption and Strong Authentication Configuration Tools
- Public Key Infrastructure Credentials Management Tools
- Enterprise User Security Configuration and Management Tools
- Duties of a Security Administrator / DBA
- Duties of an Enterprise User Security Administrator / DBA

Network Encryption and Strong Authentication Configuration Tools

Oracle Net Services can be configured to encrypt data using standard encryption algorithms, and for strong authentication methods, such as Kerberos, RADIUS, and SSL. The following sections introduce the Oracle tools you can use to configure these advanced security features for an Oracle Database:

- Oracle Net Manager
- Oracle Advanced Security Kerberos Adapter Command-Line Utilities

Oracle Net Manager

Oracle Net Manager is a graphical user interface tool, primarily used to configure Oracle Net Services for an Oracle home on a local client or server host.

Although you can use Oracle Net Manager to configure Oracle Net Services, such as naming, listeners, and general network settings, it also enables you to configure the following Oracle Advanced Security features, which use the Oracle Net protocol:

- Strong authentication (Kerberos, RADIUS, and Secure Sockets Layer)
- Network encryption (RC4, DES, Triple-DES, and AES)
- Checksumming for data integrity (MD5, SHA-1)

This section introduces you to the features of Oracle Net Manager that are used to configure Oracle Advanced Security. It contains the following topics:

- Starting Oracle Net Manager
- Navigating to the Oracle Advanced Security Profile

See Also:

- "Duties of a Security Administrator / DBA" on page 2-34 for information about the tasks you can perform with this tool that configure advanced security features.
- *Oracle Net Services Administrator's Guide* and Oracle Net Manager online help for complete documentation of this tool.

Starting Oracle Net Manager

You can start Oracle Net Manager by using Oracle Enterprise Manager Console or as a standalone application. However, you must use the standalone application to access the Oracle Advanced Security Profile where you can configure Oracle Advanced Security features.

To start Oracle Net Manager as a standalone application:

- (UNIX) From *$ORACLE_HOME*/bin, enter the following at the command line:

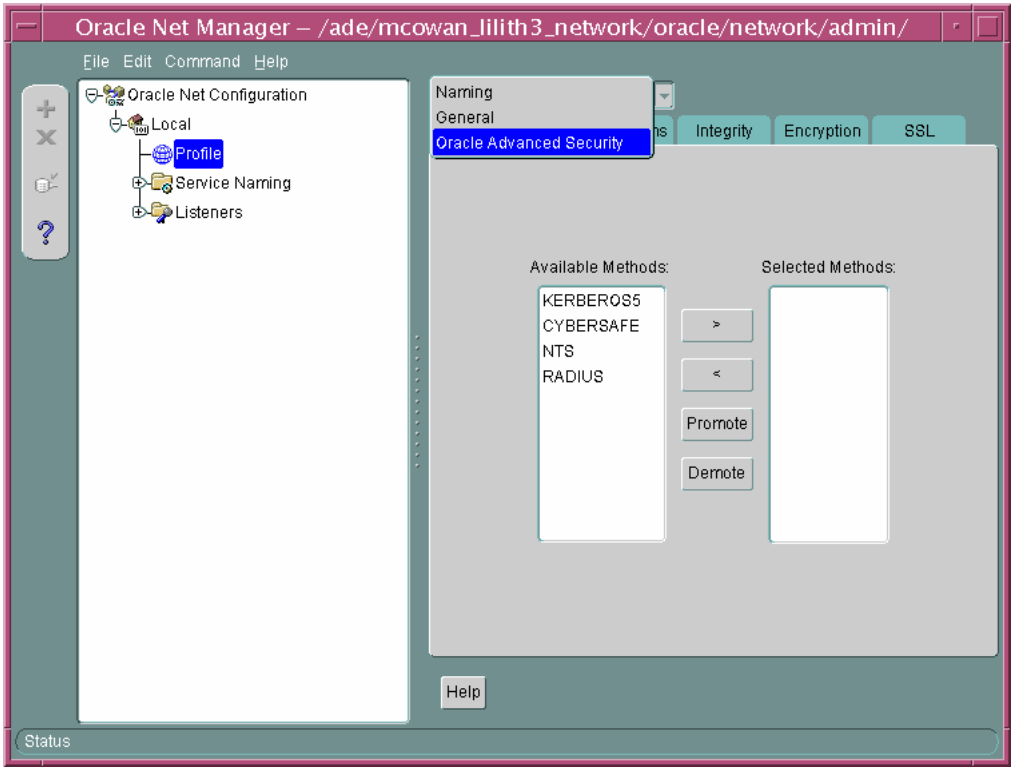
  ```
  netmgr
  ```

- (Windows) Choose **Start** > **Programs** > **Oracle -** ***HOME_NAME*** > **Configuration and Migration Tools** > **Net Manager**

Navigating to the Oracle Advanced Security Profile

The Oracle Net Manager interface window contains two panes: the navigator pane and the right pane, which displays various property sheets that enable you to configure network components. When you select a network object in the navigator pane, its associated property sheets displays in the right pane. To configure Oracle Advanced Security features, choose the **Profile** object in the navigator pane, and then select **Oracle Advanced Security** from the list in the right pane, as shown in Figure 2–1.

Figure 2–1 Oracle Advanced Security Profile in Oracle Net Manager



Oracle Advanced Security Profile Property Sheets

The Oracle Advanced Security Profile contains the following property sheets, which are described in the following sections:

- Authentication Property Sheet
- Other Params Property Sheet
- Integrity Property Sheet
- Encryption Property Sheet
- SSL Property Sheet

Authentication Property Sheet Use this property sheet to select a strong authentication method, such as Kerberos Version 5 (KERBEROS5), Windows NT native authentication (NTS), or RADIUS.

Other Params Property Sheet Use this property sheet to set other parameters for the authentication method you selected on the Authentication property sheet.

Integrity Property Sheet Use this property sheet to enable checksumming on the client or the server and to select an encryption algorithm for generating secure message digests.

Encryption Property Sheet Use this property sheet to select one or more **cipher suites** to encrypt client or server connections with native encryption algorithms.

SSL Property Sheet Use this property sheet to configure Secure Sockets Layer (SSL), including the **wallet** location and **cipher suite**, on a client or server.

Oracle Advanced Security Kerberos Adapter Command-Line Utilities

The Oracle Advanced Security Kerberos adapter provides three command-line utilities that enable you to obtain, cache, display, and remove Kerberos credentials. The following table briefly describes these utilities:

Utility Name	Description
`okinit`	Obtains Kerberos tickets from the key distribution center (**KDC**) and caches them in the user's credential cache
`oklist`	Displays a list of Kerberos tickets in the specified credential cache
`okdstry`	Removes Kerberos credentials from the specified credential cache

See Also: "Utilities for the Kerberos Authentication Adapter" on page 6-11 for complete descriptions of these utilities, their syntax, and available options.

Public Key Infrastructure Credentials Management Tools

The security provided by a public key infrastructure (PKI) depends on how effectively you store, manage, and validate your PKI credentials. The following Oracle tools are used to manage certificates, wallets, and certificate revocation lists so your PKI credentials can be stored securely and your certificate validation mechanisms kept current:

- Oracle Wallet Manager
- orapki Utility

Oracle Wallet Manager

Oracle Wallet Manager is an application that wallet owners and security administrators use to manage and edit the security credentials in their Oracle wallets. A wallet is a password-protected container that is used to store authentication and signing credentials, including private keys, certificates, and trusted certificates needed by SSL. You can use Oracle Wallet Manager to perform the following tasks:

- Create **public and private key pairs**
- Generate certificate requests
- Upload and download wallets to and from an LDAP directory
- Store and manage user credentials
- Store and manage **certificate authority** certificates (**root key certificate** and **certificate chain**)
- Create wallets to store hardware security module credentials

The following topics introduce the Oracle Wallet Manager user interface:

- Starting Oracle Wallet Manager
- Navigating the Oracle Wallet Manager User Interface
- Toolbar
- Menus

> **See Also:** Chapter 8, "Using Oracle Wallet Manager" for detailed information about using this application

Starting Oracle Wallet Manager

To start Oracle Wallet Manager:

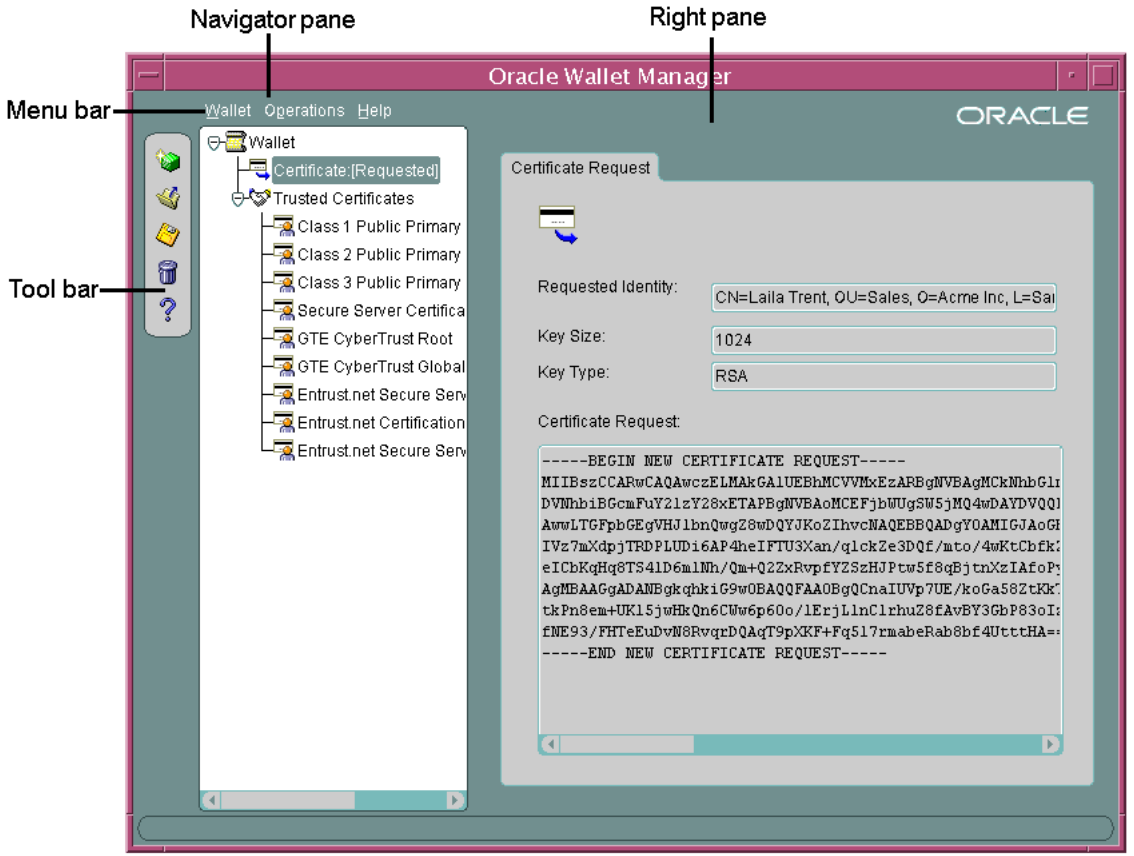
- (UNIX) From *$ORACLE_HOME*/bin, enter the following at the command line:

  owm
- (Windows) Choose **Start** > **Programs** > **Oracle - *HOME_NAME*** > **Integrated Management Tools** > **Wallet Manager**

Navigating the Oracle Wallet Manager User Interface

The Oracle Wallet Manager interface includes two panes, a toolbar, and various menu items as shown in Figure 2–2.

Figure 2–2 Oracle Wallet Manager User Interface



Navigator Pane The navigator pane provides a graphical tree view of the certificate requests and certificates stored in the Oracle home where Oracle Wallet Manager is installed. You can use the navigator pane to view, modify, add, or delete certificates and certificate requests.

The navigator pane functions the same way as it does in other Oracle graphical user interface tools, enabling you to

- Expand and contract wallet objects so that you can manage the user and trusted certificates they contain.
- Right-click a wallet, certificate, or certificate request to perform operations on it such as add, remove, import, or export.

When you expand a wallet, you see a nested list of user and trusted certificates. When you select a wallet or certificate in the navigator pane, details about your selection display in the adjacent right pane of Oracle Wallet Manager. Table 2–1 lists the main objects that display in the navigator pane.

Table 2–1 Oracle Wallet Manager Navigator Pane Objects

Object	Description
Wallet	Password-protected container that is used to store authentication and signing credentials
Certificate Request[1]	A **PKCS #10**-encoded message containing the requester's **distinguished name (DN)**, a public key, the key size, and key type. See also **certificate request**.
Certificate[1]	An **X.509** data structure containing the entity's DN, public key, and is signed by a trusted identity (**certificate authority**). See **certificate**
Trusted Certificates[1]	Sometimes called a root key certificate, is a certificate from a third party identity that is qualified with a level of trust. See **trusted certificate**

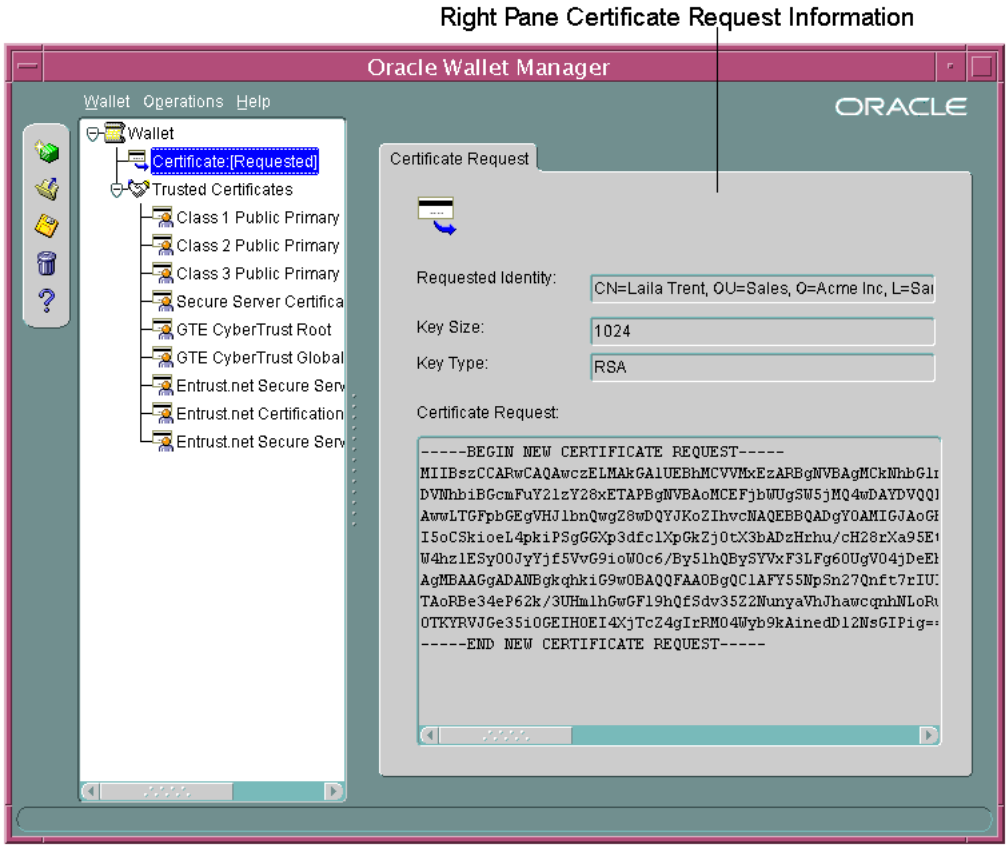
[1] These objects display only after you create a wallet, generate a certificate request, and import a certificate into the wallet.

Right Pane The right pane displays information about an object that is selected in the navigator pane. The right pane is read-only.

Figure 2–3 shows what is displayed in the right pane when a certificate request object is selected in the navigator pane. Information about the request and the requester's identity display in the **Requested Identity**, **Key Size**, and **Key Type** fields. The PKCS #10-encoded certificate request displays in the **Certificate Request**

text box. To request a certificate from a certificate authority, you can copy this request into an e-mail or export it into a file.

Figure 2–3 Certificate Request Information Displayed in Oracle Wallet Manager Right Pane



Toolbar

The toolbar contains buttons that enable you to manage your wallets. Move the mouse cursor over a toolbar button to display a description of the button's function. The toolbar buttons are listed and described in Table 2–2.

Table 2–2 Oracle Wallet Manager Toolbar Buttons

Toolbar Button	Description
New	Creates a new wallet
Open Wallet	Enables you to browse your file system to locate and open an existing wallet
Save Wallet	Saves the currently open wallet
Delete Wallet	Deletes wallet currently selected in the navigator pane
Help	Opens the Oracle Wallet Manager online help

Menus

You use Oracle Wallet Manager menus to manage your wallets and the credentials they contain. The following sections describe the options that are available under each menu.

Wallet Menu Table 2–3 describes the contents of the **Wallet** menu.

Table 2–3 Oracle Wallet Manager Wallet Menu Options

Option	Description
New	Creates a new wallet
Open	Opens an existing wallet
Close	Closes the currently open wallet
Upload Into The Directory Service	Uploads a wallet to a specified LDAP directory server. You must supply a directory password, hostname, and port information
Download From The Directory Service	Downloads a wallet from a specified LDAP directory server. You must supply a directory password, hostname, and port information.
Save	Saves the currently open wallet in the current working directory.
Save As	Enables you to browse your file system to choose a directory location in which to save the currently open wallet.
Save In System Default	Saves the currently open wallet in the system default location: ■ (UNIX) `/etc/ORACLE/WALLETS/<username>` ■ (Windows) `%USERPROFILE%\<username>`
Delete	Deletes the wallet in the current working directory. You must supply the wallet password.

Table 2–3 (Cont.) Oracle Wallet Manager Wallet Menu Options(Cont.)

Option	Description
Change Password	Changes the password for the currently open wallet. You must supply the old password before you can create a new one.
Auto Login	Sets the auto login feature for the currently open wallet. See **auto login wallet**
Exit	Exits the Oracle Wallet Manager application

Operations Menu Table 2–4 describes the contents of the **Operations** menu.

Table 2–4 Oracle Wallet Manager Operations Menu Options

Option	Description
Add Certificate Request	Generates a certificate request for the currently open wallet that you can use to request a certificate from a **certificate authority** (CA).
Import User Certificate	Imports the user certificate issued to you from the CA. You must import the issuing CA's certificate as a trusted certificate before you can import the user certificate.
Import Trusted Certificate	Imports the CA's trusted certificate.
Remove Certificate Request	Deletes the certificate request in the currently open wallet. You must remove the associated user certificate before you can delete a certificate request.
Remove User Certificate	Deletes the user certificate from the currently open wallet.
Remove Trusted Certificate	Removes the trusted certificate that is selected in the navigator pane from the currently open wallet. You must remove all user certificates that the trusted certificate signs before you can remove it.
Export User Certificate	Exports the user certificate in the currently open wallet to save in a file system directory.
Export Certificate Request	Exports the certificate request in the currently open wallet to save in a file.
Export Trusted Certificate	Exports the trusted certificate that is selected in the navigator pane to save in another location in your file system.
Export All Trusted Certificates	Exports all trusted certificates in the currently open wallet to save in another location in your file system.
Export Wallet	Exports the currently open wallet to save as a text file.

Help Menu Table 2–5 describes the contents of the **Help** menu.

Table 2–5 Oracle Wallet Manager Help Menu Options

Option	Description
Contents	Opens Oracle Wallet Manager online help.
Search for Help on	Opens Oracle Wallet Manager online help and displays the Search tab.
About Oracle Wallet Manager	Opens a window that displays the Oracle Wallet Manager version number and copyright information.

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 to create signed certificates for testing purposes.

The basic syntax for this utility is as follows:

```
orapki module command -option_1 argument ... -option_n argument
```

For example, the following command lists all CRLs in the CRL subtree in an instance of Oracle Internet Directory that is installed on `machine1.us.acme.com` and that uses port 389:

```
orapki crl list -ldap machine1.us.acme.com:389
```

See Also:

- "Certificate Revocation List Management" on page 7-40 for information about how to use `orapki` to manage CRLs in the directory.
- Appendix E, "orapki Utility" for reference information on all available `orapki` commands

Enterprise User Security Configuration and Management Tools

Enterprise users are database users who are stored and centrally managed in an LDAP directory, such as Oracle Internet Directory. Table 2–6 provides a summary of the tools that are used to configure and manage Enterprise User Security. The following subsections introduce and describe these tools.

Table 2–6 Enterprise User Security Tools Summary

Tool	Task
Database Configuration Assistant	Register and un-register databases in Oracle Internet Directory
Enterprise Security Manager and Enterprise Security Manager Console	■ Configure enterprise domains and databases in Oracle Internet Directory ■ Create users and manage their passwords ■ Manage identity management realm attributes and administrative groups that pertain to Enterprise User Security in Oracle Internet Directory
Oracle Internet Directory Self-Service Console (Delegated Administration Service)	Manage identity management realms in Oracle Internet Directory For information about this tool, refer to *Oracle Internet Directory Administrator's Guide*.
Oracle Net Configuration Assistant	Configure databases Oracle home for directory usage over the network
Oracle Wallet Manager	Manage Oracle wallets for Enterprise User Security
User Migration Utility	Perform bulk migrations of database users to Oracle Internet Directory

Database Configuration Assistant

Database Configuration Assistant is a wizard-based tool which is used to create and configure Oracle databases.

Use Database Configuration Assistant to register a database with the directory. When you register a database with the directory, Database Configuration Assistant creates a distinguished name (DN) for the database and the corresponding entry and subtree in Oracle Internet Directory

Starting Database Configuration Assistant

To start Database Configuration Assistant:

- (UNIX) From *$ORACLE_HOME*/bin, enter the following at the command line:

  ```
  dbca
  ```

- (Windows) Choose **Start** > **Programs** > **Oracle -** ***HOME_NAME*** > **Database Administration** > **Database Configuration Assistant**

See Also:

- "To register a database in the directory:" on page 12-9 for information about using this tool to register your database.
- *Oracle Database Administrator's Guide* for more information about this tool.

Enterprise Security Manager and Enterprise Security Manager Console

Oracle Advanced Security employs Enterprise Security Manager and Enterprise Security Manager Console to administer enterprise users, administrative groups, **enterprise domains**, and **enterprise roles** that are stored in Oracle Internet Directory. (Enterprise Security Manager Console can be accessed through the Enterprise Security Manager **Operations** menu. See "Enterprise Security Manager Console Overview" on page 2-22 for details.)

Enterprise users are users who are provisioned and managed centrally in an LDAP-compliant directory, such as Oracle Internet Directory, for database access. Enterprise domains are directory constructs that contain databases and enterprise roles, the access privileges that are assigned to enterprise users.

See Also: Chapter 11, "Getting Started with Enterprise User Security" for a discussion of Enterprise User Security administrative groups, enterprise domains, enterprise roles, enterprise users, shared schemas, and user-schema mappings.

This section discusses the following topics:

- Enterprise Security Manager Initial Installation and Configuration Overview
- Starting Enterprise Security Manager
- Navigating the Enterprise Security Manager User Interface
- Enterprise Security Manager Console Overview

- Logging in to Enterprise Security Manager Console
- Navigating Enterprise Security Manager Console User Interface

Enterprise Security Manager Initial Installation and Configuration Overview

The following tasks provide an overview of the initial Enterprise Security Manager installation and configuration:

- Task 1: Install Enterprise Security Manager
- Task 2: Configure an Oracle Identity Management Infrastructure

Task 1: Install Enterprise Security Manager Enterprise Security Manager is automatically installed by the Oracle Database Enterprise Edition server installation process.

> **See Also:** The Oracle Database installation documentation for your operating system.

Note: Use only the version of Enterprise Security Manager that installs with Oracle Database 10*g* Release 1 (10.1).

Task 2: Configure an Oracle Identity Management Infrastructure Enterprise User Security uses Oracle Internet Directory in which to store enterprise users. Enterprise Security Manager uses Oracle Internet Directory Delegated Administration Services to provide an administrative GUI (Enterprise Security Manager Console), and OracleAS Single Sign-On server to authenticate administrators when they log in to the console. Consequently, Oracle Internet Directory and OracleAS Single Sign-On server, which are part of the Oracle Identity Management infrastructure, must be properly installed and configured before Enterprise Security Manager can be used to manage Enterprise User Security. The following elements of Oracle Identity Management infrastructure configuration must be completed before proceeding:

- Oracle Internet Directory 10*g* (9.0.4) must be installed, running, and accessible over standard LDAP or Secure Sockets Layer LDAP (LDAP/SSL).
- Oracle Internet Directory must include an identity management realm. You can use Oracle Internet Directory Configuration Assistant to configure this on the directory server.

- OracleAS Single Sign-On server must be installed and configured to authenticate enterprise user security administrators when they log in to the Enterprise Security Manager Console, an element of Enterprise Security Manager.

  See Also:

  - *Oracle Internet Directory Administrator's Guide* for information about using Oracle Internet Directory Configuration Assistant to create or upgrade an identity management realm in the directory. This manual also contains general information about how to configure and use the directory.
  - *OracleAS Single Sign-On Administrator's Guide* for information about configuring OracleAS Single Sign-On Server.

Starting Enterprise Security Manager

To launch Enterprise Security Manager, use the following steps:

1. Depending on your operating system, use one of the following options:

   - (UNIX) From $*ORACLE_HOME*/bin, enter the following at the command line:

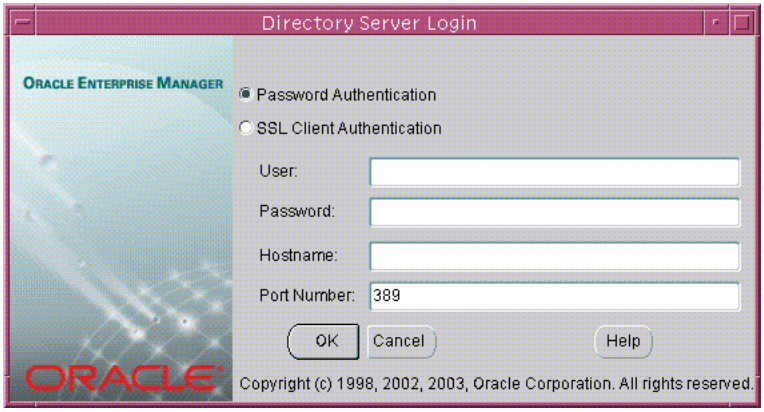
     ```
     esm
     ```

   - (Windows)

     Choose **Start > Programs > Oracle - *HOME_NAME* > Integrated Management Tools > Enterprise Security Manager**

   The directory server login window appears:

Figure 2–4 Directory Server Login Window



2. Log in to Oracle Internet Directory by selecting the authentication method and providing the hostname and port number for your directory. Table 2–7 describes the two available Enterprise Security Manager authentication methods and what each method requires:

Table 2–7 Enterprise Security Manager Authentication Methods

Authentication Method	Description
Password Authentication	Uses simple authentication requiring a **distinguished name (DN)** or a known directory user name and password[1].
SSL Client Authentication	Uses two-way SSL authentication in which both the client and server use Oracle Wallets containing digital certificates (that is, the user name and certificate). The subsequent connection is encrypted.

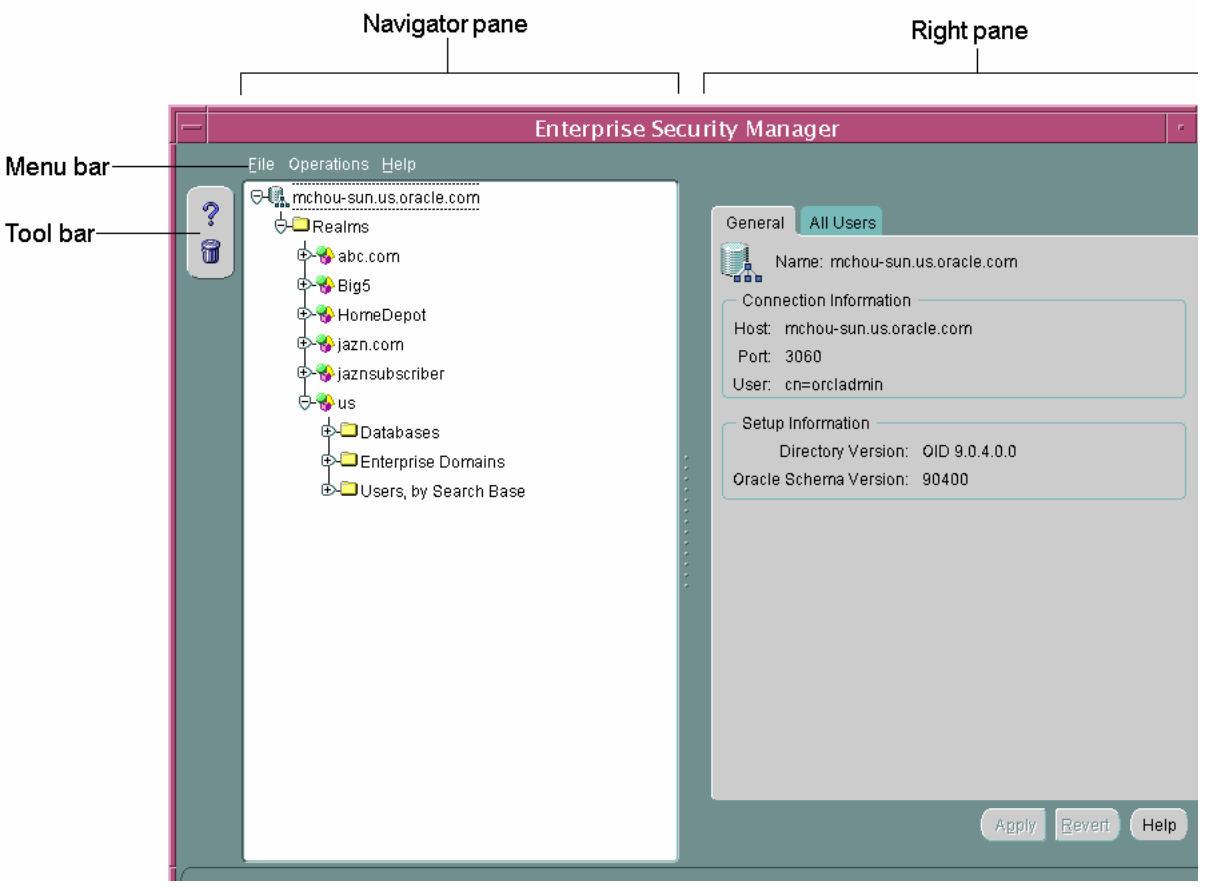
[1] Known directory user name and password can be used only for the default identity management realm in the directory.

3. After providing the directory login information, click **OK**. The main Enterprise Security Manager user interface appears.

Navigating the Enterprise Security Manager User Interface

The Enterprise Security Manager user interface includes two panes, a toolbar, and various menu items as shown in Figure 2–5.

Figure 2–5 Enterprise Security Manager User Interface



Navigator Pane The navigator pane provides a graphical tree view of your directory's identity management realms and the databases, enterprise domains, and users they contain. You can use the navigator pane to view, modify, add, or delete enterprise domains and the objects they contain.

The navigator pane enables you to

- Expand and contract identity management realms by clicking the plus and minus symbols (+ -) adjacent to the realm name in the navigation tree. This enables you to manage the enterprise domains that they contain.

- Right-click an enterprise domain to perform operations such as creating enterprise roles or deleting the domain from the identity management realm.

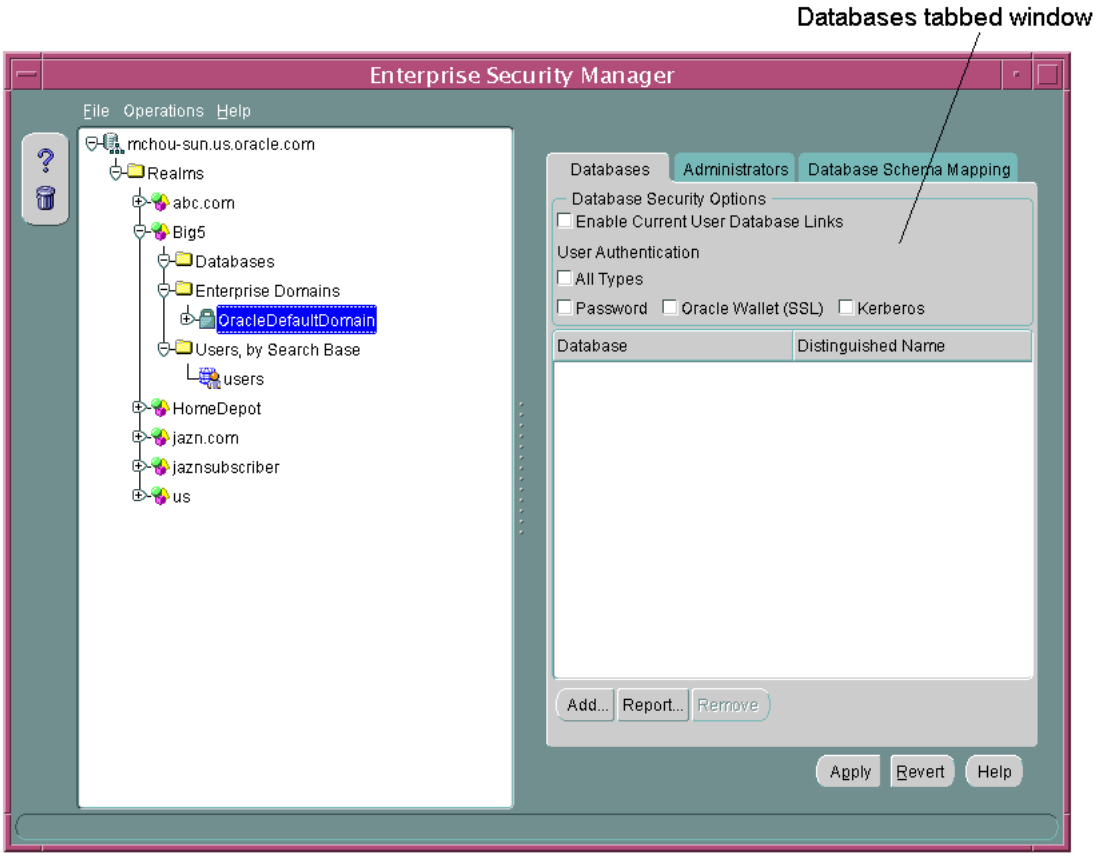
When you expand an identity management realm, you see a nested list of folders that contain enterprise user security objects. Expanding these folders enables you to view the individual objects as described in Table 2–8.

Table 2–8 Enterprise Security Manager Navigator Pane Folders

Folder	Description
Databases	When you expand this folder, you see the databases which are registered with this identity management realm. Databases are registered with a directory by using Database Configuration Assistant.
Enterprise Domains	When you expand this folder, you see the enterprise domains that this realm contains. You can also expand each enterprise domain to view the databases and enterprise roles that it contains.
Users, by Search Base	When you expand this folder, you see the users stored in the realm. The display of users is organized by *search base*, which is the node in the directory under which a collection of users resides.

Right Pane The right pane displays read-only information about an object that is selected in the navigator pane, or it displays tabbed windows that enable you to configure enterprise domains, enterprise roles, and user-schema mappings. For example, when you select an enterprise domain in the navigator pane, you can add databases to it by using the Databases tabbed window that is shown in Figure 2–6.

Figure 2–6 Enterprise Security Manager Databases Tabbed Window



The Databases tabbed window also enables you to set security options for databases which are members of an enterprise domain. See "Defining Database Membership of an Enterprise Domain" on page 13-17 for a discussion of configuring enterprise domains by using the Databases tabbed window.

Tool Bar The toolbar contains two buttons that enable you to access the Enterprise Security Manager online help and to delete directory objects.

Menus You use Enterprise Security Manager menus to create or remove enterprise domains and to manage objects within the domains, such as enterprise roles or database membership. The following sections describe the options that are available under each menu.

File Menu Table 2–9 describes the contents of the **File** menu.

Table 2–9 Enterprise Security Manager File Menu Options

Option	Description
Change Directory Connection	Causes the Directory Server Login window to reappear (see Figure 2–4 on page 2-17), enabling you to log in to another directory server.
Directory Search Options	For user searches in the directory, this menu option enables you to configure the maximum number of displayed search results, the maximum search duration, or an LDAP filter.
ESM Console URL	Enables you to specify the URL for your installation of Enterprise Security Manager Console. (See "Enterprise Security Manager Console Overview" on page 2-22)
Exit	Exits the Enterprise Security Manager application.

Operations Menu Table 2–10 describes the contents of the **Operations** menu.

Table 2–10 Enterprise Security Manager Operations Menu Options

Option	Description
Create Enterprise Domain	Creates an enterprise domain in the realm that is selected in the navigator pane.
Remove Enterprise Domain	Removes the enterprise domain that is selected in the navigator pane.
Create Enterprise Role	Creates an enterprise role in the enterprise domain that is selected in the navigator pane.
Remove Enterprise Role	Removes the enterprise role that is selected in the navigator pane.
Launch ESM Console	Brings up the Enterprise Security Manager Console in your default browser.

Help Menu Table 2–11 describes the contents of the **Help** menu.

Table 2–11 Enterprise Security Manager Help Menu Options

Option	Description
Contents	Opens the online help and displays its table of contents.

Table 2–11 (Cont.) Enterprise Security Manager Help Menu Options

Option	Description
Search for Help on	Displays the search window for the online help.
Using Help	Displays online help topics that describe how to use the online help system
About Enterprise Security Manager	Displays Enterprise Security Manager version number and copyright information

Enterprise Security Manager Console Overview

Enterprise Security Manager uses a directory management console, Enterprise Security Manager Console, to administer enterprise users and groups, and to configure an identity management realm for Enterprise User Security. By default, when you log in to a directory server with Enterprise Security Manager it uses port 7777 with the fully qualified domain name of that directory server to construct an Enterprise Security Manager Console URL. Then, when you need to launch the console, Enterprise Security Manager uses this URL to connect to it over HTTP.

For example, if an Acme Company administrator logs into an instance of Oracle Internet Directory that is hosted on a machine named `machine123`, then Enterprise Security Manager would use the following URL to connect to Enterprise Security Manager Console:

```
http://machine123.us.acme.com:7777/
```

After launching the console, administrators must log in by using their OracleAS Single Sign-On username and password pairs.

Logging in to Enterprise Security Manager Console

If you can use the URL that is constructed by default to access an instance of Enterprise Security Manager Console, then use the following steps to log in to the console.

To log in to Enterprise Security Manager Console:

1. From the Enterprise Security Manager main application window, choose **Operations** > **Launch ESM Console**.

   The Enterprise Security Manager Console login page appears, as shown in Figure 2–7.

Figure 2–7 Enterprise Security Manager Console Login Page

2. Click the **Login** icon in the upper right-corner of the page to log in with your OracleAS Single Sign-On username and password.

   After providing your OracleAS Single Sign-On credentials, you are returned to the console home page.

To change the default Enterprise Security Manager Console URL:

If you cannot use the default URL to connect to the Enterprise Security Manager Console, then you must enter the appropriate URL before you can launch the console.

1. In the Enterprise Security Manager main application, choose **File** > **ESM Console URL**. The ESM Console URL window appears as shown in Figure 2–8.

Figure 2–8 ESM Console URL Window

2. Enter the appropriate URL for connecting to Enterprise Security Manager Console, and click **OK**.

   This saves the URL information in Enterprise Security Manager so you can launch the console again without reconfiguring the URL.

Configuring Enterprise Security Manager Console for Kerberos-Authenticated Enterprise Users By default, Enterprise Security Manager Console user interface does not display the field where you can configure Kerberos principal names. The first time you create Kerberos-authenticated users in the directory, you must configure this tool to display the `krbPrincipalName` attribute in its Create User window by using the following steps:

1. Log into the Oracle Internet Directory Self-Service Console and choose the **Configuration** tab. See: *Oracle Internet Directory Administrator's Guide* for information about logging in and using the Oracle Internet Directory Self-Service Console.
2. In the Configuration page, select the **User Entry** subtab and click **Next** until the Configure User Attributes page appears.
3. In the Configure User Attributes page, click **Add New Attribute** and the Add New Attribute page appears.
4. In the Add New Attribute page, select **krbPrincipalName** from the **Directory Attribute Name** list (or the attribute that you have configured for `orclCommonKrbPrincipalAttribute` in your identity management realm) and perform the following steps on this page:
   a. Enter `Kerberos Principal Name` for the user interface label.
   b. Check **Searchable** and **Viewable**.
   c. Select **Single Line Text** from the **UI Type** list
   d. Click **Done**.
5. Click **Next** to navigate to the Configure Attribute Categories page, and click **Edit** for **Basic Information** and perform the following steps on this page:

a. Select **krbPrincipalName** in the left category list.

b. Click **Move >** to move **krbPrincipalName** to the right-hand list.

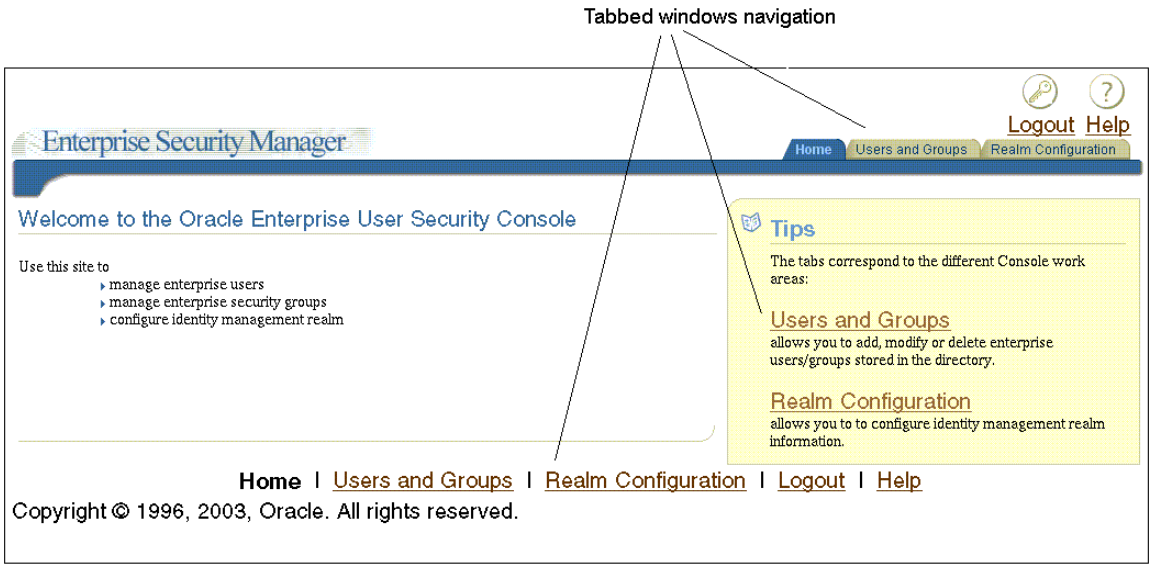
c. Click **Done**.

6. Click **Next** until you reach the last page, and then click **Finish** to save your work.

Navigating Enterprise Security Manager Console User Interface

The Enterprise Security Manager Console user interface is browser-based and uses tabbed windows instead of a navigator pane. Figure 2–9 shows the layout of the console user interface. The tabbed windows can be accessed by selecting one of the tabs at the top of the application or by selecting one of the links in the Tips box on the right. You can also access the tabbed windows by selecting one of the corresponding links at the bottom of the page.

Figure 2–9 Enterprise Security Manager Console User Interface

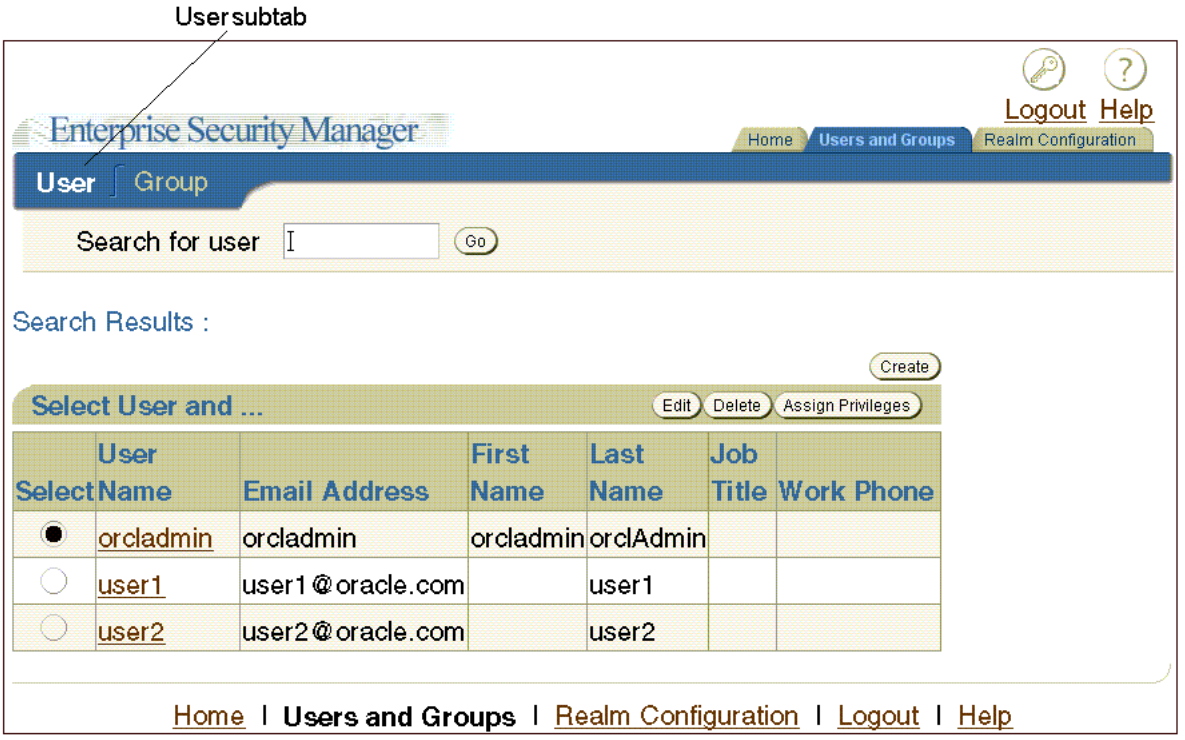


The tabbed windows are explained in the following sections:

Home Tabbed Window The Home page is your entry point to the console. You can access each tabbed window and read a brief summary of what you can do with this tool. The Home tabbed window is shown in Figure 2–9 on page 2-25.

Users and Groups Tabbed Window This tabbed window contains two subtabs: the Users subtab (shown in Figure 2–10) and the Groups subtab (shown in Figure 2–11 on page 2-28).

Figure 2–10 Enterprise Security Manager Console Users Subtab



The Users subtab (Figure 2–10) enables you to search for users in the directory by using the **Search for user** field at the top of the page. After you locate users that match your search criteria, you can select specific users and perform tasks with the buttons that are listed in Table 2–12 on page 2-27. This subtab also enables you to create new users.

Table 2–12 Enterprise Security Manager Console User Subtab Buttons

Button Name	Description
Go	After entering user search criteria in the **Search for user** field, click **Go** to display users who match your search criteria in the **Search Results** table. This button is always available.
Create	Enables you to create new enterprise users in the directory. This button is always available.
Edit	Enables you to edit a user's information in the directory. This button is available only after you have entered search criteria in the **Search for user** field and clicked **Go**.
Delete	Enables you to delete a user from the directory. This button is available only after you have entered search criteria in the **Search for user** field and clicked **Go**.
Assign Privileges	Enables you to assign directory privileges to a specified user. For example, you can assign the privilege to create new users by using this button. This button is available only after you have entered search criteria in the **Search for user** field and clicked **Go**.

The Group subtab (shown in Figure 2–11 on page 2-28) enables you to view, or to add new users or groups to the Enterprise User Security directory administrative groups. To view or edit an administrative group, select the adjacent radio button, and click **Edit** in the upper right corner of the page. When you click **Edit**, an Edit Group page for the specified group appears, displaying the following information:

- Members of the group
- Groups of which the specified administrative group is a member
- Edit history for the group

You can add members or other groups to a specified Enterprise User Security directory administrative group by clicking either **Add User** or **Add Group** in the Member region of the Edit Group page, which is shown in Figure 2–12 on page 2-29.

Figure 2–11 Enterprise Security Manager Console Group Subtab

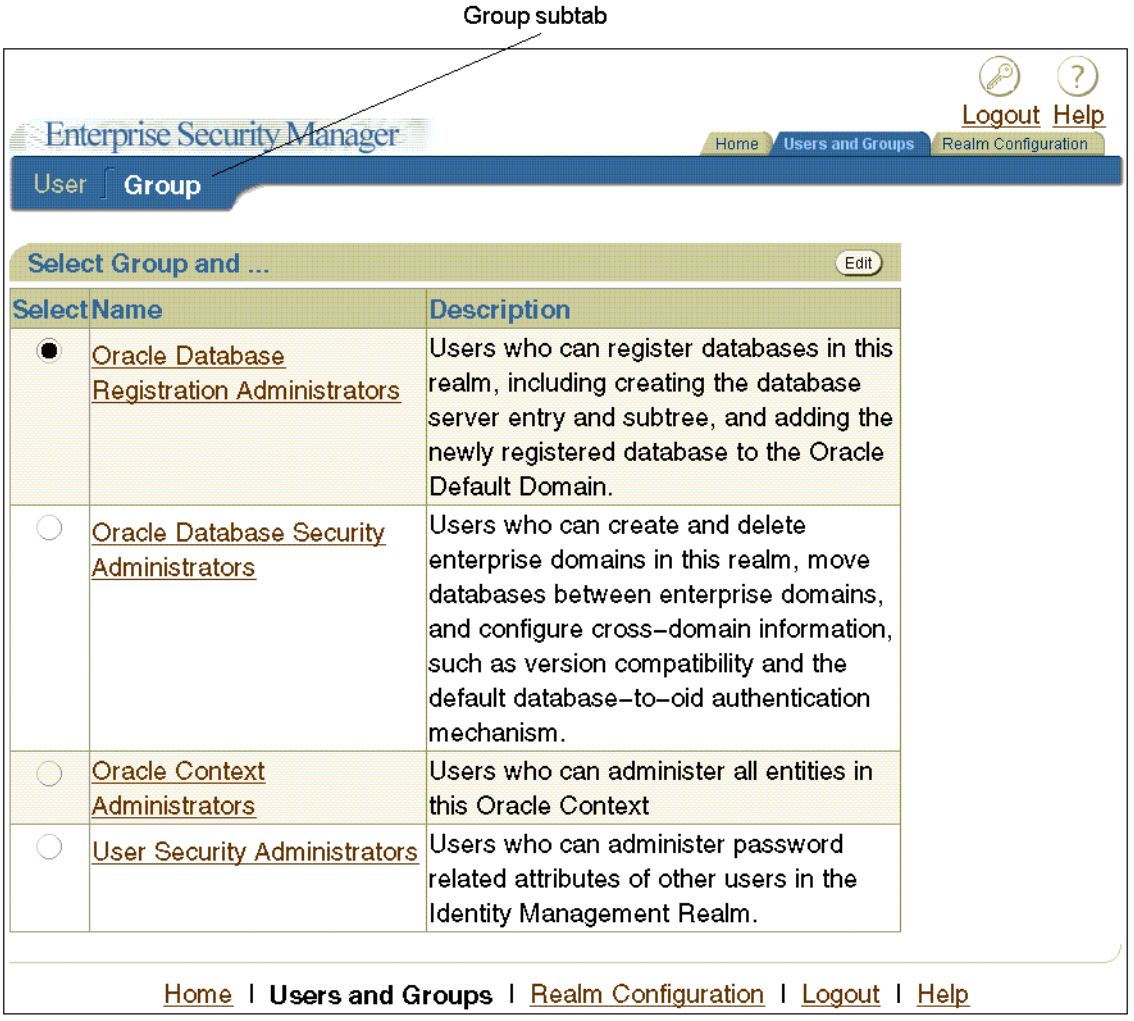
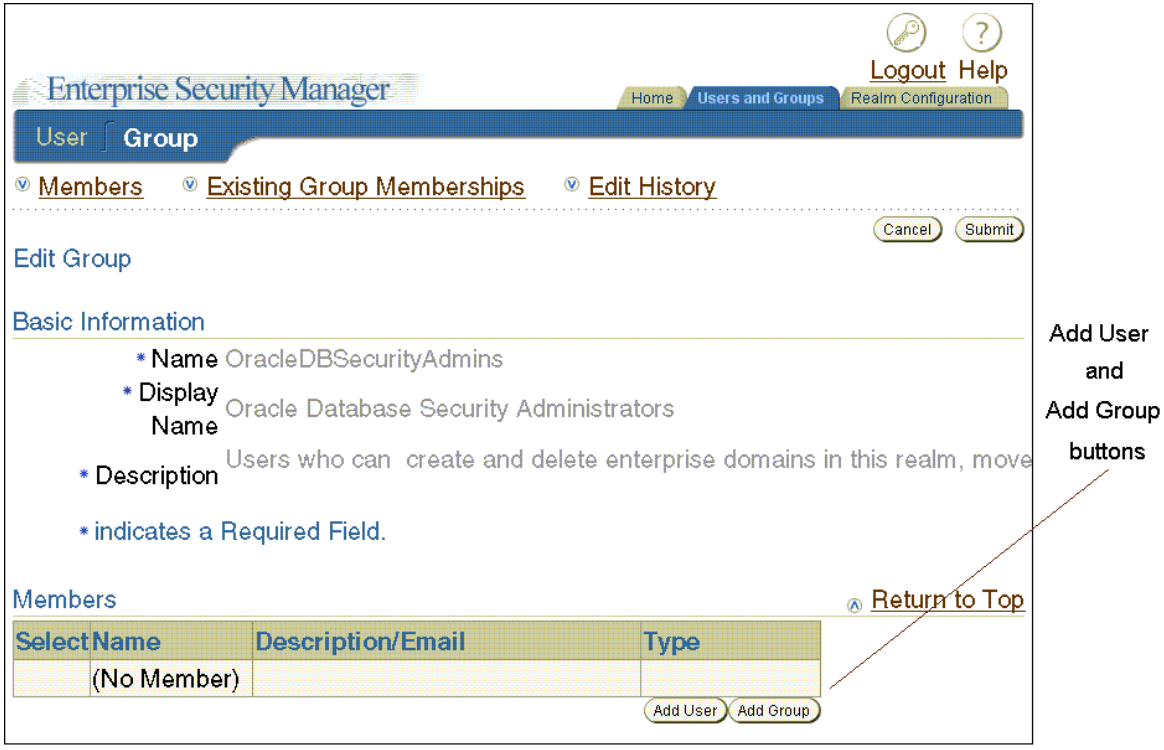


Figure 2–12 Enterprise Security Manager Console Edit Group Page



Realm Configuration Tabbed Window The Realm Configuration tabbed window, which is shown in Figure 2–13, enables you to configure identity management realm attributes that pertain to Enterprise User Security. The fields that you can edit on this page are described in Table 2–13 on page 2-30.

Figure 2–13 Enterprise Security Manager Console Realm Configuration Tabbed Window

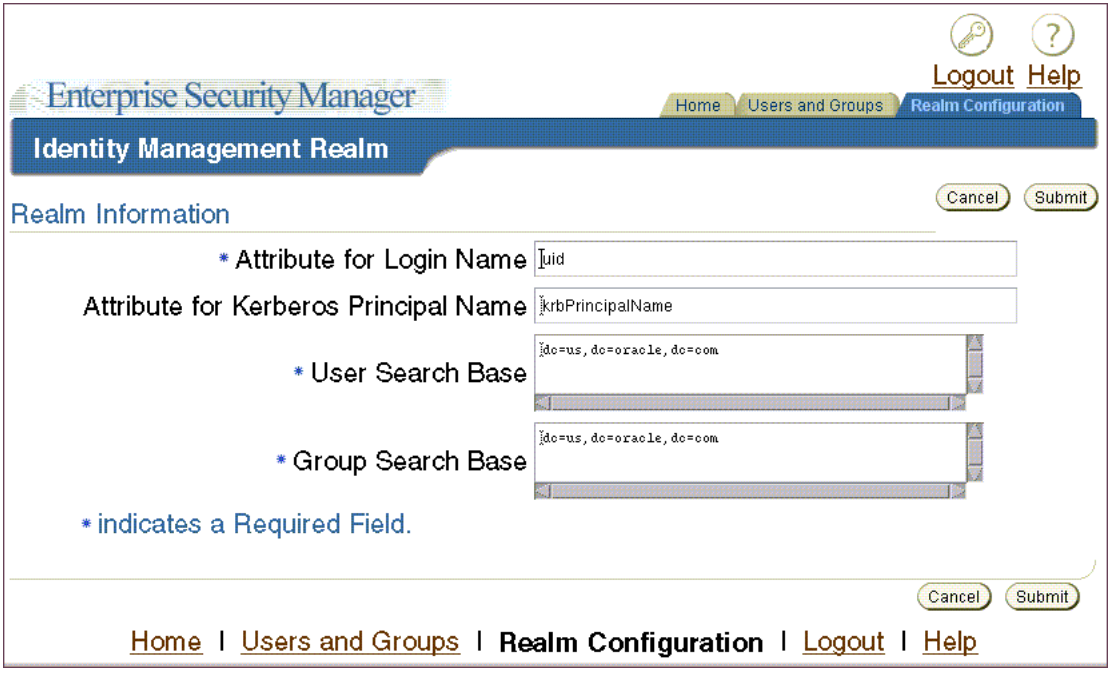


Table 2–13 Realm Configuration Tabbed Window Fields

Field	Description
Attribute for Login Name	Name of the directory attribute used to store login names.
Attribute for Kerberos Principal Name	Name of the directory attribute used to store Kerberos principal names. See also: "Configuring Enterprise Security Manager Console for Kerberos-Authenticated Enterprise Users" on page 2-24
User Search Base	Full **distinguished name (DN)** for the node under which enterprise users are stored for this realm.
Group Search Base	Full DN for the node at which user groups (not Enterprise User Security administrative groups) are stored in the directory.

Enterprise Security Manager Command-Line Utility

Enterprise Security Manager provides a command-line utility, which can be used to perform the most common tasks that the graphical user interface tool performs. Enter all Enterprise Security Manager command-line utility commands from the Oracle Enterprise Manager Oracle home.

The basic syntax for this utility is as follows:

```
esm -cmd [operation] [-option_1 -option_2 -option_3 ... -option_n]
```

For example, the following command searches for users in a directory that is installed on a host machine named `machine1.us.acme.com`:

```
esm -cmd search -U SIMPLE -D orcladmin -w Y4ilbqve -h machine1.us.acme.com
    -p 3060 -dn dc=us,dc=acme,dc=com -objectType user
```

The following table describes each option used in this example:

Command Option	Description
`-U`	Specifies which authentication type used to log in to the directory. `SIMPLE` specifies password authentication.
`-D`	Specifies the username.
`-w`	Specifies the password.
`-h`	Specifies the directory host machine name.
`-p`	Specifies the directory port number.
`-dn`	Specifies the search base.
`-objectType`	Specifies the type of object for which to search.

Accessing Enterprise Security Manager Command-Line Utility Help To view a full list of operations and options you can use with this utility, enter the following at the command line:

```
esm -cmd
```

To view help on a specific operation, enter the following at the command line:

```
esm -cmd help [operation]
```

See Also:

- "Duties of an Enterprise User Security Administrator/DBA" on page 2-35 for a list of tasks that can be performed with Enterprise Security Manager and Enterprise Security Manager Console.
- Chapter 13, "Administering Enterprise User Security" for detailed information about how to use Enterprise Security Manager and Enterprise Security Manager Console to administer enterprise users.

Oracle Net Configuration Assistant

Oracle Net Configuration Assistant is a wizard-based tool that has a graphical user interface. It is primarily used to configure basic Oracle Net network components, such as listener names and protocol addresses. It also enables you to configure your Oracle home for directory server usage. The latter use is what makes this tool important for configuring Enterprise User Security.

If you use Domain Name System (DNS) discovery (automatic domain name lookup) to locate Oracle Internet Directory on your network, then this tool is not necessary. Note that using DNS discovery is the recommended configuration. See *Oracle Internet Directory Administrator's Guide* for information about this configuration.

If you have not configured DNS discovery of Oracle Internet Directory on your network, then you must use Oracle Net Configuration Assistant to create an `ldap.ora` file for your Oracle home before you can register a database with the directory. Your database uses the `ldap.ora` file to locate the correct Oracle Internet Directory server on your network. This configuration file contains the hostname, port number, and identity management realm information for your directory server.

Starting Oracle Net Configuration Assistant

To start Oracle Net Configuration Assistant:

- (UNIX) From *`$ORACLE_HOME`*`/bin`, enter the following at the command line:

  ```
  netca
  ```

- (Windows) Choose **Start** > **Programs** > **Oracle-HOME_NAME** > **Configuration and Migration Tools** > **Net Configuration Assistant**

After you start this tool, you will be presented with the opening page that is shown in Figure 2–14 on page 2-33.

Choose the **Directory Usage Configuration** option on this page, click **Next**, and choose the directory server where you wish to store your enterprise users. Then click **Finish** to create a properly configured `ldap.ora` file for your Oracle home.

Figure 2–14 Opening Page of Oracle Net Configuration Assistant

See Also:

- "Task 5: (Optional) Configure your Oracle home for directory usage" on page 12-7 for more information about using this tool to configure your Oracle home for Enterprise User Security.
- *Oracle Net Services Administrator's Guide* and Oracle Net Configuration Assistant online help for complete documentation of this tool.

User Migration Utility

User Migration Utility is a command-line tool that enables you to perform bulk migrations of database users to Oracle Internet Directory where they are stored and managed as enterprise users. This tool performs a bulk migration in two phases: In

phase one, it populates a table with database user information. During phase two, the database user information is migrated to the directory.

This tool is automatically installed in the following location when you install an Oracle Database client:

```
$ORACLE_HOME/rdbms/bin/umu
```

The basic syntax for this utility is as follows:

```
umu parameter_keyword_1=value1:value2
    parameter_keyword_2=value
    parameter_keyword_3=value1:value2:value3
    .
    .
    .
    parameter_keyword_n=value
```

Note that when a parameter takes multiple values, they are separated with colons (:).

> **See Also:** Appendix G, "Using the User Migration Utility" for complete instructions (including usage examples) for using this tool to migrate database users to a directory and its parameters.

Duties of a Security Administrator/DBA

Most of the tasks of a security administrator involve ensuring that the connections to and from Oracle databases are secure. Table 2–14 lists the primary tasks of security administrators, the tools used to perform the tasks, and links to where the tasks are documented.

Table 2–14 Common Security Administrator/DBA Configuration and Administrative Tasks

Task	Tools Used	See Also
Configure encrypted Oracle Net connections between database servers and clients	Oracle Net Manager	"Configuring Encryption on the Client and the Server" on page 3-9
Configure checksumming on Oracle Net connections between database servers and clients	Oracle Net Manager	"Configuring Integrity on the Client and the Server" on page 3-11
Configure database clients to accept RADIUS authentication	Oracle Net	"Step 1: Configure RADIUS on the Oracle Client" on page 5-9

Table 2–14 (Cont.) Common Security Administrator/DBA Configuration and Administrative Tasks

Task	Tools Used	See Also
Configure a database to accept RADIUS authentication	Oracle Net	"Step 2: Configure RADIUS on the Oracle Database Server" on page 5-10
Create a RADIUS user and grant them access to a database session	SQL*Plus	"Task 3: Create a User and Grant Access" on page 5-17
Configure Kerberos authentication on a database client and server	Oracle Net Manager	"Task 7: Configure Kerberos Authentication" on page 6-5
Create a Kerberos database user	▪ `kadmin.local` ▪ Oracle Net Manager	▪ "Task 8: Create a Kerberos User" on page 6-10 ▪ "Task 9: Create an Externally Authenticated Oracle User" on page 6-10
Manage Kerberos credentials in the credential cache	▪ `okinit` ▪ `oklist` ▪ `okdstry`	▪ "Obtaining the Initial Ticket with the okinit Utility" on page 6-11 ▪ "Displaying Credentials with the oklist Utility" on page 6-12 ▪ "Removing Credentials from the Cache File with the okdstry Utility" on page 6-13
Create a wallet for a database client or server	▪ Oracle Wallet Manager	"Creating a New Wallet" on page 8-10
Request a user certificate from a **certificate authority** (CA) for SSL authentication	▪ Oracle Wallet Manager	▪ "Adding a Certificate Request" on page 8-21 ▪ "Importing the User Certificate into the Wallet" on page 8-22
Import a user certificate and its associated trusted certificate (CA certificate) into a wallet	▪ Oracle Wallet Manager	▪ "Importing a Trusted Certificate" on page 8-25 ▪ "Importing the User Certificate into the Wallet" on page 8-22
Configuring SSL connections for a database client	▪ Oracle Net Manager	"Task 3: Configure SSL on the Client" on page 7-23
Configuring SSL connections for a database server	▪ Oracle Net Manager	"Task 2: Configure SSL on the Server" on page 7-15
Enabling certificate validation with **certificate revocation lists**	▪ Oracle Net Manager	▪ "Configuring Certificate Validation with Certificate Revocation Lists" on page 7-37

Duties of an Enterprise User Security Administrator/DBA

Enterprise User Security administrators plan, implement, and administer enterprise users. Table 2–15 lists the primary tasks of Enterprise User Security administrators, the tools used to perform the tasks, and links to where the tasks are documented.

Table 2–15 Common Enterprise User Security Administrator Configuration and Administrative Tasks

Task	Tools Used	See Also
Create an identity management realm in Oracle Internet Directory	Oracle Internet Directory Self-Service Console (Delegated Administration Service)	*Oracle Internet Directory Administrator's Guide* for information about how to perform this task
Upgrade an identity management realm in Oracle Internet Directory	Oracle Internet Directory Configuration Assistant	*Oracle Internet Directory Administrator's Guide* and the online help for this tool
Set up DNS to enable automatic discovery of Oracle Internet Directory over the network. Note that this is the recommended configuration.	Oracle Internet Directory Configuration Assistant	*Oracle Internet Directory Administrator's Guide* (Domain Name System server discovery) and the online help for this tool
Create an `ldap.ora` file to enable directory access	Oracle Net Configuration Assistant	"Task 5: (Optional) Configure your Oracle home for directory usage" on page 12-7
Register a database in the directory	Database Configuration Assistant	"Task 6: Register the database in the directory" on page 12-8
Configure password authentication for Enterprise User Security	■ Enterprise Security Manager ■ Oracle Net Manager	"Configuring Enterprise User Security for Password Authentication" on page 12-16
Configure Kerberos authentication for Enterprise User Security	■ Oracle Net Manager ■ Enterprise Security Manager Console ■ Enterprise Security Manager	"Configuring Enterprise User Security for Kerberos Authentication" on page 12-18
Configure SSL authentication for Enterprise User Security	■ Oracle Net Manager ■ Enterprise Security Manager ■ text editor or SQL*Plus ■ Oracle Wallet Manager	"Configuring Enterprise User Security for SSL Authentication" on page 12-21
Create or modify user entries and Oracle administrative groups in the directory	Enterprise Security Manager Console	■ "Administering Identity Management Realms" on page 13-3 ■ "Administering Enterprise Users" on page 13-8
Create or modify enterprise roles and domains in the directory	Enterprise Security Manager	■ "Administering Enterprise Domains" on page 13-15 ■ "Administering Enterprise Roles" on page 13-27
Create or modify wallets for directory, databases, and clients	Oracle Wallet Manager	Chapter 8, "Using Oracle Wallet Manager"
Change a user's database or directory password	Enterprise Security Manager Console	"Setting Enterprise User Passwords" on page 13-10
Change a database's directory password	Database Configuration Assistant	"To change the database's directory password:" on page 12-9

Table 2–15 (Cont.) Common Enterprise User Security Administrator Configuration and Administrative

Task	Tools Used	See Also
Manage user wallets on the local system or update database and directory user passwords	Oracle Wallet Manager	Chapter 8, "Using Oracle Wallet Manager"
Request initial Kerberos ticket when KDC is not part of the operating system, such as Kerberos V5 from MIT	`okinit` utility	"Task 10: Get an Initial Ticket for the Kerberos / Oracle User" on page 6-11
Migrate large numbers of local or external database users to the directory for Enterprise User Security	User Migration Utility	Appendix G, "Using the User Migration Utility"

Part II

Network Data Encryption and Integrity

This part describes how to configure data encryption and integrity for your existing Oracle network, and for thin JDBC connections to the database by using the encryption features of Oracle Advanced Security. It contains the following chapters:

- Chapter 3, "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients"
- Chapter 4, "Configuring Network Data Encryption and Integrity for Thin JDBC Clients"

See Also: Oracle operating system-specific documentation

3

Configuring Network Data Encryption and Integrity for Oracle Servers and Clients

This chapter describes how to configure native Oracle Net Services data **encryption** and **integrity** for Oracle Advanced Security. It contains the following topics:

Oracle Advanced Security Encryption

This section describes data encryption algorithms available in the current release of Oracle Advanced Security:

Note: Prior to Release 8.1.7, Oracle Advanced Security provided three editions: Domestic, Upgrade, and Export—each with different key lengths. This release now 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.

About Encryption

The purpose of a secure cryptosystem is to convert **plaintext** data into unintelligible **ciphertext** based on a key, in such a way that it is very hard (computationally infeasible) to convert ciphertext back into its corresponding plaintext without knowledge of the correct key. In a symmetric cryptosystem, the same key is used both for encryption and decryption of the same data. Oracle Advanced Security provides the Advanced Encryption Standard (AES), DES, 3DES, and RC4 symmetric cryptosystems for protecting the confidentiality of Oracle Net Services traffic.

Advanced Encryption Standard

In this release, the new Federal Information Processing Standard (FIPS) encryption algorithm, Advanced Encryption Standard (AES), is supported. AES can be used by all U.S. government organizations and businesses to protect sensitive data over a network. This encryption algorithm defines three standard key lengths, which are 128-bit, 192-bit, and 256-bit. All versions operate in outer **Cipher Block Chaining (CBC)** mode.

DES Algorithm Support

Oracle Advanced Security provides the Data Encryption Standard (DES) algorithm. DES has been a U.S. government standard for many years and is sometimes mandated in the financial services industry. Because it has been a standard for so long, DES is deployed throughout the world for use in a wide variety of applications.

Triple-DES Support

Oracle Advanced Security 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 a performance penalty. The magnitude of the performance penalty depends on the speed of the processor performing the encryption. 3DES typically takes three times as long to encrypt a data block when 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.

DES40 Algorithm

The DES40 algorithm, available in every release of Oracle Advanced Security, Oracle Advanced Networking Option, and Secure Network Services, is a variant of DES in which the secret key is preprocessed to provide 40 effective key bits. It was designed to provide DES-based encryption to customers outside the U.S. and Canada at a time when the U.S. export laws were more restrictive. Now, in Oracle Advanced Security 10*g* Release 1 (10.1), DES40, DES, and 3DES are all available for export. DES40 is still supported to provide backward-compatibility for international customers.

RSA RC4 Algorithm for High Speed Encryption

The RC4 algorithm, developed by RSA Data Security Inc., has become the international standard for high-speed data encryption. RC4 is a variable key-length stream cipher that operates at several times the speed of DES, making it possible to encrypt large, bulk data transfers with minimal performance consequences.

Oracle Advanced Security 10*g* Release 1 (10.1) provides an RC4 implementation with 40-bit, 56-bit, 128-bit, and 256-bit key lengths. This provides backward-compatibility and strong encryption, with no material performance compromise.

See Also:

- "Configuring Encryption on the Client and the Server" on page 3-9.
- Table 3–2, "Valid Encryption Algorithms" on page 3-11.

Oracle Advanced Security Data Integrity

Encryption of network data provides data privacy so that unauthorized parties are not able to view plaintext data as it passes over the network. Oracle Advanced Security also provides protection against two forms of active attack:

- Data modification attack

  This type of attack occurs when an unauthorized party intercepts data in transit, alters it, and retransmits it. For example, if a bank deposit of $100 is intercepted, the monetary amount is changed to $10,000, and then the higher amount is retransmitted, then that is a data modification attack.

- Replay attack

  This type of attack occurs when an entire set of valid data is repetitively retransmitted. For example, if a bank withdrawal of $100 is intercepted and then retransmitted ten times so the final withdrawal amount equals $1,000, then that is a replay attack.

Data Integrity Algorithms Supported

Oracle Advanced Security lets you select a keyed, sequenced implementation of the Message Digest 5 (MD5) algorithm or the Secure Hash Algorithm (SHA-1) to protect against both of these forms of attack. Both of these hash algorithms create a checksum that changes if the data is altered in any way. This protection operates independently from the encryption process so you can enable data integrity with or without enabling encryption.

See Also:

- "Configuring Integrity on the Client and the Server" on page 3-11.
- Table 3–3, "Valid Integrity Algorithms" on page 3-13.

Diffie-Hellman Based Key Management

The secrecy of encrypted data depends upon the existence of a secret key shared between the communicating parties. A key is a secret exclusively shared by parties on both sides of a connection. Without the key, it is extremely difficult (computationally infeasible) to decrypt an encrypted message or to alter a cryptographic, checksummed message without detection. Providing and maintaining such secret keys is referred to as key management.

Secure key distribution is difficult in a multiuser environment. Oracle Advanced Security uses the well known **Diffie-Hellman key negotiation algorithm** to perform secure key distribution for both encryption and data integrity.

When encryption is used to protect the security of encrypted data, keys must be changed frequently to minimize the effects of a compromised key. Accordingly, the

Oracle Advanced Security key management function changes the session key with every session.

Authentication Key Fold-in

The purpose of Authentication Key Fold-in is to defeat a possible third party attack (historically called the *man-in-the-middle attack*) on the Diffie-Hellman key negotiation. It strengthens the session key significantly by combining a shared secret, known only to the client and the server, with the original session key negotiated by Diffie-Hellman.

The client and the server begin communicating using the session key generated by Diffie-Hellman. When the client authenticates to the server, they establish a shared secret that is only known to both parties. Oracle Advanced Security combines the shared secret and the Diffie-Hellman session key to generate a stronger session key designed to defeat a man-in-the-middle attack.

Note: The authentication key fold-in function is an imbedded feature of Oracle Advanced Security and requires no configuration by the system or network administrator.

How To Configure Data Encryption and Integrity

This section describes how to configure Oracle Advanced Security native Oracle Net Services encryption and integrity, and presumes the prior installation of Oracle Net Services.

The network or security administrator sets up the encryption and integrity configuration parameters. The profile on client and server systems using data encryption and integrity (`sqlnet.ora file`) must contain some or all of the parameters listed in this section, under the following topics:

- About Activating Encryption and Integrity
- About Negotiating Encryption and Integrity
- Setting the Encryption Seed (Optional)
- Configuring Encryption and Integrity Parameters Using Oracle Net Manager

See Also: Chapter 7, "Configuring Secure Sockets Layer Authentication", to configure the SSL feature for encryption, integrity, and authentication

About Activating Encryption and Integrity

In any network connection, it is possible for both the client and server to each support more than one encryption algorithm and more than one integrity algorithm. When a connection is made, the server selects which algorithm to use, if any, from those algorithms specified in the `sqlnet.ora` files.

The server searches for a match between the algorithms available on both the client and the server, and picks the first algorithm in its own list that also appears in the client list. If one side of the connection does not specify an algorithm list, all the algorithms installed on that side are acceptable. The connection fails with error message `ORA-12650` if either side specifies an algorithm that is not installed.

Encryption and integrity parameters are defined by modifying a `sqlnet.ora` file on the clients and the servers on the network.

You can choose to configure any or all of the available Oracle Advanced Security encryption algorithms (Table 3–2), and either or both of the available integrity algorithms (Table 3–3). Only one encryption algorithm and one integrity algorithm are used for each connect session.

Note: Oracle Advanced Security selects the first encryption algorithm and the first integrity algorithm enabled on the client and the server. Oracle Corporation recommends that you select algorithms and key lengths in the order in which you prefer negotiation, choosing the strongest key length first.

See Also: Appendix A, "Data Encryption and Integrity Parameters"

About Negotiating Encryption and Integrity

To negotiate whether to turn on encryption or integrity, you can specify four possible values for the Oracle Advanced Security encryption and integrity configuration parameters. The four values are listed in the order of increasing security. The value REJECTED provides the *minimum* amount of security between client and server communications, and the value REQUIRED provides the *maximum* amount of network security:

- REJECTED
- ACCEPTED

- REQUESTED
- REQUIRED

The default value for each of the parameters is ACCEPTED.

REJECTED

Select this value if you do not elect to enable the security service, even if required by the other side.

In this scenario, this side of the connection specifies that the security service is not permitted. If the other side is set to REQUIRED, the connection *terminates* with error message `ORA-12650`. If the other side is set to REQUESTED, ACCEPTED, or REJECTED, the connection continues without error and without the security service enabled.

ACCEPTED

Select this value to enable the security service if required or requested by the other side.

In this scenario, this side of the connection does not require the security service, but it is enabled if the other side is set to REQUIRED or REQUESTED. If the other side is set to REQUIRED or REQUESTED, and an encryption or integrity algorithm match is found, the connection continues without error and with the security service enabled. If the other side is set to REQUIRED and no algorithm match is found, the connection terminates with error message `ORA-12650`.

If the other side is set to REQUESTED and no algorithm match is found, or if the other side is set to ACCEPTED or REJECTED, the connection continues without error and without the security service enabled.

REQUESTED

Select this value to enable the security service if the other side permits it.

In this scenario, this side of the connection specifies that the security service is desired but not required. The security service is enabled if the other side specifies ACCEPTED, REQUESTED, or REQUIRED. There must be a matching algorithm available on the other side—otherwise the service is not enabled. If the other side specifies REQUIRED and there is no matching algorithm, *the connection fails.*

REQUIRED

Select this value to enable the security service or preclude the connection.

In this scenario, this side of the connection specifies that the security service *must be enabled*. The connection *fails* if the other side specifies REJECTED or if there is no compatible algorithm on the other side.

Table 3–1 shows whether the security service is enabled, based on a combination of client and server configuration parameters. If either the server or client has specified REQUIRED, the lack of a common algorithm *causes the connection to fail.* Otherwise, if the service is enabled, lack of a common service algorithm results in the service being *disabled.*

Table 3–1 Encryption and Data Integrity Negotiations

Client Setting	Server Setting	Encryption and Data Negotiation
REJECTED	REJECTED	OFF
ACCEPTED	REJECTED	OFF
REQUESTED	REJECTED	OFF
REQUIRED	REJECTED	Connection fails
REJECTED	ACCEPTED	OFF
ACCEPTED	ACCEPTED	OFF[1]
REQUESTED	ACCEPTED	ON
REQUIRED	ACCEPTED	ON
REJECTED	REQUESTED	OFF
ACCEPTED	REQUESTED	ON
REQUESTED	REQUESTED	ON
REQUIRED	REQUESTED	ON
REJECTED	REQUIRED	Connection fails
ACCEPTED	REQUIRED	ON
REQUESTED	REQUIRED	ON
REQUIRED	REQUIRED	ON

[1] This value defaults to OFF. Cryptography and data integrity are not enabled until the user changes this parameter by using Oracle Net Manager or by modifying the `sqlnet.ora` file.

Setting the Encryption Seed (Optional)

Several seeds are used to generate a random number on the client and on the server. One of the seeds that can be used is a user-defined encryption seed. This is set with

the `sqlnet.crypto_seed` parameter in the `sqlnet.ora` file. It can be 10 to 70 characters in length and changed at any time. The Diffie-Hellman key exchange uses the random numbers to generate unique session keys for every connect session.

Configuring Encryption and Integrity Parameters Using Oracle Net Manager

You can set up or change encryption and integrity parameter settings using Oracle Net Manager. This section describes the following topics:

- Configuring Encryption on the Client and the Server
- Configuring Integrity on the Client and the Server

See Also:

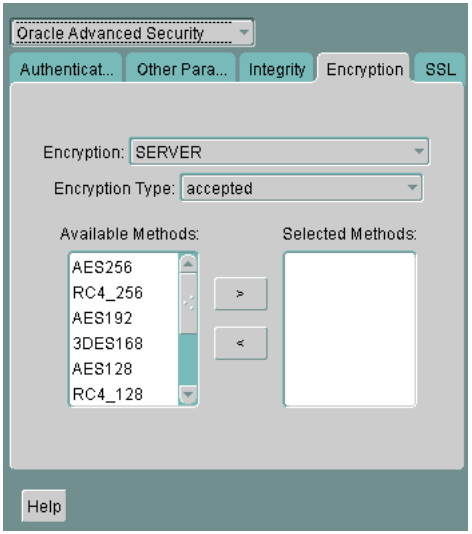
- Appendix A, "Data Encryption and Integrity Parameters", for valid encryption algorithms
- Oracle Net Manager online help, for more detailed configuration information

Configuring Encryption on the Client and the Server

Use Oracle Net Manager to configure encryption on the client and on the server (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security tabbed window appears (Figure 3–1):

Figure 3–1 Oracle Advanced Security Encryption Window



2. Choose the Encryption tab.
3. Depending upon which system you are configuring, select **CLIENT** or **SERVER** from the pull-down list.
4. From the Encryption Type list, select one of the following:
   - REQUESTED
   - REQUIRED
   - ACCEPTED
   - REJECTED
5. (Optional) In the **Encryption Seed** field, enter between 10 and 70 random characters; the encryption seed for the client should not be the same as that for the server.
6. Select an encryption algorithm in the **Available Methods** list. Move it to the **Selected Methods** list by choosing the right arrow [>]. Repeat for each additional method you want to use.
7. Choose **File** > **Save Network Configuration**. The `sqlnet.ora` file is updated.

8. Repeat this procedure to configure encryption on the other system. The `sqlnet.ora` file on the two systems should contain the following entries:

   - On the server:

     ```
     SQLNET.ENCRYPTION_SERVER = [accepted | rejected | requested | required]
     SQLNET.ENCRYPTION_TYPES_SERVER = (valid_encryption_algorithm [,valid_
     encryption_algorithm])
     ```

   - On the client:

     ```
     SQLNET.ENCRYPTION_CLIENT = [accepted | rejected | requested | required]
     SQLNET.ENCRYPTION_TYPES_CLIENT = (valid_encryption_algorithm [,valid_
     encryption_algorithm])
     ```

Valid encryption algorithms and their associated legal values are summarized by Table 3–2:

Table 3–2 Valid Encryption Algorithms

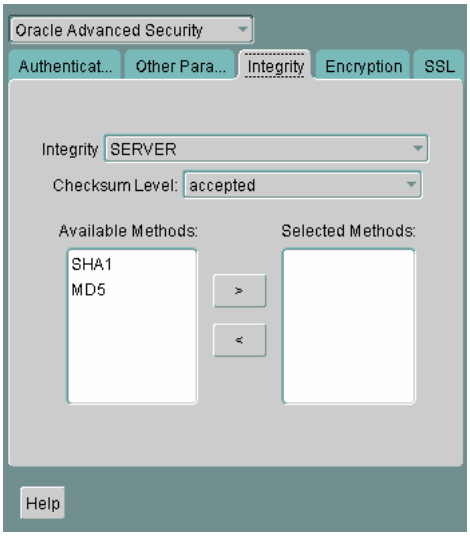
Algorithm Name	Legal Value
RC4 256-bit key	RC4_256
RC4 128-bit key	RC4_128
RC4 56-bit key	RC4_56
RC4 40-bit key	RC4_40
AES 256-bit key	AES256
AES 192-bit key	AES192
AES 128-bit key	AES128
3-key 3DES	3DES168
2-key 3DES	3DES112
DES 56-bit key	DES
DES 40-bit key	DES40

Configuring Integrity on the Client and the Server

Use Oracle Net Manager to configure data integrity on the client and on the server (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile. (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security tabbed window appears (Figure 3–2):

Figure 3–2 Oracle Advanced Security Integrity Window



2. Choose the Integrity tab.
3. Depending upon which system you are configuring, choose the **Server** or **Client** check box.
4. From the **Checksum Level** list, select one of the following checksum level values:
   - REQUESTED
   - REQUIRED
   - ACCEPTED
   - REJECTED
5. Select an integrity algorithm in the **Available Methods** list. Move it to the **Selected Methods** list by choosing the right arrow [>]. Repeat for each additional method you want to use.

6. Choose **File** > **Save Network Configuration**. The `sqlnet.ora` file is updated.
7. Repeat this procedure to configure integrity on the other system. The `sqlnet.ora` file on the two systems should contain the following entries:
   - On the server:

     ```
     SQLNET.CRYPTO_CHECKSUM_SERVER = [accepted | rejected | requested |
     required]
     SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (valid_crypto_checksum_algorithm
     [,valid_crypto_checksum_algorithm])
     ```

   - On the client:

     ```
     SQLNET.CRYPTO_CHECKSUM_CLIENT = [accepted | rejected | requested |
     required]
     SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (valid_crypto_checksum_algorithm
     [,valid_crypto_checksum_algorithm])
     ```

Valid integrity algorithms and their associated legal values are displayed by Table 3–3:

Table 3–3 Valid Integrity Algorithms

Algorithm Name	Legal Values
MD5	MD5
SHA-1	SHA1

4

Configuring Network Data Encryption and Integrity for Thin JDBC Clients

This chapter describes the Java implementation of Oracle Advanced Security, which lets thin Java Database Connectivity (JDBC) clients securely connect to Oracle Databases. This chapter contains the following topics:

- About the Java Implementation
- Configuration Parameters

> **See Also:** *Oracle Database JDBC Developer's Guide and Reference*, for information about JDBC, including examples

About the Java Implementation

The Java implementation of Oracle Advanced Security provides network encryption and integrity protection for Thin JDBC clients communicating with Oracle Databases that have Oracle Advanced Security enabled.

This section contains the following topics:

- Java Database Connectivity Support
- Securing Thin JDBC
- Implementation Overview
- Obfuscation

Java Database Connectivity Support

Java Database Connectivity (JDBC), an industry-standard Java interface, is a Java standard for connecting to a relational database from a Java program. Sun

Microsystems defined the JDBC standard and Oracle Corporation implements and extends the standard with its own JDBC drivers.

Oracle JDBC drivers are used to create JDBC applications to communicate with Oracle databases. Oracle implements two types of JDBC drivers: Thick JDBC drivers built on top of the C-based Oracle Net client, as well as a Thin (Pure Java) JDBC driver to support downloadable applets. Oracle extensions to JDBC include the following features:

- Data access and manipulation
- LOB access and manipulation
- Oracle object type mapping
- Object reference access and manipulation
- Array access and manipulation
- Application performance enhancement

Securing Thin JDBC

Because the Thin JDBC driver is designed to be used with downloadable applets used over the Internet, Oracle designed a 100% Java implementation of Oracle Advanced Security encryption and integrity algorithms for use with thin clients. Oracle Advanced Security provides the following features for Thin JDBC:

- Data encryption
- Data integrity checking
- Secure connections from Thin JDBC clients to the Oracle RDBMS
- Ability for developers to build applets that transmit data over a secure communication channel
- Secure connections from middle tier servers with Java Server Pages (JSP) to the Oracle RDBMS
- Secure connections from Oracle Database 10*g* to older versions of Oracle databases with Oracle Advanced Security installed

The Oracle JDBC Thin driver implements the Oracle O3LOGON protocol for authentication. It does not support Oracle Advanced Security SSL implementation, nor does it support third party authentication features such as RADIUS, Kerberos, and SecurID. However, the Oracle JDBC OCI (thick) driver support is the same as thick client support, where all Oracle Advanced Security features are implemented.

Oracle Advanced Security continues to encrypt and provide integrity checking of Oracle Net Services traffic between Oracle Net clients and Oracle servers using algorithms written in C. The Oracle Advanced Security Java implementation provides Java versions of the following encryption algorithms:

- RC4_256
- RC4_128
- RC4_56
- RC4_40
- DES56
- DES40

> **Note:** In Oracle Advanced Security, DES runs in Cipher Block Chaining (CBC) mode.

In addition, this implementation provides data integrity checking for Thin JDBC using Message Digest 5 (MD5), a cryptographically secure message digest.

Implementation Overview

On the server side, the negotiation of algorithms and the generation of keys function exactly the same as Oracle Advanced Security native encryption. This enables backward and forward compatibility of clients and servers.

On the client side, the algorithm negotiation and key generation occur in exactly the same manner as C-based Oracle Advanced Security encryption. The client and server negotiate encryption algorithms, generate random numbers, use Diffie-Hellman to exchange session keys, and use the Oracle Password Protocol (O3LOGON key fold-in), in the same manner as traditional Oracle Net clients. Thin JDBC contains a complete implementation of a Oracle Net client in pure Java.

Obfuscation

Java cryptography code is *obfuscated* in this release. Obfuscation protects Java classes and methods that contain encryption and decryption capabilities with obfuscation software.

Java byte code **obfuscation** is a process frequently used to protect intellectual property written in the form of Java programs. It mixes up Java symbols found in

the code. The process leaves the original program structure intact, letting the program run correctly while changing the names of the classes, methods, and variables in order to hide the intended behavior. Although it is possible to decompile and read non-obfuscated Java code, obfuscated Java code is sufficiently difficult to decompile to satisfy U.S. government export controls.

Configuration Parameters

A properties class object containing several configuration parameters is passed to the Oracle Advanced Security interface. This chapter lists the configuration parameters for the following:

- Client Encryption Level: ORACLE.NET.ENCRYPTION_CLIENT
- Client Encryption Selected List: ORACLE.NET.ENCRYPTION_TYPES_CLIENT
- Client Integrity Level: ORACLE.NET.CRYPTO_CHECKSUM_CLIENT
- Client Integrity Selected List: ORACLE.NET.CRYPTO_CHEKSUM_TYPES_CLIENT

Client Encryption Level: ORACLE.NET.ENCRYPTION_CLIENT

This parameter defines the level of security that the client wants to negotiate with the server. Table 4–1 describes this parameter's attributes.

Table 4–1 ORACLE.NET.ENCRYPTION_CLIENT Parameter Attributes

Attribute	Description
Parameter Type	String
Parameter Class	Static
Permitted Values	REJECTED; ACCEPTED; REQUESTED; REQUIRED
Default Value	ACCEPTED
Syntax	`up.put("oracle.net.encryption_client",level)`
Example	`up.put("oracle.net.encryption_client", "REQUIRED"), where up is defined as Properties up=new properties()`

Client Encryption Selected List: ORACLE.NET.ENCRYPTION_TYPES_CLIENT

This parameter defines the encryption algorithm to be used. Table 4–2 describes this parameter's attributes.

Table 4–2 ORACLE.NET.ENCRYPTION_TYPES_CLIENT Parameter Attributes

Attribute	Description
Parameter Type	String
Parameter Class	Static
Permitted Values	RC4_256; RC4_128; RC4_56; RC4_40; DES56C; DES40C
Syntax	`up.put("oracle.net.encryption_types_ client",alg)`
Example	`up.put("oracle.net.encryption_types_client", "DES40C"), where up is defined as Properties up=new Properties()`

Note: In this context, "C" refers to CBC (Cipher Block Chaining) mode.

Client Integrity Level: ORACLE.NET.CRYPTO_CHECKSUM_CLIENT

This parameter defines the level of security that it wants to negotiate with the server for data integrity. Table 4–3 describes this parameters attributes.

Table 4–3 ORACLE.NET.CRYPTO_CHECKSUM_CLIENT Parameter Attributes

Attribute	Description
Parameter Type	String
Parameter Class	Static
Permitted Values	REJECTED; ACCEPTED; REQUESTED; REQUIRED
Default Value	ACCEPTED
Syntax	`up.put("oracle.net.crypto_checksum_ client",level)`
Example	`up.put("oracle.net.crypto_checksum_client", "REQUIRED"), where up is defined as Properties up=new Properties()`

Client Integrity Selected List: ORACLE.NET.CRYPTO_CHEKSUM_TYPES_CLIENT

This parameter defines the data integrity algorithm to be used. Table 4–4 describes this parameter's attributes.

Table 4–4 ORACLE.NET.CRYPTO_CHEKSUM_TYPES_CLIENT Parameter Attributes

Attribute	Description
Parameter Type	String
Parameter Class	Static
Permitted Values	MD5
Syntax	`up.put("oracle.net.crypto_checksum_types_client",alg)`
Example	`up.put("oracle.net.crypto_checksum_types_client","MD5"), where up is defined as Properties up=new Properties()`

Part III

Oracle Advanced Security Strong Authentication

This part describes how to configure strong authentication methods for your existing Oracle network. It contains the following chapters, each of which describes a particular authentication method supported by Oracle Advanced Security:

- Chapter 5, "Configuring RADIUS Authentication"
- Chapter 6, "Configuring Kerberos Authentication"
- Chapter 7, "Configuring Secure Sockets Layer Authentication"
- Chapter 8, "Using Oracle Wallet Manager"
- Chapter 9, "Configuring Multiple Authentication Methods and Disabling Oracle Advanced Security"
- Chapter 10, "Configuring Oracle DCE Integration"

Note: Oracle Advanced Security 10*g* Release 1 (10.1) supports dynamic loading of authentication methods. As a consequence, you no longer need to specify all possible authentication methods at install time; you can implement any available authentication method at any time subsequent to the initial installation of Oracle Advanced Security.

5

Configuring RADIUS Authentication

This chapter describes how to configure an Oracle Database server for use with RADIUS (Remote Authentication Dial-In User Service). This chapter contains the following topics:

- RADIUS Overview
- RADIUS Authentication Modes
- Enabling RADIUS Authentication, Authorization, and Accounting
- Using RADIUS to Log In to a Database
- RSA ACE/Server Configuration Checklist

Note: SecurID, an authentication product of RSA Security, Inc., though not directly supported by Oracle Advanced Security, has been certified as RADIUS-compliant. You can therefore run SecurID under RADIUS.

See the RSA Security SecurID documentation for further information.

RADIUS Overview

RADIUS is a client/server security protocol widely used to enable remote authentication and access. Oracle Advanced Security uses this industry standard in a client/server network environment.

You can enable the network to use any authentication method that supports the RADIUS standard, including token cards and smart cards, by installing and configuring the RADIUS protocol. Moreover, when you use RADIUS, you can

change the authentication method without modifying either the Oracle client or the Oracle database server.

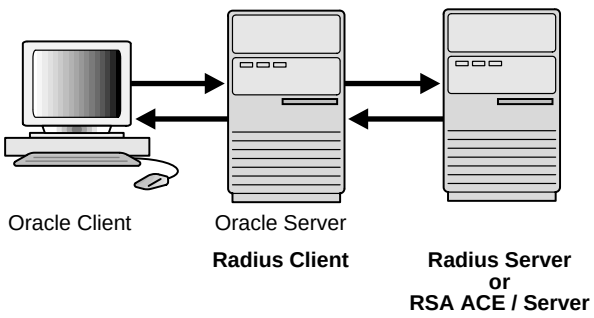
From the user's perspective, the entire authentication process is transparent. When the user seeks access to an Oracle database server, the Oracle database server, acting as the RADIUS client, notifies the RADIUS server. The RADIUS server:

- Looks up the user's security information.
- Passes authentication and authorization information between the appropriate authentication server or servers and the Oracle database server.
- Grants the user access to the Oracle database server.
- Logs session information, including when, how often, and for how long the user was connected to the Oracle database server.

Note: Oracle Advanced Security does not support RADIUS authentication over database links.

The Oracle/RADIUS environment is displayed in Figure 5–1:

Figure 5–1 RADIUS in an Oracle Environment



The Oracle database server acts as the RADIUS client, passing information between the Oracle client and the RADIUS server. Similarly, the RADIUS server passes information between the Oracle database server and the appropriate authentication servers. The authentication components are listed in Table 5–1:

Table 5–1 RADIUS Authentication Components

Component	Stored Information
Oracle client	Configuration setting for communicating through RADIUS.
Oracle database server / RADIUS client	Configuration settings for passing information between the Oracle client and the RADIUS server. The secret key file.
RADIUS server	Authentication and authorization information for all users. Each client's name or IP address. Each client's shared secret. Unlimited number of menu files enabling users already authenticated to select different login options without reconnecting.
Authentication server or servers	User authentication information such as pass codes and PINs, depending on the authentication method in use. **Note**: The RADIUS server can also be the authentication server.

A RADIUS server vendor is often the authentication server vendor as well, in which case authentication can be processed on the RADIUS server. For example, the RSA ACE / Server is both a RADIUS server and an authentication server. It thus authenticates the user's pass code.

> **See Also:** *Oracle Net Services Administrator's Guide*, for information about the `sqlnet.ora` file

RADIUS Authentication Modes

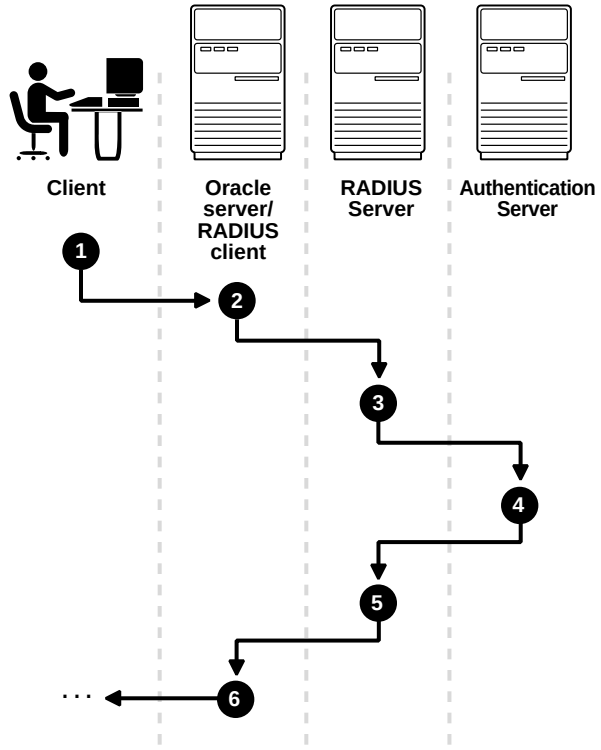
User authentication can take place in either of two ways:

- Synchronous Authentication Mode
- Challenge-Response (Asynchronous) Authentication Mode

Synchronous Authentication Mode

In the synchronous mode, RADIUS lets you use various authentication methods, including passwords and SecurID token cards. Figure 5–2 shows the sequence in which synchronous authentication occurs:

Figure 5–2 Synchronous Authentication Sequence



1. A user logs in by entering a connect string, pass code, or other value. The client system passes this data to the Oracle database server.
2. The Oracle database server, acting as the RADIUS client, passes the data from the Oracle client to the RADIUS server.
3. The RADIUS server passes the data to the appropriate authentication server, such as Smart Card or SecurID ACE for validation.
4. The authentication server sends either an Access Accept or an Access Reject message back to the RADIUS server.
5. The RADIUS server passes this response to the Oracle database server / RADIUS client.
6. The Oracle database server / RADIUS client passes the response back to the Oracle client.

Example: Synchronous Authentication with SecurID Token Cards

With SecurID authentication, each user has a token card that displays a dynamic number that changes every sixty seconds. To gain access to the Oracle database server/RADIUS client, the user enters a valid pass code that includes both a personal identification number (PIN) and the dynamic number currently displayed on the user's SecurID card. The Oracle database server passes this authentication information from the Oracle client to the RADIUS server, which in this case is the authentication server for validation. Once the authentication server (RSA ACE/Server) validates the user, it sends an "accept" packet to the Oracle database server, which, in turn, passes it to the Oracle client. The user is now authenticated and able to access the appropriate tables and applications.

See Also:

- Chapter 1, "Introduction to Oracle Advanced Security"
- "Token Cards" on page 1-11
- Documentation provided by RSA Security, Inc.

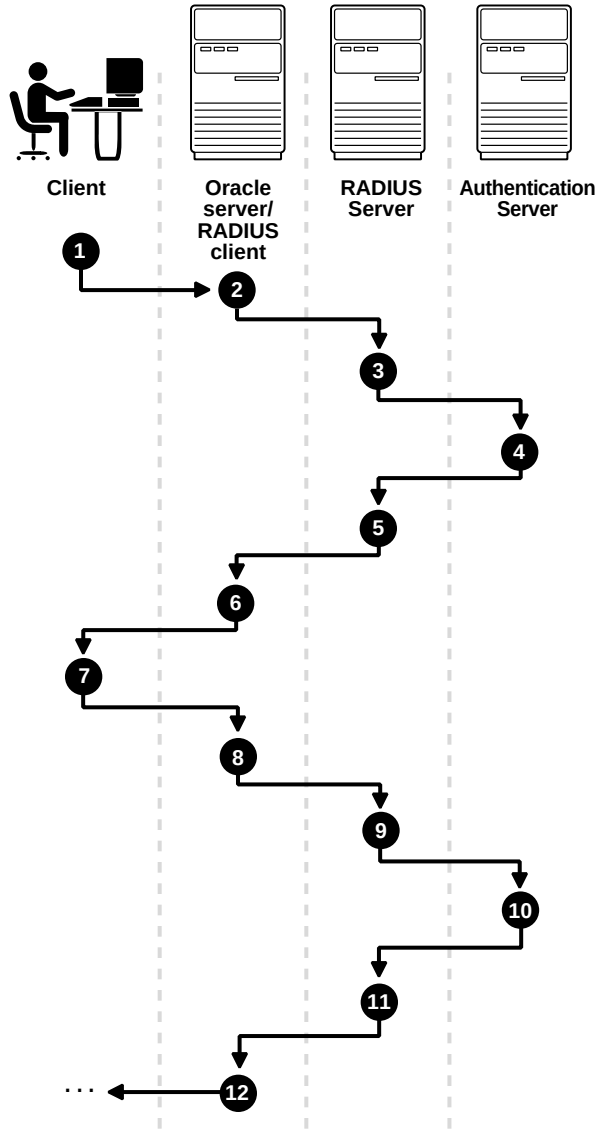
Challenge-Response (Asynchronous) Authentication Mode

When the system uses the asynchronous mode, the user does not need to enter a user name and password at the SQL*Plus CONNECT string. Instead, a graphical user interface asks the user for this information later in the process.

Figure 5–3 shows the sequence in which challenge-response (asynchronous) authentication occurs.

Note: If the RADIUS server is the authentication server, Steps 3, 4, and 5, and Steps 9, 10, and 11 in Figure 5–3 are combined.

Figure 5–3 Asynchronous Authentication Sequence



1. A user seeks a connection to an Oracle database server. The client system passes the data to the Oracle database server.

2. The Oracle database server, acting as the RADIUS client, passes the data from the Oracle client to the RADIUS server.
3. The RADIUS server passes the data to the appropriate authentication server, such as a Smart Card, SecurID ACE, or token card server.
4. The authentication server sends a challenge, such as a random number, to the RADIUS server.
5. The RADIUS server passes the challenge to the Oracle database server / RADIUS client.
6. The Oracle database server / RADIUS client, in turn, passes it to the Oracle client. A graphical user interface presents the challenge to the user.
7. The user provides a response to the challenge. To formulate a response, the user can, for example, enter the received challenge into the token card. The token card provides a dynamic password to be entered into the graphical user interface. The Oracle client passes the user's response to the Oracle database server / RADIUS client.
8. The Oracle database server / RADIUS client sends the user's response to the RADIUS server.
9. The RADIUS server passes the user's response to the appropriate authentication server for validation.
10. The authentication server sends either an Access Accept or an Access Reject message back to the RADIUS server.
11. The RADIUS server passes the response to the Oracle database server / RADIUS client.
12. The Oracle database server / RADIUS client passes the response to the Oracle client.

Example: Asynchronous Authentication with Smart Cards

With smart card authentication, the user logs in by inserting the smart card—a plastic card (like a credit card) with an embedded integrated circuit for storing information—into a hardware device which reads the card. The Oracle client sends the login information contained in the smart card to the authentication server by way of the Oracle database server / RADIUS client and the RADIUS server. The authentication server sends back a challenge to the Oracle client, by way of the RADIUS server and the Oracle database server, prompting the user for authentication information. The information could be, for example, a PIN as well as additional authentication information contained on the smart card.

The Oracle client sends the user's response to the authentication server by way of the Oracle database server and the RADIUS server. If the user has entered a valid number, the authentication server sends an "accept" packet back to the Oracle client by way of the RADIUS server and the Oracle database server. The user is now authenticated and authorized to access the appropriate tables and applications. If the user has entered incorrect information, the authentication server sends back a message rejecting the user's access.

Example: Asynchronous Authentication with ActivCard Tokens

One particular ActivCard token is a hand-held device with a keypad and which displays a dynamic password. When the user seeks access to an Oracle database server by entering a password, the information is passed to the appropriate authentication server by way of the Oracle database server/RADIUS client and the RADIUS server. The authentication server sends back a challenge to the client—by way of the RADIUS server and the Oracle database server. The user types that challenge into the token, and the token displays a number for the user to send in response.

The Oracle client then sends the user's response to the authentication server by way of the Oracle database server and the RADIUS server. If the user has typed a valid number, the authentication server sends an "accept" packet back to the Oracle client by way of the RADIUS server and the Oracle database server. The user is now authenticated and authorized to access the appropriate tables and applications. If the user has entered an incorrect response, the authentication server sends back a message rejecting the user's access.

Enabling RADIUS Authentication, Authorization, and Accounting

To enable RADIUS authentication and accounting, perform the following tasks:

- Task 1: Install RADIUS on the Oracle Database Server and on the Oracle Client
- Task 2: Configure RADIUS Authentication
- Task 3: Create a User and Grant Access
- Task 4: Configure External RADIUS Authorization (optional)
- Task 5: Configure RADIUS Accounting
- Task 6: Add the RADIUS Client Name to the RADIUS Server Database
- Task 7: Configure the Authentication Server for Use with RADIUS.
- Task 8: Configure the RADIUS Server for Use with the Authentication Server

- Task 9: Configure Mapping Roles

Task 1: Install RADIUS on the Oracle Database Server and on the Oracle Client

RADIUS is installed with Oracle Advanced Security during a typical installation of Oracle Database.

> **See Also:** Oracle Database operating system-specific installation documentation, for information about installing Oracle Advanced Security and the RADIUS adapter

Task 2: Configure RADIUS Authentication

This task includes the following steps:

- Step 1: Configure RADIUS on the Oracle Client
- Step 2: Configure RADIUS on the Oracle Database Server
- Step 3: Configure Additional RADIUS Features

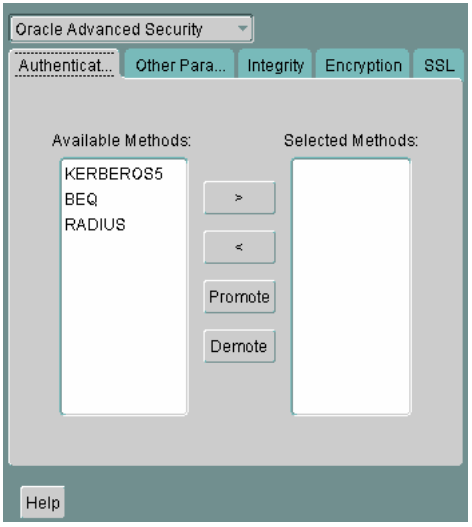
Unless otherwise indicated, perform these configuration tasks by using Oracle Net Manager or by using any text editor to modify the `sqlnet.ora` file.

Step 1: Configure RADIUS on the Oracle Client

Use Oracle Net Manager to configure RADIUS on the Oracle client (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security tabbed window appears (Figure 5–4):

Figure 5–4 Oracle Advanced Security Authentication Window



2. Choose the Authentication tab.
3. From the **Available Methods** list, select **RADIUS**.
4. Choose the right-arrow [>] to move RADIUS to the **Selected Methods** list. Move any other methods you want to use in the same way.
5. Arrange the selected methods in order of required usage by selecting a method in the Selected Methods list, and clicking **Promote** or **Demote** to position it in the list. For example, put RADIUS at the top of the list for it to be the first service used.
6. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file is updated with the following entry:

   ```
   SQLNET.AUTHENTICATION_SERVICES=(RADIUS)
   ```

Step 2: Configure RADIUS on the Oracle Database Server

- Create the RADIUS Secret Key File on the Oracle Database Server
- Configure RADIUS Parameters on the Server (sqlnet.ora file)
- Set Oracle Database Server Initialization Parameters

Create the RADIUS Secret Key File on the Oracle Database Server

1. Obtain the RADIUS secret key from the RADIUS server. For each RADIUS client, the administrator of the RADIUS server creates a shared secret key, which must be longer than 16-characters.
2. On the Oracle database server, create a directory:
   - (UNIX) `$ORACLE_HOME/network/security`
   - (Windows) *`ORACLE_HOME`*`\network\security`
3. Create the file `radius.key` to hold the shared secret copied from the RADIUS server. Place the file in the directory you just created in Step 2.
4. Copy the shared secret key and paste it (and nothing else) into the `radius.key` file created on the Oracle database server.
5. For security purposes, change the file permission of `radius.key` to read only, accessible only by the Oracle owner (Oracle relies on the file system to keep this file secret).

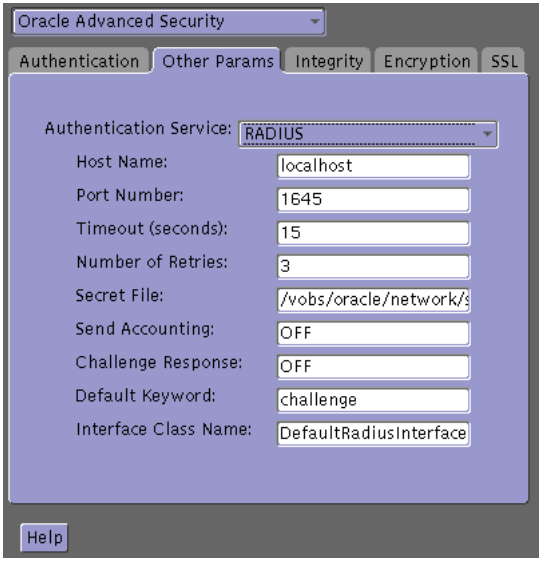
> **See Also:** The RADIUS server administration documentation, for information about obtaining the secret key

Configure RADIUS Parameters on the Server (`sqlnet.ora` file)

Use Oracle Net Manager to configure RADIUS parameters on the server (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile. (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security tabbed window appears (Figure 5–4).
2. Choose the Authentication tab.
3. From the **Available Methods** list, select **RADIUS**.
4. Move RADIUS to the **Selected Methods** list by choosing the right-arrow [>].
5. To arrange the selected methods in order of desired use, select a method in the Selected Methods list, and choose **Promote** or **Demote** to position it in the list. For example, if you want RADIUS to be the first service used, put it at the top of the list.
6. Choose the Other Params tab. The Other Params window appears (Figure 5–5):

Figure 5–5 Oracle Advanced Security Other Params Window



7. From the Authentication Service list, select **RADIUS**.
8. In the **Host Name** field, accept the `localhost` as the default primary RADIUS server, or enter another host name.
9. Ensure that the default value of the **Secret File** field is valid.
10. Choose **File** > **Save Network Configuration**.

    The `sqlnet.ora` file is updated with the following entries:

```
SQLNET.AUTHENTICATION_SERVICES=RADIUS
SQLNET.RADIUS_AUTHENTICATION=RADIUS_server_{hostname|IP_address}
```

Set Oracle Database Server Initialization Parameters

Configure the initialization parameter file, located in

- (UNIX) `$ORACLE_BASE/admin/`*`db_name`*`/pfile`
- (Windows) *`ORACLE_BASE`*`\admin\`*`db_name`*`\pfile`

with the following values:

```
REMOTE_OS_AUTHENT=FALSE
```

```
OS_AUTHENT_PREFIX=""
```

Caution: Setting REMOTE_OS_AUTHENT to TRUE can enable a security breach because it lets someone using a non-secure protocol, such as TCP, perform an operating system-authorized login (formerly called an OPS$ login).

See Also: *Oracle Database Reference* and the *Oracle Database Administrator's Guide*, for information about setting initialization parameters on an Oracle Database server

Step 3: Configure Additional RADIUS Features

- Change Default Settings
- Configure Challenge-Response
- Set Parameters for an Alternate RADIUS Server

Change Default Settings

Use Oracle Net Manager to change default settings (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security tabbed window appears (Figure 5–5).
2. Choose the Other Params tab.
3. From the **Authentication Service** list, select **RADIUS**.
4. Change the default setting for any of the following fields:

Field	Description
Port Number	Specifies the listening port of the primary RADIUS server. The default value is 1645.
Timeout (seconds)	Specifies the time the Oracle database server waits for a response from the primary RADIUS server. The default is 15 seconds.

Field	Description
Number of Retries	Specifies the number of times the Oracle database server resends messages to the primary RADIUS server. The default is three retries. For instructions on configuring RADIUS accounting, see: Task 5: Configure RADIUS Accounting on page 5-19.
Secret File	Specifies the location of the secret key on the Oracle database server. The field specifies the location of the secret key file, not the secret key itself. For information about specifying the secret key, see: Create the RADIUS Secret Key File on the Oracle Database Server on page 5-11.

5. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file is updated with the following entries:

```
SQLNET.RADIUS_AUTHENTICATION_PORT=(PORT)
SQLNET.RADIUS_AUTHENTICATION_TIMEOUT=
(NUMBER OF SECONDS TO WAIT FOR response)
SQLNET.RADIUS_AUTHENTICATION_RETRIES=
(NUMBER OF TIMES TO RE-SEND TO RADIUS server)
SQLNET.RADIUS_SECRET=(path/radius.key)
```

Configure Challenge-Response

The challenge-response (asynchronous) mode presents the user with a graphical interface requesting first a password, then additional information—for example, a dynamic password that the user obtains from a token card. With the RADIUS adapter, this interface is Java-based to provide optimal platform independence.

> **Note:** Third party vendors of authentication devices must customize this graphical user interface to fit their particular device. For example, a smart card vendor would customize the Java interface so that the Oracle client reads data, such as a dynamic password, from the smart card. When the smart card receives a challenge, it responds by prompting the user for more information, such as a PIN.

See Also: Appendix C, "Integrating Authentication Devices Using RADIUS", for information about how to customize the challenge-response user interface

To configure challenge-response:

1. If you are using JDK 1.1.7 or JRE 1.1.7, set the JAVA_HOME environment variable to the JRE or JDK location on the system where the Oracle client is run:
   - On UNIX, enter this command at the prompt:

     ```
     % setenv JAVA_HOME /usr/local/packages/jre1.1.7B
     ```

   - On Windows, choose **Start**> **Settings** > **Control Panel** > **System** > **Environment**, and set the JAVA_HOME variable as follows:

     ```
     c:\java\jre1.1.7B
     ```

     Note: This step is not required for any other JDK / JRE version.

2. Navigate to the Oracle Advanced Security profile in Oracle Net Manager (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security Other Params window appears (Figure 5–5).
3. From the **Authentication Service** list, select **RADIUS**.
4. In the **Challenge Response** field, enter **ON** to enable challenge-response.
5. In the **Default Keyword** field, accept the default value of the challenge or enter a keyword for requesting a challenge from the RADIUS server.

Note: The keyword feature is provided by Oracle and supported by some, but not all, RADIUS servers. You can use this feature only if your RADIUS server supports it.

By setting a keyword, you let the user avoid using a password to verify identity. If the user does *not* enter a password, the keyword you set here is passed to the RADIUS server which responds with a challenge requesting, for example, a driver's license number or birth date. If the user *does* enter a password, the RADIUS server may or may not respond with a challenge, depending upon the configuration of the RADIUS server.

6. In the **Interface Class Name** field, accept the default value of **`DefaultRadiusInterface`** or enter the name of the class you have created to handle the challenge-response conversation. If other than the default RADIUS interface is used, you also must edit the `sqlnet.ora` file to enter `SQLNET.RADIUS_CLASSPATH=(location)`, where `location` is the complete path name of the jar file. It defaults to *`$ORACLE_HOME`*`/network/jlib/netradius.jar: `*`$ORACLE_HOME`*`/JRE/lib/vt.jar`
7. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file is updated with the following entries:

```
SQLNET.RADIUS_CHALLENGE_RESPONSE=([ON | OFF])
SQLNET.RADIUS_CHALLENGE_KEYWORD=(KEYWORD)
SQLNET.RADIUS_AUTHENTICATION_INTERFACE=(name of interface including the
package name delimited by "/" for ".")
```

Set Parameters for an Alternate RADIUS Server

If you are using an alternate RADIUS server, set these parameters in the `sqlnet.ora` file using any text editor.

```
SQLNET.RADIUS_ALTERNATE=(hostname or ip address of alternate radius server)
SQLNET.RADIUS_ALTERNATE_PORT=(1812)
SQLNET.RADIUS_ALTERNATE_TIMEOUT=(number of seconds to wait for response)
SQLNET.RADIUS_ALTERNATE_RETRIES=(number of times to re-send to radius server)
```

Task 3: Create a User and Grant Access

To grant user access:

1. Launch SQL*Plus and execute these commands to create and grant access to a user identified externally on the Oracle database server.

```
SQL> CONNECT system/manager@database_name;
SQL> CREATE USER username IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO USER username;
SQL> EXIT
```

   If you are using Windows, you can use the Security Manager tool in the Oracle Enterprise Manager.

   See Also:

   - *Oracle Database Administrator's Guide*
   - *Oracle Database Heterogeneous Connectivity Administrator's Guide*

2. Enter the same user in the RADIUS server's users file.

   See Also: Administration documentation for the RADIUS server

Task 4: Configure External RADIUS Authorization (optional)

If you require external RADIUS authorization for RADIUS users who connect to an Oracle database, then you must perform the following steps to configure the Oracle server, the Oracle client, and the RADIUS server:

To configure the Oracle server (RADIUS client):

1. Add the `OS_ROLE` parameter to the `init.ora` file and set this parameter to `TRUE` as follows:

```
OS_ROLE=TRUE
```

   Then restart the database so the system can read the change to the `init.ora` file.

2. Set the RADIUS challenge-response mode to `ON` for the server if you have not already done so by following the steps listed in "Configure Challenge-Response" on page 5-14.

3. Add externally identified users and roles.

To configure the Oracle client (where users log in):

Set the RADIUS challenge-response mode to ON for the client if you have not already done so by following the steps listed in "Configure Challenge-Response" on page 5-14.

To configure the RADIUS server:

1. Add the following attributes to the RADIUS server attribute configuration file:

ATTRIBUTE NAME	CODE	TYPE
VENDOR_SPECIFIC	26	Integer
ORACLE_ROLE	1	String

2. Assign a Vendor ID for Oracle in the RADIUS server attribute configuration file that includes the SMI Network Management Private Enterprise Code of 111.

   For example, enter the following in the RADIUS server attribute configuration file:

   ```
   VALUE      VENDOR_SPECIFIC      ORACLE      111
   ```

3. Using the following syntax, add the ORACLE_ROLE attribute to the user profile of the users who will use external RADIUS authorization:

   ORA_*databaseSID_rolename*[_[A]|[D]]

   where:

   - ORA designates that this role is used for Oracle purposes
   - databaseSID is the Oracle system identifier that is configured in the database server's init.ora file
   - rolename is the name of role as it is defined in the data dictionary. For example, SYSDBA
   - A is an optional character that indicates the user has administrator's privileges for this role
   - D is an optional character that indicates this role is to be enabled by default

Ensure that RADIUS groups which map to Oracle roles adhere to the ORACLE_ROLE syntax.

For example:

```
USERNAME      USERPASSWD="user_password",
              SERVICE_TYPE=login_user,
              VENDOR_SPECIFIC=ORACLE,
              ORACLE_ROLE=ORA_ora920_sysdba
```

See Also: The RADIUS server administration documentation for information about configuring the server.

Task 5: Configure RADIUS Accounting

RADIUS accounting logs information about access to the Oracle database server and stores it in a file on the RADIUS accounting server. Use this feature only if both the RADIUS server and authentication server support it.

Set RADIUS Accounting on the Oracle Database Server

Use Oracle Net Manager to enable or disable RADIUS accounting (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile. (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Other Params window appears (Figure 5–5).
2. From the **Authentication Service** list, select **RADIUS**.
3. In the **Send Accounting** field, enter **ON** to enable accounting or **OFF** to disable accounting.
4. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file is updated with the following entry:

   ```
   SQLNET.RADIUS_SEND_ACCOUNTING= ON
   ```

Configure the RADIUS Accounting Server

RADIUS Accounting consists of an accounting server residing on either the same host as the RADIUS authentication server or on a separate host.

See Also: Administration documentation for the RADIUS server, for information about configuring RADIUS accounting

Task 6: Add the RADIUS Client Name to the RADIUS Server Database

You can use virtually any RADIUS server that complies with the standards in the Internet Engineering Task Force (IETF) RFC #2138, *Remote Authentication Dial In User Service (RADIUS)* and RFC #2139 *RADIUS Accounting*. Because RADIUS servers vary, consult the documentation for your particular RADIUS server for any unique interoperability requirements.

Perform the following steps to add the RADIUS client name to a Livingston RADIUS server:

1. Open the clients file, which can be found at `/etc/raddb/clients`. The following text and table appear:

   ```
   @ (#) clients 1.1 2/21/96 Copyright 1991 Livingston Enterprises Inc
   This file contains a list of clients which are allowed to make
   authentication requests and their encryption key. The first field is a valid
   hostname. The second field (separated by blanks or tabs) is the encryption
   key.
   Client Name                    Key
   ```

2. In the `CLIENT NAME` column, enter the host name or IP address of the host on which the Oracle database server is running. In the `KEY` column, type the shared secret.

   The value you enter in the CLIENT NAME column, whether it is the client's name or IP address, depends on the RADIUS server.

3. Save and close the clients file.

See Also: Administration documentation for the RADIUS server

Task 7: Configure the Authentication Server for Use with RADIUS

See the authentication server documentation for instructions about configuring the authentication servers.

See Also: "Related Documentation" on page -xxix, which contains a list of possible resources.

Task 8: Configure the RADIUS Server for Use with the Authentication Server

See the RADIUS server documentation.

Task 9: Configure Mapping Roles

If the RADIUS server supports vendor type attributes, you can manage roles by storing them in the RADIUS server. The Oracle database server downloads the roles when there is a CONNECT request using RADIUS.

To use this feature, configure roles on both the Oracle database server and the RADIUS server.

Perform these steps to configure roles on the Oracle database server:

1. Use a text editor to set the `OS_ROLES` parameter in the initialization parameters file on the Oracle database server.
2. Stop and restart the Oracle database server.
3. Create each role the RADIUS server is to manage on the Oracle database server with IDENTIFIED EXTERNALLY.

   To configure roles on the RADIUS server, refer to Table 5–1 and use the following syntax:

   ```
   ORA_DatabaseName.DatabaseDomainName_RoleName
   ```

   Example:

   ```
   ORA_USERDB.US.ORACLE.COM_MANAGER
   ```

Table 5–2 RADIUS Configuration Parameters

Parameter	Description
DatabaseName	The name of the Oracle database server for which the role is being created. This is the same as the value of the DB_NAME initialization parameter.
DatabaseDomainName	The name of the domain to which the Oracle database server belongs. The value is the same as the value of the DB_DOMAIN initialization parameter.
RoleName	The name of the role created in the Oracle database server.

4. Configure RADIUS challenge-response mode.

See Also:

- Challenge-Response (Asynchronous) Authentication Mode on page 5-5
- Configure Challenge-Response on page 5-14

These sections describe how to configure challenge-response mode.

Using RADIUS to Log In to a Database

If you are using the synchronous authentication mode, launch SQL*Plus and enter the following command at the prompt:

```
CONNECT username/password@database_alias
```

Note that you can log in with this command only when challenge-response is not turned to ON.

If you are using the challenge-response mode, launch SQL*Plus and, at the prompt, enter the command that follows:

```
CONNECT /@database_alias
```

Note that you can log in with this command only when challenge-response is turned to ON.

Note: The challenge-response mode can be configured for all login cases.

RSA ACE/Server Configuration Checklist

If you are using an RSA ACE/Server as a RADIUS server, check the following items before making your initial connection:

- Ensure that the host agent in the RSA ACE/Server is set up to send a node secret. In version 5.0, this is done by leaving the SENT Node secret box unchecked. If the RSA ACE/Server fails to send a node secret to the agent, then a node verification failure message will be written to the RSA ACE/Server log.
- If you are using RSA SecurID tokens, then ensure that the token is synchronized with the RSA ACE/Server.

See Also: RSA ACE/Server documentation for specific information about troubleshooting.

6

Configuring Kerberos Authentication

This chapter describes how to configure Oracle Advanced Security for Oracle Database for use with Kerberos authentication—and how to configure Kerberos to authenticate Oracle database users. This chapter contains the following topics:

Enabling Kerberos Authentication

To enable Kerberos authentication:

- Task 1: Install Kerberos
- Task 2: Configure a Service Principal for an Oracle Database Server
- Task 3: Extract a Service Table from Kerberos
- Task 4: Install an Oracle Database Server and an Oracle Client
- Task 5: Install Oracle Net Services and Oracle Advanced Security
- Task 6: Configure Oracle Net Services and Oracle Database
- Task 7: Configure Kerberos Authentication
- Task 8: Create a Kerberos User
- Task 9: Create an Externally Authenticated Oracle User
- Task 10: Get an Initial Ticket for the Kerberos / Oracle User

Task 1: Install Kerberos

Install Kerberos on the system that functions as the authentication server.

> **See Also:** Notes about building and installing Kerberos from Kerberos version 5 source distribution for information about how to install Kerberos.

Task 2: Configure a Service Principal for an Oracle Database Server

To enable the Oracle database server to validate the identity of clients that authenticate themselves using Kerberos, you must create a **service principal** for Oracle Database.

The name of the principal should have the following format:

```
kservice/kinstance@REALM
```

Each of the fields in the service principal specify the following values:

Service Principal Field	Description
kservice	A case-sensitive string that represents the Oracle service; this can be the same as the database service name.
kinstance	This is typically the fully qualified name of the system on which Oracle Database is running.
REALM	The domain name of the database server. REALM must always be uppercase and is typically the DNS domain name.

Note: The utility names in this section are executable programs. However, the Kerberos user name `krbuser` and the realm `SOMECO.COM` are examples only.

For example, if `kservice` is `oracle`, the fully qualified name of the system on which Oracle Database is running is `dbserver.someco.com` and the realm is `SOMECO.COM`. The principal name is:

```
oracle/dbserver.someco.com@SOMECO.COM
```

It is a convention to use the DNS domain name as the name of the realm. To create the **service principal**, run `kadmin.local`. On UNIX, run this command as the root user, by using the following syntax:

```
# cd /kerberos-install-directory/sbin
# ./kadmin.local
```

To add a **principal** named `oracle/dbserver.someco.com@SOMECO.COM` to the list of server principals known by Kerberos, enter the following:

```
kadmin.local:addprinc -randkey oracle/dbserver.someco.com@SOMECO.COM
```

Task 3: Extract a Service Table from Kerberos

Extract the **service table** from Kerberos and copy it to the Oracle database server / Kerberos client system.

For example, use the following steps to extract a service table for `dbserver.someco.com`:

1. Enter the following to extract the service table:

```
kadmin.local:  ktadd -k /tmp/keytab oracle/dbserver.someco.com

Entry for principal oracle/dbserver.someco.com with kvno 2, encryption
DES-CBC-CRC added to the keytab WRFILE: 'WRFILE:/tmp/keytab

kadmin.local:  exit

oklist -k -t /tmp/keytab
```

2. After the service table has been extracted, verify that the new entries are in the table in addition to the old ones. If they are not, or you need to add more, use `kadmin.local` to append to them.

   If you do not enter a realm when using `ktadd`, it uses the realm of the current host and displays it in the command output, as shown in Step 1.

3. If the Kerberos service table is on the same system as the Kerberos client, you can move it. If the service table is on a different system from the Kerberos client, you must transfer the file with a program such as FTP. If using FTP, transfer the file in binary mode.

   The following example shows how to move the service table on a UNIX platform:

```
# mv /tmp/keytab /etc/v5srvtab
```

   The default name of the service file is `/etc/v5srvtab`.

4. Verify that the owner of the Oracle database server executable can read the service table (`/etc/v5srvtab` in the previous example). To do so, set the file owner to the Oracle user, or make the file readable by the group to which Oracle belongs.

   Caution: Do not make the file readable to all users. This can cause a security breach.

Task 4: Install an Oracle Database Server and an Oracle Client

Install the Oracle database server and client software.

See Also: Oracle Database operating system-specific installation documentation

Task 5: Install Oracle Net Services and Oracle Advanced Security

Install Oracle Net Services and Oracle Advanced Security on the Oracle database server and Oracle client systems.

See Also: Oracle Database operating system-specific installation documentation

Task 6: Configure Oracle Net Services and Oracle Database

Configure Oracle Net Services on the Oracle database server and client.

See Also:

- Oracle Database operating system-specific installation documentation
- *Oracle Net Services Administrator's Guide.*

Task 7: Configure Kerberos Authentication

Perform these tasks to set required parameters in the Oracle database server and client `sqlnet.ora` files:

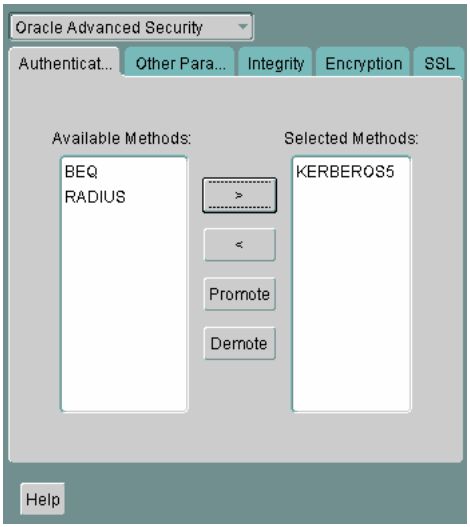
- Step 1: Configure Kerberos on the Client and on the Database Server
- Step 2: Set the Initialization Parameters
- Step 3: Set sqlnet.ora Parameters (optional)

Step 1: Configure Kerberos on the Client and on the Database Server

Use Oracle Net Manager to perform the following steps to configure Kerberos authentication service parameters on the client and on the database server (See "Starting Oracle Net Manager" on page 2-2):

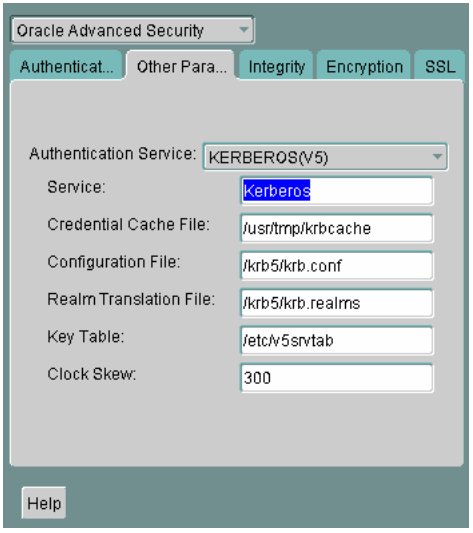
1. Navigate to the Oracle Advanced Security profile. (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security window appears (Figure 6–1):

Figure 6–1 Oracle Advanced Security Authentication Window (Kerberos)



2. Choose the Authentication tab.
3. From the **Available Methods** list, select **KERBEROS5**.
4. Move **KERBEROS5** to the **Selected Methods** list by clicking the right arrow (**>**).
5. Arrange the selected methods in order of use. To do this, select a method in the Selected Methods list, then click **Promote** or **Demote** to position it in the list. For example, if you want KERBEROS5 to be the first service used, move it to the top of the list.
6. Choose the Other Params tab (Figure 6–2).

Figure 6–2 Oracle Advanced Security Other Params Window (Kerberos)



7. From the **Authentication Service** list, select **KERBEROS(V5)**.
8. Type **Kerberos** into the **Service** field. This field defines the name of the service Oracle Database uses to obtain a Kerberos **service ticket**. When you provide the value for this field, the other fields are enabled.
9. Optionally enter values for the following fields:
   - Credential Cache File
   - Configuration File
   - Realm Translation File
   - Key Table
   - Clock Skew

   See Also: Oracle Net Manager online help, and "Step 3: Set sqlnet.ora Parameters (optional)" on page 6-8, for more information about the fields and the parameters they configure
10. Choose **File** > **Save Network Configuration**.

The sqlnet.ora file is updated with the following entries:

```
SQLNET.AUTHENTICATION_SERVICES=(KERBEROS5)
SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=kservice
```

Step 2: Set the Initialization Parameters

To set parameters in the initialization parameter file:

1. Add the following parameter to the initialization parameter file:

   ```
   REMOTE_OS_AUTHENT=FALSE
   ```

   Caution: Setting REMOTE_OS_AUTHENT to TRUE can enable a security breach, because it lets someone using a non-secure protocol, such as TCP, perform an operating system-authorized login (formerly called an OPS$ login).

2. Because Kerberos user names can be long, and Oracle user names are limited to 30 characters, Oracle Corporation strongly recommends that you set the value of OS_AUTHENT_PREFIX to null as follows:

   ```
   OS_AUTHENT_PREFIX=""
   ```

   Setting this parameter to null overrides the default value of OPS$.

Step 3: Set sqlnet.ora Parameters (optional)

In addition to the required parameters, you can optionally set the following parameters in the sqlnet.ora file on the client and the Oracle database server:

Parameter:	SQLNET.KERBEROS5_CC_NAME=*pathname_to_credentials_cache_file*
Description:	Specifies the complete path name to the Kerberos credentials cache (CC) file. The default value is operating system-dependent. For UNIX, it is /tmp/krb5cc_userid. You can also set this parameter by using the KRB5CCNAME environment variable, but the value set in the sqlnet.ora file takes precedence over the value set in KRB5CCNAME.
Example:	SQLNET.KERBEROS5_CC_NAME=/usr/tmp/krbcache

Parameter: SQLNET.KERBEROS5_CLOCKSKEW=*number_of_seconds_accepted_as_network_delay*

Description: This parameter specifies how many seconds can pass before a Kerberos credential is considered out-of-date. It is used when a credential is actually received by either a client or a database server. An Oracle database server also uses it to decide if a credential needs to be stored to protect against a replay attack. The default is 300 seconds.

Example: SQLNET.KERBEROS5_CLOCKSKEW=1200

Parameter: SQLNET.KERBEROS5_CONF=*pathname_to_Kerberos_configuration_file*

Description: This parameter specifies the complete path name to the Kerberos configuration file. The configuration file contains the realm for the default KDC (key distribution center) and maps realms to KDC hosts. The default is operating system-dependent. For UNIX, it is /krb5/krb.conf.

Example: SQLNET.KERBEROS5_CONF=*/krb/krb.conf*

Parameter: SQLNET.KERBEROS5_CONF_MIT=[TRUE|FALSE]

Description: This parameter specifies whether the new MIT Kerberos configuration format is used. If the value is set to TRUE, it will parse the file according to the new configuration format rules. When the value is set to FALSE, the default (non-MIT) configuration is used. The default is FALSE.

Example: SQLNET.KERBEROS5_CONF_MIT=False

Parameter: SQLNET.KERBEROS5_KEYTAB=*pathname_to_Kerberos_principal/key_table*

Description: This parameter specifies the complete path name to the Kerberos principal/secret key mapping file. It is used by the Oracle database server to extract its key and decrypt the incoming authentication information from the client. The default is operating system-dependent. For UNIX, it is /etc/v5srvtab.

Example: SQLNET.KERBEROS5_KEYTAB=/etc/v5srvtab

Parameter: SQLNET.KERBEROS5_REALMS=*pathname_to_Kerberos_realm_translation_file*

Description: This parameter specifies the complete path name to the Kerberos realm translation file. The translation file provides a mapping from a host name or domain name to a realm. The default is operating system-dependent. For UNIX, it is `/etc/krb.realms`.

Example: `SQLNET.KERBEROS5_REALMS=/krb5/krb.realms`

Task 8: Create a Kerberos User

To create Oracle users that Kerberos can authenticate, perform this task on the Kerberos authentication server where the administration tools are installed. The realm must already exist.

> **Note:** The utility names in this section are executable programs. However, the Kerberos user name `krbuser` and realm `SOMECO.COM` are examples only; they can vary among systems.

Run `/krb5/admin/kadmin.local` as root to create a new Kerberos user, such as `krbuser`.

The following example is UNIX-specific:

```
# ./kadmin.local
kadmin.local: addprinc krbuser
Enter password for principal: "krbuser@SOMECO.COM": (password does not display)
Re-enter password for principal: "krbuser@SOMECO.COM": (password does not
display)
kadmin.local: exit
```

Task 9: Create an Externally Authenticated Oracle User

Run SQL*Plus on the Oracle database server to create the Oracle user that corresponds to the Kerberos user. In the following example, `OS_AUTHENT_PREFIX` is set to null (""). The Oracle user name is in uppercase enclosed in double quotation marks as shown in the following example:

```
SQL> CONNECT / AS SYSDBA;
SQL> CREATE USER "KRBUSER@SOMECO.COM" IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO "KRBUSER@SOMECO.COM";
```

Task 10: Get an Initial Ticket for the Kerberos/Oracle User

Before you can connect to the database, you must ask the Key Distribution Center (KDC) for an **initial ticket**. To do so, run the following on the client:

```
% okinit username
```

If, when making a database connection, a reference such as the following follows a database link, you must use the forwardable flag (`-f`) option:

```
sqlplus /@oracle
```

Executing `okinit -f` enables credentials that can be used across database links. Run the following commands on the Oracle client:

```
% okinit -f
Password for krbuser@SOMECO.COM:password
```

Utilities for the Kerberos Authentication Adapter

Three utilities are shipped with the Oracle Kerberos authentication adapter. These utilities are intended for use on an Oracle client with Oracle Kerberos authentication support installed. Use the following utilities for these specified tasks:

- Obtaining the Initial Ticket with the okinit Utility
- Displaying Credentials with the oklist Utility
- Removing Credentials from the Cache File with the okdstry Utility

Obtaining the Initial Ticket with the okinit Utility

The `okinit` utility obtains and caches Kerberos tickets. This utility is typically used to obtain the ticket-granting ticket, using a password entered by the user to decrypt the credential from the key distribution center (KDC). The ticket-granting ticket is then stored in the user's credential cache.

The options available with `okinit` are listed in Table 6–1:

Table 6–1 Options for the okinit Utility

Option	Description
`-f`	Ask for a forwardable ticket-granting ticket. This option is necessary to follow database links.

Table 6–1 (Cont.) Options for the okinit Utility

Option	Description
`-l`	Specify the lifetime of the ticket-granting ticket and all subsequent tickets. By default, the ticket-granting ticket is good for eight (8) hours, but shorter or longer-lived credentials may be desired. Note that the KDC can ignore this option or put site-configured limits on what can be specified. The lifetime value is a string that consists of a number qualified by w (weeks), d (days), h (hours), m (minutes), or s (seconds), as in the following example: `okinit -l 2wld6h20m30s` The example requests a ticket-granting ticket that has a life time of 2 weeks, 1 day, 6 hours, 20 minutes, and 30 seconds.
`-c`	Specify an alternative credential cache. For UNIX, the default is `/tmp/krb5cc_`*`uid`*. You can also specify the alternate credential cache by using the `SQLNET.KERBEROS5_CC_NAME` parameter in the `sqlnet.ora` file.
`-?`	List command line options.

Displaying Credentials with the oklist Utility

Run the `oklist` utility to display the list of tickets held; available `oklist` options are listed in Table 6–2:

Table 6–2 Options for the oklist Utility

Option	Description
`-f`	Show flags with credentials. Relevant flags are I, credential is a ticket-granting ticket, F, credential is forwardable, and f, credential is forwarded.
`-c`	Specify an alternative credential cache. In UNIX, the default is `/tmp/krb5cc_`*`uid`*. The alternate credential cache can also be specified by using the `SQLNET.KERBEROS5_CC_NAME` parameter in the `sqlnet.ora` file.
`-k`	List the entries in the service table (default `/etc/v5srvtab`) on UNIX. The alternate service table can also be specified by using the `SQLNET.KERBEROS5_KEYTAB` parameter in the `sqlnet.ora` file.

The show flag option (`-f`) displays additional information, as shown in the following example:

```
% oklist -f
27-Jul-1999 21:57:51   28-Jul-1999 05:58:14
krbtgt/SOMECO.COM@SOMECO.COM
Flags: FI
```

Removing Credentials from the Cache File with the okdstry Utility

Use the `okdstry` utility to remove credentials from the credentials cache file:

```
$ okdstry -f
```

where the `-f` command option lets you specify an alternative credential cache. For UNIX, the default is `/tmp/krb5cc_`*`uid`*. You can also specify the alternate credential cache by using the `SQLNET.KRB5_CC_NAME` parameter in the `sqlnet.ora` file.

Connecting to an Oracle Database Server Authenticated by Kerberos

You can now connect to an Oracle database server without using a user name or password. Enter a command similar to the following:

```
$ sqlplus /@net_service_name
```

where *`net_service_name`* is an Oracle Net Services service name. For example:

```
$ sqlplus /@oracle_dbname
```

> **See Also:** Chapter 1, "Introduction to Oracle Advanced Security", for information about external authentication and *Oracle Database Heterogeneous Connectivity Administrator's Guide*

Configuring Interoperability with a Windows 2000 Domain Controller KDC

Oracle Advanced Security, which complies with MIT Kerberos, can interoperate with tickets that are issued by a Kerberos Key Distribution Center (KDC) on a Windows 2000 domain controller to enable Kerberos authentication with an Oracle database. To configure Kerberos authentication that uses a Windows 2000 domain controller KDC, perform the following tasks:

- Task 1: Configuring an Oracle Kerberos Client to Interoperate with a Windows 2000 Domain Controller KDC

Task 1: Configuring an Oracle Kerberos Client to Interoperate with a Windows 2000 Domain Controller KDC

The following steps must be performed on the Oracle Kerberos client.

Step 1: Creating Client Kerberos Configuration Files to Use a Windows Domain Controller KDC

Create the following Kerberos client configuration files that refer to the Windows 2000 domain controller as the Kerberos KDC. In the examples that follow, the Windows 2000 domain controller is running on a node named `sales3854.us.acme.com`.

- **`krb.conf`** file

  For example:

  ```
  SALES3854.US.ACME.COM
  SALES3854.US.ACME.COM sales3854.us.acme.com admin server
  ```

- **`krb5.conf`** file

  For example:

  ```
  [libdefaults]
  default_realm=SALES.US.ACME.COM
  [realms]
  SALES.US.ACME.COM= {
      kdc=sales3854.us.acme.com:88
      }
  [domain_realm]
  .us.acme.com=SALES.US.ACME.COM
  ```

- **`krb5.realms`** file

  For example:

  ```
  us.acme.com SALES.US.ACME.COM
  ```

Step 2: Specifying Oracle Configuration Parameters in the sqlnet.ora File

Configuring an Oracle client to interoperate with a Windows 2000 domain controller KDC uses the same `sqlnet.ora` file parameters that are listed in "Step 1: Configure Kerberos on the Client and on the Database Server" on page 6-5.

Set the following parameters in the `sqlnet.ora` file on the client:

```
SQLNET.KERBEROS5_CONF=pathname_to_Kerberos_configuration_file
SQLNET.KERBEROS5_CONF_MIT=TRUE
SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=Kerberos_service_name
SQLNET.AUTHENTICATION_SERVICES=(BEQ,KERBEROS5)
```

Note: Ensure that the `SQLNET.KERBEROS5_CONF_MIT` parameter is set to `TRUE` because the Windows 2000 operating system is designed to interoperate only with security services that are based on MIT Kerberos version 5.

Step 3: Specifying the Listening Port Number

The Windows 2000 domain controller KDC listens on UDP/TCP port 88. Ensure that the system file entry for `kerberos5` is set to UDP/TCP port 88 as follows:

- (UNIX)

  Ensure that the `kerberos5` entry in the `/etc/services` file is set to 88

Task 2: Configuring a Windows 2000 Domain Controller KDC to Interoperate with an Oracle Client

The following steps must be performed on the Windows 2000 domain controller.

See Also: Microsoft documentation for information about how to create users in Active Directory.

Step 1: Creating the User

Create a new user for the Oracle client in Microsoft Active Directory.

Step 2: Creating the Oracle Database Principal

1. Create a new user for the Oracle database in Microsoft Active Directory.

For example, if the Oracle database runs on the host `sales3854.us.acme.com`, then use Active Directory to create a user with the username `sales3854.us.acme.com` and the password `oracle`.

Note: Do not create a user as `host/hostname.dns.com`, such as `oracle/sales3854.us.acme.com`, in Active Directory. Microsoft's KDC does not support multipart names like an MIT KDC does. An MIT KDC allows multipart names to be used for service principals because it treats all principals as usernames. However, Microsoft's KDC does not.

---

2. Use the `Ktpass` command line utility to extract the keytab file with the following syntax:

   ```
   Ktpass -princ service/hostname@NT-DNS-REALM-NAME -mapuser account -pass
   password -out keytab.file
   ```

   Using the database user created in the previous step, the following is an example of `Ktpass` usage:

   ```
   C:> Ktpass -princ oracle/sales3854.us.acme.com@SALES.US.COM -mapuser
   sales3854 -pass oracle -out C:\temp\v5srvtab
   ```

   This utility is part of the Windows 2000 Support Tools and can be found on the Windows 2000 distribution media in the `\support\reskit\netmgmt\security` folder.

3. Copy the extracted keytab file to the host computer where the Oracle database is installed.

   For example, the keytab that was created in the previous step can be copied to `/krb5/v5svrtab`.

   See Also: Detailed information about Windows 2000 interoperability with Kerberos 5 that is available at the following URL:

   ```
   http://www.microsoft.com/WINDOWS2000/techinfo/planning/security/
   kerbsteps.asp
   ```

Task 3: Configuring an Oracle Database to Interoperate with a Windows 2000 Domain Controller KDC

The following steps must be performed on the host computer where the Oracle database is installed.

Step 1: Setting Configuration Parameters in the sqlnet.ora File

Specify values for the following parameters in the sqlnet.ora file for the database server:

```
SQLNET.KERBEROS5_CONF=pathname_to_Kerberos_configuration_file
SQLNET.KERBEROS5_KEYTAB=pathname_to_Kerberos_principal/key_table
SQLNET.KERBEROS5_CONF_MIT=TRUE
SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=Kerberos_service_name
SQLNET.AUTHENTICATION_SERVICES=(BEQ,KERBEROS5)
```

Note: Ensure that the `SQLNET.KERBEROS5_CONF_MIT` parameter is set to `TRUE` because the Windows 2000 operating system is designed to interoperate only with security services that are based on MIT Kerberos version 5.

Step 2: Creating an Externally Authenticated Oracle User

Follow the task information for "Task 9: Create an Externally Authenticated Oracle User" on page 6-10 to create an externally authenticated Oracle user. Ensure that the username is created in all uppercase characters. For example, `ORAKRB@SALES.US.ACME.COM`.

See Also: ■"Task 7: Configure Kerberos Authentication" on page 6-5 for information about using Oracle Net Manager to set the `sqlnet.ora` file parameters.

Task 4: Getting an Initial Ticket for the Kerberos/Oracle User

Before a client can connect to the database, the client must request an **initial ticket**. To request an initial ticket, follow the task information for "Task 10: Get an Initial Ticket for the Kerberos / Oracle User" on page 6-11.

Troubleshooting

This section lists some common configuration problems and explains how to resolve them.

- If you cannot get your ticket-granting ticket using OKINIT:
  - Ensure that the default realm is correct by examining the `krb.conf` file.
  - Ensure that the KDC is running on the host specified for the realm.
  - Ensure that the KDC has an entry for the user principal and that the passwords match.
  - Ensure that the `krb.conf` and `krb.realms` files are readable by Oracle.
- If you have an initial ticket, but still cannot connect:
  - After trying to connect, check for a service ticket.
  - Check that the `sqlnet.ora` file on the database server side has a service name that corresponds to a service known by Kerberos.
  - Check that the clocks on all systems involved are set to times that are within a few minutes of each other (or change the `SQLNET.KERBEROS5_CLOCKSKEW` parameter in the `sqlnet.ora` file).
- If you have a service ticket and you still cannot connect:
  - Check the clocks on the client and database server.
  - Check that the `v5srvtab` file exists in the correct location and is readable by Oracle (remember to set the `sqlnet.ora` parameters).
  - Check that the `v5srvtab` file has been generated for the service named in the `sqlnet.ora` file on the database server side.
- If everything seems to work fine, but then you issue another query and it fails:
  - Check that the initial ticket is forwardable. (You must have obtained the initial ticket by running the `okinit` utility.)
  - Check the expiration date on the credentials. If the credentials have expired, then close the connection and run `okinit` to get a new initial ticket.

7

Configuring Secure Sockets Layer Authentication

This chapter describes how to configure and use the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols which are supported by Oracle Advanced Security. It contains the following topics:

SSL and TLS in an Oracle Environment

Secure Sockets Layer (SSL) is an industry standard protocol originally designed by Netscape Communications Corporation for securing network connections. SSL uses RSA public key cryptography in conjunction with symmetric key cryptography to provide authentication, encryption, and data integrity.

This section discusses the following topics:

- Difference between SSL and TLS
- About Using SSL
- How SSL Works in an Oracle Environment: The SSL Handshake

Difference between SSL and TLS

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

See Also: *The TLS Protocol Version 1.0* [RFC 2246] at the IETF Web site, which can be found at the following URL:

`http://www.ietf.org`

Note: To simplify discussion, this document uses the term "SSL" where either SSL or TLS may be appropriate because SSL is the most widely recognized term. However, where distinctions occur between how you use or configure these protocols, this document specifies what is appropriate for either SSL or TLS.

About Using SSL

Oracle Advanced Security supports authentication by using digital certificates over SSL in addition to the native encryption and data integrity capabilities of these protocols.

By using Oracle Advanced Security SSL functionality to secure communications between clients and servers, you can

- Use SSL to encrypt the connection between clients and servers
- Authenticate any client or server, such as Oracle Application Server 10*g*, to any Oracle database server that is configured to communicate over SSL

You can use SSL features by themselves or in combination with other authentication methods supported by Oracle Advanced Security. For example, you can use the encryption provided by SSL in combination with the authentication provided by Kerberos. SSL supports any of the following authentication modes:

- Only the server authenticates itself to the client
- Both client and server authenticate themselves to each other
- Neither the client nor the server authenticates itself to the other, thus using the SSL encryption feature by itself

See Also:

- *The SSL Protocol*, Version 3.0, published by the Internet Engineering Task Force, for a more detailed discussion of SSL
- Chapter 1, "Introduction to Oracle Advanced Security", for more information about authentication methods

How SSL Works in an Oracle Environment: The SSL Handshake

When a network connection over SSL is initiated, the client and server perform an SSL handshake that includes the following steps:

- The client and server establish which **cipher suites** to use. This includes which encryption algorithms are used for data transfers.
- The server sends its certificate to the client, and the client verifies that the server's certificate was signed by a trusted CA. This step verifies the identity of the server.
- Similarly, if client authentication is required, the client sends its own certificate to the server, and the server verifies that the client's certificate was signed by a trusted CA.
- The client and server exchange key 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 and the negotiated cipher suite.

The authentication process consists of the following steps:

1. On a client, the user initiates an Oracle Net connection to the server by using SSL.
2. SSL performs the handshake between the client and the server.
3. If the handshake is successful, the server verifies that the user has the appropriate **authorization** to access the database.

Public Key Infrastructure in an Oracle Environment

A public key infrastructure (PKI) is a substrate of network components that provide a security underpinning, based on trust assertions, for an entire organization. A PKI exists so that disparate network entities can access its security services, which use public-key cryptography, on an as-needed basis. Oracle provides a complete PKI that is based on RSA Security, Inc., Public-Key Cryptography Standards, and which interoperates with Oracle servers and clients.

About Public Key Cryptography

Traditional private-key or symmetric-key cryptography requires a single, secret key that is shared by two or more parties to a secure communication. This key is used to both encrypt and decrypt secure messages sent between the parties, requiring prior, secure distribution of the key to each party. The problem with this method is that it is difficult to securely transmit and store the key.

Public-key cryptography provides a solution to this problem, by employing **public and private key pair**s and a secure method for key distribution. The freely available **public key** is used to encrypt messages that can *only* be decrypted by the holder of the associated **private key**. The private key is securely stored, together with other security credentials, in an encrypted container called a **wallet**.

Public-key algorithms can guarantee the secrecy of a message, but they don't necessarily 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 own public key for a legitimate key (the **man-in-the-middle** attack).

In order to avoid such an 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), which is a third party that is trusted by both of the communicating parties.

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 wallets.

Public Key Infrastructure Components in an Oracle Environment

Public key infrastructure (PKI) components in an Oracle environment include the following:

- Certificate Authority
- Certificates
- Certificate Revocation Lists
- Wallets
- Hardware security modules

Certificate Authority

A certificate authority (CA) is a trusted third party that certifies the identity of entities, such as users, databases, administrators, clients, and servers. When an entity requests certification, the CA verifies its identity and grants a certificate, which is signed with the CA's private key.

Different CAs may have different identification requirements when issuing certificates. Some CAs may verify a requester's identity with a driver's license, some may verify identity with the requester's fingerprints, while others 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, network entities exchange certificates and check that each 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, Oracle Advanced Security automatically installs trusted certificates from VeriSign, RSA, Entrust, and GTE CyberTrust when you create a new wallet.

Oracle Application Server Certificate Authority, part of Oracle Identity Management Infrastructure, is a new Oracle PKI component available in Oracle Application Server 10*g* (9.0.4).

See Also: "Wallets" on page 7-8

Certificates

A certificate is created when an entity's public key is signed by a trusted certificate authority (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 expiration date—as well as a serial number and **certificate chain** information. It can also contain information about the privileges associated with the certificate.

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.

Certificate Revocation Lists

Typically, when a CA signs a certificate binding a public key pair to a user identity, the certificate is valid 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 alert the user population when it is no longer acceptable to use a particular public key to verify its associated user identity.

When servers or clients receive user certificates in an Oracle environment, they can validate the certificate by checking its expiration date, signature, and revocation status. Certificate revocation status is checked by validating it against published CRLs. If certificate revocation status checking is turned on, then the server searches for the appropriate CRL depending on how this feature has been configured. The server searches for CRLs in the following locations:

1. Local file system
2. Oracle Internet Directory
3. **CRL Distribution Point**, a location specified in the CRL Distribution Point (CRL DP) X.509, version 3, certificate extension when the certificate is issued.

See Also: "Certificate Validation with Certificate Revocation Lists" on page 7-35 for information about configuring and managing this PKI component

---

Note: To use CRLs with other Oracle products, refer to the specific product documentation. This implementation of certificate validation with CRLs is only available in the Oracle Database 10*g* Release 1 (10.1) SSL adapter.

Wallets

A wallet is a container that is used to store authentication and signing credentials, including private keys, certificates, and trusted certificates needed by SSL. In an Oracle environment, every entity that communicates over SSL must have a wallet containing an X.509 version 3 certificate, private key, and list of trusted certificates (with the exception of Diffie-Hellman).

Security administrators use Oracle Wallet Manager to manage security credentials on the server. Wallet owners use it to manage security credentials on clients. Specifically, you use Oracle Wallet Manager to do the following:

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

Note: Installation of Oracle Advanced Security 10*g* Release 1 (10.1) also installs Oracle Wallet Manager release 10.1.

See Also:

- Chapter 8, "Using Oracle Wallet Manager"
- "Creating a New Wallet" on page 8-10
- "Managing Trusted Certificates" on page 8-25

Hardware security modules

Oracle Advanced Security uses these devices for the following functions:

- Store cryptographic information, such as private keys
- Perform cryptographic operations to off load RSA operations from the server, freeing the CPU to respond to other transactions

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

- (Server-side) Hardware boxes where keys are stored in the box, but managed by using tokens.
- (Client-side) Smart card readers, which support storing private keys on tokens.

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

Note: Currently only nCipher devices are certified with Oracle Advanced Security. Certificate with other vendors is in progress.

See Also: "Configuring Your System to Use Hardware Security Modules" on page 7-48 for details configuration details.

SSL Combined with Other Authentication Methods

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

- Architecture: Oracle Advanced Security and SSL
- How SSL Works with Other Authentication Methods

> **See Also:** Appendix A, "Data Encryption and Integrity Parameters" for information about how to configure SSL with other supported authentication methods, including an example of a `sqlnet.ora` file with multiple authentication methods specified.

Architecture: Oracle Advanced Security and SSL

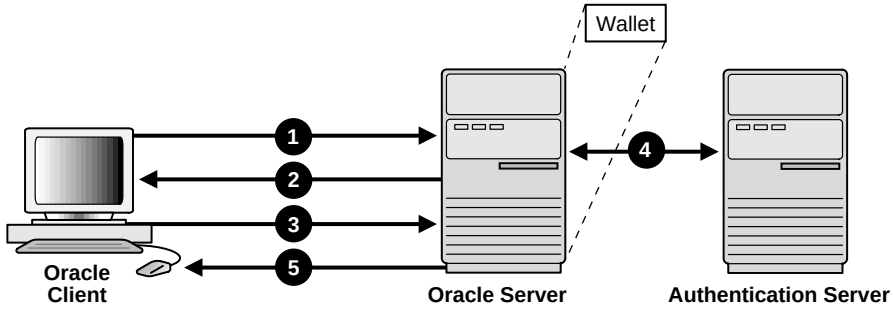
Figure 1–5 on page 1-15, which displays the Oracle Advanced Security implementation architecture, shows that Oracle Advanced Security operates at the **session layer** on top of SSL and uses TCP/IP at the **transport layer**. This separation of functionality lets you employ SSL concurrently with other supported protocols.

> **See Also:** *Oracle Net Services Administrator's Guide*, for information about stack communications in an Oracle networking environment

How SSL Works with Other Authentication Methods

Figure 7–1 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 an alternative authentication method is used to authenticate the client.

Figure 7–1 SSL in Relation to 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.

See Also: "How SSL Works in an Oracle Environment: The SSL Handshake" on page 7-4

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 like 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 and lets the SSL packet pass through to permitted SSL ports, rejecting all others.

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 its traffic.
- The database listener requires access to a **certificate** in order 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. This approach prohibits the database server from using randomly chosen SSL ports, because the SSL ports on the firewall must match those chosen by the database. You can avoid this condition by deploying Oracle Connection Manager, an application included with Oracle Database Enterprise Edition.

Oracle Connection Manager lets you 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 through the firewall.

Note: Although Oracle Connection Manager can be used to avoid opening up multiple SSL ports through the firewall, consider the following:

- The internal connection, between Oracle Connection Manager and the database, is not an SSL connection. You should encrypt such connections, using Oracle Advanced Security native encryption.
- Because such connections do not use SSL, clients cannot use certificate-based authentication.

See Also: *Oracle Net Services Administrator's Guide* for information about Oracle Connection Manager

SSL Usage Issues

Consider the following issues when using SSL:

- SSL use enables secure communication with other Oracle products, such as Oracle Internet Directory.
- 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.

Note:

- U.S. government regulations prohibit double encryption. Accordingly, if you configure Oracle Advanced Security to use SSL encryption and another encryption method concurrently, the connection fails (you also cannot configure SSL authentication concurrently with non-SSL authentication).
- If you configure SSL encryption, you must disable non-SSL encryption. To disable such encryption, see: "Disabling Oracle Advanced Security Authentication" on page 9-2.

See Also:

- "Configuring Your System to Use Hardware Security Modules" on page 7-48 for information about improving SSL performance with hardware accelerators
- "Enabling SSL" on page 7-15

Enabling SSL

To enable SSL:

- Task 1: Install Oracle Advanced Security and Related Products
- Task 2: Configure SSL on the Server
- Task 3: Configure SSL on the Client
- Task 4: Log on to the Database

Task 1: Install Oracle Advanced Security and Related Products

Install Oracle Advanced Security on both the client and server. When you do this, the Oracle Universal Installer automatically installs SSL libraries and Oracle Wallet Manager on your system.

> **See Also:** Oracle Database platform-specific installation documentation

Task 2: Configure SSL on the Server

During installation, Oracle sets defaults on both the Oracle database server and on the Oracle client for all SSL parameters except the location of the Oracle wallet. To configure SSL on the server, perform these steps:

- Step 1: Confirm Wallet Creation on the Server
- Step 2: Specify the Database Wallet Location on the Server
- Step 3: Set the SSL Cipher Suites on the Server (Optional)
- Step 4: Set the Required SSL Version on the Server (Optional)
- Step 5: Set SSL Client Authentication on the Server (Optional)
- Step 6: Set SSL as an Authentication Service on the Server (Optional)
- Step 7: Create Listening Endpoint that Uses TCP/IP with SSL on the Server

> **See Also:** Appendix B, "Authentication Parameters" for the dynamic parameter names

Step 1: Confirm Wallet Creation on the Server

Before proceeding with the next step, you must confirm that a wallet has been created. To confirm that your wallet is ready, open it by using Oracle Wallet

Manager. The wallet should contain a certificate with a status of "Ready" and auto login turned on. If auto login is not on, then select it from the Wallet menu and re-save the wallet. This turns auto login on.

See Also:

- "Opening an Existing Wallet" on page 8-13
- "Creating a New Wallet" on page 8-10
- "Using Auto Login" on page 8-19

Step 2: Specify the Database Wallet Location on the Server

Use Oracle Net Manager to specify required configuration parameters for the server (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile. (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security SSL window appears (Figure 7–5).
2. Choose the SSL tab and select **Configure SSL for: Server**.
3. In the **Wallet Directory** box, enter the directory in which the Oracle wallet is located, or click **Browse** to find it by searching the file system.

   Note that if you are configuring the database-to-directory SSL connection for Enterprise User Security, then Database Configuration Assistant automatically creates a database wallet while registering the database with the directory. You must use that wallet to store the database PKI credentials for SSL-authenticated Enterprise User Security.

   Important:

   - Use Oracle Wallet Manager to create the wallet. See "Creating a New Wallet" on page 8-10.
   - Use Oracle Net Manager to set the wallet location in the `sqlnet.ora` file.

   Be sure to enter the same wallet location when you create it and when you set the location in the `sqlnet.ora` file.

4. Choose **File** > **Save Network Configuration**.

The sqlnet.ora and listener.ora files are updated with the following entries:

```
wallet_location =
 (SOURCE=
  (METHOD=File)
  (METHOD_DATA=
   (DIRECTORY=wallet_location)))
```

Note: The listener uses the wallet defined in listener.ora (it can use any database wallet). When SSL is configured for a server using Net Manager, the wallet location is entered into the listener.ora and the sqlnet.ora files. The listener.ora file is not relevant to the Oracle client.

To change the listener wallet location (so that the listener has its own wallet), you can edit listener.ora to enter the new location.

Step 3: Set the SSL Cipher Suites on the Server (Optional)

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

When you install Oracle Advanced Security, the SSL cipher suites listed in Table 7–1 are set for you by default and negotiated in the order they are listed. You can override the default order by setting the SSL_CIPHER_SUITES parameter. For example, if you use Oracle Net Manager to add the cipher suite SSL_RSA_WITH_RC4_128_SHA, all other cipher suites in the default setting are ignored.

You can prioritize the cipher suites. When the client negotiates with servers regarding which cipher suite to use, it follows the prioritization you set. When you prioritize the cipher suites, consider the following:

- Server and client must be configured to use compatible cipher suites for a successful connection.
- The level of security you want to use. For example, triple-DES encryption is stronger than DES.
- The impact on performance. For example, triple-DES encryption is slower than DES.

- Prioritize cipher suites starting with the strongest and moving to the weakest to ensure the highest level of security possible.

Note: If you set a cipher suite employing Diffie-Hellman anonymous authentication on the server, then you must also set the same cipher suite on the client. Otherwise, the connection fails.

If you use a cipher suite employing Diffie-Hellman anonymous, then you must set the `SSL_CLIENT_AUTHENTICATION` parameter to FALSE. See: "Step 5: Set SSL Client Authentication on the Server (Optional)" on page 7-21.

Table 7–1 lists the SSL cipher suites supported in the current release of Oracle Advanced Security. These cipher suites are set by default when you install Oracle Advanced Security. This table also lists the authentication, encryption, and data integrity types each cipher suite uses.

Table 7–1 Oracle Advanced Security Cipher Suites

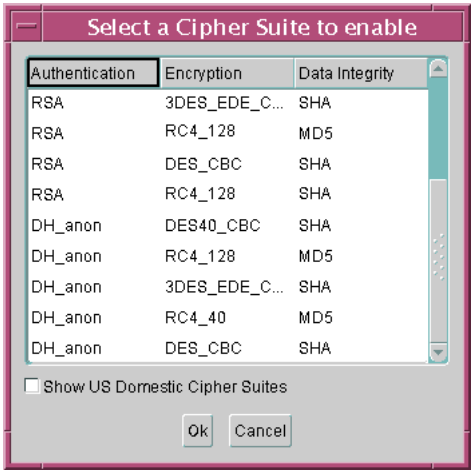
Cipher Suites	Authentication	Encryption	Data Integrity
SSL_RSA_WITH_3DES_EDE_CBC_SHA	RSA	3DES EDE CBC	SHA-1
SSL_RSA_WITH_RC4_128_SHA	RSA	RC4 128	SHA-1
SSL_RSA_WITH_RC4_128_MD5	RSA	RC4 128	MD5
SSL_RSA_WITH_DES_CBC_SHA	RSA	DES CBC	SHA-1
SSL_DH_anon_WITH_3DES_EDE_CBC_SHA	DH anon	3DES EDE CBC	SHA-1
SSL_DH_anon_WITH_RC4_128_MD5	DH anon	RC4 128	MD5
SSL_DH_anon_WITH_DES_CBC_SHA	DH anon	DES CBC	SHA-1
SSL_RSA_EXPORT_WITH_RC4_40_MD5	RSA	RC4 40	MD5
SSL_RSA_EXPORT_WITH_DES40_CBC_SHA	RSA	DES40 CBC	SHA-1
SSL_RSA_WITH_AES_128_CBC_SHA[1]	RSA	AES 128 CBC	SHA-1
SSL_RSA_WITH_AES_256_CBC_SHA[1]	RSA	AES 256 CBC	SHA-1

[1] AES ciphers work with Transport Layer Security (TLS 1.0) only

To specify cipher suites for the server:

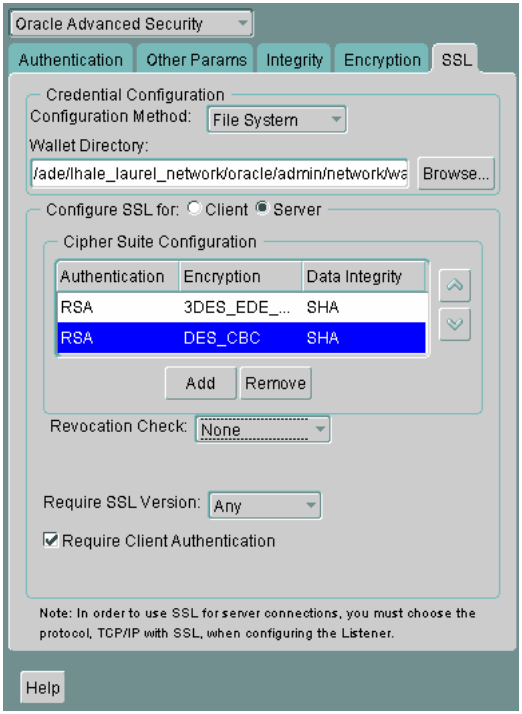
1. Navigate to the **SSL** tab of the Oracle Advanced Security window in Oracle Net Manager, and select **Configure SSL for: Server**.
2. Click **Add**. A dialog box displays available cipher suites (Figure 7–2).

Figure 7–2 SSL Cipher Suites Window



3. Select a suite and click **OK**. The **Cipher Suite Configuration** list is updated (Figure 7–3):

Figure 7–3 Oracle Advanced Security SSL Window (Server)



4. Use the up and down arrows to prioritize the cipher suites.
5. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file is updated with the following entry:

   ```
   SSL_CIPHER_SUITES= (SSL_cipher_suite1 [,SSL_cipher_suite2])
   ```

Step 4: Set the Required SSL Version on the Server (Optional)

You can set the `SSL_VERSION` parameter in the `sqlnet.ora` file. This parameter defines the version of SSL that must run on the systems with which the server communicates. You can require these systems to use any valid version. The default setting for this parameter in `sqlnet.ora` is `undetermined`, which is set by selecting **Any** from the list in the SSL tab of the Oracle Advanced Security window.

To set the SSL version for the server:

1. Navigate to the **SSL** tab of the Oracle Advanced Security window in Oracle Net Manager, and select **Configure SSL for: Server**.
2. In the **Require SSL Version:** list, the default is **Any**. Accept this default or select the SSL version you want to use.
3. Choose **File** > **Save Network Configuration**.

   If you chose **Any**, then the `sqlnet.ora` file is updated with the following entry:

   ```
   SSL_VERSION=UNDETERMINED
   ```

Note: SSL 2.0 is not supported on the server side.

Step 5: Set SSL Client Authentication on the Server (Optional)

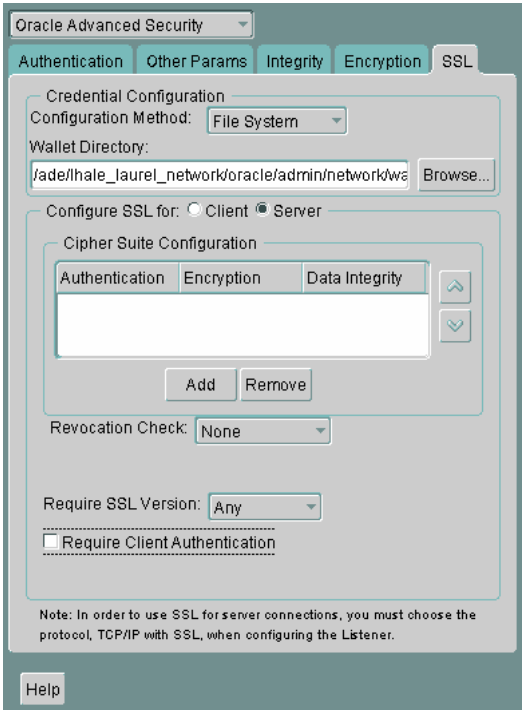
The `SSL_CLIENT_AUTHENTICATION` parameter in the `sqlnet.ora` file controls whether the client is authenticated using SSL. The default value is `TRUE`.

You must set this parameter to `FALSE` if you are using a cipher suite that contains Diffie-Hellman anonymous authentication (`DH_anon`). Also, you can set this parameter to `FALSE` for the client to authenticate itself to the server by using any of the non-SSL authentication methods supported by Oracle Advanced Security, such as Kerberos or RADIUS.

To set SSL_CLIENT_AUTHENTICATION to FALSE on the server:

1. Navigate to the **SSL** tab of the Oracle Advanced Security window in Oracle Net Manager, and select **Configure SSL for: Server**. The Oracle Advanced Security SSL window for server configuration appears (Figure 7–4).

Figure 7–4 Oracle Advanced Security SSL Window (Server)



2. Uncheck **Require Client Authentication**.
3. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file is updated with the following entry:

   ```
   SSL_CLIENT_AUTHENTICATION=FALSE
   ```

Step 6: Set SSL as an Authentication Service on the Server (Optional)

The `SQLNET.AUTHENTICATION_SERVICES` parameter in the `sqlnet.ora` file sets the SSL authentication service.

Set this parameter if you want to use SSL authentication in conjunction with another authentication method supported by Oracle Advanced Security. For example, use this parameter if you want the server to authenticate itself to the client by using SSL and the client to authenticate itself to the server by using Kerberos.

To set the SQLNET.AUTHENTICATION_SERVICES parameter on the server:
Add TCP/IP with SSL (TCPS) to this parameter in the `sqlnet.ora` file by using a text editor. For example, if you want to use SSL authentication in conjunction with RADIUS authentication, set this parameter as follows:

```
SQLNET.AUTHENTICATION_SERVICES = (TCPS, radius)
```

If you do not want to use SSL authentication in conjunction with another authentication method, then do not set this parameter.

Step 7: Create Listening Endpoint that Uses TCP/IP with SSL on the Server

Configure the listener with a TCP/IP with SSL listening endpoint in the `listener.ora` file. Oracle Corporation recommends using port number 2484 for typical Oracle Net clients.

See Also:

- *Oracle Net Services Administrator's Guide.* for detailed information about configuring the `listener.ora` file
- "Certificate Validation with Certificate Revocation Lists" on page 7-35 for information about configuring your system to validate certificates with certificate revocation lists

Task 3: Configure SSL on the Client

To configure SSL on the client:

- Step 1: Confirm Client Wallet Creation
- Step 2: Configure Oracle Net Service Name to Include Server DNs and Use TCP/IP with SSL on the Client
- Step 3: Specify Required Client SSL Configuration (Wallet Location)
- Step 4: Set the Client SSL Cipher Suites (Optional)
- Step 5: Set the Required SSL Version on the Client (Optional)
- Step 6: Set SSL as an Authentication Service on the Client (Optional)

See Also: Appendix B, "Authentication Parameters", for the dynamic parameter names.

Step 1: Confirm Client Wallet Creation

Before proceeding with the next step, you must confirm that a wallet has been created on the client and that the client has a valid certificate.

Note: Oracle Corporation recommends that you use Oracle Wallet Manager to remove the **trusted certificate** in your Oracle wallet associated with each **certificate authority** that you do not use.

See Also:

- Chapter 8, "Using Oracle Wallet Manager", for general information about wallets
- "Opening an Existing Wallet" on page 8-13, for information about opening an existing wallet
- "Creating a New Wallet" on page 8-10, for information about creating a new wallet

Step 2: Configure Oracle Net Service Name to Include Server DNs and Use TCP/IP with SSL on the Client

You must specify the server's **distinguished name (DN)** and `TCPS` as the protocol in the client network configuration files to enable server DN matching and TCP/IP with SSL connections. Server DN matching prevents the database server from faking its identity to the client during connections by matching the server's global database name against the DN from the server certificate.

You must manually edit the client network configuration files, `tnsnames.ora` and `listener.ora`, to specify the server's DN and the TCP/IP with SSL protocol. The `tnsnames.ora` file can be located on the client or in the LDAP directory. If it is located on the client, then it typically resides in the same directory as the `listener.ora` file. Depending on your operating system, these files reside in the following directory locations:

- (UNIX) *ORACLE_HOME*`/network/admin/`
- (Windows) *ORACLE_BASE*`\`*ORACLE_HOME*`\network\admin\`

To edit the `tnsnames.ora` and `listener.ora` files, use the following steps:

1. In the client `tnsnames.ora` file, add the `SSL_SERVER_CERT_DN` parameter and specify the database server's DN as follows:

```
(SECURITY=
    (SSL_SERVER_CERT_DN="cn=finance,cn=OracleContext,c=us,o=acme"))
```

The client uses this information to obtain the list of DNs it expects for each of the servers, enforcing the server's DN to match its service name. Example 7–1 shows an entry for the `Finance` database in the `tnsnames.ora` file.

Alternatively, the administrator can ensure that the common name (CN) portion of the server's DN matches the service name.

2. Also in the client `tnsnames.ora` file, enter `tcps` as the `PROTOCOL` in the `ADDRESS` parameter. This specifies that the client will use TCP/IP with SSL to connect to the database that is identified in the `SERVICE_NAME` parameter. Example 7–1 also shows an entry that specifies TCP/IP with SSL as the connecting protocol in the `tnsnames.ora` file.
3. In the `listener.ora` file, enter `tcps` as the `PROTOCOL` in the `ADDRESS` parameter. Example 7–2 shows an entry that specifies TCP/IP with SSL as the protocol.

Example 7–1 Sample tnsnames.ora File with Server Certificate DN and TCP/IP with SSL Specified

```
finance=
(DESCRIPTION=
    (ADDRESS_LIST=
        (ADDRESS= (PROTOCOL = tcps) (HOST = finance_server) (PORT = 1575)))
    (CONNECT_DATA=
        (SERVICE_NAME= Finance.us.acme.com))
    (SECURITY=
        (SSL_SERVER_CERT_DN="cn=finance,cn=OracleContext,c=us,o=acme"))
```

Example 7–2 Sample listener.ora File with TCP/IP with SSL Specified as the Protocol

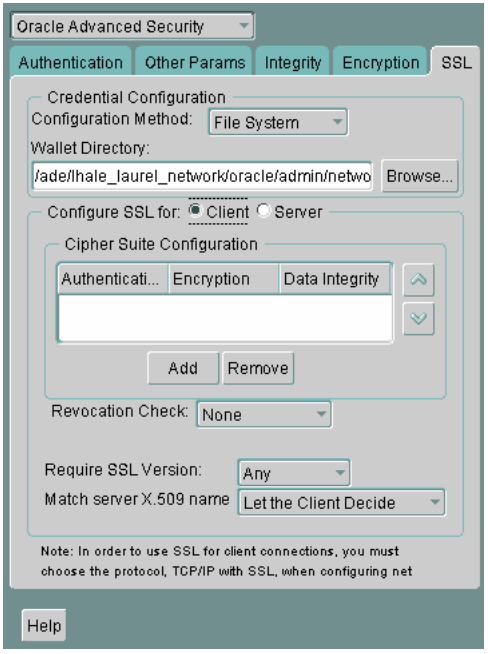
```
LISTENER=
    (DESCRIPTION_LIST=
        (DESCRIPTION=
            (ADDRESS= (PROTOCOL = tcps) (HOST = finance_server) (PORT = 1575))))
```

Step 3: Specify Required Client SSL Configuration (Wallet Location)

Use Oracle Net Manager to specify required configuration parameters for the client (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile. (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security SSL window appears (Figure 7–5):

Figure 7–5 Oracle Advanced Security SSL Window (Client)



2. Choose the **SSL** tab.
3. Select **Configure SSL for: Client**.
4. In the Wallet Directory box, enter the directory in which the Oracle wallet is located, or click **Browse** to find it by searching the file system.
5. From the Match server X.509 name list, choose one of the following options:
   - **Yes:** Requires that the server's **distinguished name (DN)** match its service name. SSL ensures that the certificate is from the server and connections succeed only if there is a match.

Note: This check can be made only when RSA ciphers are selected, which is the default setting.

- **No (default):** SSL checks for a match between the DN and the service name, but does not enforce it. Connections succeed regardless of the outcome, but an error is logged if the match fails.
- **Let Client Decide:** Enables the default.

Note: The following alert appears when you select No:

Security Alert

Not enforcing the server X.509 name match allows a server to potentially fake its identity. Oracle Corporation recommends selecting YES for this option so that connections are refused when there is a mismatch.

---

6. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file on the client is updated with the following entries:

```
SSL_CLIENT_AUTHENTICATION =TRUE
wallet_location =
 (SOURCE=
  (METHOD=File)
  (METHOD_DATA=
   (DIRECTORY=wallet_location)))

SSL_SERVER_DN_MATCH=(ON/OFF)
```

   See Also:

   For information about the server match parameters:

   - "SSL X.509 Server Match Parameters" on page B-10

   For information about using Oracle Net Manager to configure TCP/IP with SSL:

   - *Oracle Net Services Administrator's Guide*
   - *Oracle Net Services Reference Guide*

Step 4: Set the Client SSL Cipher Suites (Optional)

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

When you install Oracle Advanced Security,the SSL cipher suites listed in Table 7–1 are set for you by default. This table lists them in the order they are tried when two entities are negotiating a connection. You can override the default by setting the `SSL_CIPHER_SUITES` parameter. For example, if you use Oracle Net Manager to add the cipher suite `SSL_RSA_WITH_RC4_128_SHA`, all other cipher suites in the default setting are ignored.

You can prioritize the cipher suites. When the client negotiates with servers regarding which cipher suite to use, it follows the prioritization you set. When you prioritize the cipher suites, consider the following:

- The level of security you want to use. For example, triple-DES encryption is stronger than DES.
- The impact on performance. For example, triple-DES encryption is slower than DES. See "Configuring Your System to Use Hardware Security Modules" on page 7-48 for information about using SSL hardware accelerators with Oracle Advanced Security.
- Administrative requirements:

  The cipher suites selected for a client must be compatible with those required by the server. For example, in the case of an Oracle Call Interface (OCI) user, the server requires the client to authenticate itself. You cannot, in this case, use a cipher suite employing Diffie-Hellman anonymous authentication which disallows the exchange of certificates.

You typically prioritize cipher suites starting with the strongest and moving to the weakest.

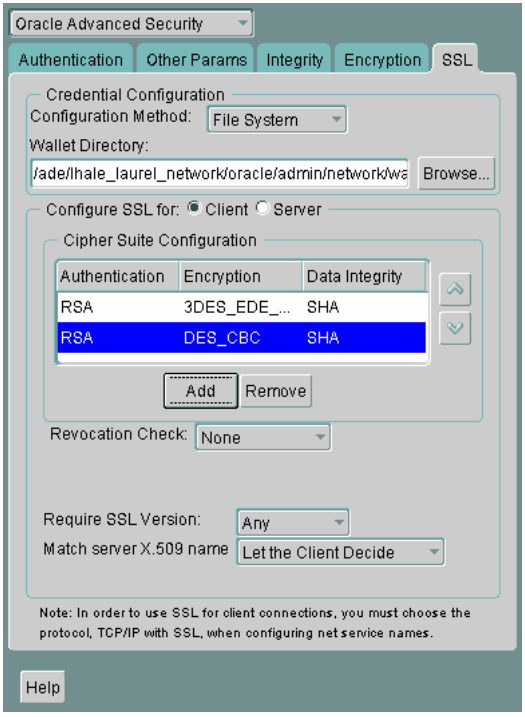
Table 7–1 lists the SSL cipher suites supported in the current release of Oracle Advanced Security. These cipher suites are set by default when you install Oracle Advanced Security. This table also lists the authentication, encryption, and data integrity types each cipher suite uses.

Note: If the `SSL_CLIENT_AUTHENTICATION` parameter is set to `true` in the `sqlnet.ora` file, then disable all cipher suites that use Diffie-Hellman anonymous authentication. Otherwise, the connection fails.

To specify client cipher suites:

1. Navigate to the **SSL** tab of the Oracle Advanced Security window in Oracle Net Manager, and select **Configure SSL for Client**.
2. In the Cipher Suite Configuration region, click **Add**. A dialog box displays available cipher suites (Figure 7–2).
3. Select a suite and click **OK**. The **Cipher Suite Configuration** list is updated (Figure 7–6):

Figure 7–6 Oracle Advanced Security SSL Window (Client)



4. Use the up and down arrows to prioritize the cipher suites.
5. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file is updated with the following entry:

   ```
   SSL_CIPHER_SUITES= (SSL_cipher_suite1 [,SSL_cipher_suite2])
   ```

Step 5: Set the Required SSL Version on the Client (Optional)

You can set the `SSL_VERSION` parameter in the `sqlnet.ora` file. This parameter defines the version of SSL that must run on the systems with which the client communicates. You can require these systems to use any valid version. The default setting for this parameter in `sqlnet.ora` is `undetermined`, which is set by selecting **Any** from the list in the **SSL** tab of the Oracle Advanced Security window. When **Any** is selected, TLS 1.0 is tried first, then SSL 3.0 and SSL 2.0 are tried in that order. Ensure that the client SSL version is compatible with the version the server uses.

To set the required SSL version for the client:

1. Navigate to the **SSL** tab of the Oracle Advanced Security window in Oracle Net Manager, and select **Configure SSL for: Client**. (See Figure 7–5).
2. In the **Require SSL Version** list, the default setting is **Any**. Accept this default or select the SSL version you want to configure.
3. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file is updated. If you selected **Any**, then it is updated with the following entry:

   ```
   SSL_VERSION=UNDETERMINED
   ```

Step 6: Set SSL as an Authentication Service on the Client (Optional)

The `SQLNET.AUTHENTICATION_SERVICES` parameter in the `sqlnet.ora` file sets the SSL authentication service. Typically, the `sqlnet.ora` file is located in the same directory as the other network configuration files. Depending on your platform, the `sqlnet.ora` file is in the following directory location:

- (UNIX) *`ORACLE_HOME`*`/network/admin`
- (Windows) *`ORACLE_BASE`*`\`*`ORACLE_HOME`*`\network\admin\`

Set the `SQLNET.AUTHENTICATION_SERVICES` parameter if you want to use SSL authentication in conjunction with another authentication method supported by

Oracle Advanced Security. For example, use this parameter if you want the server to authenticate itself to the client by using SSL and the client to authenticate itself to the server by using RADIUS.

To set the client SQLNET.AUTHENTICATION_SERVICES parameter:
Add TCP/IP with SSL (TCPS) to this parameter in the `sqlnet.ora` file by using a text editor. For example, if you want to use SSL authentication in conjunction with RADIUS authentication, set this parameter as follows:

```
SQLNET.AUTHENTICATION_SERVICES = (TCPS, radius)
```

If you do not want to use SSL authentication in conjunction with another authentication method, then do not set this parameter.

Task 4: Log on to the Database

If you are using SSL authentication for the client (`SSL_CLIENT_AUTHENTICATION=true` in the `listener.ora` file), then launch SQL*Plus and enter the following:

```
CONNECT/@net_service_name
```

If you are not using SSL authentication (`SSL_CLIENT_AUTHENTICATION=false` in the `listener.ora` file), launch SQL*Plus and enter the following:

```
CONNECT username/password@net_service_name
```

> **See Also:** "Certificate Validation with Certificate Revocation Lists" on page 7-35 for information about configuring the client for certificate validation with certificate revocation lists

Troubleshooting SSL

The following section lists the most common errors you may receive while using the Oracle Advanced Security SSL adapter.

It may be necessary to enable Oracle Net tracing to determine the cause of an error. For information about setting tracing parameters to enable Oracle Net tracing, see *Oracle Net Services Administrator's Guide*.

ORA-28759: Failure to Open File

Cause: The system could not open the specified file. Typically, this error occurs because the wallet cannot be found.

Action: Check the following:

- Ensure that the correct wallet location is specified in the `sqlnet.ora` file. Note: this should be the same directory location where you saved the wallet.
- Enable Oracle Net tracing to determine the name of the file that cannot be opened and the reason.
- Ensure that auto login was enabled when you saved the wallet. See "Using Auto Login" on page 8-19

ORA-28786: Decryption of Encrypted Private Key Failure

Cause: An incorrect password was used to decrypt an encrypted private key. Frequently, this happens because an **auto login wallet** is not being used.

Action: Use Oracle Wallet Manager to turn the auto login feature on for the wallet. Then re-save the wallet. See "Using Auto Login" on page 8-19.

ORA-28858: SSL Protocol Error

Cause: This is a generic error that can occur during SSL handshake negotiation between two processes.

Action: Enable Oracle Net tracing and attempt the connection again to produce trace output. Then contact Oracle customer support with the trace output.

ORA-28859 SSL Negotiation Failure

Cause: An error occurred during the negotiation between two processes as part of the SSL protocol. This error can occur when two sides of the connection do not support a common cipher suite.

Action: Check the following:

- Use Oracle Net Manager to ensure that the SSL versions on both the client and the server match, or are compatible. For example, if the server accepts only SSL 3.0 and the client accepts only TLS 1.0, then the SSL connection will fail.
- Use Oracle Net Manager to check what cipher suites are configured on the client and the server, and ensure that compatible cipher suites are set on both. See "Step 4: Set the Client SSL Cipher Suites (Optional)" on page 7-28 for details about setting compatible cipher suites on the client and the server. Note: if you do not configure any cipher suites, then all available cipher suites are enabled.

ORA-28862: SSL Connection Failed

Cause: This error occurred because the peer closed the connection.

Action: Check the following:

- Ensure that the correct wallet location is specified in the `sqlnet.ora` file so the system can find the wallet.
- Use Oracle Net Manager to ensure that cipher suites are set correctly in the `sqlnet.ora` file. (Sometimes this error occurs because the `sqlnet.ora` has been manually edited and the cipher suite names are misspelled. Note that case sensitive string matching is used with cipher suite names.)
- Use Oracle Net Manager to ensure that the SSL versions on both the client and the server match, or are compatible. Sometimes this error occurs because the SSL version specified on the server and client do not match. For example, if the server accepts only SSL 3.0 and the client accepts only TLS 1.0, then the SSL connection will fail.
- For more diagnostic information, enable Oracle Net tracing on the peer.

ORA-28865: SSL Connection Closed

Cause: The SSL connection closed because of an error in the underlying transport layer, or because the peer process quit unexpectedly.

Action: Check the following:

- Use Oracle Net Manager to ensure that the SSL versions on both the client and the server match, or are compatible. Sometimes this error occurs because the SSL version specified on the server and client do not match. For example, if the server accepts only SSL 3.0 and the client accepts only TLS 1.0, then the SSL connection will fail.
- If you are using a Diffie-Hellman anonymous cipher suite and the `SSL_CLIENT_AUTHENTICATION` parameter is set to `true` in the server's `listener.ora` file, then the client does not pass its certificate to the server. When the server does not receive the client's certificate, it (the server) cannot authenticate the client so the connection is closed. To resolve this use another cipher suite, or set this `listener.ora` parameter to false.
- Enable Oracle Net tracing and check the trace output for network errors.
- See Actions listed for "ORA-28862: SSL Connection Failed" on page 7-32

ORA-28868: Peer Certificate Chain Check Failed

Cause: When the peer presented the **certificate chain**, it was checked and that check failed. This failure can be caused by a number of problems, including:

- One of the certificates in the chain is expired.

- A certificate authority for one of the certificates in the chain is not recognized as a **trust point**.
- The signature in one of the certificates cannot be verified.

Action: See "Opening an Existing Wallet" on page 8-13 to use Oracle Wallet Manager to open your wallet and check the following:

- Ensure that all of the certificates installed in your wallet are current (not expired).
- Ensure that a certificate authority's certificate from your peer's certificate chain is added as a **trusted certificate** in your wallet. See "Importing a Trusted Certificate" on page 8-25 to use Oracle Wallet Manager to import a trusted certificate.

ORA-28885: No Certificate with Required Key Usage Was Found

Cause: Your certificate was not created with the appropriate X.509 Version 3 key usage extension.

Action: Use Oracle Wallet Manager to check the certificate's key usage. See Table 8–1, "KeyUsage Values" on page 8-5.

ORA-29024: Certificate Validation Failure

Cause: The certificate sent by the other side could not be validated. This may occur if the certificate has expired, has been revoked, or is invalid for another reason.

Action: Check the following:

- Check the certificate to determine whether it is valid. If necessary, get a new certificate, inform the sender that her certificate has failed, or resend.
- Check to ensure that the server's wallet has the appropriate **trust point**s to validate the client's certificate. If it does not, then use Oracle Wallet Manager to import the appropriate trust point into the wallet. See "Importing a Trusted Certificate" on page 8-25 for details.
- Ensure that the certificate has not been revoked and that certificate revocation list (CRL) checking is turned on. See "Configuring Certificate Validation with Certificate Revocation Lists" on page 7-37

ORA-29223: Cannot Create Certificate Chain

Cause: A **certificate chain** cannot be created with the existing **trust point**s for the certificate being installed. Typically, this error is returned when the peer

does not give the complete chain and you do not have the appropriate trust points to complete it.

Action: Use Oracle Wallet Manager to install the trust points that are required to complete the chain. See "Importing a Trusted Certificate" on page 8-25

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 **certificate authority** (CA) has digitally signed the certificate
- The certificate's digital signature corresponds to the independently-calculated hash value of the certificate itself and the certificate signer's (CA's) public key
- 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 certificate revocation list (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. (See **certificate revocation lists**)

This section contains the following topics:

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 CRL Checking Works

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. Typically, CRL definitions are valid for a few days. 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. When the system finds a CRL that matches the certificate CA's DN, 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. See "Renaming CRLs with a Hash Value for Certificate Validation" on page 7-41

2. Oracle Internet Directory

   If the server cannot locate the CRL on the local file system and directory connection information has been configured in an `ldap.ora` file, then the server searches in the directory. It searches the CRL subtree by using the CA's **distinguished name (DN)** and the DN of the CRL subtree.

   See "To create an ldap.ora file for your Oracle home:" on page 12-7 (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.) Also note that if you store CRLs in the directory, then you must use the `orapki` utility to periodically update them. See "Uploading CRLs to Oracle Internet Directory" on page 7-42

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 appropriate CRL that contains revocation information for that certificate is downloaded. Currently, Oracle Advanced Security supports downloading CRLs over HTTP and LDAP.

Note:

- For performance reasons, only user certificates are checked.
- Oracle recommends that you store CRLs in the directory rather than the local file system.

Configuring Certificate Validation with Certificate Revocation Lists

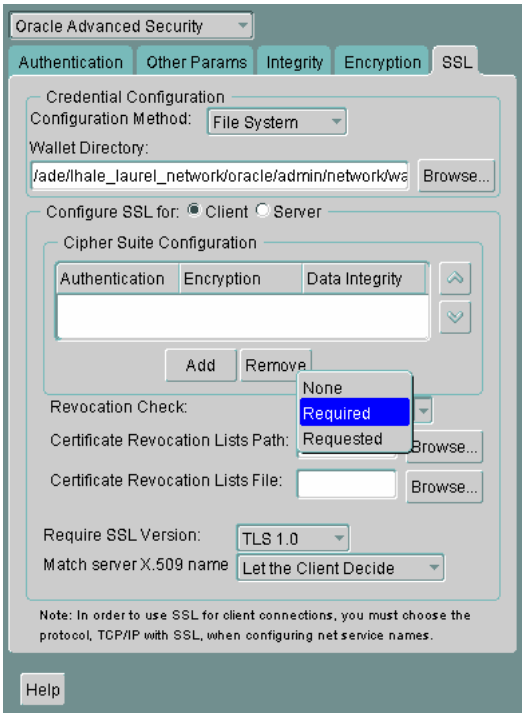
The `SSL_CERT_REVOCATION` parameter must be set to `REQUIRED` or `REQUESTED` in the `sqlnet.ora` file to enable certificate revocation status checking. By default this parameter is set to `NONE` indicating that certificate revocation status checking is turned off.

Note: If you want to store CRLs on your local file system or in Oracle Internet Directory, then you must use the command line utility, `orapki`, to rename CRLs in your file system or upload them to the directory. See: "Certificate Revocation List Management" on page 7-40 for information about using `orapki`.

To enable certificate revocation status checking for the client or the server:

1. Navigate to the **SSL** tab of the Oracle Advanced Security window in Oracle Net Manager, and select either **Client** or **Server** for the **Configure SSL for:** field.

Figure 7–7 Oracle Advanced Security SSL Window with Certificate Revocation Checking Selected



2. Choose one of the following options from the **Revocation Check** list (see Figure 7–7):

   - **REQUIRED**

     Requires certificate revocation status checking. The SSL connection is rejected if a certificate is revoked or no CRL is found. SSL connections are accepted only if it can be verified that the certificate has not been revoked.

   - **REQUESTED**

     Performs certificate revocation status checking if a CRL is available. The SSL connection is rejected if a certificate is revoked. SSL connections are accepted if no CRL is found or if the certificate has not been revoked.

Note: For performance reasons, only user certificates are checked for revocation.

---

3. (Optional) If CRLs are stored on your local file system, then set one or both of the following fields that specify where they are stored. These fields are available only when **Revocation Check** is set to **REQUIRED** or **REQUESTED**.

   - **Certificate Revocation Lists Path:**

     Enter the path to the directory where CRLs are stored, or click **Browse** to find it by searching the file system. Specifying this path sets the `SSL_CRL_PATH` parameter in the `sqlnet.ora` file. If a path is not specified for this parameter, then the default is the wallet directory. Both DER-encoded (binary format) and **PEM**-encoded (BASE64) CRLs are supported.

   - **Certificate Revocation Lists File:**

     Enter the path to a comprehensive CRL file (where PEM-encoded (BASE64) CRLs are concatenated in order of preference in one file), or click **Browse** to find it by searching the file system. Specifying this file sets the `SSL_CRL_FILE` parameter in the `sqlnet.ora` file. If this parameter is set, then the file must be present in the specified location, or else the application will error out during startup.

Note: If you want to store CRLs in a local file system directory by setting the **Certificate Revocation Lists Path**, then you must use the `orapki` utility to rename them so the system can locate them. See "Renaming CRLs with a Hash Value for Certificate Validation" on page 7-41

---

4. (Optional) If CRLs are fetched from Oracle Internet Directory, then directory server and port information must be specified in an `ldap.ora` file. See "To create an ldap.ora file for your Oracle home:" on page 12-7

Note: When configuring your ldap.ora file, you should specify only a non-SSL port for the directory. CRL download is done as part of the SSL protocol, and making an SSL connection within an SSL connection is not supported.

Oracle Advanced Security CRL functionality will not work if the Oracle Internet Directory non-SSL port is disabled.

5. Choose **File** > **Save Network Configuration**. The `sqlnet.ora` file is updated.

To disable certificate revocation status checking:

1. Navigate to the **SSL** tab of the Oracle Advanced Security window in Oracle Net Manager, and select **Configure SSL for: Server**.
2. Choose **NONE** from the **Revocation Check** list.
3. Choose **File** > **Save Network Configuration**. The `sqlnet.ora` file is updated with the following entry:

```
SSL_CERT_REVOCATION=NONE
```

See Also: "Troubleshooting Certificate Validation" on page 7-45 for information about resolving certificate validation errors.

Certificate Revocation List Management

Before you can 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) where your system can use them. Oracle Advanced Security provides a command-line utility, `orapki`, that you can use to perform the following tasks:

- Displaying orapki Help
- 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.

> **See Also:** Appendix A, "Syntax for Command-Line Tools" in *Oracle Internet Directory Application Developer's Guide* for information about LDAP command-line tools and their syntax.

Displaying orapki Help

You can display all the `orapki` commands that are available for managing CRLs by entering the following at the command line:

```
orapki crl help
```

This command displays all available CRL management commands and their options.

> **Note:** Using the `-summary`, `-complete`, or `-wallet` command options is always optional. A command will still run if these command options are not specified.

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 appropriate 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 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 the appropriate 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 Windows file systems:

```
orapki crl hash -crl crl_filename [-wallet wallet_location] -copy crl_directory
[-summary]
```

where `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 where the CRL is located.

Using `-wallet` and `-summary` are optional. Specifying `-wallet` causes the tool to verify the validity of the CRL against the CA's certificate 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 where they can be centrally managed, greatly reducing the administrative overhead of CRL management and use.

The user who uploads CRLs to the directory by using `orapki` must be a member of the directory group `CRLAdmins` (`cn=CRLAdmins,cn=groups,%s_OracleContextDN%`). 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]
```

where *`crl_location`* is the file name or URL where the CRL is located, *`hostname`* and *`ssl_port`* (SSL port with no authentication) are for 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 CA's certificate 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 where the CRL is stored in the directory.

Note:

- The `orapki` utility will prompt 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 is useful for browsing 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
```

where the *`hostname`* and `ssl_`*`port`* are for 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
```

where *crl_location* is 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. See: "Listing CRLs Stored in Oracle Internet Directory" on page 7-43.

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 CA's certificate.

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 graphical user interface 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

The user who deletes CRLs from the directory by using `orapki` must be a member of the directory group `CRLAdmins`. See "Uploading CRLs to Oracle Internet Directory" on page 7-42 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]
```

where *issuer_name* is the name of the CA who issued the CRL, the *hostname* and *ssl_port* are for 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. See "Uploading CRLs to Oracle Internet Directory" on page 7-42 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
```

Troubleshooting Certificate Validation

To determine whether certificates are being validated against CRLs, you can enable Oracle Net tracing. When a revoked certificate is validated by using CRLs, then you will see the following entries in the Oracle Net tracing file without error messages logged between `entry` and `exit`:

```
nzcrlVCS_VerifyCRLSignature: entry
nzcrlVCS_VerifyCRLSignature: exit

nzcrlVCD_VerifyCRLDate: entry
nzcrlVCD_VerifyCRLDate: exit

nzcrlCCS_CheckCertStatus: entry
nzcrlCCS_CheckCertStatus: Certificate is listed in CRL
nzcrlCCS_CheckCertStatus: exit
```

Note that when certificate validation fails, the peer in the SSL handshake sees an `ORA-29024: Certificate Validation Failure`. If this message displays, see "ORA-29024: Certificate Validation Failure" on page 7-34 for information about how to resolve the error.

See Also: *Oracle Net Services Administrator's Guide* for information about setting tracing parameters to enable Oracle Net tracing

Oracle Net Tracing File Error Messages Associated with Certificate Validation

The following trace messages, relevant to certificate validation, may be logged between the `entry` and `exit` entries in the Oracle Net tracing file. Oracle SSL looks for CRLs in multiple locations, so there may be multiple errors in the trace.

Check the following list of possible error messages for information about how to resolve them.

CRL signature verification failed with RSA status

Cause: The CRL signature cannot be verified.

Action: Ensure that the downloaded CRL is issued by the peer's CA and that the CRL was not corrupted when it was downloaded. Note that the `orapki` utility verifies the CRL before renaming it with a hash value or before uploading it to the directory. See "Certificate Revocation List Management" on page 7-40 for information about using `orapki` for CRL management.

CRL date verification failed with RSA status

Cause: The current time is later than the time listed in the next update field. You should not see this error if CRL DP is used. The systems searches for the CRL in the following order:

1. File system
2. Oracle Internet Directory
3. CRL DP

The first CRL found in this search may not be the latest.

Action: Update the CRL with the most recent copy.

CRL could not be found

Cause: The CRL could not be found at the configured locations. This will return error ORA-29024 if the configuration specifies that certificate validation is require.

Action: Ensure that the CRL locations specified in the configuration are correct by performing the following steps:

1. Use Oracle Net Manager to check if the correct CRL location is configured. See "Configuring Certificate Validation with Certificate Revocation Lists" on page 7-37

2. If necessary, use the `orapki` utility to configure CRLs for system use as follows:
   - For CRLs stored on your local file system, see "Renaming CRLs with a Hash Value for Certificate Validation" on page 7-41
   - CRLs stored in the directory, see "Uploading CRLs to Oracle Internet Directory" on page 7-42

OID hostname or port number not set

Cause: Oracle Internet Directory (OID) connection information is not set. Note that this is not a fatal error. The search continues with CRL DP.

Action: If you want to store the CRLs in Oracle Internet Directory, then use Oracle Net Configuration Assistant to create and configure an `ldap.ora` file for your Oracle home. See "To create an ldap.ora file for your Oracle home:" on page 12-7

Fetch CRL from CRL DP: No CRLs found

Cause: The CRL could not be fetched by using the CRL DP. This happens if the certificate does not have a location specified in its CRL DP extension, or if the URL specified in the CRL DP extension is incorrect.

Action: Manually download the CRL. Then depending on whether you want to store it on your local file system or in Oracle Internet Directory, perform the following steps:

If you want to store the CRL on your local file system:

1. Use Oracle Net Manager to specify the path to the CRL directory or file. See "Configuring Certificate Validation with Certificate Revocation Lists" on page 7-37
2. Use the `orapki` utility to configure the CRL for system use. See "Renaming CRLs with a Hash Value for Certificate Validation" on page 7-41

If you want to store the CRL in Oracle Internet Directory:

1. Use Oracle Net Configuration Assistant to create and configure an `ldap.ora` file with directory connection information. See "To create an ldap.ora file for your Oracle home:" on page 12-7
2. Use the `orapki` utility to upload the CRL to the directory. See "Uploading CRLs to Oracle Internet Directory" on page 7-42

Configuring Your System to Use Hardware Security Modules

Oracle Advanced Security supports hardware security modules that use APIs which conform to the RSA Security, Inc., PKCS #11 specification. Typically, these hardware devices are used to securely store and manage private keys in tokens or smart cards, or to accelerate cryptographic processing.

This section contains the following topics:

- General Guidelines for Using Hardware Security Modules with Oracle Advanced Security
- Configuring Your System to Use nCipher Hardware Security Modules
- Troubleshooting Using Hardware Security Modules

General Guidelines for Using Hardware Security Modules with Oracle Advanced Security

The following general guidelines apply if you are using a hardware security module with Oracle Advanced Security:

1. Contact your hardware device vendor to obtain the necessary hardware, software, and PKCS #11 libraries.
2. Install the hardware, software, and libraries where appropriate for the hardware security module you are using.
3. Test your hardware security module installation to ensure that it is operating correctly. Refer to your device documentation for instructions.
4. Create a wallet of the type `PKCS11` by using Oracle Wallet Manager and specify the absolute path to the PKCS #11 library (including the library name) if you wish to store the private key in the token. Oracle `PKCS11` wallets contain information that points to the token for private key access.

You can use the wallet containing PKCS #11 information just as you would use any Oracle wallet, except the private keys are stored on the hardware device and the cryptographic operations are performed on the device as well.

See Also: "Creating a Wallet to Store Hardware Security Module Credentials" on page 8-11

Configuring Your System to Use nCipher Hardware Security Modules

Hardware security modules made by nCipher Corporation are certified to operate with Oracle Advanced Security. These modules provide a secure way to store keys and off load cryptographic processing. Primarily, these devices provide the following benefits:

- Off load of cryptographic processing to free your server to respond to more requests
- Secure private key storage on the device
- Administration of keys controlled through the use of smart cards

> **Note:** You must contact your nCipher representative to obtain certified hardware and software to use with Oracle Advanced Security.

Oracle Components Required To Use an nCipher Hardware Security Module

To use an nCipher hardware security module, you need the following components:

- nCipher Hardware Security Module
- Supporting nCipher PKCS #11 library for your platform as follows:
  - (UNIX 32 bit): `libcknfast.so` library
  - (UNIX 64 bit): `libcknfast-64.so` library
  - (Windows): `cknfast.dll` library

> **Note:** You must contact your nCipher representative to have the hardware security module or the secure accelerator installed and to acquire the necessary library.
>
> These tasks must be performed before you can use an nCipher hardware security module with Oracle Advanced Security.

About Installing an nCipher Hardware Security Module

To use the secure accelerator, you must provide the absolute path to the directory that contains the nCipher PKCS #11 library (including the library name) when you create the wallet by using Oracle Wallet Manager. This enables the library to be loaded at runtime. Typically, the nCipher card is installed at the following locations:

- (UNIX) `/opt/nfast`
- (Windows) `C:\nfast`

The nCipher PKCS #11 library is located at the following file system directory locations for typical installations:

- (UNIX 32 bit): `/opt/nfast/toolkits/pkcs11/libcknfast.so`
- (UNIX 64 bit): `/opt/nfast/toolkits/pkcs11/libcknfast-64.so`
- (Windows): `C:\nfast\toolkits\pkcs11\cknfast.dll`

Note: Use the 32-bit library version when using the 32-bit release of Oracle Database and use the 64-bit library version when using the 64-bit release of Oracle Database. For example, use the 64-bit nCipher PKCS #11 library for the Oracle Database for Solaris Operating System (SPARC 64-bit).

Troubleshooting Using Hardware Security Modules

To detect whether the module is being used, you can turn on Oracle Net tracing. If the wallet contains PKCS #11 information and the private key on the module is being used, then you will see the following entries in the Oracle Net tracing file without error messages logged between `entry` and `exit`:

```
nzpkcs11_Init: entry
nzpkcs11CP_ChangeProviders: entry
nzpkcs11CP_ChangeProviders: exit
nzpkcs11GPK_GetPrivateKey: entry
nzpkcs11GPK_GetPrivateKey: exit
nzpkcs11_Init: exit
...
nzpkcs11_Decrypt: entry
nzpkcs11_Decrypt: exit

nzpkcs11_Sign: entry
nzpkcs11_Sign: exit
```

See Also: *Oracle Net Services Administrator's Guide* for information about setting tracing parameters to enable Oracle Net tracing

Error Messages Associated with Using Hardware Security Modules

The following errors are associated with using PKCS #11 hardware security modules:

ORA-43000: PKCS11: library not found

Cause: The system cannot locate the PKCS #11 library at the location specified when the wallet was created. This happens only when the library is moved after the wallet is created.

Action: Copy the PKCS #11 library back to its original location (where it was when the wallet was created).

ORA-43001: PKCS11: token not found

Cause: The smart card that was used to create the wallet is not present in the hardware security module slot.

Action: Ensure that the smart card that was used when the wallet was created is present in the hardware security module slot.

ORA-43002: PKCS11: passphrase is wrong

Cause: This can occur when

- An incorrect password is specified at wallet creation, or
- The PKCS #11 device password is changed after the wallet is created and not updated in the wallet by using Oracle Wallet Manager.

Action: Depending on the cause, take one of the following actions:

- If you see this error during wallet creation, then check to ensure that you have the correct password and re-enter it.
- If the password changed after wallet creation, then use Oracle Wallet Manager to open the wallet and enter a new password.

See Also: "Creating a Wallet to Store Hardware Security Module Credentials" on page 8-11

Note: The nCipher log file is in the directory where the module is installed at the following location:

```
/log/logfile
```

See Also: nCipher documentation for further information about troubleshooting.

8

Using Oracle Wallet Manager

Security administrators use Oracle Wallet ManagerOracle Wallet Manager to manage public key security credentials on Oracle clients and servers. The wallets it creates can be read by Oracle Database, Oracle Application Server 10*g*, and the Oracle Identity Management infrastructure.

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

- Oracle Wallet Manager Overview
- Starting Oracle Wallet Manager
- How To Create a Complete Wallet: Process Overview
- Managing Wallets
- Managing Certificates

See Also:

- "Public Key Infrastructure in an Oracle Environment" on page 7-5, which discusses all of the Oracle PKI components.
- Appendix E, "orapki Utility" for information about the `orapki` command line utility, which can be used to create wallets and issue certificates for testing purposes.

Oracle Wallet Manager Overview

Oracle Wallet Manager is an application that wallet owners use to manage and edit the security credentials in their Oracle wallets. A wallet is a password-protected container that is used to store authentication and signing credentials, including private keys, certificates, and trusted certificates needed by SSL. You can use Oracle Wallet Manager to perform basic tasks such as creating wallets, generating certificate requests, and opening wallets to access PKI-based services. In addition, Oracle Wallet Manager can save credentials to hardware security modules by using APIs which comply to the Public-Key Cryptography Standards #11 (**PKCS #11**) specification. Oracle Wallet Manager can be used to upload wallets to and download them from an LDAP directory. Oracle Wallet Manager can also be used to import third-party **PKCS #12**-format wallets, and export Oracle wallets to a third-party environment.

Oracle Wallet Manager provides the following features:

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

See Also: "Public Key Infrastructure in an Oracle Environment" on page 8-2

Wallet Password Management

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

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

Strong Wallet Encryption

Oracle Wallet Manager stores private keys associated with X.509 certificates and uses Triple-DES encryption.

Microsoft Windows Registry Wallet Storage

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

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

Options Supported:

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

See Also: *Oracle Database Platform Guide for Windows*

Backward Compatibility

Oracle Wallet Manager is backward-compatible to Release 8.1.7.

Public-Key Cryptography Standards (PKCS) Support

RSA Laboratories, a division of RSA Security, Inc., has developed, in cooperation with representatives from industry, academia, and government, a family of basic

cryptography standards called Public-Key Cryptography Standards, or PKCS for short. These standards have been developed to establish interoperability between computer systems that use public-key technology to secure data across intranets and the Internet.

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

Oracle Wallet Manager wallets can be enabled to store credentials on hardware security modules that use 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 Third-Party Wallets" on page 8-13
- "Exporting Oracle Wallets to Third-Party Environments" on page 8-14
- "Creating a Wallet to Store Hardware Security Module Credentials" on page 8-11
- To view PKCS standards documents, navigate to the following URL: `http://www.rsasecurity.com/rsalabs/PKCS`

Multiple Certificate Support

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

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

Oracle Wallet Manager supports multiple certificates for a single digital entity, where each certificate can be used for a set of Oracle PKI certificate usages, but the same certificate cannot be used for all such usages (See Table 8–2 and Table 8–3 for

legal usage combinations). There must be a one-to-one mapping between certificate requests and certificates. The same certificate request can be used to obtain multiple certificates; however, more than one certificate for each certificate request cannot be installed in the same wallet at the same time.

Oracle Wallet Manager uses the X.509 Version 3 KeyUsage extension to define Oracle PKI certificate usages (Table 8–1):

Table 8–1 KeyUsage Values

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

When installing a certificate (user certificate or **trusted certificate**), Oracle Wallet Manager maps the KeyUsage extension values to Oracle PKI certificate usages as specified in Table 8–2 and Table 8–3.

Table 8–2 Oracle Wallet Manager Import of User Certificates to an Oracle Wallet

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

Table 8–2 Oracle Wallet Manager Import of User Certificates to an Oracle Wallet

KeyUsage Value	Critical?[1]	Usage
2 alone, or 2 + any combination excluding 5	na	Accept certificate for SSL or S/MIME encryption use.
5 alone, or any combination including 5	na	Accept certificate for CA certificate signing use.
Any settings not listed previously	Yes	Not importable.
	No	Certificate is importable for SSL or S/MIME encryption use.

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

Table 8–3 Oracle Wallet Manager Import of Trusted Certificates to an Oracle Wallet

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

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

You should obtain certificates from the certificate authority with the correct KeyUsage value for the required Oracle PKI certificate usage. A single wallet can contain multiple **key pairs** for the same usage. Each certificate can support multiple Oracle PKI certificate usages, as indicated by Table 8–2 and Table 8–3. 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.

If you do not have a certificate with SSL usage, then an ORA-28885 error (`No certificate with required key usage found`) is returned.

LDAP Directory Support

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

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. If a directory contains Oracle8*i* (or prior) users, they are automatically upgraded to use the wallet upload and download feature on first use.

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

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

See Also:

- Uploading a Wallet to an LDAP Directory on page 8-15.
- Downloading a Wallet from an LDAP Directory on page 8-16
- Multiple Certificate Support on page 8-4, for more information about Oracle PKI certificate usage.

Starting Oracle Wallet Manager

To start Oracle Wallet Manager:

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

How To Create a Complete Wallet: Process Overview

Wallets provide a necessary repository in which you can securely store your user certificates and the **trust points** you need to validate the certificates of your peers.

The following steps provide an overview of the complete wallet creation process:

1. Use Oracle Wallet Manager to create a new wallet:
   - See "Required Guidelines for Creating Wallet Passwords" on page 8-9 for information about creating a wallet password
   - See "Creating a New Wallet" on page 8-10 for information about creating standard wallets (store credentials on your file system) and hardware security module wallets.
2. Generate a certificate request. Note that when you create a new wallet with Oracle Wallet Manager, the tool automatically prompts you to create a certificate request. See "Adding a Certificate Request" on page 8-21 for information about creating a certificate request.
3. Send the certificate request to the 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. See "Exporting a User Certificate Request" on page 8-25. Note that the certificate request becomes part of your 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**, then you can import these certificates in the following order. (Note that user certificates and trusted certificates in the PKCS #7 format can be imported at the same time.)
   - First import the CA's trusted certificate into your wallet. See "Importing a Trusted Certificate" on page 8-25 Note that this step may be 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.
   - After you have successfully imported the trusted certificate, then import the user certificate that the CA sent to you into your wallet. See "Importing the User Certificate into the Wallet" on page 8-22
5. (Optional) Set the auto login feature for your wallet. See "Using Auto Login" on page 8-19.

   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 process, you have a wallet that contains a user certificate and its associated trust points.

Managing Wallets

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

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

Required Guidelines for Creating Wallet Passwords

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

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

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

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

See Also:

- Wallet Password Management on page 8-2.
- "Using Auto Login" on page 8-19

Creating a New 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

Unless you have a hardware security module (a PKCS #11 device), then you should use a standard wallet that stores credentials in a directory on your file system.

To create a standard wallet, perform the following tasks:

1. Choose **Wallet > New** from the menu bar. The New Wallet dialog box appears.
2. Follow the "Required Guidelines for Creating Wallet Passwords" on page 8-9 and enter a password in the **Wallet Password** field. This password protects unauthorized use of your credentials.
3. Re-enter that password in the **Confirm Password** field.
4. Choose **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 appears:

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

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 add a certificate request. See "Adding a Certificate Request" on page 8-21.

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

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. 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 PKCS #11 credentials on a hardware security module, perform the following tasks:

1. Choose **Wallet > New** from the menu bar; the New Wallet dialog box appears.
2. Follow the "Required Guidelines for Creating Wallet Passwords" on page 8-9 and enter a password in the **Wallet Password** field.
3. Re-enter that password in the **Confirm Password** field.
4. Choose **PKCS11** from the **Wallet Type** list, and click **OK** to continue. The New PKCS11 Wallet window appears.
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 where the PKCS11 library is stored, or click **Browse** to find it by searching the file system.
7. Enter the **SmartCard password**, and choose **OK**.

   The smart card password, which is different from the wallet password, is stored in the wallet.
8. An alert is displayed, and informs you that a new empty wallet has been created. It prompts you to decide whether you want to add a certificate request. See "Adding a Certificate Request" on page 8-21.

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

Note: If you change the smart card password or move the PKCS #11 library, an error message displays when you try to open the wallet. Then you are prompted to enter the new smart card password or the new path to the library.

Opening an Existing Wallet

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

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

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

Closing a Wallet

To close an open wallet in the currently selected directory:

Choose **Wallet > Close**.

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

Importing Third-Party Wallets

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

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

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

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

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

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

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

See Also: "Importing a Trusted Certificate" on page 8-25

Exporting Oracle Wallets to Third-Party Environments

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

To export a wallet to third-party environments:

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

Note:

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

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

You can export a wallet to a text-based PKI format if you want to put a wallet into a tool that does not support PKCS #12. Individual components are formatted according to the standards listed in Table 8–4. Within the wallet, only those certificates with SSL key usage are exported with the wallet.

To export a wallet to text-based PKI format:

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

Table 8–4 PKI Wallet Encoding Standards

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

Uploading a Wallet to an LDAP Directory

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

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

To upload a wallet:

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

   Wallet needs to be saved before uploading.

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

   - **If at least one certificate has SSL key usage:** When prompted, enter the LDAP directory server hostname and port information, then click **OK**. Oracle Wallet Manager attempts connection to the LDAP directory server using SSL.A message appears indicating whether the wallet was uploaded successfully or it failed.

- **If no certificates have SSL key usage:** When prompted, enter the user's **distinguished name (DN)**, the LDAP server hostname and port information, and click **OK**. Oracle Wallet Manager attempts connection to the LDAP directory server using simple password authentication mode, assuming that the wallet password is the same as the directory password.

  If the connection fails, a dialog box prompts for the directory password of the specified DN. Oracle Wallet Manager attempts connection to the LDAP directory server using this password and displays a warning message if the attempt fails. Otherwise, Oracle Wallet Manager displays a status message at the bottom of the window indicating that the upload was successful.

Downloading a Wallet from an LDAP Directory

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

See Also:

- "Saving Changes" on page 8-17
- "Saving the Open Wallet to a New Location" on page 8-17
- "Saving in System Default" on page 8-17

To download a wallet from an LDAP directory:

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

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

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

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

Saving Changes

To save your changes to the current open wallet:

Choose **Wallet > Save**.

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

Saving the Open Wallet to a New Location

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

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

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

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

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

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

Saving in System Default

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

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

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

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

Note:

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

Deleting the Wallet

To delete the current open wallet:

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

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

Changing the Password

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

Note: If you are using a wallet with auto login enabled, you must regenerate the auto login wallet after changing the password. See "Using Auto Login" on page 8-19

To change the password for the current open wallet:

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

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

See Also:

- "Required Guidelines for Creating Wallet Passwords" on page 8-9
- "Wallet Password Management" on page 8-2, for password policy restrictions.

Using Auto Login

The Oracle Wallet Manager auto login feature creates an obfuscated copy of the wallet and enables PKI-based access to services without a password until the auto login feature is disabled for the wallet. 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:

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

Disabling Auto Login

To disable auto login:

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

Managing Certificates

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

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

- Managing User Certificates
- Managing Trusted Certificates

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

Managing User Certificates

User certificates can be used by end users, smart cards, or applications, such as Web servers. Server certificates are a type of user certificate. For example, if a CA issues a certificate for a Web server, placing its **distinguished name (DN)** in the Subject field, then the Web server is the certificate owner, thus the "user" for this user certificate. User certificates do not validate other user certificates, except when they are used as a trusted certificate in a *user-centric* trust model.

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

Managing user certificates involves the following tasks:

- Adding a Certificate Request

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

Adding a Certificate Request

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

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

To create a PKCS #10 certificate request:

1. Choose **Operations > Add Certificate Request**. The Add Certificate Request dialog box appears.
2. Enter the information specified in Table 8–5.
3. Choose **OK**. A message informs you that a certificate request was successfully created. You can either copy the certificate request text from the body of this dialog panel and paste it into an e-mail message to send to a certificate authority, or you can export the certificate request to a file.
4. Choose **OK** to return to the Oracle Wallet Manager main window. The status of the certificate changes to **[Requested]**.

See Also: "Exporting a User Certificate Request" on page 8-25

Table 8–5 Certificate Request: Fields and Descriptions

Field Name	Description
Common Name	Mandatory. Enter the name of the user's or service's identity. Enter a user's name in first name / last name format. Example: Eileen.Sanger
Organizational Unit	Optional. Enter the name of the identity's organizational unit. Example: Finance.

Table 8–5 (Cont.) Certificate Request: Fields and Descriptions

Field Name	Description
Organization	Optional.Enter the name of the identity's organization. Example: XYZ Corp.
Locality / City	Optional. Enter the name of the locality or city in which the identity resides.
State / Province	Optional. Enter the full name of the state or province in which the identity resides. Enter the full state name, because some certificate authorities do not accept two–letter abbreviations.
Country	Mandatory. Choose to view a list of country abbreviations. Select the country in which the organization is located.
Key Size	Mandatory. Choose to view a list of key sizes to use when creating the public / private key pair. See Table 8–6 to evaluate key size.
Advanced	Optional. Choose **Advanced** to view the Advanced Certificate Request dialog panel. Use this field to edit or customize the identity's distinguished name (DN). For example, you can edit the full state name and locality.

Table 8–6 lists the available key sizes and the relative security each size provides. Typically, CAs use key sizes of 1024 or 2048. When certificate owners wish to keep their keys for a longer duration, they choose 3072 or 4096 bit keys.

Table 8–6 Available Key Sizes

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

Importing the User Certificate into the Wallet

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

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

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

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

Keyboard shortcuts for copying and pasting certificates:

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

To import a file that contains the user certificate:

The file containing the user certificate should have been saved in either text (BASE64) or binary (`der`) format.

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

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

Removing a User Certificate from a Wallet

To remove a user certificate from a wallet:

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

Removing a Certificate Request

You must remove a certificate before removing its associated request.

To remove a certificate request:

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

Exporting a User Certificate

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

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

Exporting a User Certificate Request

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

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

Managing Trusted Certificates

Managing trusted certificates includes the following tasks:

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

Importing a Trusted Certificate

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

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

To 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. Choose **Operations > Import Trusted Certificate...** from the menu bar. The Import Trusted Certificate dialog panel appears.

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

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

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

Keyboard shortcuts for copying and pasting certificates:

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

To import a file that contains the trusted certificate:

The file containing the trusted certificate should have been saved in either text (BASE64) or binary (`der`) format.

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

Removing a Trusted Certificate

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

To remove a trusted certificate from a wallet:

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

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

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

Exporting a Trusted Certificate

To export a trusted certificate to another file system location:

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

Exporting All Trusted Certificates

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

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

9

Configuring Multiple Authentication Methods and Disabling Oracle Advanced Security

This chapter describes how to configure multiple authentication methods under Oracle Advanced Security, and how to use conventional user name and password authentication, even if you have configured another authentication method. This also chapter describes how to configure your network so that Oracle clients can use a specific authentication method, and Oracle servers can accept any method specified.

This chapter contains the following topics:

Connecting with User Name and Password

To connect to an Oracle database server using a user name and password when an Oracle Advanced Security authentication method has been configured, disable the external authentication (See: "Disabling Oracle Advanced Security Authentication" on page 9-2).

With the external authentication disabled, a user can connect to a database using the following format:

```
% sqlplus username/password@net_service_name
```

For example:

```
% sqlplus scott/tiger@emp
```

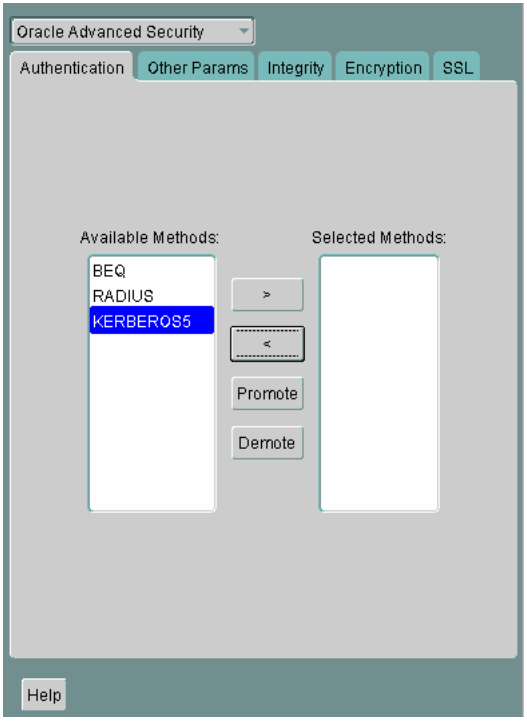
Note: You can configure multiple authentication methods, including both externally authenticated users and password authenticated users, on a single database.

Disabling Oracle Advanced Security Authentication

Use Oracle Net Manager to disable authentication methods (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile. (See "Navigating to the Oracle Advanced Security Profile" on page 2-3). The Oracle Advanced Security tabbed window appears (Figure 9–1):

Figure 9–1 Oracle Advanced Security Authentication Window



2. Choose the Authentication tab.
3. Sequentially move all authentication methods from the Selected Method list to the Available Methods list by selecting a method and choosing the left arrow [<].
4. Choose File > Save Network Configuration.

   The `sqlnet.ora` file is updated with the following entry:

   ```
   SQLNET.AUTHENTICATION_SERVICES = (NONE)
   ```

Configuring Multiple Authentication Methods

Many networks use more than one authentication method on a single security server. Accordingly, Oracle Advanced Security lets you configure your network so that Oracle clients can use a specific authentication method, and Oracle database servers can accept any method specified.

You can set up multiple authentication methods on both client and server systems either by using Oracle Net Manager, or by using any text editor to modify the `sqlnet.ora` file.

Use Oracle Net Manager to add authentication methods to both clients and servers (See "Starting Oracle Net Manager" on page 2-2):

1. Navigate to the Oracle Advanced Security profile. (See "Navigating to the Oracle Advanced Security Profile" on page 2-3) The Oracle Advanced Security tabbed window appears (Figure 9–1).
2. Choose the Authentication tab.
3. Select a method listed in the Available Methods list.
4. Sequentially move selected methods to the Selected Methods list by choosing the right arrow [>].
5. Arrange the selected methods in order of desired use. To do this, select a method in the Selected Methods list, and choose Promote or Demote to position it in the list.
6. Choose File > Save Network Configuration.

   The `sqlnet.ora` file is updated with the following entry, listing the selected authentication methods:

   ```
   SQLNET.AUTHENTICATION_SERVICES = (RADIUS|KERBEROS5)
   ```

Note:

- SecurID functionality is available through RADIUS; RADIUS support is built into the RSA ACE/Server.
- See Also: Chapter 5, "Configuring RADIUS Authentication"

Configuring Oracle Database for External Authentication

This section describes the parameters you must set to configure Oracle Database for network authentication, using the following tasks:

- Setting the SQLNET.AUTHENTICATION_SERVICES Parameter in sqlnet.ora
- Verifying that REMOTE_OS_AUTHENT Is Not Set to TRUE
- Setting OS_AUTHENT_PREFIX to a Null Value

See Also:

- The corresponding chapter in this guide for information about configuring a particular authentication method
- Appendix B, "Authentication Parameters"

Setting the SQLNET.AUTHENTICATION_SERVICES Parameter in sqlnet.ora

The following parameter must be set in the `sqlnet.ora` file for all clients and servers to enable each to use a supported authentication method:

```
SQLNET.AUTHENTICATION_SERVICES=(oracle_authentication_method)
```

For example, for all clients and servers using Kerberos authentication, the `sqlnet.ora` parameter must be set as follows:

```
SQLNET.AUTHENTICATION_SERVICES=(KERBEROS5)
```

Verifying that REMOTE_OS_AUTHENT Is Not Set to TRUE

To verify that REMOVE_OS_AUTHENT is not set to TRUE, add the following parameter to the initialization file—in each database instance—when you configure the authentication method:

```
REMOTE_OS_AUTHENT=FALSE
```

Caution: Setting `REMOTE_OS_AUTHENT` to `TRUE` can cause a security exposure, because it lets someone using a non-secure protocol, such as TCP, perform an operating system-authorized login (formerly referred to as an `OPS$` login).

If REMOTE_OS_AUTHENT is set to FALSE, and the server cannot support any of the authentication methods requested by the client, the authentication service negotiation fails and the connection terminates.

If the parameter is set as follows in the sqlnet.ora file on either the client or server, the database attempts to use the supplied user name and password to login the user:

```
SQLNET.AUTHENTICATION_SERVICES=(NONE)
```

If REMOTE_OS_AUTHENT is set to FALSE, however, the connection fails.

Setting OS_AUTHENT_PREFIX to a Null Value

Authentication service-based user names can be long, and Oracle user names are limited to 30 characters. Oracle Corporation strongly recommends that you enter a null value for the OS_AUTHENT_PREFIX parameter in the initialization file used for the database instance as follows:

```
OS_AUTHENT_PREFIX=""
```

Note: The default value for OS_AUTHENT_PREFIX is OPS$; however, you can set it to any string.

Attention: If a database already has the OS_AUTHENT_PREFIX **set to a value other than** NULL **(" "),** ***do not change it,*** **since it can inhibit previously created, externally identified users from connecting to the Oracle server.**

To create a user, launch SQL*Plus and enter the following:

```
SQL> CREATE USER os_authent_prefix username IDENTIFIED EXTERNALLY;
```

When OS_AUTHENT_PREFIX is set to a null value (" "), enter the following to create the user king:

```
SQL> CREATE USER king IDENTIFIED EXTERNALLY;
```

The advantage of creating a user in this way is that the administrator no longer needs to maintain different user names for externally identified users. This is true for all supported authentication methods.

See Also:

- *Oracle Database Administrator's Guide*
- *Oracle Database Heterogeneous Connectivity Administrator's Guide*

10

Configuring Oracle DCE Integration

Oracle **DCE** Integration enables Oracle applications and tools to access Oracle Database servers in a distributed computing environment. This chapter briefly describes the **Distributed Computing Environment (DCE)**, the Oracle DCE Integration product, and how to configure it. It contains the following topics:

- Introduction to Oracle DCE Integration
- Configuring DCE for Oracle DCE Integration
- Configuring Oracle Database and Oracle Net Services for Oracle DCE Integration
- Connecting to an Oracle Database Server in the DCE Environment
- Connecting Clients Outside DCE to Oracle Servers in DCE

See Also: "Related Documentation" on page -xxix.

Introduction to Oracle DCE Integration

The **Distributed Computing Environment (DCE)** from the Open Group is a set of integrated network services that works across multiple systems to provide a distributed environment. The network services include remote procedure calls (RPCs), directory service, security service, threads, distributed file service, diskless support, and distributed time service.

DCE is the middleware between distributed applications and the operating system / network services and is based on a client / server model of computing. By using the services and tools that DCE provides, users can create, use, and maintain distributed applications that run across a heterogeneous environment.

Oracle DCE Integration enables Oracle applications and tools to access Oracle database servers in a DCE environment.

System Requirements

Oracle DCE Integration requires Oracle Net Services and Oracle Database. It is based on the Open Software Foundation (OSF) DCE protocol (V1.1 and later).

Note that OSF has merged with X / OPEN, another standards group, to form The Open Group. This group is committed to continuing DCE support.

Backward Compatibility

Oracle servers running DCE Integration 2.3.2 and later are backward compatible with *clients* running SQL*Net / DCE 2.1.6 or 2.2.3; however, Release 2.1.6 clients cannot take advantage of external roles.

A client running DCE Integration 2.3.2 or later cannot connect to a SQL*Net / DCE 2.1.6 or 2.2.3 *server.* A DCE Integration Release 2.3.2 or later client requires a Release 2.3.2 or later server in order to connect to a database.

Components of Oracle DCE Integration

Oracle DCE Integration has two components: DCE Communication / Security and DCE CDS Native Naming.

- DCE Communication / Security
- DCE Cell Directory Services Native Naming

DCE Communication/Security

This component has three principal features:

Authenticated RPC Oracle DCE Integration provides authenticated Remote Procedure Call (RPC) as the transport mechanism that enables multi-vendor interoperability. RPC also uses some of the other DCE services, including directory and security services, to provide location transparency and secure distributed computing.

Integrated Security and Single Sign-On Oracle DCE Integration works with the DCE Security service to provide security within DCE cells. It enables a user logged onto DCE to securely access any Oracle database without having to specify a user name or password. This is sometimes called **external authentication** to the database, or **single sign-on (SSO)**. Clients and servers that are not running DCE authentication services can interoperate with systems that have DCE security by specifying an Oracle password.

Data Privacy and Integrity Oracle DCE Integration uses the multiple levels of security that DCE provides to ensure data authenticity, privacy, and integrity. Users have a range of choices, from no protection to full encryption for each connection, with a guarantee that no data is modified in transit.

Note: For parts of the network that do not use DCE, you can use the other security and authentication services that are part of Oracle Advanced Security. These services work with SQL*Net release 2.1 and later or with Oracle Net Services. They provide message integrity and data encryption services in non-DCE environments, letting administrators ensure that all network traffic is protected against unauthorized viewing or modification, regardless of the start or end point.

DCE Cell Directory Services Native Naming

The DCE **Cell Directory Services (CDS)** Native Naming component includes naming and location transparency.

DCE Integration registers Oracle Database connect descriptors in the DCE CDS, letting them be transparently accessed across the entire DCE environment. Users can connect to Oracle database servers in a DCE environment using familiar Oracle service names.

The DCE CDS offers a distributed, replicated repository service for name, address, and attributes of objects across the network. Because servers register their name and address information in the CDS, Oracle clients can make location-independent connections to Oracle Database servers. Services can be relocated without any changes to the client configuration. An Oracle utility is provided to load the Oracle service names with corresponding connect descriptors into CDS. After this is done, Oracle connect descriptors can be viewed from a central location with standard DCE tools.

For location of services across multiple cells, either of the following options can be used:

- DCE **Global Directory Service (GDS)**
- Internet Domain Naming Service (DNS)

See Also:

- To configure DCE to use CDS naming, see "Configuring DCE for Oracle DCE Integration" on page 10-5
- To configure Oracle clients and servers to use CDS, see "Configuring Oracle Database and Oracle Net Services for Oracle DCE Integration" on page 10-8
- For information about how Oracle Native Naming works with other Oracle name services, see the *Oracle Net Services Administrator's Guide*.

Flexible DCE Deployment

Oracle Advanced Security provides flexibility in your use of DCE services. You have the following options:

- You can use full DCE integration in your environment to integrate with all the DCE Secure Core services (RPC, directory, security, threads).
- You can use only the DCE directory services by using the DCE CDS Native Naming adapter, along with any conventional protocol adapter, such as TCP/IP.

Release Limitations

The following are limitations in 10*g* Release 1 (10.1) of Oracle Advanced Security:

- Only one listener address that uses the DCE protocol is permitted for each node.
- Database links must specify a user name and password to connect.
- This release of DCE Integration does not support the Oracle Multi-Protocol Interchange.
- This release does not work with the Oracle shared server.

Configuring DCE for Oracle DCE Integration

The following tasks, performed by the DCE cell administrator, assume that a DCE cell has been configured and the systems being used are part of that cell:

- Task 1: Create New Principals and Accounts
- Task 2: Install the Key of the Server into a Keytab File
- Task 3: Configure DCE CDS for Use by Oracle DCE Integration

Task 1: Create New Principals and Accounts

Use the following procedure model to add server principals:

```
% dce_login cell_admin password
% rgy_edit
Current site is: registry server at /.../cell1/subsys/dce/sec/master
rgy_edit=>do p
Domain changed to: principal
rgy_edit=> add oracle
rgy_edit=> do a
Domain changed to: account
rgy_edit=> add oracle -g none -o none -pw oracle_password -mp cell_admin_
password
rgy_edit=> quit
bye
```

In this example, a DCE principal named `oracle` is created. The principal has a corresponding account with a password set to `oracle_password`. The account does not belong to any DCE group or DCE profile.

Note: Perform this task on the server only once after DCE Integration has been installed. Do not perform this task on client systems.

Task 2: Install the Key of the Server into a Keytab File

Install the key of the server into a keytab file, dcepa.key. This file contains the password of the principal under which the Oracle Net listener starts. The Oracle Net listener reads this file to authenticate itself to DCE. To generate the keytab file, enter the following:

```
% dce_login cell_admin password
% rgy_edit
Current site is: registry server at /.../cell1/subsys/dce/sec/master
rgy_edit=> ktadd -p oracle -pw Oracle_password -f
$ORACLE_HOME/dcepa/admin/dcepa.key
rgy_edit=>quit
bye
```

Note:

- Perform this task on the server only once after DCE Integration has been installed. Do not perform this task on client systems.
- Remember to substitute the full path name for the $ORACLE_HOME variable. If the specified directories do not exist, create them before running the command. To create the directories. enter the following:

  ```
  mkdir $ORACLE_HOME/dcepa
  mkdir $ORACLE_HOME/dcepa/admin
  ```

Task 3: Configure DCE CDS for Use by Oracle DCE Integration

1. Create Oracle directories in the CDS namespace by entering the following after installing DCE Integration for the first time in a cell. Create directories on all CDS replicas:

   ```
   % dce_login cell_admin

   Enter Password:(password not displayed)
   $ cdscp
   ```

```
cdscp> create dir /.:/subsys/oracle
cdscp> create dir /.:/subsys/oracle/names
cdscp> create dir /.:/subsys/oracle/service_registry
cdscp> exit
```

Note:

- The directory `/.:/subsys/oracle/names` contains objects that map Oracle Net service names to connect descriptors, which are used by the CDS naming adapter.
- The directory `/.:/subsys/oracle/service_registry` contains objects that map the service name in DCE addresses to the network endpoint that is used by both DCE protocol adapter clients and servers.

2. Give servers permission to create objects in the CDS namespace by entering the following, which adds the principal `oracle` to the CDS-server group:

```
$ dce_login cell_admin
Enter Password:   (password not displayed)
$ rgy_edit
rgy_edit=> domain group
Domain changed to: group
rgy_edit=> member subsys/dce/cds-server -a oracle
rgy_edit=> exit
```

3. Load Oracle service names into CDS as described in "Configuring Oracle Database and Oracle Net Services for Oracle DCE Integration" on page 10-8.

Configuring Oracle Database and Oracle Net Services for Oracle DCE Integration

This section describes how to configure an Oracle database server and Oracle Net Services to use Oracle DCE Integration after it has been successfully installed. It contains the following topics:

DCE Address Parameters

DCE addresses in the `listener.ora` and `tnsnames.ora` configuration files are defined by DCE parameters, illustrated in the following:

```
ADDRESS=(PROTOCOL=DCE)(SERVER_PRINCIPAL=server_name)(CELL_NAME=cell_name)
(SERVICE=dce_service_name))
```

These parameters are described by Table 10–1:

Table 10–1 DCE Address Parameters and Definitions

Component	Description
PROTOCOL	A mandatory field that identifies the DCE RPC protocol.
SERVER_PRINCIPAL	A mandatory field for the server and an optional field for the client. The server authenticates itself to DCE as this principal. This field is mandatory in the listener configuration file (`listener.ora`) and specifies the principal the server will start under. This field is optional in your local naming configuration file (`tnsnames.ora`) and specifies the principal of the server the client must connect to. If not specified, then one-way authentication is used. In this case, the client does not care what principal the server is running under.

Table 10–1 (Cont.) DCE Address Parameters and Definitions

Component	Description
CELL_NAME	An optional parameter. If present, it specifies the DCE cell name of the database. If this parameter is not set, the cell name defaults to the local cell (useful for single-cell environments). Optionally, the SERVICE parameter (described in the following section) may specify the complete path (including the cell name) to the service, making this parameter unnecessary.
SERVICE	A mandatory field for both server and client. For the server, this is the service registered with CDS. For the client, this is the service name used when querying CDS for the location of the Oracle DCE servers. The default directory for storing service names in CDS is `/.../cellname/subsys/oracle /service_registry`. This service name can fully specify the path in CDS.

You can specify a service as follows:

```
SERVICE=/.../cell_name/subsys/oracle/service_registry/dce_service_name
```

Alternatively, you can specify:

```
SERVICE=dce_service_name
```

if `CELL_NAME=cell_name` is also specified.

In this case, the cell name defaults to the local cell. However, this way of specifying service names only works if you are operating within a single cell.

Note: The *`dce_service_name`* in the service field might not be the same as that used by Oracle Net Services. The service name used by Oracle Net is mapped to the connect descriptor in a local naming configuration file (`tnsnames.ora`). The *`dce_service_name`* is part of the address within the connect descriptor.

Task 1: Configure the Server

To configure a server for DCE Integration, do the following:

1. Configure the listener configuration file (`listener.ora`) with DCE address information for all servers.

2. For servers in distributed systems that require database link connections to other servers, configure the `sqlnet.ora` and `protocol.ora` files with DCE address information.

   Note: In this release, the configuration files `listener.ora`, `sqlnet.ora`, `tnsnames.ora`, and `protocol.ora` are located in the $ORACLE_HOME/network/admin directory.

   For a database server to receive connections from Oracle Net clients in a DCE environment, there must be an Oracle Net listener active on the server platform. This process listens for connections on a network address that is defined in the `listener.ora` configuration file.

   The `SERVER_PRINCIPAL` parameter designates what DCE principal the listener should be running under. In the following sample, the listener is running under principal `oracle`.

   The following is a sample DCE address as it would appear in the `listener.ora` file.

```
LSNR_DCE=
  (ADDRESS=
      (PROTOCOL=DCE)
      (SERVER_PRINCIPAL=oracle)
      (CELL_NAME=cell1)
      (SERVICE=dce_svc))
SID_LIST_LSNR_DCE=
      (SID_DESC=
      (SID_NAME=ORASID)
       (ORACLE_HOME=/private/oracle9))
```

Task 2: Create and Name Externally Authenticated Accounts

To use DCE authentication for logging on to an Oracle database, you must create database accounts that are authenticated externally. To enable secure external authentication, do the following:

Note: The privileges shown in this section are the *minimum access privileges necessary*. The actual set of privileges needed depends upon the instance or application.

1. Verify that these lines are in the initialization parameter file:

```
REMOTE_OS_AUTHENT=FALSE
OS_AUTHENT_PREFIX=""
```

2. Verify that the initialization parameter file does not have a multi-threaded server (MTS) entry for DCE. For example, an entry such as the following is *not* permitted:

```
mts_dispatchers="(PROTOCOL=dce)(DISPATCHERS=3)"
```

> **Note:** The MTS_DISPATCHERS initialization parameter is obsolete in 10*g* Release 1 (10.1). See *Oracle Database Upgrade Guide* for further details.

3. Ensure that you are logged on as a member of the DBA group. Restart the database instance for the changes to take effect.

4. At the SQL*Plus prompt, define users. Before doing so, decide whether you are, or ever will be, operating in a multi-cell DCE environment in which you let Oracle access across cell boundaries. The way you define users depends on whether they are connecting within a single cell or across cell boundaries.

   Local Cell:

   If users are connecting within a local cell, use the following format:

```
SQL> CREATE USER server_principal IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO server_principal;
```

   For example:

```
SQL> CREATE USER oracle IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO oracle;
```

   The entire CELL_NAME/SERVER_PRINCIPAL string must be 30 characters or less *(this is an Oracle Database restriction—not a restriction of the DCE adapter).*

   For example:

```
SQL> CREATE USER "CELL1/ORACLE" IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO "CELL1/ORACLE";
```

   Multiple Cells:

If connecting to the database across multiple cells, specify both the *cell_name* and the *server_principal*, as illustrated in the following:

```
SQL> CREATE USER "CELL_NAME/SERVER_PRINCIPAL" IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO "CELL_NAME/SERVER_PRINCIPAL";
```

You must enclose the externally-identified account name in double quotation marks, because the slash is a reserved character. Also, if the account (user) name is double-quoted, it must be capitalized.

For example:

```
SQL> CREATE USER "CELL1/ORACLE" IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO "CELL1/ORACLE";
```

When using this format, set the following parameter in the `protocol.ora` configuration file to FALSE:

```
dce.local_cell_usernames=false
```

References to an Oracle account created in this manner must include the schema / account in the correct format. Consider requests for access to tables from another account. When a user references the tables in another account created within a local cell, the command might appear as follows:

```
SQL> SELECT * FROM oracle.emp
```

If a user wants to access tables in another account created for connections across cells, the command might appear as follows:

```
SQL> SELECT * FROM "CELL1/ORACLE" .emp
```

> **See Also:** *Oracle Database Heterogeneous Connectivity Administrator's Guide*, for more information about external authentication

Task 3: Set up DCE Integration External Roles

To set up external roles for DCE Integration, and enable connection to an Oracle database as SYSOPER or SYSDBA with DCE credentials, do the following:

1. Set the following parameter in the initialization parameter file:

   ```
   OS_ROLES=TRUE
   ```

2. Restart the database.

3. Ensure that the DCE groups that map to Oracle roles adhere to the following syntax:

```
ORA_global_name_role[_[a][d]]
```

Table 10–2 describes the syntax components:

Table 10–2 Setting Up External Role Syntax Components

Component	Definition
ORA	Designates that this group is used for Oracle purposes
GLOBAL_NAME	The global name for the database
ROLE	The name of the role, as defined in the data dictionary
A or a	Optional character indicating that the user has admin privileges for this role
D or d	Optional character indicating the role is to be enabled by default at connect time

See Also: *Oracle Database Administrator's Guide* for more information about external roles

4. Authenticate to DCE a user who is a member of a DCE group by entering the following commands:

```
dce_login
klist
```

Sample Output:

```
% dce_login oracle
```

Enter Password:

```
% klist
dce identity information:
Warning: Identity information is not certified
Global Principal: /.../ilab1/oracle
Cell:      001c3f90-01f5-1f72-ba65-02608c2c84f3 /.../ilab1
Principal: 00000068-0568-2f72-bd00-02608c2c84f3 oracle
Group:     0000000c-01f5-2f72-ba01-02608c2c84f3 none
```

```
Local Groups:
0000000c-01f5-2f72-ba01-02608c2c84f3 none
0000006a-0204-2f72-b901-02608c2c84f3 subsys/dce/cds-server
00000078-daf4-2fe1-a201-02608c2c84f3 ora_dce222_dba
00000084-89c8-2fe8-a201-02608c2c84f3 ora_dce222_connect_d
00000087-8a13-2fe8-a201-02608c2c84f3 ora_dce222_resource_d
00000080-f681-2fe1-a201-02608c2c84f3 ora_dce222_role1_ad
.
.
.
```

5. Connect to the database as usual.

The following sample output lists external roles (DBA, CONNECT, RESOURCE, and ROLE1) that have been mapped to DCE groups:

```
SQL> SELECT * FROM session_roles;

ROLE
------------------------------
CONNECT
RESOURCE
ROLE1

SQL> SET ROLE all;

Role set.

SQL> SELECT * FROM session_roles;

ROLE
------------------------------
DBA
EXP_FULL_DATABASE
IMP_FULL_DATABASE
CONNECT
RESOURCE
ROLE1

6 rows selected.

SQL> EXIT
```

Task 4: Configure DCE for SYSDBA and SYSOPER Connections to Oracle Databases

To configure DCE so that you can connect to an Oracle database as SYSOPER or SYSDBA with DCE credentials, do the following:

1. Create DCE groups that map to Oracle DBA and OPERATOR roles. DCE group names should adhere to the syntax described by Task 3: Set up DCE Integration External Roles on page 10-12. Add the externally authenticated user `oracle` as a member of the group(s).

```
$ dce_login cell_admin cell_admin_password
$ rgy_edit
rgy_edit=> domain group
Domain changed to: group
rgy_edit=> add ora_dce222_dba_ad
rgy_edit=> add ora_dce222_operator_ad
rgy_edit=> member ora_dce222_dba_ad -a oracle
rgy_edit=> member ora_dce222_operator_ad -a oracle
```

2. Add the GLOBAL_NAME parameter to the DCE address or TNS service name in the local configuration file `tnsnames.ora`.

```
ORADCE=
    (ADDRESS=
               (PROTOCOL=DCE)
               (SERVER_PRINCIPAL=oracle)
               (CELL_NAME=cell1)
               (SERVICE=dce_svc))
 (CONNECT_DATA=
             (SID=ORASID)
             (GLOBAL_NAME=dce222)))
```

3. Create the database user `oracle` as described by Task 2: Create and Name Externally Authenticated Accounts on page 10-10.

4. Get DCE credentials for the externally authenticated user:

```
$ dce_login oracle oracle_password
$klist
DCE Identity Information:
        Warning: Identity information is not certified
        Global Principal: /.../dce.dlsun685.us.oracle.com/oracle
        Cell:      00af8052-7e94-11d2-b261-9019b88baa77
/.../dce.dlsun685.us.ora
cle.com
        Principal: 0000006d-88b9-21d2-9300-9019b88baa77 oracle
```

```
        Group:     0000000c-7e94-21d2-b201-9019b88baa77 none
        Local Groups:
                0000000c-7e94-21d2-b201-9019b88baa77 none
                0000006a-7e94-21d2-ad01-9019b88baa77 subsys/dce/cds-server
                00000076-8b53-21d2-9301-9019b88baa77 ora_dce222_dba_ad
                00000077-8b53-21d2-9301-9019b88baa77 ora_dce222_operator_ad

Identity Info Expires: 1999-12-04-10:28:22
Account Expires:       never
Passwd Expires:        never

Kerberos Ticket Information:
Ticket cache: /opt/dcelocal/var/security/creds/dcecred_43ae2600
Default principal: oracle@dce.dlsun685.us.oracle.com
Server: krbtgt/dce.dlsun685.us.oracle.com@dce.dlsun685.us.oracle.com
        valid 1999-12-04-00:28:22 to 1999-12-04-10:28:22
Server: dce-rgy@dce.dlsun685.us.oracle.com
        valid 1999-12-04-00:28:22 to 1999-12-04-10:28:22
Server: dce-ptgt@dce.dlsun685.us.oracle.com
        valid 1999-12-04-00:28:26 to 1999-12-04-02:28:26
Client: dce-ptgt@dce.dlsun685.us.oracle.com     Server:
krbtgt/dce.dlsun685.us.o
racle.com@dce.dlsun685.us.oracle.com
        valid 1999-12-04-00:28:26 to 1999-12-04-02:28:26
Client: dce-ptgt@dce.dlsun685.us.oracle.com     Server:
dce-rgy@dce.dlsun685.us.
oracle.com
        valid 1999-12-04-00:28:27 to 1999-12-04-02:28:26
```

Note: List output shows the DCE group membership of `oracle`.

5. Connect to the Oracle database as SYSDBA or SYSOPER.

   For example:

   ```
   SQL> connect /@oradce as SYSDBA
   ```

Task 5: Configure the Client

To configure a client for DCE Integration, you must configure the following Oracle Net files with DCE address and parameter information:

- protocol.ora
- sqlnet.ora

Typically, CDS is used for name resolution. Thus, a local naming configuration file (tnsnames.ora) is not used, except when loading names and addresses into CDS.

Parameters in protocol.ora

There are four DCE parameters located in the protocol.ora file. Each parameter begins with the prefix DCE. to distinguish it from parameters relevant to other protocols. If default values are used for these four parameters, DCE Integration does not require a protocol.ora file. The parameters and their current defaults follow:

- DCE.AUTHENTICATION=*dce_secret*
- DCE.PROTECTION=*pkt_integ*
- DCE.TNS_ADDRESS_OID=1.3.22.1.5.1
- DCE.LOCAL_CELL_USERNAMES=TRUE

Configuration parameters are not case-sensitive; you can enter them in either uppercase or lowercase.

DCE.AUTHENTICATION

The DCE.AUTHENTICATION parameter is optional. It indicates the authentication value to be used for each DCE RPC. The client DCE_AUTHENTICATION value must be the same as the server DCE_AUTHENTICATION value. If this entry is not specified, cell-wide default authentication is used. The options follow:

Option	Description
NONE	No authentication
DCE_SECRET	DCE shared-secret key authentication (Kerberos)
DCE_SECRET	Default authentication level and recommended value
DEFAULT	Cell default

DCE.PROTECTION

DCE.PROTECTION is an optional field that specifies the data integrity protection levels for data transmission. The client DCE_PROTECTION level must be equal to or

greater than the server DCE_PROTECTION level. If this entry is not specified, cell-wide default protection is used. The options follow:

Option	Description
NONE	Perform no protection for the current connection
DEFAULT	Use the default cell-wide protection level
CONNECT	Perform protection only when the client establishes a relationship with the server
CALL	Perform protection only at the beginning of each remote procedure call when the server receives the request
PKT	Ensure that all data received is from the expected client
PKT_INTEG	Ensure and verify that none of the data transferred between the client and server has been modified
PRIVACY	Perform protection as specified by all of the previous levels and also encrypt each RPC argument value and all user data in each call

DCE.TNS_ADDRESS_OID

DCE.TNS_ADDRESS_OID is an optional parameter that enables you to specify an alternative to the default value as follows:

```
DCE.TNS_ADDRESS_OID=1.3.22.1.x.x
```

> **See Also:** "Step 2: Modify the CDS Attributes File and Restart the CDS" on page 10-20.

DCE.LOCAL_CELL_USERNAMES

DCE.LOCAL_CELL_USERNAMES is an optional parameter that defines the format used to specify the principal name (username), with or without the cell name. The choice you make for this parameter should be determined by whether or not users are making connections across cells—with unique names. The default for DCE.LOCAL_CELL_USERNAMES is now TRUE (it was set to FALSE in the DCE Integration 2.1.6 release).

The associated options follow:

Option	Description
TRUE	The default value. Select TRUE if using just the SERVER_PRINCIPAL format, without the CELL_NAME. An example of a user specified in this format is as follows: oracle TRUE is an appropriate option if users are making connections within a single cell, or if naming conventions in the network assure that users in different cells do not have duplicate names.
FALSE	Select FALSE when using the CELLNAME/SERVER_PRINCIPAL format. An example of a user specified in this format is as follows: CELL1/ORACLE FALSE is an appropriate option if users are making connections across cells and there can be users in different cells with identical name

Task 6: Configure Clients to Use DCE CDS Naming

Clients typically use **Cell Directory Services (CDS)** to resolve Oracle service names to addresses. Perform the following steps to configure CDS:

- Step 1: Enable CDS for use in Performing Name Lookup
- Step 2: Modify the CDS Attributes File and Restart the CDS
- Step 3: Create a tnsnames.ora File for Loading Oracle Connect Descriptors into CDS
- Step 4: Load Oracle Connect Descriptors into CDS
- Step 5: Delete or Rename the tnsnames.ora File
- Step 6: Modify the sqlnet.ora File to Resolve Names in CDS

Note: Upon completion of this task, you can connect to an Oracle database in your DCE environment.

Step 1: Enable CDS for use in Performing Name Lookup

To use CDS for name resolution, the DCE Integration CDS Naming Adapter must be installed on all clients and servers that use CDS. Also, the CDS namespace must have been configured for use by DCE Integration.

> **See Also:** DCE Integration installation instructions, and "Task 3: Configure DCE CDS for Use by Oracle DCE Integration" on page 10-6.

For example, a service name such as ORADCE and its network address can be stored in DCE CDS.

Users can typically connect to Oracle services using the familiar Oracle service name if there are no domains or the database is in the user's default domain, as in the following example:

```
sqlplus /@ORADCE
```

This example assumes that DCE externally-authenticated accounts are in use.

As an alternative name resolution service, use a local naming configuration file, tnsnames.ora, when CDS is inaccessible. To do so, locate names and addresses of all Oracle servers in the local tnsnames.ora file.

Step 2: Modify the CDS Attributes File and Restart the CDS

On all DCE machines where CDS naming is used, add the object ID for the CDS attribute TNS_Address to the CDS attributes file. (The object ID must be the same across all machines.)

1. Add a line in the following format to the /opt/dcelocal/etc/cds_attributes file:

   ```
   1.3.22.1.5.1    TNS_Address    char
   ```

   The first four digits of this TNS_Address attribute value, 1.3.22.1.x.y, are fixed, under DCE naming conventions. If the default TNS_Address object ID value 1.3.22.1.5.1 already exists in the cds_attributes file, you must specify a value for the object ID that is not already in use.

   If you are unable to use the default value for the object ID, then you must specify the object ID in the protocol.ora file on the client.

   If you had to specify a value other than the default value 1.3.22.1.5.1, then you must add the following parameter to the protocol.ora file:

   ```
   DCE.TNS_ADDRESS_OID=1.3.22.1.x.y
   ```

   Make sure that the object ID value in the cds_attributes file matches the value specified in the DCE.TNS_ADDRESS_OID parameter in the protocol.ora file.

2. Restart CDS on the system.

   The command to restart CDS varies between different operating systems. On the Solaris platform, for example, you can use the following command to restart CDS:

   ```
   /opt/dcelocal/etc/rc.dce restart
   ```

Step 3: Create a tnsnames.ora File for Loading Oracle Connect Descriptors into CDS

To load the Oracle service names and addresses into CDS, create or modify a local naming configuration file, `tnsnames.ora`. This file is used to map service names to addresses for use by Oracle Net.

This section describes the parameters that must be included in the `tnsnames.ora` file. The file contains a list of Oracle service names mapped to connect descriptors of destinations or endpoints in the network. The sample DCE address in the following section shows a network address for an Oracle server with the Oracle service name ORADCE. It is used to connect to the service registered as DCE_SVC in the CDS directory

```
 /.../cell_name/subsys/oracle/names.
ORADCE=(DESCRIPTION=(ADDRESS=(PROTOCOL=DCE)(SERVER_PRINCIPAL=oracle)(CELL_
NAME=cell1)(SERVICE=DCE_SVC))(CONNECT_DATA=(SID=ORASID)))
```

Note: In this example, the Oracle service name and the DCE service name are different, although they are frequently the same.

Parameter Name	Type	Mandatory?	Description
PROTOCOL=DCE	keyword value pair	Yes	Appears in the address sections of (i) `listener.ora`, a listener configuration file, and (ii) `tnsnames.ora`, a local naming configuration file.
SERVER_ PRINCIPAL	DCE Parameter	No	Appears in `tnsnames.ora`
SERVICE	DCE Parameter	Yes	The value given for the DCE parameter (SERVICE=*dce_service_name*) must be the same in `listener.ora` and `tnsnames.ora`

Parameter Name	Type	Mandatory?	Description
SID	Oracle Parameter	Yes	Identifies the Oracle system ID; each SID value must be unique on a node. This parameter is used locally only, and is not used in DCE CDS.

See Also: *Oracle Net Services Administrator's Guide*, for information about `tnsnames.ora`, the local naming configuration file.

Step 4: Load Oracle Connect Descriptors into CDS

A separate utility called `tnnfg` is provided with Oracle DCE Integration to load connect descriptors into CDS. If you configure a new service name and address in `tnsnames.ora`, `tnnfg` adds the new service name and address to CDS. If you change the address for a particular service name, `tnnfg` updates the address for a particular service name.

To load the Oracle service names or aliases from `tnsnames.ora` into CDS, enter the following at the system prompt:

```
% dce_login cell_admin
% tnnfg dceload full_pathname_to_tnsnames.ora
% Enter Password:(password will not display)
```

Be sure to enter the full path name of the `tnsnames.ora` file, and ensure that the `sqlnet.ora` file exists in the same directory as the `tnsnames.ora` file.

Step 5: Delete or Rename the tnsnames.ora File

You can keep `tnsnames.ora` available as a backup in case CDS becomes unavailable. To assure that CDS is routinely searched instead of `tnsnames.ora`, configure the `NAMES.DIRECTORY_PATH` parameter in a profile (`sqlnet.ora`), as described by "Step 6: Modify the sqlnet.ora File to Resolve Names in CDS" (the next section).

Step 6: Modify the sqlnet.ora File to Resolve Names in CDS

The parameters required in a profile (`sqlnet.ora`) depend upon the version of SQL*Net or Oracle Net Services you are using.

For a client or server to use DCE CDS Naming, the administrator must do the following:

1. Ensure that the CDS Naming Adapter has been installed on that node.
2. Add the following parameter to the `sqlnet.ora` file:

   ```
   NAMES.DIRECTORY_PATH=(cds, tnsnames, onames)
   ```

   The first name resolution service listed as a value for this parameter is used. If it is unavailable for any reason, the next name resolution service is used, and so forth.

Connecting to an Oracle Database Server in the DCE Environment

This section describes how to connect to an Oracle database after installing Oracle DCE Integration, and configuring both DCE and Oracle to use Oracle DCE Integration in the following topics:

- Starting the Listener
- Connecting to an Oracle Database by Using DCE Authentication for Single Sign-On
- Connecting to an Oracle Database by Using Password Authentication

Starting the Listener

To start the listener, do the following:

1. Enter the following commands:

   ```
   % dce_login principal_name password
   % lsnrctl start listener_name
   ```

   For example, if the listener name is LSNR_DCE in the `listener.ora` file, enter the following:

   ```
   % dce_login oracle orapwd
   % lsnrctl start LSNR_DCE
   ```

2. Verify that the server has registered its binding handler with rpcd:

   ```
   % rpccp show mapping
   ```

   Look for the line that includes the `dce_service_name` that is part of the listener address.

3. Verify that the service has been created by searching for the `dce_service_name` as follows:

```
% cdscp show object "/.:/subsys/oracle/service_registry/dce_service_name"
```

For example:

The following command shows you the mapping in the CDS namespace that the listener has chosen for the endpoint:

```
% cdscp show object "/.:/subsys/oracle/service_registry/dce_svc"

            SHOW
          OBJECT    /.../subsys/oracle/service_registry/dce_svc
             AT   1999-05-15-17:10:52
RPC_ClassVersion = 0100
         CDS_CTS = 1999-05-16-00:05:01.221106100/aa-00-04-00-3e-8c
         CDS_UTS = 1999-05-16-00:05:01.443343100/aa-00-04-00-3e-8c
       CDS_Class = RPC_Server
CDS_ClassVersion = 1.0
      CDS_Towers = :
           Tower = ncacn_ip_tcp:144.25.23.57[]
```

Connecting to an Oracle Database by Using DCE Authentication for Single Sign-On

After externally-identified accounts have been set up, you can take advantage of DCE authentication to log in to Oracle without providing any username or password information. To use this single sign-on capability, just log in to DCE using a command like the following:

```
% dce_login principal_name password
```

For example:

```
% dce_login oracle orapwd
```

Note: You only need to enter the `dce_login` command once. If you are already logged into DCE, you do not need to log in again.

You can now connect to an Oracle server without using a username or password. Enter a command like the following:

```
% sqlplus /@net_service_name
```

where *net_service_name* is the database service name.

For example:

```
% sqlplus /@ORADCE
```

Connecting to an Oracle Database by Using Password Authentication

From a client, you can still connect with a user name and password:

```
% sqlplus username/password@net_service_name
```

where *net_service_name* is the Oracle Net service name.

For example:

```
% sqlplus scott/tiger@ORADCE
```

Connecting Clients Outside DCE to Oracle Servers in DCE

Clients without access to DCE and CDS can still connect to Oracle servers in DCE using TCP/IP or some other protocol if a listener is configured to do this. If a listener has been configured in the `listener.ora` file on the server, non-DCE clients can use normal Oracle Database and Oracle Net Services procedures to connect to an Oracle server in DCE.

> **Note:** In this case, DCE security is not available to clients. Also, service names are resolved to network addresses and located in a `tnsnames.ora` file on the client, not using the CDS name server.

The following section contains these topics, which include samples of `listener.ora` and `tnsnames.ora` files as they would be configured if a client from outside of DCE wanted to connect to Oracle database servers in a DCE environment:

- Sample Parameter Files
- Using tnsnames.ora for Name Lookup When CDS Is Inaccessible

Sample Parameter Files

At least the following two Oracle parameter files are needed for successful client/server communications; create and modify these files using a text editor:

The parameter files are described in the following sections:

The listener.ora File

The `listener.ora` file resides on the listener node. It defines listener characteristics and the addresses at which the listener listens.

In the following example, each element is displayed on a separate line, to show the file's structure. This is the recommended format, but you do not have to put each element on a separate line. Be sure to include all the appropriate parentheses, and to indent if you must continue an element on the next line.

This example assumes the UNIX operating system and the TCP/IP protocol for one listener, and the DCE protocol for another listener. A single listener can have multiple addresses. For example, instead of having two separate listeners for different database instances on a server node, you could have *one listener for both*, listening on both TCP/IP and on DCE. However, performance is improved with separate listeners.

```
LSNR_TCP=
      (ADDRESS_LIST=
          (ADDRESS=
                 (PROTOCOL=IPC)
                 (KEY=DB1)
          )
          (ADDRESS=
                 (PROTOCOL=tcp)
                 (HOST=rose)
                 (PORT=1521)
          ))

  SID_LIST_LSNR_TCP=
         (SID_DESC=
                (SID_NAME=ORASID)
                (ORACLE_HOME=/usr/jprod/Oracle Database)
         )
LSNR_DCE=
  (ADDRESS=
     (PROTOCOL=DCE)
     (SERVER_PRINCIPAL=oracle)
     (CELL_NAME=cell1)
     (SERVICE=dce_svc))
   SID_LIST_LSNR_DCE=
     (SID_DESC=
```

```
       (SID_NAME=ORASID)
       (ORACLE_HOME=/usr/prod/oracle8))
#For all listeners, the following parameters list sample
#default values.

PASSWORDS_LISTENER=
STARTUP_WAIT_TIME_LISTENER=0
CONNECT_TIMEOUT_LISTENER=10
TRACE_LEVEL_LISTENER=OFF
TRACE_DIRECTORY_LISTENER=/usr/prod/Oracle Database/network/trace
TRACE File_LISTENER=listener.trc
LOG_DIRECTORY_LISTENER=/usr/prod/Oracle Database/network/log
LOG_FILE_LISTENER=listener.log
```

The tnsnames.ora File

This file resides on both the client and the server nodes. It lists the service names and addresses of all services on the network.

The following sample `tnsnames.ora` file maps the service name ORATCP to the connect descriptor that includes a TCP/IP address and the service name ORADCE to a connect descriptor that includes a DCE address.

```
ORATCP = (DESCRIPTION=
          (ADDRESS=
            (PROTOCOL=TCP)
            (HOST=rose)
            (PORT=1521)
           )
           (CONNECT_DATA=
             (SID=DB1)
           )
          )
ORADCE=(DESCRIPTION=
         (ADDRESS=
           (PROTOCOL=DCE)
           (SERVER_PRINCIPAL=oracle)
           (CELL_NAME=cell1)
           (SERVICE=dce_svc)
          )
          (CONNECT_DATA=
                 (SID=ORASID)
          )
          )
```

To access the DB1 database, a user can use ORATCP to identify the appropriate connect descriptor.

For example:

```
sqlplus scott/tiger@oratcp
```

Using tnsnames.ora for Name Lookup When CDS Is Inaccessible

Typically, names are resolved into network addresses by CDS. Although the main purpose of the tnsnames.ora file (in the context of native naming adapters) is to load Oracle service names and network addresses into CDS, it could be used temporarily as a backup name resolution service if CDS is inaccessible.

SQL*Net Release 2.2 and Earlier

To use the tnsnames.ora file for name lookup and resolution, remove (or comment out) the "native name" parameters from the sqlnet.ora file on the client. To comment out the lines, add a pound sign (#) at the beginning of each line.

For example:

```
#native_names.use_native=true
#native_names.directory_path=(dce)
```

SQL*Net Release 2.3 and Oracle Net Services

You can use tnsnames.ora for name lookup and resolution when DCE CDS is unavailable if you have TNSNAMES listed as a value for the NAMES.DIRECTORY_PATH parameter in the sqlnet.ora file on the client.

For example:

```
names.directory_path=(dce, tnsnames)
```

This parameter enables you to list more than one names resolution method. The methods are tried in order. In this example, DCE is attempted first. If it is unsuccessful, TNSNAMES is tried next.

Part IV

Enterprise User Security

This part describes Oracle Database directory and security integration functionality, which enables single sign-on in a client/server environment. It contains the following chapters, which describe how to set up enterprise user security in an Oracle distributed database environment:

- Chapter 11, "Getting Started with Enterprise User Security"
- Chapter 12, "Enterprise User Security Configuration Tasks and Troubleshooting"
- Chapter 13, "Administering Enterprise User Security"

11

Getting Started with Enterprise User Security

Enterprise User Security, a critical component of Oracle Identity Management, lets you create and administer large numbers of users in a secure, **LDAP**-compliant directory service. The following topics in this chapter explain what Enterprise User Security is and how it works:

Introduction to Enterprise User Security

This section provides an overview of Enterprise User Security, explaining the benefits, how enterprise users access resources across a distributed database system, and how they are authenticated. It contains the following topics:

The Challenges of User Management

Administrators must keep user information up to date and secure for the entire enterprise. This task becomes more difficult as the number of applications and users increases. Typically, each user has multiple accounts on different databases, which means each user must remember multiple passwords. The results of these conditions are too many passwords for users to remember and too many accounts for administrators to effectively manage.

With thousands of users accessing database accounts, administrators must devote substantial resources to user administration. Common information used by multiple applications, such as usernames, telephone numbers, and system roles and privileges, is typically fragmented across the enterprise, contributing to data that is redundant, inconsistent, and difficult to manage.

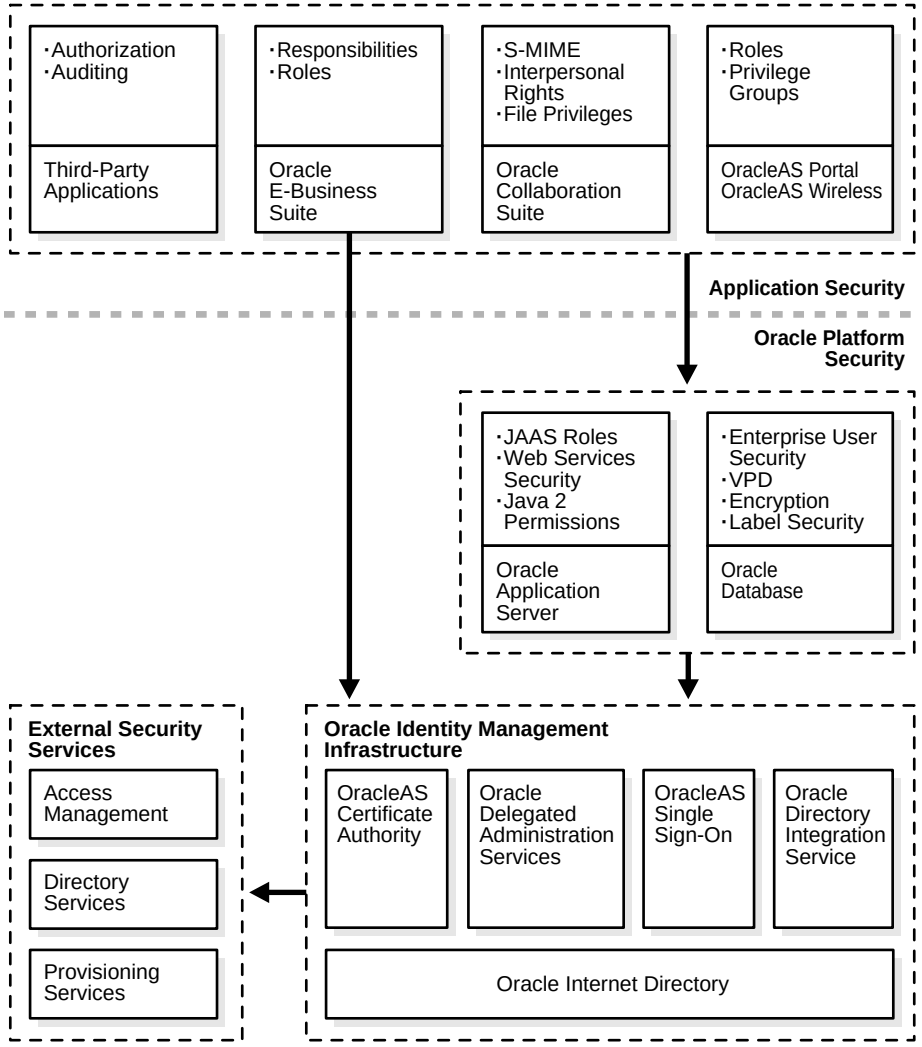
In addition to user and account management problems, these conditions produce security problems as well. For example, any time a user leaves a company or changes jobs, that user's privileges should be changed the same day in order to guard against their misuse. However, in a large enterprise, if you have too many user accounts distributed over multiple databases, an administrator may be unable to make such timely changes. If your users have too many passwords, then they may write them down (making them easy for others to copy), choose passwords that are easy to remember (making them easy for others to guess), or choose the same password for multiple applications (making a compromised password a greater security risk). All of these user efforts to keep track of their multiple passwords can compromise the security of the enterprise.

Enterprise User Security: The Big Picture

Enterprise User Security addresses user, administrative, and security challenges by relying on the identity management services supplied by Oracle Internet Directory, an LDAP-compliant directory service. Identity management is the process by which the complete security life cycle for network entities is managed in an organization. It typically refers to the management of an organization's application users, where steps in the security life cycle include account creation, suspension, privilege modification, and account deletion.

Figure 11–1 shows how Enterprise User Security fits into the Oracle security architecture, which uses the Oracle Identity Management infrastructure as its foundation.

Figure 11–1 Enterprise User Security and the Oracle Security Architecture



Users benefit from Enterprise User Security through **single sign-on (SSO)** or **single password authentication**, depending on the configuration chosen by the administrator. Using single sign-on, users need to authenticate only once and subsequent authentications take place transparently. This functionality requires SSL, and should not be confused with OracleAS Single Sign-On, a component of Oracle Identity Management infrastructure.

Single password authentication lets users authenticate to multiple databases with a single global password although each connection requires a unique authentication. The password is securely stored in the centrally located, LDAP-compliant directory, and protected with security mechanisms including encryption and **Access Control Lists (ACLs)**. This approach improves usability by reducing the number of passwords to remember and manage, and by eliminating the overhead of setting up SSL.

Enterprise User Security requires Oracle Internet Directory 10*g* (9.0.4). Other LDAP-compliant directory services are supported by using Oracle Internet Directory Integration Platform to synchronize them with Oracle Internet Directory.

This section contains the following topics:

- How Oracle Internet Directory Implements Identity Management
- Enterprise Users Compared to Database Users
- About Enterprise User Schemas
- How Enterprise Users Access Database Resources with Database Links
- How Enterprise Users Are Authenticated

> **See Also:** [*Oracle Internet Directory Administrator's Guide*, for information about using Oracle Directory Integration Platform with other directories.

> **Note:** Microsoft Active Directory is only supported for Oracle databases on Windows platforms.

How Oracle Internet Directory Implements Identity Management

Oracle Internet Directory uses the concept of identity management realms to organize information in the directory information tree (DIT), which is a hierarchical tree-like structure consisting of directory object entries. In a directory, each collection of information about an object is called an entry. This object may be a person, but it can also be information about a networked device, such as configuration information. To name and identify the location of directory objects in the DIT, each entry is assigned a unique distinguished name (DN). The DN of an entry consists of the entry itself and its parent entries, connected in ascending order, from the entry itself up to the root (top) entry in the DIT.

About Identity Management Realms An identity management realm is a subtree of directory entries, all of which are governed by the same administrative policies. For example, all employees in an enterprise who have access to the intranet may belong to one realm, while all external users who access the public applications of the enterprise may belong to another realm. Use of different realms enables an enterprise to isolate user populations and enforce different administrative policies, such as password policies or naming policies, in each realm.

About Identity Management Realm-Specific Oracle Contexts Each identity management realm has a realm-specific Oracle Context (realm Oracle Context) that stores Oracle product information for that realm. A realm Oracle Context stores application data, how users are named and located, how users must be authenticated, group locations, and privilege assignments—all specific to the particular identity management realm in which the realm Oracle Context is located.

See Also:

- *Oracle Internet Directory Administrator's Guide* for information about Oracle Internet Directory and its architecture.
- "About Enterprise User Security Directory Entries" on page 11-11 for information about Oracle Internet Directory entries that are used for Enterprise User Security.

Enterprise Users Compared to Database Users

Database users are typically defined in the database by using the `CREATE USER` statement as follows:

```
CREATE USER username IDENTIFIED BY password;
```

This creates a database user, associated with a user schema, who can access the database and be authenticated by using a password with the `CONNECT` command as follows:

```
connect username/password@database_service_name
```

Database users must be created in each database they need to access, and they can choose a different password for each database. Database user privileges are controlled by local roles in each database.

In contrast, enterprise users are provisioned and managed centrally in an LDAP-compliant directory, such as Oracle Internet Directory, for database access. Enterprise users have a unique identity in the directory called the **distinguished**

name (DN). When enterprise users log on to a database, the database authenticates those users by using their DN.

Enterprise users are defined in the database as global users. Global users can have their own schemas, or they can share a global schema in the databases they access. You can create enterprise users by using the GLOBALLY clause in the CREATE USER statement in two different ways.

You can specify a user's directory DN with an AS clause, which is shown in the following statement:

```
CREATE USER username IDENTIFIED GLOBALLY AS '<DN of directory user entry>';
```

In this case, they have a schema allocated exclusively to them.

Alternatively, you can specify a null string with the AS clause as the following statement shows:

```
CREATE USER username IDENTIFIED GLOBALLY AS '';
```

When you specify a null string with the AS clause, the directory maps authenticated users to the appropriate database schema. In this case, multiple users can be mapped to a shared schema based on the mapping information set up and stored in Oracle Internet Directory.

When enterprise users connect over SSL to the database, they do not use a password. Instead they use the following CONNECT command, which looks up the wallet location based on information in the client's `sqlnet.ora` file:

```
connect /@database_service_name
```

Password-authenticated enterprise users use the same CONNECT statement to connect to the database as regular database users. For example, password-authenticated enterprise users connect to the database by using the following syntax:

```
connect username/password@database_service_name
```

When the database receives a connection request from an enterprise user, the database refers to the directory for user authentication and authorization (role) information.

See Also:

- "Creating New Enterprise Users" on page 13-9
- *Oracle Database Security Guide* for more information about global users.
- *Oracle Internet Directory Administrator's Guide* for information about defining users in the directory.

About Enterprise User Schemas

Enterprise users can retain their individual database schemas (exclusive schemas) or share schemas if the enterprise security administrator maps them to a shared schema.

Exclusive-Schema Enterprise Users If users want to retain their individual schemas in the databases that they access, then

- Create enterprise users in the directory, and
- Create a global user schema for each user in each database that they access.

Creating separate accounts for each enterprise user on each database that they access results in significant overhead. Instead, creating enterprise users who access a single, generic shared schema in each database increases the efficiency of the enterprise user solution.

Shared Schema Enterprise Users To receive the real benefit of the enterprise user solution, you can use shared schemas for your enterprise users. For this strategy

- Create enterprise users in the directory,
- Create a single shared schema in each database, and
- Create a single shared schema mapping in Oracle Internet Directory.

Mapping enterprise users to a generic, shared schema on each of the databases that they access greatly reduces the overhead of creating separate schemas for each enterprise user.

Shared schema enterprise users can be mapped to generic, shared schemas on all of the databases that they access, or they can have exclusive schemas on some databases and shared schemas on others. The shared schema mappings are stored in the directory.

See Also: "About Using Shared Schemas for Enterprise User Security" on page 11-19 for more information about creating and using shared schemas for enterprise users.

How Enterprise Users Access Database Resources with Database Links

Database links are network objects stored in the local database or in the network definition that identify a remote database, a communication path to that database, and optionally, a username and password. Once defined, the database link is used to access the remote database. Oracle Database supports connected user links, fixed user links, and current user links.

Enterprise users can use all three types of database links. Connected user links are accessed by a local user who has an account on the remote server. Fixed user links contain a username and password as part of the link definition. Current user database links allow enterprise users to access objects on remote databases without passing authentication information during link execution, or storing authentication information in the link definition. They require SSL for the database network connections, which means public key infrastructure (PKI) credentials must be obtained and maintained for the databases. Current user database links can be used to connect to the remote database only as an enterprise user.

See Also:

- "About Using Current User Database Links for Enterprise User Security" on page 11-23 for detailed information about creating and using current user database links.
- *Oracle Database Administrator's Guide* for information about all of the different types of database links supported by Oracle Database

How Enterprise Users Are Authenticated

Enterprise User Security supports the following authentication methods:

- Password-based authentication
- SSL-based authentication
- Kerberos-based authentication

Each authentication method has advantages and disadvantages. Table 11–1 summarizes the criteria for selecting which authentication method is best for your Enterprise User Security implementation.

Table 11–1 Enterprise User Security Authentication: Selection Criteria

Password Authentication	SSL Authentication	Kerberos Authentication
Password-based authentication.	Provides strong authentication over SSL.	Provides strong authentication by using Kerberos, version 5 tickets.
Provides centralized user and password management.	Provides centralized user and PKI credential / wallet management.	Provides centralized user and Kerberos credential management.
Separate authentications required for each database connection.	Supports **single sign-on (SSO)** using SSL.	Supports **single sign-on (SSO)** using Kerberos, version 5 encrypted tickets and authenticators, and authentication forwarding.
Retains users' current authentication methods.	Initial configuration maybe more difficult because PKI credentials must be generated for all users. (Dependent on administrators' PKI knowledge)	Initial configuration maybe more difficult because Kerberos must be installed and configured to authenticate database users.
User identity can be used in two-tier or multitier applications. OracleAS Single Sign-On users and enterprise users use the same stored password.	Compatible with either a two-tier or multitier environment.	Compatible with either a two-tier or multitier environment.
Supports Oracle Release 7.3 and later clients with an Oracle Database 10*g*.	Supports Oracle8*i* and later clients with an Oracle Database 10*g*.	Supports Oracle Database 10*g* clients and later with an Oracle Database 10*g*.
Supports current user database links only if the connection between databases is over SSL.	Supports current user database links.	Supports current user database links only if the connection between databases is over SSL.
Can use third-party directories to store users if synchronized with Oracle Internet Directory.[1]	Can use third-party directories to store users if synchronized with Oracle Internet Directory.[2]	Can use third-party directories to store users if synchronized with Oracle Internet Directory.[3]

[1] If third-party directory is Microsoft Active Directory, then when user passwords change, they must be changed in both Active Directory and in Oracle Internet Directory.

[2] Must modify the Directory Integration Services agent to synchronize user PKCS #12 attributes.

[3] If third-party directory is Microsoft Active Directory, then login to Windows gives you single sign-on login to databases. However, you must modify the Directory Integration Services agent for other third-party directories to synchronize the `KrbPrincipalName` attribute. This synchronization is automatic for Microsoft Active Directory.

Note: Enterprise User Security supports three-tier environments. Oracle Database 10*g* **proxy authentication** features enable (i) proxy of user names and passwords through multiple tiers, and (ii) proxy of X.509 certificates and distinguished names through multiple tiers.

See Also:

- Chapter 12, "Enterprise User Security Configuration Tasks and Troubleshooting" for information about configuring the various authentication types for enterprise user security.
- *Oracle Database Security Guide*, for information about using proxy authentication.

About Enterprise User Security Directory Entries

In a directory, each collection of information about an object is called an entry. For Enterprise User Security, elements such as users, roles, and databases are directory objects and information about these objects are stored as entries in the directory.

Each entry in the directory is uniquely identified by a distinguished name (DN). The DN tells you exactly where the entry resides in the directory entry hierarchy, which is commonly called the **directory information tree (DIT)**.

Note: In the Oracle Database 10*g* release, databases must be registered in a complete **identity management realm** of Oracle Internet Directory.

See Also: *Oracle Internet Directory Administrator's Guide* for a complete discussion of directory entries.

The following sections describe directory entries related to Enterprise User Security:

Enterprise Users

An **enterprise user** is one that is defined and managed in a directory. Each enterprise user has a unique identity across an enterprise. Enterprise user entries can reside at any location within the identity management realm, except within the realm Oracle Context.

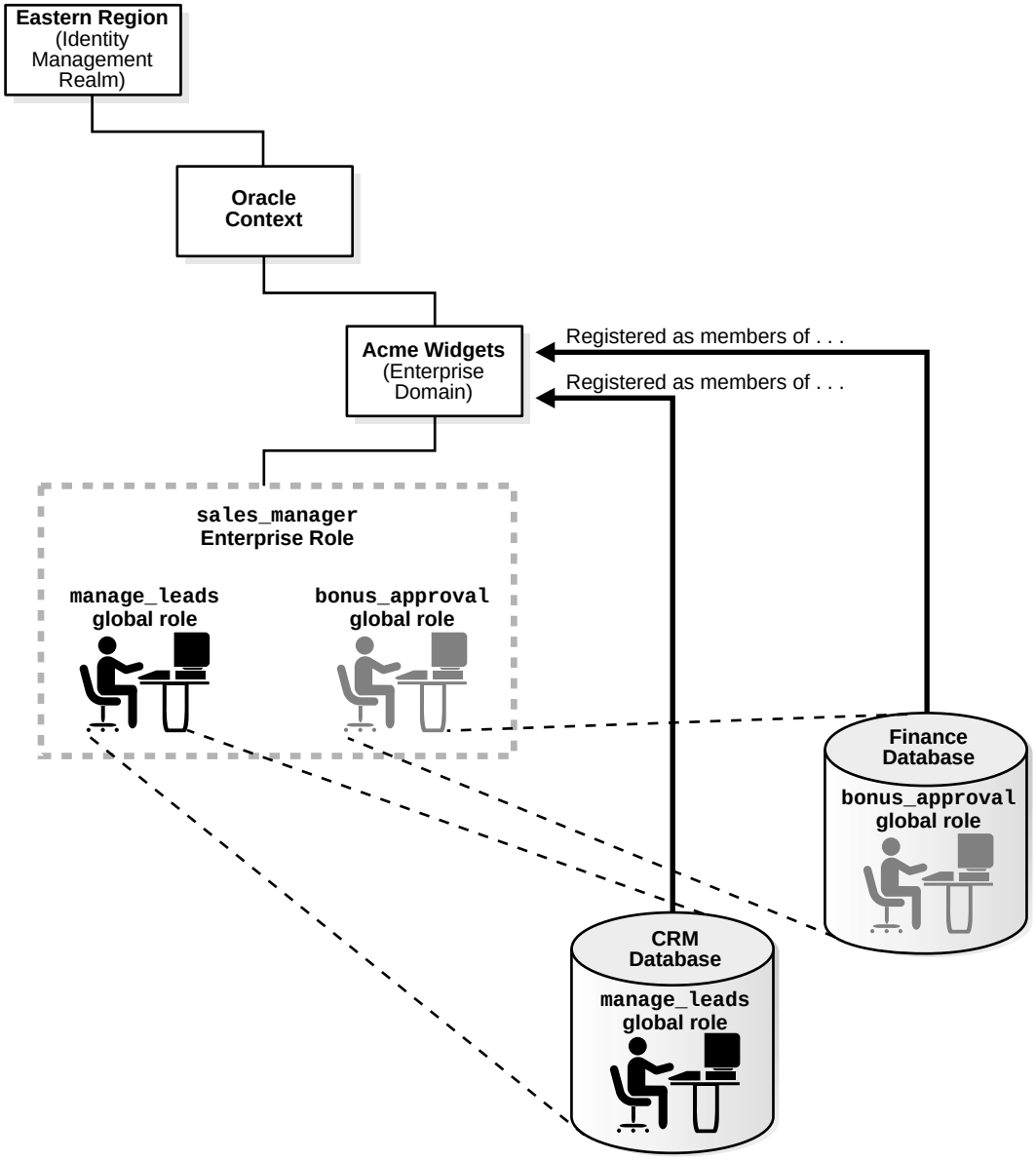
The entries described in the following sections can only reside within a **realm Oracle Context**.

Enterprise Roles

Enterprise users can be assigned an **enterprise role**, which determines their access privileges on databases. These enterprise roles are stored and managed in a directory. Figure 11–3 shows an example of an enterprise role called Manager under the OracleDefaultDomain.

An enterprise role can consist of one or many **global role**s, each one of which is defined in a specific database. A global role includes privileges contained in a database, but the global role is managed in a directory. An enterprise role is thus a container of global roles. For example, the enterprise role `sales_manager` could contain the global role `manage_leads` with its privileges on the Customer Relationship Management (CRM) database, and the `bonus_approval` global role with its privileges on the Finance database. Figure 11–2 illustrates this example.

Figure 11–2 Example of Enterprise Roles



An **enterprise role** can be assigned to one or more enterprise users. For example, you could assign the enterprise role `sales_manager` to a number of enterprise users who hold the same job. This information is protected in the directory, and only a directory administrator can manage users and assign their roles. A user can be granted local roles and privileges in a database in addition to enterprise roles.

Enterprise role entries are stored in **enterprise domain** subtrees. Each enterprise role contains information about associated global roles on each database server and the associated enterprise users. The **Enterprise Domain Administrator** creates and manages enterprise roles by using Enterprise Security Manager.

> **See Also:** "Administering Enterprise Roles" on page 13-27 for information about using Enterprise Security Manager to create and manage enterprise roles.

Note: The database obtains a user's global roles from the directory as part of the login process. If you change a user's global roles in the directory, then those changes do not take effect until the next time the user logs in to the database.

Enterprise Domains

An **enterprise domain** is a group of databases and enterprise roles. An example of a domain could be the engineering division in an enterprise or a small enterprise itself. Figure 11–3 shows an example of an enterprise domain called Services that resides under the OracleDBSecurity entry in an identity management realm. It is here, at the enterprise domain level, that the **Enterprise Domain Administrator**, using Enterprise Security Manager, assigns enterprise roles to users and manages enterprise security. An enterprise domain subtree in a directory is composed of three types of entries: enterprise role entries, user-schema mappings, and the enterprise domain administrator's group for that domain. Enterprise domains are used to manage information that applies to multiple databases. All user-schema mappings entries contained in an enterprise domain apply to all databases in the domain. If you need to apply different user-schema mappings to individual databases, then use Database Server entries, which are discussed in the following section.

Enterprise roles apply to specific databases in the domain, as explained in the previous section. Enterprise roles, domain-level mappings, and the domain administrators group are all administered by using Enterprise Security Manager.

See Also: "Administering Enterprise Domains" on page 13-15

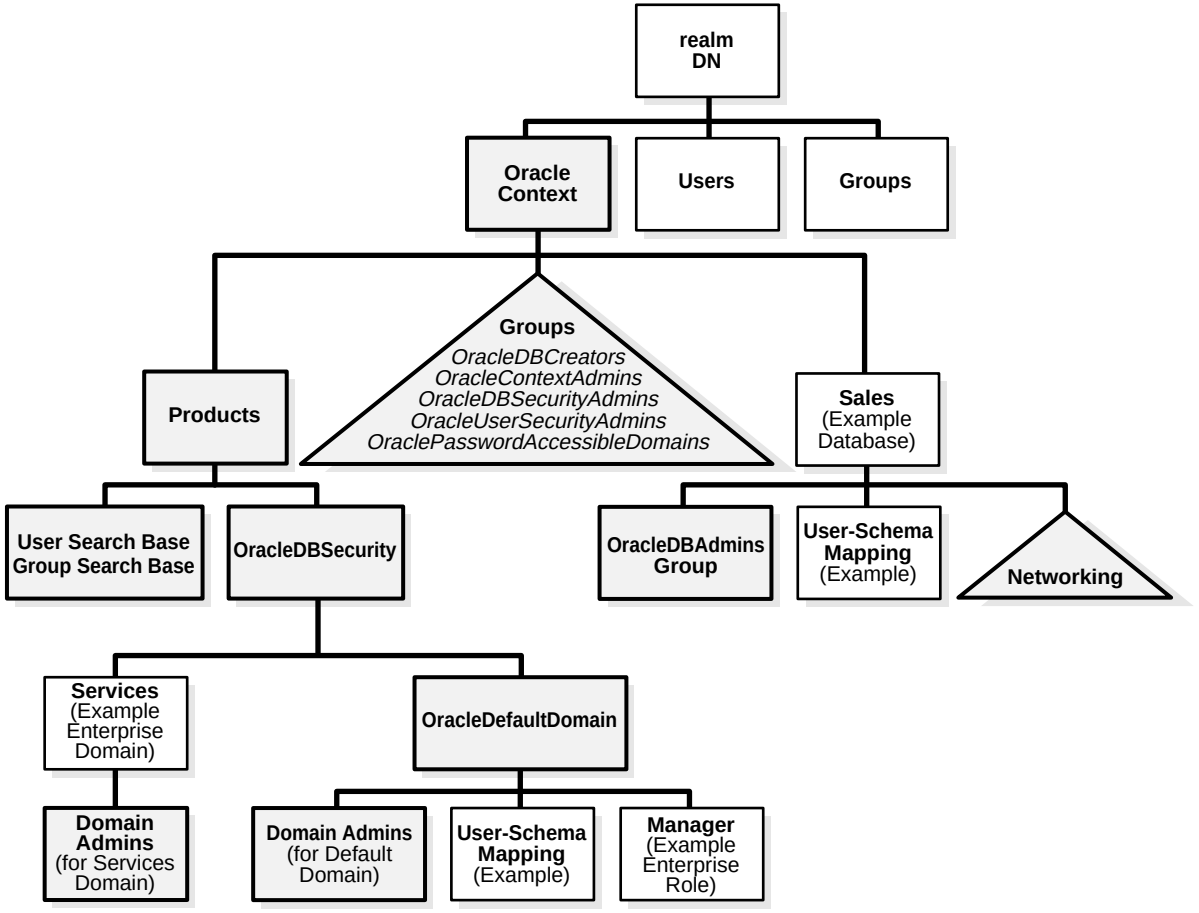
Database Server Entries

A database server entry (represented as "Sales" in Figure 11–3) contains information about one database server. It is created by the Database Configuration Assistant during database registration. A database server entry is the parent of database-level mapping entries that contain mapping information between full or partial user DNs and Oracle shared schema names (user-schema mappings). Database-level mapping entries are created by the **Database Administrator** by using Enterprise Security Manager. This tool is also used to manage the database administrator's group, which contains administrators for a specific database. The directory entry for this group is located under the database server entry in the DIT.

See Also:

- "Task 6: Register the database in the directory" on page 12-8
- "Managing Database Administrators" on page 13-25

Figure 11–3 Related Entries in a Realm Oracle Context



User-Schema Mappings

A **user-schema mapping** entry contains mapping information between a DN and an Oracle database schema. The users referenced in the mapping are connected to the specified schema when they connect to the database. User-schema mapping entries can apply only to one database or they can apply to all databases in a domain, depending on where they reside in the realm Oracle Context.

See Also:

- "How Enterprise Users Are Mapped to Schemas" on page 11-20
- "Managing Enterprise Domain Database Schema Mappings" on page 13-20

Administrative Groups

An identity management realm contains administrative groups that are related to Enterprise User Security. Figure 11–3 shows these administrative groups in a realm in the triangle labeled "Groups." Each administrative group includes an **Access Control Lists (ACLs)** that controls access to the group itself. ACLs elsewhere in the directory may refer to these groups, which allows directory administrators access to perform necessary administrative tasks. The administrative user who creates the realm automatically becomes the first member of each of these groups, thus gaining the associated privileges provided by each group, but can be removed.

The relevant administrative groups in a realm are described in Table 11–2 on page 11-18.

Note: Observe the following practices. Using other methods may break the security configuration for Enterprise User Security objects and may break enterprise user functionality as well.

- Do not modify the ACLs for the objects contained in a realm Oracle Context. Modified realm Oracle Context object ACLs are not supported.
- Use only Oracle tools, such as Enterprise Security Manager Console, Enterprise Security Manager, and Database Configuration Assistant, to modify Enterprise User Security directory entries.

Table 11–2 Administrative Groups in a Realm Oracle Context

Administrative Group	Description
OracleDBCreators (Called "Database Registration Admins" in Release 9.2 and earlier versions of Enterprise Security Manager)	**DN**: (`cn=OracleDBCreators,cn=OracleContext...`) **Default owner**: OracleContextAdmins During default realm Oracle Context creation, Oracle Internet Directory Configuration Assistant sets up the following access rights / permissions for these group members: ▪ `Add` permission for database service objects in the realm Oracle Context ▪ `Modify` permission for the Default Domain OracleDBCreators create new databases and register them in the directory by using Database Configuration Assistant
OracleContextAdmins (Called "Full Context Management" group in Release 9.2 and earlier versions of Enterprise Security Manager)	**DN**: (`cn=OracleContextAdmins,cn=Groups,cn=OracleContext...`) **Default owner**: The user who created the identity management realm. (If it is the realm created during installation, then it is `orcladmin`.) OracleContextAdmins have full access to all groups and entries within its associated realm Oracle Context.
OracleDBSecurityAdmins (Called "Database Security Management" group in Release 9.2 and earlier versions of Enterprise Security Manager)	**DN**: (`cn=OracleDBSecurityAdmins,cn=OracleContext...`) **Default owner**: All group members. During default realm Oracle Context creation, Oracle Internet Directory Configuration Assistant sets up the following access rights / permissions for these group members: ▪ All privileges in the OracleDBSecurity subtree ▪ Modify privileges for membership in this group OracleDBSecurityAdmins have permissions on all of the domains in the enterprise and perform the following tasks: ▪ Sets Enterprise User Security configurations for the realm, such as the default database-to-directory authentication method ▪ Group owner administers the OracleDBSecurityAdmins group ▪ Creates and deletes enterprise domains ▪ Moves databases from one domain to another within the enterprise
OracleUserSecurityAdmins (Called "Directory User Management" in Release 9.2 and earlier versions of Enterprise Security Manager)	**DN**: (`cn=OracleUserSecurityAdmins,cn=Groups,cn=OracleContext...`) **Default owner**: The user who created the identity management realm. By default, an ACL is set at the directory root in Oracle Internet Directory that sets up the relevant permissions so OracleSecurityAdmins can administer Oracle user security. For example, by default, they can read wallet password hints and modify user passwords.
OraclePasswordAccessible Domains	**DN**: (`cn=OraclePasswordAccessibleDomains,cn=Groups,cn=OracleContext...`) **Default owner**: Same as OracleDBSecurityAdmins Group members are enterprise domains, which contain databases enabled for password-authorized enterprise users.

About Using Shared Schemas for Enterprise User Security

The following sections describe shared schemas, and how to set them up:

- Overview of Shared Schemas Used in Enterprise User Security
- How Shared Schemas Are Configured for Enterprise Users
- How Enterprise Users Are Mapped to Schemas

Overview of Shared Schemas Used in Enterprise User Security

Users do not necessarily require individual accounts or schemas set up in each database. Alternatively, they can connect to a **shared schema** and be granted access to objects that are associated with target applications. For example, suppose that users Tom, Dick, and Harriet require access to the Payroll application on the Finance database. They do not need to create unique objects in the database, and therefore do not need their own schemas, but they do need access to the objects in the Payroll schema.

Oracle Database supports mapping multiple users stored in an enterprise directory to a shared schema on an individual database. This separation of users from schemas reduces administration costs by reducing the number of user accounts on databases. It means that you do not need to create an account for each user (user schema) in addition to creating the user in the directory. Instead, you can create a user in the enterprise directory, and map that user to a shared schema which other enterprise users can also be mapped to. For example, if Tom, Dick and Harriet all access both the Sales and the Finance databases, you do not need to create an account for each user on each of these databases. Instead, you can create a single shared schema on each database, such as GUEST, that all three users can access. Then individual access to objects in the Sales or Finance database can be granted to these three users by using enterprise roles. A typical environment can have up to 5,000 enterprise users mapped to one shared schema and each user can be assigned a set of enterprise roles.

Oracle recommends that you create a separate shared schema that contains no objects to use as an entry point. Then grant access to application objects in other schemas through enterprise roles. Otherwise, application objects can be inadvertently or maliciously deleted or altered.

In summary, shared schemas provide the following benefits:

- Shared schemas eliminate the need to have a dedicated database schema on each database for each enterprise user.

- Each enterprise user can be mapped to a shared schema on each database the user needs to access. The user connects to the shared schema when the user connects to a database.
- Shared schemas lower the cost of managing users in an enterprise.

How Shared Schemas Are Configured for Enterprise Users

To configure shared schemas, the local database administrator (DBA) must create at least one database schema in a database. Enterprise users can be mapped to this schema.

In the following example, the administrator creates a shared schema and maps users to it:

1. The administrator creates a global shared schema called EMPLOYEE and the global role HRMANAGER on the HR database.
2. The administrator uses Enterprise Security Manager to create and manage enterprise users and roles in the directory. For example, the administrator creates enterprise user Harriet and an enterprise role named MANAGER. The administrator then assigns the HR database global role of HRMANAGER to the enterprise role MANAGER.
3. The administrator assigns enterprise roles to enterprise users in the directory. For example, the administrator assigns the enterprise role MANAGER to Harriet.
4. The administrator uses Enterprise Security Manager to map the user Harriet in the directory to the shared schema *EMPLOYEE* on the HR database.

When Harriet connects to the HR database, she is automatically connected to the EMPLOYEE schema and is given the global role of HRMANAGER. Multiple enterprise users can be mapped to the same shared schema. For example, the enterprise security administrator can create another enterprise user Scott and map Scott to the EMPLOYEE schema. From that point on, both Harriet and Scott automatically use the EMPLOYEE schema when connecting to the HR database, but each can have different roles and can be individually audited.

> **See Also:** *Oracle Database Security Guide* for more information about auditing.

How Enterprise Users Are Mapped to Schemas

Global schemas (those created with CREATE USER IDENTIFIED GLOBALLY AS '') can be owned by one enterprise user (exclusive schema) or shared among

multiple enterprise users (shared schema). The mapping between a single enterprise user and his or her exclusive schema is stored in the database as an association between the user DN and the schema name. The mapping between enterprise users and a shared schema is done in the directory by means of one or more mapping objects. A mapping object is used to map the **distinguished name (DN)** of a user to a database schema that the user will access. You create a mapping object by using Enterprise Security Manager. This mapping can be one of the following:

- Entry-level (full DN) mapping

  This method associates the DN of a single directory user with a particular schema on a database. It results in one mapping entry for each user.

- Subtree-level (partial DN) mapping

  This method lets multiple enterprise users share part of their DN to access the same shared schema. This method is useful if multiple enterprise users are already grouped under some common root in the directory tree. The subtree that these users share can be mapped to a shared schema on a database. For example, you can map all enterprise users in the subtree for the engineering division to one shared schema, BUG_APP_USER, on the bug database. Note that the root of the subtree is not mapped to the specified schema.

  When an enterprise user connects to a database, the database retrieves a DN for the user, either from the network (in the case of SSL) or from the directory (in the case of password- and Kerberos-authenticated enterprise users).

  When determining which schema to connect the user to, the database uses the user DN and the following precedence rules:

  1. It looks for an exclusive schema locally (in the database).
  2. If it does not find an exclusive schema locally, then it searches the directory. Within the directory, it looks under the server entry, first for an entry-level mapping, then for a subtree-level mapping.
  3. If it does not find a mapping entry under the server entry, then it looks under the enterprise domain entry, first for an entry-level mapping, then for a subtree-level mapping.
  4. If it does not find an exclusive schema locally, or an applicable mapping entry in the database, then the database refuses the connection. Otherwise, the database connects the user to the appropriate schema.

For example, suppose that Harriet is trying to connect to the HR database, but the database does not find Harriet's exclusive schema (in the database). In this case, the following steps occur:

1. The HR database looks up a user schema mapping with Harriet's DN in the directory. The directory has a mapping of Harriet to the shared schema EMPLOYEE and returns this schema.
2. The database logs Harriet in and connects her to the EMPLOYEE schema.
3. The database retrieves this user's global roles for this database from the directory.
4. The database also retrieves from its own tables any local roles and privileges associated with the database schema to which the user is mapped.
5. The database uses both the global and the local roles to determine the information that the user can access.

Continuing this example, assume that the enterprise role MANAGER contains the global roles ANALYST on the HR database, and USER on the Payroll database. When Harriet, who has the enterprise role MANAGER, connects to the HR database, she uses the schema *EMPLOYEE* on that database.

- Her privileges on the HR database are determined by:
  - The global role ANALYST
  - Any local roles and privileges associated with the *EMPLOYEE* schema on the HR database
- When Harriet connects to the Payroll database, her privileges are determined by:
  - The global role USER
  - Any local roles and privileges associated with the EMPLOYEE schema on the Payroll database

You can grant privileges to a specified group of users by granting roles and privileges to a database schema. Every user sharing such a schema gets these local roles and privileges in addition to personal enterprise roles. However, you should exercise caution when doing this, because every user who is mapped to this shared schema can exercise the privileges assigned to it. Accordingly, Oracle does not recommend granting roles and privileges to a shared schema.

See Also: "Task 1: Create Global Schemas and Global Roles in the Database" on page 12-12 for detailed information about how to create shared schemas for enterprise users.

About Using Current User Database Links for Enterprise User Security

Oracle Database supports current user database links over an SSL-authenticated network connection. Current user database links let you connect to a second database as yourself, or as another user when used from within a stored procedure owned by that user. Such access is limited to the scope of the procedure. The security advantage of current user database links is that the other user's credentials are not stored in the database link definition, and are not sent across the network connection between databases. Instead, security of these links is based on mutual trust, mutual authentication, and a secure network connection between the databases themselves.

For example, a current user database link lets Harriet, a user of the Finance database, procedurally access the Accounts Payable database by connecting as the enterprise user Scott.

For Harriet to access a current user database link to connect to the schema Scott, Scott must be a global schema (created as `IDENTIFIED GLOBALLY`) in both databases. Harriet, however, can be a user identified in one of three ways:

- By a password
- `GLOBALLY`
- `EXTERNALLY`

To create Scott as a global user in the first database, Finance, you must enter

```
CREATE USER Scott IDENTIFIED GLOBALLY as 'CN=Scott,O=nmt'
```

so that Scott has an exclusive schema. Then Scott can map to a shared schema in the second database, Accounts Payable. In order for the current user database link to work, the schema created for Scott in the first database cannot be shared with other users.

Current user database links operate only between trusted databases within a single enterprise domain—databases within the domain trust each other to authenticate users. You specify an enterprise domain as trusted by using Enterprise Security Manager. When you use Enterprise Security Manager to enable current user database links for a domain, they will work for all databases within that domain. However, each database in the domain must have its own PKI credentials and use

SSL to authenticate to the other databases. To specify a database as untrusted that is part of a trusted enterprise domain, use the PL/SQL package `DBMS_DISTRIBUTED_TRUST_ADMIN`. To obtain a list of trusted servers, use the `TRUSTED_SERVERS` view.

Note: Oracle Advanced Security does not support RADIUS authentication over database links.

See Also:

- "What is Meant by Trusted Databases" on page 11-26
- *Oracle Database Heterogeneous Connectivity Administrator's Guide*, for additional information about current user database links
- *Oracle Database SQL Reference*, for more information about SQL syntax
- *PL/SQL Packages and Types Reference*, for information about the PL/SQL package `DBMS_DISTRIBUTED_TRUST_ADMIN`
- *Oracle Database Reference*, for information about the `TRUSTED_SERVERS` view
- Chapter 7, "Configuring Secure Sockets Layer Authentication"
- Chapter 8, "Using Oracle Wallet Manager" for information about creating wallets

Enterprise User Security Deployment Considerations

Consider the following issues before deploying Enterprise User Security:

- Security Aspects of Centralizing Security Credentials
- Security of Password-Authenticated Enterprise User Database Login Information
- Considerations for Defining Database Membership in Enterprise Domains
- Considerations for Choosing Authentication Types between Clients, Databases, and Directories for Enterprise User Security

Security Aspects of Centralizing Security Credentials

Beyond the general benefits that flow from the centralization of enterprise users and their associated credentials, there are a number of security-related benefits and risks that should be reviewed.

Security Benefits Associated with Centralized Security Credential Management

Centralizing management makes it easier and faster to administer users, credentials, and roles, and to quickly revoke a user's privileges on all applications and databases across the enterprise. With centralized management, the administrator can delete a user in one place to revoke all global privileges, minimizing the risk of retaining unintended privileges.

Centralizing management makes it possible to centralize an organization's security expertise. Specialized, security-aware administrators can manage all aspects of enterprise user security, including directory security, user roles and privileges, and database access. This is a substantial improvement over the traditional model, where DBAs are typically responsible for everything on the databases they manage, including security.

Security Risks Associated with Centralized Security Credential Management

While Oracle Internet Directory is a secure repository, there is a security challenge and inherent risk in centralizing credentials in any publicly accessible repository. Although centralized credentials can be protected at least as securely as distributed credentials, the very nature of centralization increases the consequences of inadvertent credential exposure to unauthorized parties. It is therefore imperative to limit the privileges of administrators, to set restrictive Access Control Lists (ACLs) in the directory, and to implement good security practices in the protection of security credentials when they are temporarily outside of the directory.

Security of Password-Authenticated Enterprise User Database Login Information

In all secure password-based authentication methods, a server authenticates a client with a password verifier, typically a hashed version of the password that must be rigorously protected. Password-based authentication to an Oracle database is no different. There is a password verifier, and it must be protected as well. This is true if the verifier is stored locally in the database or centrally in the directory. Note that a password verifier cannot be used to derive its original password.

An enterprise user's database password can be stored in a central directory service for access by multiple databases. It can be viewed and shared by all trusted databases to which the user has access. Although the password verifier stored in the directory is not the **cleartext** password, it is still necessary to protect it from casual or unauthorized access. It is therefore extremely important to define password-related ACLs in the directory that are as restrictive as possible, while still enabling necessary access and usability. (Note that Oracle Database supports all verifier types that are supported by Oracle Internet Directory.)

Oracle tools help set up ACLs in the directory to protect these password verifiers during identity management realm creation. The approach that Oracle recommends is intended to balance security and usability considerations. If you require maximum security and can set up wallets for all users, you should require only SSL connections from users to databases. This SSL-only approach circumvents the entire directory password protection issue.

The following sections provide more information about trusted databases and protecting database password verifiers in the directory.

What is Meant by Trusted Databases

SSL provides strong authentication so databases are ensured of each others identity. With password-authenticated Enterprise User Security where database password verifiers are stored centrally in a directory and shared among multiple databases, each database that allows password-authenticated enterprise users to log in must be a trusted database. Each database has access to the shared password verifiers so it is important that each database can be trusted to observe the following security precautions:

- Each database must be trusted to protect itself from tampering with the server code so a malicious user cannot misuse the database identity to gain access to password verifiers in the directory.
- Each database must be trusted to protect its PKI and other credentials from theft so a malicious user cannot use them to gain access to the password verifiers stored in the directory.

Protecting Database Password Verifiers

The OraclePasswordAccessibleDomains group in each identity management realm is created automatically when the realm is created, and can be managed by using Enterprise Security Manager. Enterprise domains with member databases that must view users' database password verifiers in the directory are placed into this group.

For a selected realm, determine which databases can accept password-authenticated connections. Use Enterprise Security Manager to place the domains containing those databases into the OraclePasswordAccessibleDomains group. An ACL on the user subtree permits access to the directory attribute that holds the password verifier used by the database.

All other users are denied access to this attribute. An ACL that prevents anonymous read access to the password verifier attributes is at the root of the directory tree.

Note that for usability, by default the OracleDefaultDomain is a member of the OraclePasswordAccessibleDomains group. It can be removed, if desired.

See Also:

- "Managing Password Accessible Domains" on page 13-23
- *Oracle Internet Directory Administrator's Guide* if you are not storing your users in the subtree of an identity management realm. This manual describes how to configure ACLs so password-authenticated users can connect to databases.

Considerations for Defining Database Membership in Enterprise Domains

Consider the following criteria when defining the database membership of a domain:

- Current user **database link**s operate only between databases within a single **enterprise domain**. Use of these links requires mutual trust between these databases and between the DBAs who administer them.
- Accepted authentication types for enterprise users are defined at the domain level. Database membership in a domain should therefore be defined accordingly. If one or more databases are intended to only support SSL-based certificate authentication, they cannot be combined in the same domain with password-authenticated databases.
- Enterprise roles are defined at the domain level. To share an **enterprise role** across multiple databases, the databases must be members of the same domain.

Considerations for Choosing Authentication Types between Clients, Databases, and Directories for Enterprise User Security

Enterprise User Security supports the authentication types listed in Table 11–3 for connections between clients, databases, and directories.

Table 11–3 Enterprise User Security: Supported Authentication Types for Connections between Clients, Databases, and Directories

Connection	Supported Authentication Types
Clients-to-Databases	Passwords, SSL, and Kerberos
Databases-to-Databases (Current User Database Links)	SSL only
Databases-to-Directories	SSL and Passwords

However, some combinations of authentication types for connections make more sense than others. For example, it is unusual to require a high level of security for client-to-database connections by using SSL for all user connections, but then configuring the database to authenticate to the directory by using passwords. Although this configuration is supported, it does not provide consistent security for connections. Ideally, the database-directory connection should be at least as secure as that between users and databases.

Typical Configurations

The following combinations of authentication types between clients, databases, and directories are typical:

- Password authentication for all connections with no need for current user database links
- SSL authentication for all connections
- Kerberos authentication for client-to-database connections, and password authentication for database-to-directory connections

12

Enterprise User Security Configuration Tasks and Troubleshooting

This chapter describes the sequence of steps involved to configure Enterprise User Security from the initial database and directory preparation through connecting to the database as either a password-, Kerberos-, or SSL-authenticated enterprise user. In addition, a troubleshooting section is also included that will help you when testing your Enterprise User Security implementation.

This chapter contains the following topics:

Enterprise User Security Configuration Overview

Configuring Enterprise User Security essentially consists of creating shared schemas and global roles in databases that you want to be accessible to enterprise users. Then you configure the identity management realm in the directory to reflect those database roles and schemas, and, finally, associate directory users with them.

Regardless of the authentication method you choose—password, SSL, or Kerberos—you must still create the global database objects and configure the identity management realm as described.

The primary difference between configuration for the various authentication types lies with network connection configuration. You must consider the following three connections:

- Client-to-database
- Database-to-directory
- Database-to-database (current user database links can be secured by SSL only)

Enterprise User Security supports many combinations of authentication types between databases, directories, and clients. The three most common implementations of Enterprise User Security, which will be described in this chapter, use the following authentication methods for client / database and database / directory connections:

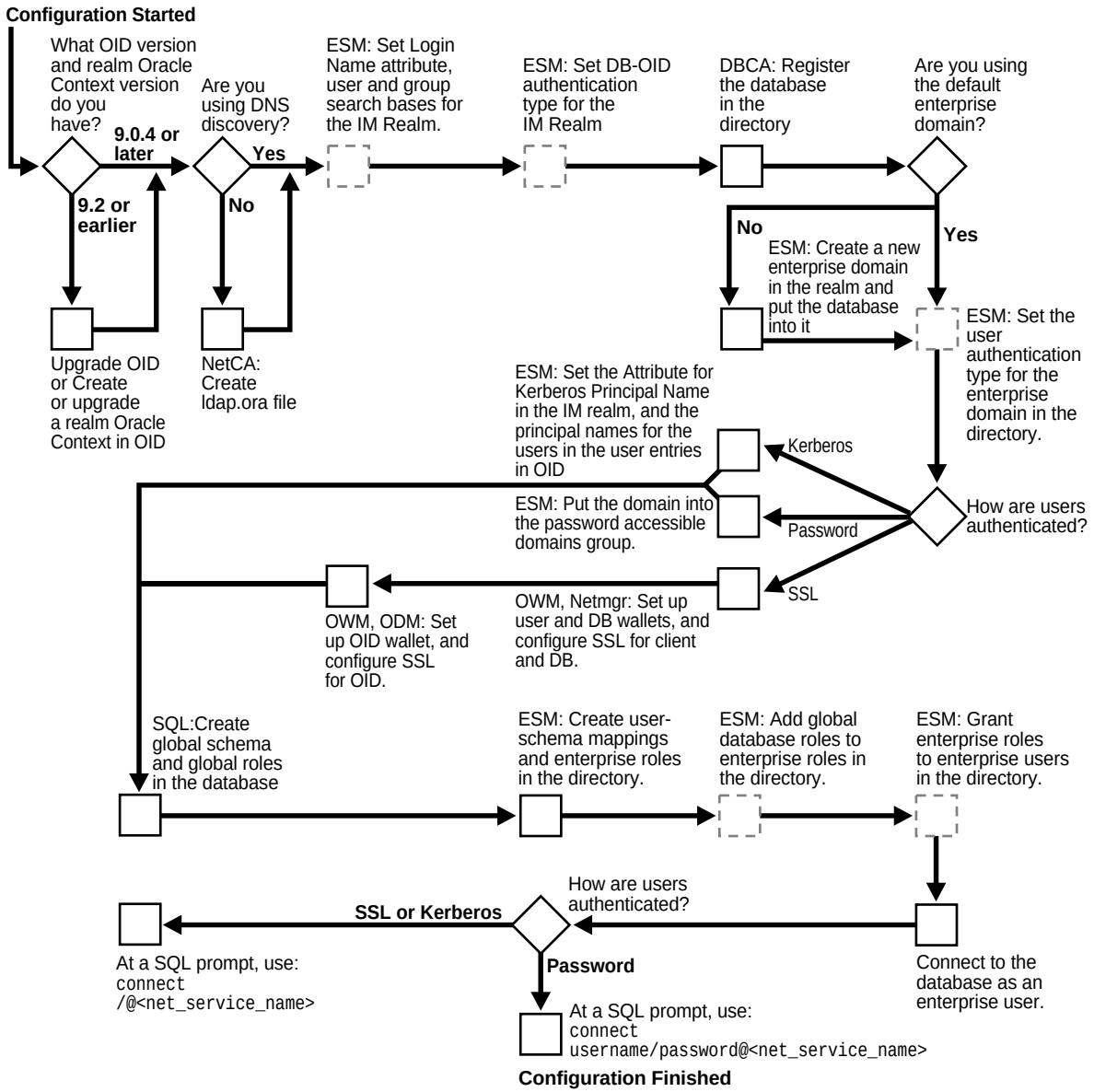
- Passwords for both connections
- SSL for both connections
- Kerberos for client / database connections and passwords for database / directory connections

Primarily, your network environment—whether all clients, databases, and directories reside within the same network behind a firewall, or are distributed across several networks and perhaps exposed to the Internet—determines what authentication type you choose for Enterprise User Security network connections. Security and integrity of enterprise data depend on secure network connections.

Secondarily, the configuration complexity, additional software, and ongoing maintenance required by more rigorous authentication types, such as SSL and Kerberos, should also be considered when choosing which "flavor" of Enterprise User Security to use.

Figure 12–1 shows the configuration process for Enterprise User Security. It is a step-by-step process with decision points based on your implementation and how your users are authenticated. Note that the steps which are represented with broken lines are optional steps in the configuration process.

Figure 12–1 Enterprise User Security Configuration Flow Chart



For brevity, some product names and features have been abbreviated in this flow chart. The following table lists the abbreviations used and their corresponding meaning:

Abbreviation	Meaning
DBCA	Database Configuration Assistant
ESM	Enterprise Security Manager
IM Realm	Identity Management Realm
Netmgr	Oracle Net Manager
ODM	Oracle Directory Manager
OID	Oracle Internet Directory
OWM	Oracle Wallet Manager
SQL	SQL*Plus

See Also: Chapter 11, "Getting Started with Enterprise User Security" for information about the realm Oracle Context, its administrative groups, and entries that pertain to Enterprise User Security.

Enterprise User Security Configuration Roadmap

The rest of this section provides detailed descriptions of these configuration steps, which should be performed in the following order:

1. "Preparing the Directory for Enterprise User Security" on page 12-5
2. "Configuring Enterprise User Security Objects in the Database and the Directory" on page 12-11
3. Complete your Enterprise User Security configuration by performing the steps necessary for your authentication method:
   - "Configuring Enterprise User Security for Password Authentication" on page 12-16
   - "Configuring Enterprise User Security for Kerberos Authentication" on page 12-18

– "Configuring Enterprise User Security for SSL Authentication" on page 12-21

Preparing the Directory for Enterprise User Security

This is the first phase in configuring Enterprise User Security and must be performed before you can configure any other part of this feature.

Enterprise User Security, 10g Release 1 (10.1) requires Oracle Internet Directory, Release 9.0.4, or later, which installs with the required version of the Oracle schema. This schema is backward compatible. After you have installed Oracle Internet Directory, perform the following directory usage configuration tasks:

- Task 1: (Optional) Create an identity management realm in the directory
- Task 2: (Optional) Set identity management realm properties
- Task 3: Identity administrative users in the directory
- Task 4: (Optional) Set the default database-to-directory authentication type for the identity management realm
- Task 5: (Optional) Configure your Oracle home for directory usage
- Task 6: Register the database in the directory

Task 1: (Optional) Create an identity management realm in the directory

If necessary, use Oracle Internet Directory Self-Service Console (Delegated Administration Service) to create an identity management realm in the directory. You can also use this tool to upgrade an Oracle9i Oracle Context to a 9.0.4 version Identity Management Realm.

You must have a version 9.0.4 identity management realm to use an Oracle Database 10g. Version 9.0.4 realms are backward compatible to Oracle9i, so you can register Oracle9i and version 10g Oracle Databases in the same realm and place them in the same domain, if desired.

Task 2: (Optional) Set identity management realm properties

If you do not want to use the default settings, then use Enterprise Security Manager Console to set the user search base, group search base, attribute for login name (nickname attribute), and to set up the necessary context administrators in the identity management realm you plan to use in the directory. To perform this task, see "Setting Properties of an Identity Management Realm" on page 13-5.

Note: By default in a version 9.0.4 identity management realm, the user search base is set to `cn=Users,cn=realm_name`, the group search base is set to `cn=Groups,cn=realm_name`, and the attribute for login name is set to the user's id (`uid`). In previous releases, this used to be `cn`.

Task 3: Identity administrative users in the directory

Identify administrative users in the directory who are authorized to perform the following tasks:

- Register databases
- Administer database security
- Create and manage enterprise domains

If administrative users do not already exist who can perform these tasks, then see Chapter 13, "Administering Enterprise User Security" to create them.

Note: Although one administrator can perform all Enterprise User Security administrative tasks, you can create many different kinds of administrators so security tasks can be assigned to different people. Separating security tasks in this way results in a more secure enterprise environment, but requires coordination between the different administrators.

Task 4: (Optional) Set the default database-to-directory authentication type for the identity management realm

By default, the identity management realm database-to-directory authentication type is set to passwords. If you do not want to use this default setting, then use Enterprise Security Manager to change it. For example, if you are using a public key infrastructure (PKI), then you would need to set this to SSL. See "Setting the Default Database-to-Directory Authentication Type for an Identity Management Realm" on page 13-6.

Note:

- This default realm-wide setting can be overridden on a database by setting the `LDAP_DIRECTORY_ACCESS` initialization parameter. See *Oracle Database Reference* for more information about this parameter.
- If you are using SSL, then see *Oracle Internet Directory Administrator's Guide* for information about setting up SSL with two-way authentication for Oracle Internet Directory.

Task 5: (Optional) Configure your Oracle home for directory usage

If you use Domain Name System (DNS) discovery (automatic domain name lookup) to locate the directory on your network, then this step is not necessary. (See *Oracle Internet Directory Administrator's Guide* for information about DNS server discovery.) If you are not using DNS discovery, then you must use Oracle Net Configuration Assistant to create an `ldap.ora` file for your Oracle home. This configuration file specifies the directory host and port information, and the location of the identity management realm so the database can connect to the directory. (See "Starting Oracle Net Configuration Assistant" on page 2-32)

To create an ldap.ora file for your Oracle home:

1. In the Oracle Net Configuration Assistant welcome page, choose **Directory Service Usage Configuration**, and click **Next**.
2. Select one of the options on the Directory Usage Configuration page that is appropriate for your environment. Then follow the prompts in the wizard and refer to the online help to create an `ldap.ora` file for your Oracle home.

Note:

- If you are using SSL authentication for your database-to-directory connection, then the SSL port entered in the `ldap.ora` file must support two-way authentication. This requires a PKI digital certificate and wallet for Oracle Internet Directory.
- If you are using password authentication for your database-to-directory connection, then the SSL port entered in the `ldap.ora` file must support SSL with no authentication. (The directory SSL port on which the Diffie-Hellman-based SSL server is running.) This does not require a wallet or certificate for Oracle Internet Directory.

Task 6: Register the database in the directory

After you have configured your Oracle home for directory usage, use Database Configuration Assistant to register the database in the directory. Registration creates an entry in the directory so the database can bind, or log in, to it.

When a database is registered in the directory, Database Configuration Assistant performs the following configuration tasks:

- Creates a new database service entry and subtree, and assigns a DN to it in the Oracle Context for the identity management realm you are using.
- Adds the database to the default enterprise domain.
- Sets the `LDAP_DIRECTORY_ACCESS` parameter in the database initialization parameter file to the default authentication type for the specified identity management realm. This parameter determines whether and how the database attempts authentication to the directory. The allowable settings are `NONE`, `PASSWORD`, or `SSL`. The default setting is `PASSWORD`.
- Creates a database wallet, containing the database DN in the form `cn=<short_database_name>,cn=OracleContext,<realm_DN>` where `short_database_name` is the first part of the fully qualified domain name for a database. For example, if you have a database named `db1.us.oracle.com`, then the short database name is `db1`.
- Randomly generates a database password for directory access, storing it in the database wallet and in the directory.

- After creating the wallet, Database Configuration Assistant stores it at *ORACLE_HOME*/admin/*Oracle_SID*/wallet in UNIX environments and at *ORACLE_BASE**ORACLE_HOME*\admin*Oracle_SID*\wallet in Windows environments. If a database wallet already exists, then Database Configuration Assistant uses it and updates the wallet password.
- Enables auto login for the database wallet.
- Restarts the database, which is required for the new initialization parameter to take effect, if you choose this option in Database Configuration Assistant.

You must be a member of the OracleDBCreators group, the OracleContextAdmins group, or you must be the directory superuser to perform this task.

To register a database in the directory:

1. See "Starting Database Configuration Assistant" on page 2-14 to start this tool.
2. After starting Database Configuration Assistant, select **Configure database options in a database** and choose **Next**.
3. Select a database and choose **Next**.
4. Choose **Yes, Register the Database**. Enter the directory credentials for a user in the OracleDBCreators group.
5. Enter a password for the database wallet.

> **Note:** Remember the database wallet password you entered in Step 5. It cannot be retrieved after you finish database registration, but can be changed later by using Oracle Wallet Manager. See "About the Database Wallet and Password" on page 12-10 for further information about this database wallet.

6. Choose **Finish** if you are only registering a database. Choose **Next** if you want to configure additional database features.

To change the database's directory password:

1. After starting Database Configuration Assistant, select **Configure database options in a database**, and choose **Next**.
2. Select a database and choose **Next**.
3. Choose **Regenerate database password**.

4. Choose **Finish** if you are only registering a database. Choose **Next** if you want to configure additional database features.

To cancel database registration:

> **Note:** Depending on user permissions, Database Configuration Assistant may be unable to remove a database from its domain in the directory. If it cannot, then use Enterprise Security Manager to remove it from the enterprise domain.

If you must unregister a database from the directory, then use Database Configuration Assistant and follow the same steps used for registering it, except choose the unregister option. When you unregister a database from the directory, Database Configuration Assistant performs the following configuration tasks:

- Removes the database entry and subtree from the directory.
- Sets the `LDAP_DIRECTORY_ACCESS` parameter to `NONE`.
- Removes the database from its enterprise domain (if the user has sufficient permissions).
- Does not remove the database wallet. See "Managing Wallets" on page 8-9, to use Oracle Wallet Manager to delete the wallet.

About the Database Wallet and Password The database requires the wallet even if no SSL (Secure Sockets Layer) is used to secure the connection between the database and the directory. If SSL is used, then this wallet should be used to store the database's digital PKI certificate.

The wallet password you enter when using Database Configuration Assistant to register a database in the directory is the password to the wallet itself, and is not the database's directory login credentials.

You can change this wallet password later by using Oracle Wallet Manager. However, if you forget this wallet password, you must delete the database wallet that was created, unregister the database from the directory, and reregister the database in the directory so another database wallet can be generated.

> **See Also:** Chapter 8, "Using Oracle Wallet Manager" for information about using Oracle Wallet Manager to change wallet passwords and, in general, to manage public key infrastructure (PKI) credentials.

After you have prepared the directory for Enterprise User Security, then you can create the Enterprise User Security database and directory objects as described in "Configuring Enterprise User Security Objects in the Database and the Directory" on page 12-11.

See Also:

- *Oracle Internet Directory Administrator's Guide* for information about configuring an identity management realm in the directory.
- *Oracle Database Reference* for information about changing the value of the `LDAP_DIRECTORY_ACCESS` initialization parameter.

Configuring Enterprise User Security Objects in the Database and the Directory

This is the second phase of configuration steps required to implement Enterprise User Security. The configuration steps in this section assume the following recommended setup:

- You have prepared your database and your directory by completing the tasks described in "Preparing the Directory for Enterprise User Security" on page 12-5.
- Your users are stored in an identity management realm Users subtree.
- You use the OracleDefaultDomain, which is the default **enterprise domain** that Database Configuration Assistant uses when you register databases in the directory.

Note that databases must be in an enterprise domain that is in an identity management realm in order for enterprise user logins to work.

If you do not use the OracleDefaultDomain or store your users in an identity management realm Users subtree, then see the following documentation:

- *Oracle Internet Directory Administrator's Guide* for information about creating a new identity management realm or modifying an existing one, and for information about setting access control lists on directory objects.
- "Creating a New Enterprise Domain" on page 13-16 to create another domain in which to put your database. Then substitute your new domain name for OracleDefaultDomain in the following configuration steps.

To configure Enterprise User Security objects in the database and directory perform the following tasks:

- Task 1: Create Global Schemas and Global Roles in the Database
- Task 2: Configure User-Schema Mappings for the Enterprise Domain
- Task 3: Create Enterprise Roles in the Enterprise Domain
- Task 4: Add Global Database Roles to Enterprise Roles
- Task 5: Grant Enterprise Roles to Enterprise Users for Database Access
- Task 6: Configure Enterprise User Security for the Authentication Method You Require

Task 1: Create Global Schemas and Global Roles in the Database

Although this step can also be completed by using Oracle Enterprise Manager, the following examples use SQL*Plus directly:

1. Create a shared schema for enterprise users. The following syntax example creates a shared schema named guest:

   ```
   SQL> CREATE USER guest IDENTIFIED GLOBALLY AS '';
   ```

   If you do not want to use a shared schema, then specify a user DN between the single quotation marks to create an exclusive schema.

2. Grant the `CREATE SESSION` privilege to the shared schema created in Step 1 so users can connect to it. The following syntax example grants the `CREATE SESSION` privilege to the guest shared schema:

   ```
   SQL> GRANT CREATE SESSION TO guest;
   ```

Alternatively, you can grant the `CREATE SESSION` privilege to a global role, which you grant to specific users through an **enterprise role**. See Step 3.

3. Create global roles for the database to hold relevant privileges. The following syntax examples create the `emprole` and `custrole` global roles:

```
SQL> CREATE ROLE emprole IDENTIFIED GLOBALLY;
SQL> CREATE ROLE custrole IDENTIFIED GLOBALLY;
```

Global roles are associated with enterprise roles, which will be created later, and then are allocated to enterprise users.

4. Grant privileges to the new global roles that were created in Step 3. The following syntax example grants the `SELECT` privilege to `emprole` and `custrole` global roles on the products table:

```
SQL> GRANT select ON products TO custrole, emprole;
```

See Also: *Oracle Database SQL Reference* for information about the syntax used for these steps.

Task 2: Configure User-Schema Mappings for the Enterprise Domain

Use Enterprise Security Manager (see "Starting Enterprise Security Manager" on page 2-16) to configure user-schema mappings for the OracleDefaultDomain by using the following steps:

1. Select the **OracleDefaultDomain** in the navigator pane.
2. Choose the Database Schema Mapping tabbed window and click **Add...**.
3. In the Add Database Schema Mappings dialog box enter the appropriate DN and the shared schema name that you created in Task 1 on page 12-12. Refer to the Enterprise Security Manager online help for information about how to enter these values.
4. Choose **OK**. The new user-schema mappings apply to all databases in the enterprise domain.

For more information about this task, see "Managing Enterprise Domain Database Schema Mappings" on page 13-20.

Note: You also can create user-schema mappings under a database in an enterprise domain which only apply to that database.

Task 3: Create Enterprise Roles in the Enterprise Domain

Use Enterprise Security Manager to create enterprise roles in the OracleDefaultDomain by using the following steps:

1. Right-click the OracleDefaultDomain in the navigator pane and choose **Create Enterprise Role...**.

   The Create Enterprise Role dialog box appears with the appropriate realm Oracle Context and enterprise domain displayed.

2. Enter the enterprise role name in the **Role Name** field.

3. Click **OK**. The new enterprise role is added under the domain in the navigator pane.

For more information about this task, see "Creating a New Enterprise Role" on page 13-27.

Task 4: Add Global Database Roles to Enterprise Roles

Use Enterprise Security Manager to add the global database roles that you created in Task 1 on page 12-12 to the enterprise roles that you created in Task 3 by using the following steps:

1. Select the enterprise role name in the navigator pane.

2. Choose the Database Global Roles tabbed window and click **Add...**.

3. In the Add Global Database Roles dialog box, select the database from which to obtain global roles. A database logon window appears, prompting you for a username and password to authenticate to the database so global roles can be fetched. Typically, this is a DBA logon to the database.

Note: You can use the database name that appears by default in the Service field to connect to the database if your Oracle home has **LDAP** as one of its selected Oracle Net naming methods, or if this name appears as a TNS alias in your local Oracle Net configuration. Otherwise, you can overwrite the Service field with any other TNS alias (from the database `tnsnames.ora` file), or by using a connect string in the following format:

<host>`:`*<port>*`:`*<oracle_SID>*

For example: `machine111:1521:sales_db`

---

4. Click **OK**. Enterprise Security Manager connects to the selected database, fetches the global roles supported on that database, and displays them in the Add Global Database Roles dialog box.
5. Select one or more global roles and click **OK**. The selected global roles appear in the Database Global Roles window.
6. Click **Apply**. The new global roles are added to the enterprise role.

For more information about this task, see "Assigning Database Global Role Membership to an Enterprise Role" on page 13-28.

Task 5: Grant Enterprise Roles to Enterprise Users for Database Access

Use Enterprise Security Manager to grant enterprise roles that you created in Task 3 on page 12-14 to the enterprise users by using the following steps:

1. In the navigator pane, select an enterprise role in the appropriate identity management realm.
2. Select the Users tab adjacent to the main application window and click **Add...**.
3. In the Add Enterprise Users dialog box top panel, select a directory entry as a user search base, or edit the Selection field to manually define the user search base.
4. In the middle Search Criteria panel, check **Include Subtrees** to enable searching for all users within the search, including subtrees.
5. Enter any known user name in the **Show Names Containing** field. This limits the search to users in the directory who have a common name value that contains or starts with the specified text.
6. Click **Search Now**. If there are any users in the directory that match your search criteria, then they are listed in the bottom panel.
7. Choose a desired user by selecting the user in the bottom panel and clicking **OK**, or by double-clicking the user. Multiple users can be granted the enterprise role by selecting a range of users and clicking **OK**.

   The Add Enterprise Users dialog box automatically closes and you are returned to the main application window.
8. The user names you added appear in the Users tab. Click **Apply** to grant the enterprise role to the users.

For more information about this task, see "Granting Enterprise Roles to Users" on page 13-31.

Task 6: Configure Enterprise User Security for the Authentication Method You Require

Based on the authentication method you have chosen, go to one of the following sections to complete your Enterprise User Security configuration:

- "Configuring Enterprise User Security for Password Authentication" on page 12-16
- "Configuring Enterprise User Security for Kerberos Authentication" on page 12-18
- "Configuring Enterprise User Security for SSL Authentication" on page 12-21

> **See Also:** Table 11–1, "Enterprise User Security Authentication: Selection Criteria" on page 11-10 for a comparison of the benefits provided by password, Kerberos, and SSL authentication for Enterprise User Security.

Configuring Enterprise User Security for Password Authentication

By default, new enterprise domains are configured to accept all supported user authentication types (password, Kerberos, and SSL). If you want enterprise users to be authenticated by passwords, then you must configure that as described in the following tasks.

The configuration steps in this section assume the following:

- You have prepared your directory by completing the tasks described in "Preparing the Directory for Enterprise User Security" on page 12-5.
- You have configured your Enterprise User Security objects in the database and the directory by completing the tasks described in "Configuring Enterprise User Security Objects in the Database and the Directory" on page 12-11.
- You have configured an SSL instance with no authentication for Oracle Internet Directory as described in *Oracle Internet Directory Administrator's Guide*. If you are using an `ldap.ora`, also ensure that the port number for this SSL with no authentication instance is listed there as your directory SSL port.

To configure Enterprise User Security for password authentication, perform the following tasks:

Task 1: (Optional) Enable the Enterprise Domain to Accept Password Authentication

By default, the OracleDefaultDomain is configured to accept password authentication. If this has been changed, then use Enterprise Security Manager to enable password authentication for the OracleDefaultDomain and add it to the Password-Accessible Domains List by using the following steps:

1. Select the OracleDefaultDomain in the navigator pane.
2. Choose the Databases tabbed window and select **Password** or **All Types** from the **User Authentication** methods listed.
3. Click **Apply**.
4. Select the identity management realm in the navigator pane.
5. Choose the Accessible Domains tabbed window and click **Add**.
6. In the Add Accessible Enterprise Domains dialog box, select the **OracleDefaultDomain** from the list of enterprise domains, and click **OK**. The OracleDefaultDomain is added to the password-accessible domains list.

For more information about this task, see "Managing Password Accessible Domains" on page 13-23.

Task 2: Add the Enterprise Domain to the Password-Accessible Domains List

Use Enterprise Security Manager to add the OracleDefaultDomain to the Password-Accessible Domains List by using the following steps:

1. Select the identity management realm in the navigator pane.
2. Choose the Accessible Domains tabbed window and click **Add**.
3. In the Add Accessible Enterprise Domains dialog box, select the **OracleDefaultDomain** from the list of enterprise domains, and click **OK**. The OracleDefaultDomain is added to the password-accessible domains list.

For more information about this task, see "Managing Password Accessible Domains" on page 13-23.

Task 3: Connect as a Password-Authenticated Enterprise User

For an enterprise user whose directory login name is `hscortea` and whose password is `welcome`, enter the following to connect to the database by using SQL*Plus:

```
SQL> connect hscortea/welcome@<Oracle Net Service Name>
```

The database authenticates the enterprise user (`hscortea`) by verifying the username / password combination against the directory entry associated with this user. Then it identifies the proper schema and retrieves the user's global roles. If successful, the connection to the database is established.

If your connection succeeds, then the system responds `Connected to:...`. This is the confirmation message of a successful connect and setup. If an error message displays, then see "ORA-# Errors for Password-Authenticated Enterprise Users" on page 12-26.

If you do connect successfully, then check that the appropriate global roles were retrieved from the directory by entering the following at the SQL*Plus prompt:

```
select * from session_roles
```

If the global roles were not retrieved from the directory, then see "NO-GLOBAL-ROLES Checklist" on page 12-33.

You have completed password-authenticated Enterprise User Security configuration.

See Also:

- "Troubleshooting Enterprise User Security" on page 12-26 for information about diagnosing and resolving errors.
- Chapter 13, "Administering Enterprise User Security" for information about configuring the identity management realm, and about creating and managing enterprise domains, enterprise roles, and enterprise users.

Configuring Enterprise User Security for Kerberos Authentication

The configuration steps in this section assume the following:

- You have registered your databases with the Kerberos authentication server and configured your Oracle Net Services as described in Chapter 6, "Configuring Kerberos Authentication".

- You have prepared your directory by completing the tasks described in "Preparing the Directory for Enterprise User Security" on page 12-5.
- You have configured your Enterprise User Security objects in the database and the directory by completing the tasks described in "Configuring Enterprise User Security Objects in the Database and the Directory" on page 12-11.
- You have configured an SSL instance with no authentication for Oracle Internet Directory as described in *Oracle Internet Directory Administrator's Guide*. If you are using an `ldap.ora`, also ensure that the port number for this SSL with no authentication instance is listed there as your directory SSL port.

To configure Enterprise User Security for Kerberos authentication, perform the following tasks:

- Task 1: Configure the Enterprise Security Manager Console to display the Kerberos principal name attribute
- Task 2: (Optional) Configure the Kerberos Principal Name Directory Attribute for the Identity Management Realm
- Task 3: Specify the Enterprise User's Kerberos Principal Name in the krbPrincipalName Attribute
- Task 4: (Optional) Enable the Enterprise Domain to Accept Kerberos Authentication
- Task 5: Connect as a Kerberos-Authenticated Enterprise User

Task 1: Configure the Enterprise Security Manager Console to display the Kerberos principal name attribute

Use Oracle Internet Directory Self-Service Console to configure the Enterprise Security Manager Console to display the Kerberos principal name attribute. For more information about this task, see "Configuring Enterprise Security Manager Console for Kerberos-Authenticated Enterprise Users" on page 2-24.

Task 2: (Optional) Configure the Kerberos Principal Name Directory Attribute for the Identity Management Realm

Use Enterprise Security Manager Console to enter the directory attribute used to store the Kerberos principal name for the identity management realm you are using in the directory. By default Kerberos principal names are stored in the `krbPrincipalName` attribute, but can be changed to correspond to your directory configuration by changing `orclCommonKrbPrincipalAttribute` in the identity management realm. For more information about this task, see "Setting Login Name,

Kerberos Principal Name, User Search Base, and Group Search Base Identity Management Realm Attributes" on page 13-5.

Note: By default, Enterprise Security Manager Console user interface does not display the field where you can configure Kerberos principal names. The first time you create Kerberos-authenticated users in the directory, you must configure the console to display the `krbPrincipalName` attribute in its Create User window. See "Configuring Enterprise Security Manager Console for Kerberos-Authenticated Enterprise Users" on page 2-24 for details.

Task 3: Specify the Enterprise User's Kerberos Principal Name in the krbPrincipalName Attribute

Use Enterprise Security Manager Console to specify the enterprise user's Kerberos principal name (*`Kerberos_username`*`@`*`Kerberos_realm`*) in the `krbPrincipalName` attribute of the enterprise user's directory entry. For more information about this task, see "Creating New Enterprise Users" on page 13-9.

Task 4: (Optional) Enable the Enterprise Domain to Accept Kerberos Authentication

By default, the OracleDefaultDomain is configured to accept all types of authentication. If this has been changed, or you are using another domain then use Enterprise Security Manager to enable Kerberos authentication for your enterprise domain by using the following steps:

1. Select the enterprise domain in the navigator pane.
2. Choose the Databases tabbed window and select **Kerberos** or **All Types** from the **User Authentication** methods listed.
3. Click **Apply**.

For more information about this task, see "Managing Database Security Options for an Enterprise Domain" on page 13-19.

Task 5: Connect as a Kerberos-Authenticated Enterprise User

If the **KDC** is not part of the operating system, such as Kerberos V5 from MIT, then the user must get an initial ticket with the `FORWARDABLE` flag set by using the `okinit` utility. See "Obtaining the Initial Ticket with the okinit Utility" on page 6-11.

If the KDC is part of the operating system, such as Windows 2000 or some versions of Linux or UNIX, then the operating system automatically picks up the user's ticket (with the FORWARDABLE flag set) from the cache when the user logs in.

The user connects to the database by launching SQL*Plus and entering the following at the command line:

```
SQL> connect /@<net_service_name>
```

The database uses Kerberos to authenticate the user. The database authenticates itself to the directory by password.

If your connection succeeds, then the system responds `Connected to:...`. This is the confirmation message of a successful connect and setup. If an error message displays, then see "ORA-# Errors for Kerberos-Authenticated Enterprise Users" on page 12-29.

If you do connect successfully, then check that the appropriate global roles were retrieved from the directory by entering the following at the SQL*Plus prompt:

```
select * from session_roles
```

If the global roles were not retrieved from the directory, then see "NO-GLOBAL-ROLES Checklist" on page 12-33.

You have completed Kerberos-authenticated Enterprise User Security configuration.

See Also:

- "Troubleshooting Enterprise User Security" on page 12-26 for information about diagnosing and resolving errors.
- Chapter 13, "Administering Enterprise User Security" for information about configuring the identity management realm, and information about creating and managing enterprise domains, enterprise roles, and enterprise users.

Configuring Enterprise User Security for SSL Authentication

The configuration steps in this section assume the following:

- You have obtained the appropriate PKI credentials and used Oracle Wallet Manager to create wallets for the directories, the databases, and the clients that you want to include in your Enterprise User Security implementation.
- You have confirmed that the following DNs are identical:

  - Database certificate DN (stored in the database wallet)
  - Database directory entry DN
  - Database wallet DN (not the certificate)

  See "Viewing the Database DN in the Wallet and in the Directory" on page 12-24. Note that Database Configuration Assistant sets the database directory entry DN and the database wallet DN to be identical when registering the database in the directory.

- You have enabled SSL for your client-database Oracle Net connections as described in "Enabling SSL" on page 7-15. Ensure you included the following steps when you enabled SSL:
  - Enabled SSL for your database listener on TCPS and provided a corresponding TNS name.
  - Stored your database PKI credentials in the database wallet that Database Configuration Assistant automatically created during database registration.
- You have configured an SSL instance with two-way authentication for Oracle Internet Directory as described in *Oracle Internet Directory Administrator's Guide*.
- You have prepared your directory by completing the tasks described in "Preparing the Directory for Enterprise User Security" on page 12-5.
- You have configured your Enterprise User Security objects in the database and the directory by completing the tasks described in "Configuring Enterprise User Security Objects in the Database and the Directory" on page 12-11.

To configure Enterprise User Security for SSL authentication, perform the following tasks:

- Task 1: Enable the Enterprise Domain to Accept SSL Authentication
- Task 2: Set the LDAP_DIRECTORY_ACCESS Initialization Parameter to SSL
- Task 3: Connect as an SSL-Authenticated Enterprise User

Task 1: Enable the Enterprise Domain to Accept SSL Authentication

Use Enterprise Security Manager to enable SSL authentication for the **enterprise domain** (OracleDefaultDomain) by using the following steps:

1. Select the enterprise domain in the navigator pane.
2. Choose the Databases tabbed window and select **Oracle Wallet (SSL)** or **All Types** from the **User Authentication** methods listed.

3. Click **Apply**.

For more information about this task, see "Managing Database Security Options for an Enterprise Domain" on page 13-19.

Task 2: Set the LDAP_DIRECTORY_ACCESS Initialization Parameter to SSL

You can change this initialization parameter either by editing your database initialization parameter file, or by issuing an `ALTER SYSTEM` SQL command with the `SET` clause.

For example, the following `ALTER SYSTEM` command changes the `LDAP_DIRECTORY_ACCESS` parameter value to `SSL` in the server parameter file:

```
ALTER SYSTEM SET LDAP_DIRECTORY_ACCESS=SSL SCOPE=SPFILE
```

See Also:

- *Oracle Database Administrator's Guide* for information about editing initialization parameters.
- *Oracle Database Reference* for information about the `LDAP_DIRECTORY_ACCESS` initialization parameter.
- *Oracle Database SQL Reference* for information about using the `ALTER SYSTEM` command with the `SET` clause.

Task 3: Connect as an SSL-Authenticated Enterprise User

Connecting as an SSL-authenticated enterprise user involves ensuring that you have the appropriate Oracle wallet features configured, and that you do not have a wallet location specified in the client `sqlnet.ora` file. If the client `sqlnet.ora` file contains a wallet location, then multiple users cannot share that file. Only the server `sqlnet.ora` file must have a value for the wallet location parameter.

To connect as an SSL-authentication enterprise user, perform the following steps:

1. Use Oracle Wallet Manager to download a user wallet from the directory. See "Downloading a Wallet from an LDAP Directory" on page 8-16.
2. Use Oracle Wallet Manager to enable auto login for the user wallet. Enabling auto login generates a single sign-on (`.sso`) file and enables authentication to the SSL adapter. See "Using Auto Login" on page 8-19.
3. Set the `TNS_ADMIN` environment variable (to point to the client's `sqlnet.ora` file) for the client if the client Oracle home points to a server Oracle home. (Because a server must have a wallet location set in its `sqlnet.ora` file and a

client cannot have a wallet location specified there, the server and client cannot share `sqlnet.ora` files.)

If you have a separate client Oracle home, then you do not need to set the `TNS_ADMIN` environment variable.

4. Launch SQL*Plus and enter the following at the command line:

```
SQL> /@connect_identifier
```

where `connect_identifer` is the Oracle Net service name you set up when you configured SSL for the database client.

If your connection succeeds, then the system responds `Connected to:...`. This is the confirmation message of a successful connect and setup. If an error message displays, then see "ORA-# Errors for SSL-Authenticated Enterprise Users" on page 12-32.

If you do connect successfully, then check that the appropriate global roles were retrieved from the directory by entering the following at the SQL*Plus prompt:

```
select * from session_roles
```

If the global roles were not retrieved from the directory, then see "NO-GLOBAL-ROLES Checklist" on page 12-33.

You have completed SSL-authenticated Enterprise User Security configuration.

Note: For security purposes, ensure that you disable auto login for the user wallet after logging out from the enterprise user session with the database. This is especially important if the client machine is shared by more than one user. See "Disabling Auto Login" on page 8-19 for information about disabling this Oracle Wallet feature.

Viewing the Database DN in the Wallet and in the Directory

For SSL-authenticated Enterprise User Security to work, the database DNs in the database wallet, the database directory entry, and the database certificate must be identical. When you use Database Configuration Assistant to register your database in the directory, this tool automatically creates identical DNs for the database wallet and the database directory entry. To request a database certificate with the proper DN, you must view either the directory entry DN or the wallet DN.

To view the database DN so you can request a certificate with the appropriate DN use one of the following options:

- Use Oracle Directory Manager to look in the directory under the realm Oracle Context for `cn=<short_database_name>,cn=OracleContext,<realm_DN>` where *`short_database_name`* is the first part of the fully qualified domain name for a database. For example, if you have a database named `db1.us.oracle.com`, then the short database name is `db1`.
- Use the following `mkstore` utility syntax on the command line:

  ```
  mkstore -wrl <wallet_location> -viewEntry ORACLE.SECURITY.DN
  ```

  where *`wallet_location`* is the path to the database wallet.

  See Also:

  - "Troubleshooting Enterprise User Security" on page 12-26 for information about diagnosing and resolving errors.
  - Chapter 13, "Administering Enterprise User Security" for information about configuring the identity management realm, and information about creating and managing enterprise domains, enterprise roles, and enterprise users.

Enabling Current User Database Links

Current user database links require SSL-enabled network connections between the databases. Before you can enable current user database links, you must enable SSL, create Oracle wallets, and obtain PKI credentials for all databases involved.

Then use Enterprise Security Manager to enable current user database links between databases within the **enterprise domain** in the directory by using the following steps:

1. Select the enterprise domain in the navigator pane.
2. Choose the Databases tabbed window and check **Enable Current User Database Links**.
3. Click **Apply**.

For more information about this task, see "Managing Database Security Options for an Enterprise Domain" on page 13-19.

Troubleshooting Enterprise User Security

This section describes potential problems and associated corrective actions in the following topics:

ORA-# Errors for Password-Authenticated Enterprise Users

If you receive an ORA-# error while using password-authenticated Enterprise User Security, then locate the error in the following section and take the recommended action.

ORA-1017: Invalid username/password; login denied

Action: See "USER-SCHEMA ERROR Checklist" on page 12-34

ORA-28030: Problem accessing LDAP directory service

Cause: Indicates a problem with the connection between the database and the directory.

Action: Check the following:

1. Check that there is a correct `wallet_location` value in the database's `sqlnet.ora` file. If not, then use Oracle Net Manager to enter one.
2. If Domain Name System (DNS) server discovery of Oracle Internet Directory is not used, check that there is a correct `ldap.ora` file in `$LDAP_ADMIN`, `$ORACLE_HOME/ldap/admin`, `$TNS_ADMIN`, or `$ORACLE_HOME/network/admin`. (See *Oracle Internet Directory Administrator's Guide* for information about DNS server discovery.)
3. Check that the SSL port used (by way of either DNS discovery or an `ldap.ora` file) supports SSL with no authentication.
4. Check that the `LDAP_DIRECTORY_ACCESS` parameter is set to `PASSWORD` in the database initialization parameters file.

5. Use Database Configuration Assistant to reset the database password used to authenticate the database to Oracle Internet Directory. This resets it both locally in the database wallet, and remotely in the database entry in Oracle Internet Directory.
6. Check that the database wallet has auto login enabled. Either use Oracle Wallet Manager, or check that there is a `cwallet.sso` file in *`$ORACLE_HOME`*`/admin/<`*`ORACLE_SID`*`>/wallet/`.
7. Use the password stored in the database wallet to check that the database can bind to Oracle Internet Directory:
   - Use the `mkstore` command line utility to retrieve the database password from the wallet by using the following syntax:

     ```
     mkstore -wrl <database wallet location> -viewEntry
     ORACLE.SECURITY.PASSWORD
     ```

   - Use the password returned from `mkstore` in the following `ldapbind`:

     ```
     ldapbind -h <directory host> -p <non-SSL directory port> -D
     "<database DN>" -w <password returned by mkstore>
     ```

8. Check to ensure the database belongs to only one enterprise domain.

Note: The `mkstore` utility is for troubleshooting purposes only. The name and functionality of this tool may change in the future. In 10*g* Release 1 (10.1), Oracle supports only the `viewEntry` mode.

ORA-28271: No permission to read user entry in LDAP directory service

Action: Check the following:

1. Use Enterprise Security Manager to check that a user search base containing this user is listed in the user search base attribute of the realm that you are using.
2. Check the ACL on the User Search Base in Oracle Internet Directory to ensure that the `verifierServices` group has read permission on the user entry, and that this permission is not prevented by an ACL between the User Search Base entry and the user entry in the directory tree.
3. Check that the enterprise domain is in the password-accessible domains group for that realm Oracle Context.

ORA-28272: Domain policy does not allow password-authenticated GLOBAL users

Action: Use Enterprise Security Manager to set the user authentication policy for this enterprise domain to **Password** or **ALL**.

ORA-28273: No mapping for user login name to LDAP distinguished name exists

Action: Check the following:

1. Check that a user entry exists in Oracle Internet Directory for your user.
2. Use Enterprise Security Manager to check that a user search base containing this user is listed in the identity management realm that you are using.
3. Check that the user entry contains the right login name:
   - Use Enterprise Security Manager Console to find the login name attribute that is configured for the directory in your realm, and
   - Check that the name provided during the attempted user database login is the value for that attribute in the user directory entry.
4. If you have an exclusive schema for the global user in the database, then check that the DN in the database matches the DN of the user entry in Oracle Internet Directory.

ORA-28274: No ORACLE password attribute corresponding to user login name exists

Action: Check the following:

1. Check that the user entry in the directory has the `orcluser` object class. If it does not, then perform the following steps:
   - Use Oracle Internet Directory Self-Service Console to check that the default object classes for new user creation include `orcluser`, and then
   - Use Enterprise Security Manager Console or Oracle Internet Directory Self-Service Console to re-create the user, or
   - Add the `orcluser` and the `orcluserV2` object classes.
2. Check that there is a value for the attribute `orclpassword` in the user entry. If there is no value, then reset the user's directory password (`userpassword` attribute). This should prompt Oracle Internet Directory to regenerate the database password verifier for the user.

3. Use Enterprise Security Manager to check that the user search base containing this user is listed in the user search base attribute of the realm that you are using.
4. Use Enterprise Security Manager to check that the enterprise domain is in the password accessible domains group.
5. Check that the ACL on the user search base attribute allows read and search access to the `orclpassword` attributes by the `verifierServices` group. This is set properly by default, but may have been altered.

ORA-28275: Multiple mappings for user login name to LDAP distinguished name exist

Cause: There are multiple user DNs in the directory within the user search base whose login name for the user matches what was provided during the database connection.

Action: Use Enterprise Security Manager Console to make the login name value unique (no two users share the same login name) within all user search bases associated with the realm Oracle Context.

ORA-28277: LDAP search, while authenticating global user with passwords, failed

Action: Check that the relevant directory instance is up and running.

ORA-28278: No domain policy registered for password-based GLOBAL users

Cause: The database cannot read the enterprise domain information that it needs.

Action: See "DOMAIN-READ-ERROR Checklist" on page 12-35

ORA-28862: SSL handshake failed

Action: Check that you are using a non-SSL connect string.

ORA-# Errors for Kerberos-Authenticated Enterprise Users

If you receive an ORA-# error while using Kerberos-authenticated Enterprise User Security, then locate the error in the following section and take the recommended action.

ORA-1017: Invalid username/password; login denied

Action: See "USER-SCHEMA ERROR Checklist" on page 12-34

ORA-28030: Problem accessing LDAP directory service

Cause: Indicates a problem with the connection between the database and the directory.

Action: See the actions listed for resolving "ORA-28030: Problem accessing LDAP directory service" on page 12-26 in the troubleshooting section for password-authenticated enterprise users.

ORA-28271: No permission to read user entry in LDAP directory service

Action: See the actions listed for resolving "ORA-28271: No permission to read user entry in LDAP directory service" on page 12-27 in the troubleshooting section for password-authenticated enterprise users.

ORA-28292: No domain policy registered for Kerberos-based authentication

Action: Perform the following actions:

1. Use Enterprise Security Manager to set the user authentication policy for this enterprise domain to **KERBEROS** or **ALL**.
2. See "DOMAIN-READ-ERROR Checklist" on page 12-35

ORA-28290: Multiple entries found for the same Kerberos principal name

Cause: The Kerberos principal name for this user is not unique within the user search base containing this user.

Action: Use Oracle Internet Directory Self-Service Console to change the Kerberos principal name, or to change the other copies so that it is unique.

ORA-28291: No Kerberos principal value found

Action: Check the following:

1. Check that the user entry in the directory has the `krbprincipalname` attribute.

   If it does not have the `krbprincipalname` attribute, then check the following:

   - Check that the default attributes for new user creation by using Oracle Internet Directory Self-Service Console include `krbprincipalname`, and then
   - Use Enterprise Security Manager Console or Oracle Internet Directory Self-Service Console to create the user again, or
   - Add the `orclcommonattributes` object class.

2. Check that there is a value for the attribute `krbprincipalname` in the user entry. If there is no value, then use Oracle Internet Directory Self-Service Console to enter one.
3. Use Enterprise Security Manager to check that the user search base containing this user is listed in the realm Oracle Context that you are using.
4. Check that the ACL on the user search base attribute allows read and search access to the `krbprincipalname` attributes by the `verifierServices` group. This is set properly by default, but may have been altered.

ORA-28293: No matched Kerberos principal found in any user entry.

Action: Check the following:

1. Check that a user entry exists in Oracle Internet Directory for your user.
2. Use Enterprise Security Manager or `ldapsearch` to check that a user search base containing this user is listed in the identity management realm that you are using.
3. Check that the user entry in the directory contains the correct Kerberos principal name by using the following steps:
   - Use Enterprise Security Manager Console to find the Kerberos principal name attribute that is configured for the directory in your realm, and
   - Check that the correct Kerberos principal name appears in that attribute in the user's directory entry.
4. If you have an exclusive schema for the global user in the database, check that the DN in the database matches the DN of the user entry in Oracle Internet Directory.

ORA-28300: No permission to read user entry in LDAP directory service

Action: Check that the database wallet contains the correct credentials for the database-to-directory connection. The wallet DN should be the DN of the database in Oracle Internet Directory. To retrieve the credentials, perform the following steps:

1. Use the `mkstore` command line utility to retrieve the database password for the wallet by using the following syntax:

```
mkstore -wrl <database wallet location> -viewEntry
ORACLE.SECURITY.PASSWORD -viewEntry ORACLE.SECURITY.DN
```

2. If these values are incorrect, reset the database wallet by using Database Configuration Assistant.
3. Use the DN and the password returned by `mkstore` in the following `ldapbind`:

   ```
   ldapbind -h <directory host> -p <non-SSL directory port> -D "<database DN>" -w <password>
   ```

Note: The `mkstore` utility is for troubleshooting purposes only. The name and functionality of this tool may change in the future. In 10*g* Release 1 (10.1), Oracle supports only the `viewEntry` mode.

ORA-28302: User does not exist in the LDAP directory service

Action: Check that the user entry is present in the directory.

ORA-# Errors for SSL-Authenticated Enterprise Users

If you receive an ORA-# error while using SSL-authenticated Enterprise User Security, then locate the error in the following section and take the recommended action.

ORA-1017: Invalid username/password; login denied

Action: See "USER-SCHEMA ERROR Checklist" on page 12-34

ORA-28030: Problem accessing LDAP directory service

Cause: Indicates a problem with the connection between the database and the directory.

Action: Check the following:

1. Check that there is a correct `wallet_location` value in the database's `sqlnet.ora` file. If not, then use Oracle Net Manager to enter one.
2. If Domain Name System (DNS) server discovery of Oracle Internet Directory is not used, check that there is a correct `ldap.ora` file in `$LDAP_ADMIN`, `$ORACLE_HOME/ldap/admin`, `$TNS_ADMIN`, or `$ORACLE_HOME/network/admin`. (See *Oracle Internet Directory Administrator's Guide* for information about DNS server discovery.)
3. Check that the SSL port used (by way of DNS discovery or an `ldap.ora` file) supports SSL with two-way authentication.

4. Check that the LDAP_DIRECTORY_ACCESS parameter is set to SSL in the database initialization parameters file.
5. Check that the database wallet has auto login enabled. Either use Oracle Wallet Manager, or check that there is a cwallet.sso file in *$ORACLE_HOME*/admin/*<ORACLE_SID>*/wallet/.
6. Use the mkstore command line utility to check that the database wallet has the database DN in it by using the following syntax:

   ```
   mkstore -wrl <database_wallet_location> -viewEntry ORACLE.SECURITY.DN
   ```

   If the wallet does not contain the database DN, then use Database Configuration Assistant to re-register the database with Oracle Internet Directory.
7. Check that the database can bind to Oracle Internet Directory by using its wallet with the following ldapbind:

   ```
   ldapbind -h <directory_host> -p <directory_SSLport> -U 3 -W
   "file:<database wallet_location>" -P <wallet_password>
   ```

8. Check to ensure the database belongs to only one enterprise domain.

Note: The mkstore utility is for troubleshooting purposes only. The name and functionality of this tool may change in the future. In 10*g* Release 1 (10.1), Oracle supports only the viewEntry mode.

ORA-28301: Domain policy has not been registered for SSL authentication

Action: Use Enterprise Security Manager to set the user authentication policy for this enterprise domain to include SSL.

ORA-28862: SSL handshake failed

Action: See Chapter 7, "Configuring Secure Sockets Layer Authentication" for information about configuring your SSL connection.

NO-GLOBAL-ROLES Checklist

If the enterprise user can connect to the database, but a `select * from session_roles` returns no global roles, then check the following:

1. Check that the global role has been created in the database. To create global roles, use the following syntax:

   ```
   CREATE ROLE <role_name> IDENTIFIED GLOBALLY;
   ```

2. Use Enterprise Security Manager to check that the global role is included in an enterprise role in the directory.
3. Use Enterprise Security Manager to check that the enterprise role is assigned to the user in the directory.
4. If these checks are OK, then see the "DOMAIN-READ-ERROR Checklist" on page 12-35.

USER-SCHEMA ERROR Checklist

If your database cannot read the user schema, then check the following:

1. If this is an SSL-authenticated enterprise user, then ensure that the correct user wallet is being used by checking the following:
   - There is no `WALLET_LOCATION` parameter value in the client `sqlnet.ora` file, and
   - The `TNS_ADMIN` parameter is set properly so that the correct `sqlnet.ora` file is being used.
2. Check that the schema was created in the database as a global user by using the following syntax:

   ```
   CREATE USER username IDENTIFIED GLOBALLY AS ' ';
   ```

   or by using the following syntax:

   ```
   CREATE USER username IDENTIFIED GLOBALLY AS '<DN>';
   ```

3. If the following is true:
   - The user schema is an exclusive schema (created with the `CREATE USER username IDENTIFIED GLOBALLY AS '<user_DN>';` syntax), and
   - This is an SSL-authenticated user.

   Then ensure that the DN in the user wallet matches the DN that was used in the `CREATE USER` statement.

   Use Oracle Wallet Manager to view the DN in the user wallet.

Use the following syntax to view the DN that was used with the CREATE USER statement:

```
SELECT EXTERNAL_NAME FROM DBA_USERS WHERE USERNAME='<schema>';
```

4. If you are using a shared schema, then check the following:

   – Use Enterprise Security Manager to ensure that you have created a user-schema mapping either for the entire enterprise domain, or for the database.

   – If the user-schema mapping is intended to apply to this database (not to the entire enterprise domain), then check that the database can read its own entry and subtree in the directory.

     To check this, enter the following `ldapsearch` command for your database-to-directory connection type:

     * If the database connects to the directory over SSL, then use

       ```
       ldapsearch -h <directory_host> -p <directory_SSLport> -U 3 -W
       "file:<database_wallet_path>" -P <wallet_password> -b "<database_
       DN>" "objectclass=*"
       ```

       where `<wallet_password>` is the password to the wallet, which enables you to open or change the wallet.

     * If the database connects to the directory by using password authentication, then use

       ```
       ldapsearch -h <directory_host> -p <directory_port> -D <database_DN>
       -w <database_directory_password> -b "<database_DN>" "objectclass=*"
       ```

       where `<database_directory_password>` is the password in the database wallet, which is the database's password to Oracle Internet Directory.

     You should see the database entry and the relevant mapping.

   – If the user-schema mapping applies to the entire enterprise domain rather than to only this individual database, then see "DOMAIN-READ-ERROR Checklist" on page 12-35.

DOMAIN-READ-ERROR Checklist

If your database cannot read its enterprise domain information in Oracle Internet Directory, then check the following:

1. Use Enterprise Security Manager to check that the database is a member of exactly one enterprise domain, and add it to one if it is not.
2. Check that the database can see its domain by entering one of the following at the command line:
   - If the database connects to the directory over SSL, then use

     ```
     ldapsearch -h <directory_host> -p <directory_SSLport> -U 3 -W
     "file:<database_wallet_path>" -P <wallet_password> -b "cn=OracleContext,
     <realm_DN>" "objectclass=orclDBEnterpriseDomain"
     ```

     where <wallet_password> is the password to the wallet, which enables you to open or change the wallet.
   - If the database connects to the directory by using password authentication, then use

     ```
     ldapsearch -h <directory_host> -p <directory_port> -D <database_DN> -w
     <database_directory_password> -b "cn=OracleContext, <realm_DN>"
     "objectclass=orclDBEnterpriseDomain"
     ```

     where <database_directory_password> is the password in the database wallet, which is the database's password to Oracle Internet Directory.

   This ldapsearch should return exactly one enterprise domain.

   If no domain is returned, and Enterprise Security Manager shows the database as a member of a domain, then restart the database. Restarting the database updates the cached value for the enterprise domain.

   If more than one domain is returned, then use Enterprise Security Manager to remove the database from the additional domain.
3. Check that the database can read the enterprise domain subtree, and thus can read its enterprise roles and mappings, by entering one of the following at the command line:
   - If the database connects to the directory over SSL, then use

     ```
     ldapsearch -h <directory_host> -p <directory_SSLport> -U 3 -W
     "file:<database_wallet_path>" -P <wallet_password> -b "cn=OracleContext,
     <realm_DN>" "objectclass=orclDBEnterpriseRole"
     ```

     where <wallet_password> is the password to the wallet, which enables you to open or change the wallet.

– If the database connects to the directory by using password authentication, then use

```
ldapsearch -h <directory_host> -p <directory_port> -D <database_DN> -w
<database_directory_password> -b "cn=OracleContext, <realm_DN>"
"objectclass=orclDBEnterpriseRole"
```

where `<database_directory_password>` is the password in the database wallet, which is the database's password to Oracle Internet Directory.

This `ldapsearch` should return all of the enterprise roles that you have created for this domain. If it does not, then use Enterprise Security Manager to create enterprise roles and mappings.

4. Use Enterprise Security Manager to set or reset the user authentication policy for the relevant enterprise domain. See "Managing Database Security Options for an Enterprise Domain" on page 13-19 for information about setting the user authentication policy for an enterprise domain.

13

Administering Enterprise User Security

This chapter describes how to use Enterprise Security Manager to administer Enterprise User Security in Oracle Databases. This chapter contains the following topics:

Enterprise User Security Administration Tools Overview

Enterprise Security Manager and Enterprise Security Manager Console are the two main tools provided for administering Enterprise User Security.

Use Enterprise Security Manager to create and manage

- Enterprise domains
- Enterprise roles

Use Enterprise Security Manager Console to create, manage, and configure

- Enterprise users
- Enterprise User Security administrative groups
- Identity management realm properties.

These tools are introduced in Chapter 2, "Configuration and Administration Tools Overview" where you can find information about starting each tool and navigating its interface.

In particular, refer to the following topics to get started using Enterprise User Security administration tools:

Tool	Introductory Topics
Enterprise Security Manager	■ "Enterprise Security Manager and Enterprise Security Manager Console" on page 2-14 ■ "Enterprise Security Manager Initial Installation and Configuration Overview" on page 2-15 ■ "Starting Enterprise Security Manager" on page 2-16
Enterprise Security Manager Console	■ "Enterprise Security Manager Console Overview" on page 2-22 ■ "Logging in to Enterprise Security Manager Console" on page 2-22 ■ "Navigating Enterprise Security Manager Console User Interface" on page 2-25

Administering Identity Management Realms

An identity management realm is a subtree of directory entries, all of which are governed by the same administrative policies. A realm Oracle Context is a subtree in a directory identity management realm that contains the data used by any installed Oracle product that uses the directory. Enterprise Security Manager is one such product. It lets you manage database and security-related information in an identity management realm.

This section describes how to use Enterprise Security Manager to administer directory identity management realm properties that pertain to Enterprise User Security. It contains the following topics:

- Identity Management Realm Versions
- Setting Properties of an Identity Management Realm
  - Setting Login Name, Kerberos Principal Name, User Search Base, and Group Search Base Identity Management Realm Attributes
  - Setting the Default Database-to-Directory Authentication Type for an Identity Management Realm
- Managing Identity Management Realm Administrators

Note: Do not create users within a realm Oracle Context.

See Also:

- "How Oracle Internet Directory Implements Identity Management" on page 11-5 for a discussion of identity management realms and realm Oracle Contexts and how they are related to one another.
- "About Enterprise User Security Directory Entries" on page 11-11 for a discussion of the Oracle Internet Directory entries that are used for Enterprise User Security.

Identity Management Realm Versions

Enterprise User Security can only use an identity management realm supplied by Oracle Internet Directory 10*g* (9.0.4) or later, which ships with Oracle Application Server 10*g* (9.0.4). You can manage Enterprise User Security directory entries in a version 9.0.4 identity management realm by using Enterprise Security Manager for Oracle Database 10*g*.

Enterprise Security Manager displays all existing version 9.0.4 identity management realms in its main application tree.

Note: Enterprise User Security did not require identity management realms in Oracle8*i*, nor in Oracle9*i*. In those previous releases, only an Oracle Context was used. For Oracle Database 10*g* Enterprise User Security, full identity management realms and their associated realm Oracle Contexts must be used.

Setting Properties of an Identity Management Realm

An identity management realm has a number of properties that can be viewed and managed by using Enterprise Security Manager. These properties are described in Table 13–1.

Table 13–1 Identity Management Realm Properties

Property	Description
Attribute for Login Name	Name of the directory attribute used to store login names. By default, login names are stored in the `uid` attribute, but can be changed to correspond to your directory configuration. In prior releases, this was the `cn` attribute.
Attribute for Kerberos Principal Name	Name of the directory attribute used to store Kerberos principal names. By default, Kerberos principal names are stored in the `krbPrincipalName` directory attribute, but can be changed to correspond to your directory configuration by changing `orclCommonKrbPrincipalAttribute` in the identity management realm.
User Search Base	Full distinguished name (DN) for the node at which enterprise users are stored in the directory.
Group Search Base	Full DN for the node at which user groups are stored for this identity management realm in the directory.
Version Compatibility	This property is no longer used. However, you should ensure that it is not set to `81000`, since release 8.1.7 and earlier databases cannot be in the same realm with *10g* Release 1 (10.1) databases.

Setting Login Name, Kerberos Principal Name, User Search Base, and Group Search Base Identity Management Realm Attributes

Setting these identity management realm attributes enables the database to locate Enterprise User Security entries.

To set Login Name, Kerberos Principal Name, User Search Base, and Group Search Base identity management realm attributes:

1. Navigate to the Enterprise Security Manager Console home page. (Choose **Launch Enterprise Security Manager Console** from the Operations menu and log in by using your OracleAS Single Sign-On username and password.)
2. Choose the Realm Configuration tab.

3. In the Realm Information window, enter the appropriate information into the available fields.
4. Click **Submit** to save your changes to the directory.

Setting the Default Database-to-Directory Authentication Type for an Identity Management Realm

Setting the default database-to-directory authentication type, enters a value for the LDAP_DIRECTORY_ACCESS initialization parameter. This parameter is set on individual databases when they are registered in Oracle Internet Directory.

To set the default database-to-directory authentication type for an identity management realm:

1. Select the identity management realm in the left navigator pane.
2. Choose the General tab in the right main window.
3. In the Realm Attribute Settings region of the General tabbed window, choose either **PASSWORD** or **SSL** from the **Database to Directory** list.
4. Click **Apply** to save your changes to the directory.

Managing Identity Management Realm Administrators

An identity management realm contains administrative groups that have varying levels of privileges. The administrative groups for an identity management realm, which pertain to Enterprise User Security, are defined in Table 13–2. For more information about these groups, see "Administrative Groups" on page 11-17.

Table 13–2 Enterprise User Security Identity Management Realm Administrators

Administrative Group	Definition
Oracle Database Registration Administrators (OracleDBCreators)	Registers new databases in the realm.
Oracle Database Security Administrators (OracleDBSecurityAdmins)	Has all privileges on the OracleDBSecurity directory subtree. Creates, modifies, and can read all Enterprise User Security directory objects.
Oracle Context Administrators (OracleContextAdmins)	Has full access to all groups and entries within its associated realm.
User Security Administrators (OracleUserSecurityAdmins)	Has relevant permissions necessary to administer security aspects for enterprise users in the directory. For example, OracleUserSecurityAdmins can modify user passwords.

To manage identity management realm administrators:

1. Navigate to the Enterprise Security Manager Console home page. (Choose **Launch Enterprise Security Manager Console** from the Operations menu and log in by using your OracleAS Single Sign-On username and password.)
2. Choose the Users and Groups tab.
3. In the Users and Groups tabbed window, choose the Group subtab.
4. In the Group subtab window, select the administrative group you wish to edit, and click **Edit**.
5. In the Edit Group window, enter group information into the appropriate fields. You can change group owners, add users to or remove them from groups, and view group membership.
6. Click **Submit** to save your changes to the directory.

Administering Enterprise Users

Enterprise Security Manager manages one directory server at a time, identified at the top of the main application tree. It lets you manage enterprise users and data that is relevant to Enterprise User Security in the identity management.

This section describes how to use Enterprise Security Manager to administer enterprise users. It contains the following topics:

- Creating New Enterprise Users
- Setting Enterprise User Passwords
- Defining an Initial Enterprise Role Assignment
- Browsing Users in the Directory

Creating New Enterprise Users

Use Enterprise Security Manager to create users in the directory.

> **Note:** Before creating new enterprise users, you must define the user search base in the directory. See "Setting Login Name, Kerberos Principal Name, User Search Base, and Group Search Base Identity Management Realm Attributes" on page 13-5

To create new enterprise users:

1. Select **Launch Enterprise Security Manager Console** from the Operations menu. The Enterprise Security Manager Console home page appears (Figure 13–1). Log in with your OracleAS Single Sign-On username and password.

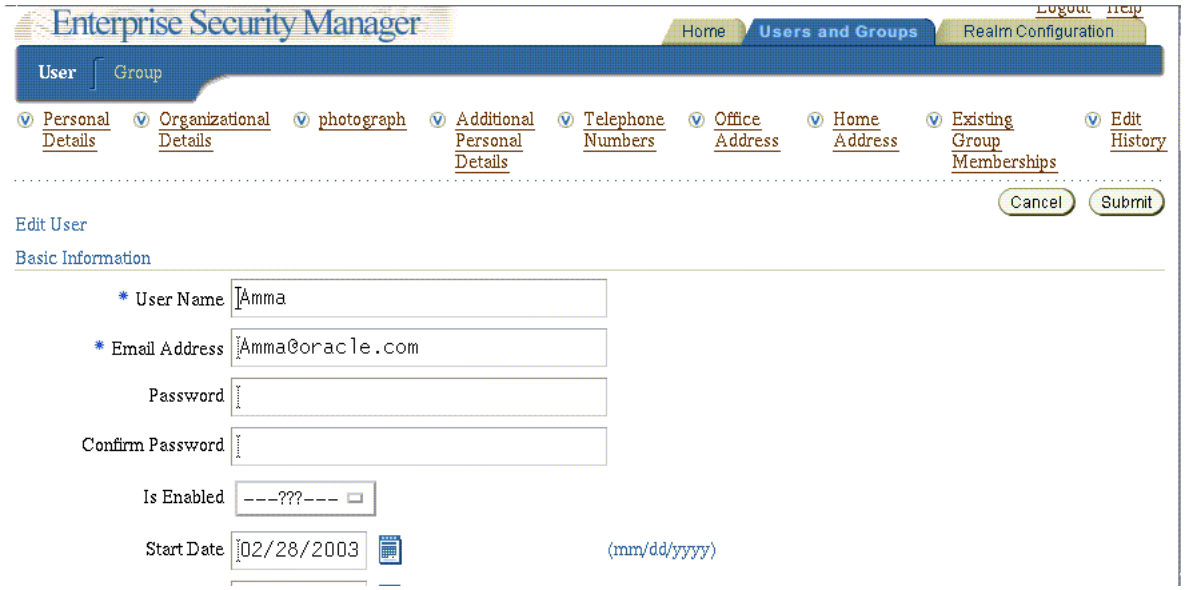
Figure 13–1 Enterprise Security Manager Console Home Page

2. Choose the Users and Groups tab.
3. In the Users and Groups tabbed window, choose the User subtab, if it is not already displayed.
4. In the User subtab window, click **Create** (located on the upper right corner of the Search Results table). Note that if your users are authenticated to the database by using Kerberos credentials, and the krbPrincipalName attribute is not there, then see "Configuring Enterprise Security Manager Console for Kerberos-Authenticated Enterprise Users" on page 2-24 for information about how to configure this.
5. Enter the appropriate user information in the Create User window and click **Submit** to create a new enterprise user.

Setting Enterprise User Passwords

You can set and maintain enterprise user passwords in the Basic Information region of the Enterprise Security Manager Console Edit User window (Figure 13–2).

Figure 13–2 Enterprise Security Manager Console Edit User Window: Basic Information



The enterprise user password is used for:

- Directory logon
- Database logon, to databases that support password authentication for global users

To set the password for an enterprise user:

1. Navigate to the Enterprise Security Manager Console home page. (Choose **Launch Enterprise Security Manager Console** from the Operations menu and log in using your OracleAS Single Sign-On username and password.)
2. Choose the Users and Groups tab.
3. In the Users and Groups tabbed window, choose the User subtab, if it is not already displayed.
4. In the User subtab window, enter part of the enterprise user's username (login name) or e-mail address, and click **Go**.

   A list of all users that match your search criteria displays.
5. Select the user for whom you wish to create a new password, and click **Edit**.
6. In the Edit User window, enter the new password, and click **Submit**.

Defining an Initial Enterprise Role Assignment

When you create a new enterprise user, you can grant any previously configured enterprise roles to the new user.

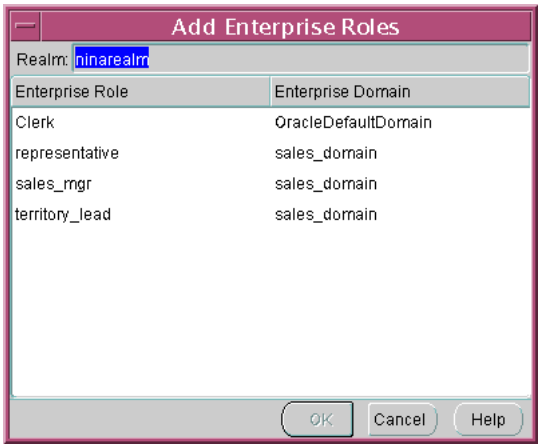
See Also: "Administering Enterprise Roles" on page 13-27

To assign existing enterprise roles to a new enterprise user:

1. In the left navigator pane, choose the Users icon under the Users, By Search Base folder, which display under the identity management realm you are using. The list of users displays in the right main window.
2. Select a user in the main window, and click **Edit...**. An Edit User window displays.
3. Choose the Enterprise Roles tab of the Edit User window, and click **Add...**.

   The Add Enterprise Roles window appears (Figure 13–3):

Figure 13–3 Enterprise Security Manager: Add Enterprise Roles Window



4. Select the correct identity management realm, then select any enterprise roles in your realm to assign to the new user, and choose **OK**.

Browsing Users in the Directory

Enterprise Security Manager lets you browse the directory for all users currently stored there in two ways—by using Enterprise Security Manager Console, or by using the All Users tab in the main application window.

To browse enterprise users in the directory by using Enterprise Security Manager Console:

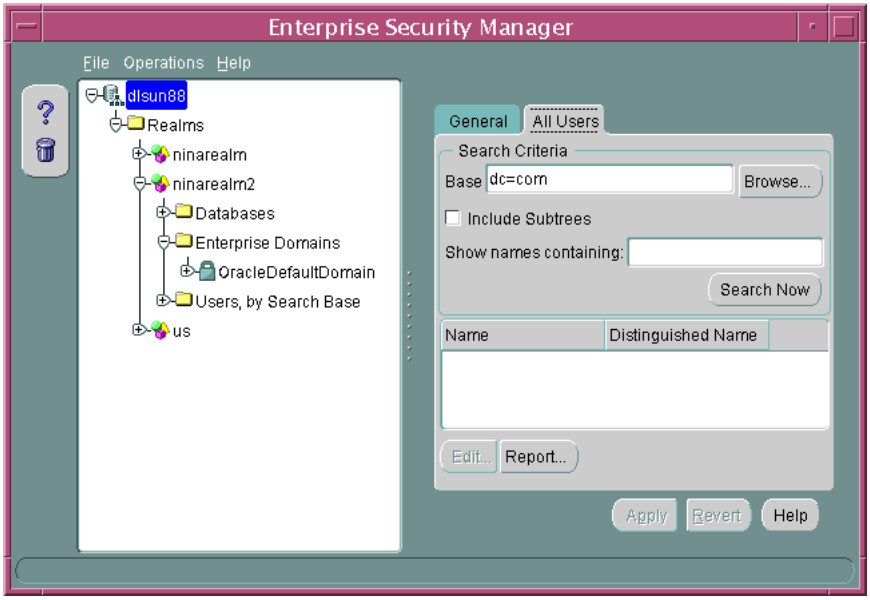
1. Navigate to the Enterprise Security Manager Console home page. (Choose **Launch Enterprise Security Manager Console** from the Operations menu and log in by using your OracleAS Single Sign-On username and password.)
2. Choose the Users and Groups tab.
3. In the Users and Groups tabbed window, choose the User subtab, if it is not already displayed.
4. In the User subtab window, enter part of the enterprise user's username (login name) or e-mail address, and click **Go**. To display all users, do not enter search criteria.

A list of all users that match your search criteria displays. You can browse through the displayed users and select one to **Edit**, **Delete**, or **Assign Privileges**. If you need to create a new user, click **Create**.

To browse enterprise users in the directory by using the All Users tab in the main application window:

1. Select the directory in the left navigator pane.
2. Choose the All Users tab in the right main window (Figure 13–4):

Figure 13–4 Enterprise Security Manager: Main Window (All Users Tab)



3. Define the search criteria and click Search Now. The window displays the results of the search. Table 13–3 summarizes the search criteria and their respective effects on the search results.

Table 13–3 Directory Search Criteria

Search Criteria	Effect on the Search
Base	This is the base entry point in the directory where the search is performed. Only users under this base are returned by the search.
Include Subtrees	This determines whether to show *all* users found in the entire subtree under the selected base, or to only show only those users that exist directly under that base location (one level only).
Show names containing	This *limits the search* to those users whose directory entries have a common name that starts with the characters you specify. This is useful if you do not know the exact name or base of the target users.

Note that you can also browse enterprise users in the directory by selecting ***realm_name*** > **User, by Search Base** > **Users** in the left navigation pane of the main application window.

Administering Enterprise Domains

An identity management realm contains an enterprise domain called `OracleDefaultDomain`. The `OracleDefaultDomain` is part of the realm when it is first created in the directory. When a new database is registered into a realm, it automatically becomes a member of the `OracleDefaultDomain` in that realm. You can create and remove your own enterprise domains but you must not remove the `OracleDefaultDomain` from a realm.

This section describes how to use Enterprise Security Manager to administer enterprise domains in the directory. It contains the following topics:

- Creating a New Enterprise Domain
- Defining Database Membership of an Enterprise Domain
- Managing Database Security Options for an Enterprise Domain
- Managing Enterprise Domain Administrators
- Managing Enterprise Domain Database Schema Mappings
- Managing Password Accessible Domains
- Managing Database Administrators

Creating a New Enterprise Domain

If you do not want to use the `OracleDefaultDomain`, then you can create a new enterprise domain in your identity management realm.

To create a new enterprise domain in an identity management realm:

1. Start by using one of the following methods:
   - Select **Create Enterprise Domain** from the Operations menu.
   - Select a realm from the main application tree with a right mouse-click.

   The Create Enterprise Domain window appears (Figure 13–5):

Figure 13–5 Enterprise Security Manager: Create Enterprise Domain Window

2. In the Create Enterprise Domain window, select the appropriate **Realm** from the list (Figure 13–5).

   Note: If you invoked the Create Enterprise Domain window by right-clicking the realm in the main application tree, the name of that realm is already selected.

3. Enter the name of the new enterprise domain, in the **Domain Name** field.
4. Choose **OK**. The new enterprise domain is created in the realm, and appears on the main application tree.

To remove an enterprise domain:

1. Select the target enterprise domain from the main application tree.
2. Use either of the following methods:

- Select **Remove Enterprise Domain** from the Operations menu.
- Select an enterprise domain from the main application tree with a right mouse-click.

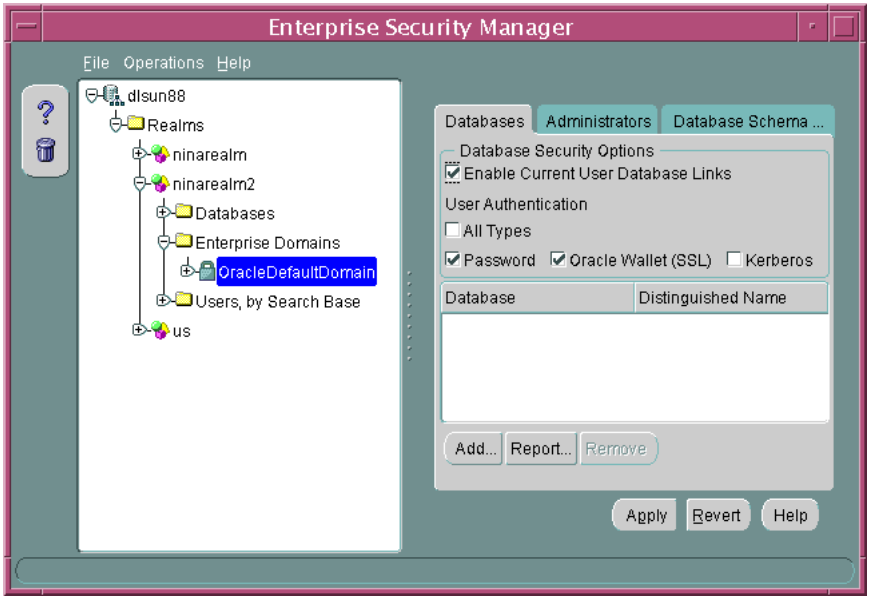
3. Enterprise Security Manager asks you to confirm removal of the enterprise domain from the realm. Choose **OK** to remove it.

Note: You cannot remove an enterprise domain from an identity management realm if that enterprise domain contains any enterprise roles.

Defining Database Membership of an Enterprise Domain

Use the navigation tree of the main Enterprise Security Manager window to select a specific enterprise domain. You can then use the Databases tab to manage database membership of an enterprise domain in a realm (Figure 13–6):

Figure 13–6 Enterprise Security Manager: Databases Tab (Database Membership)



To remove a database from an enterprise domain:

1. Select a specific database for removal, and choose **Remove...**. The database is removed from the list.
2. Choose **Apply**. The database is removed from the enterprise domain.

To add a database to an enterprise domain:

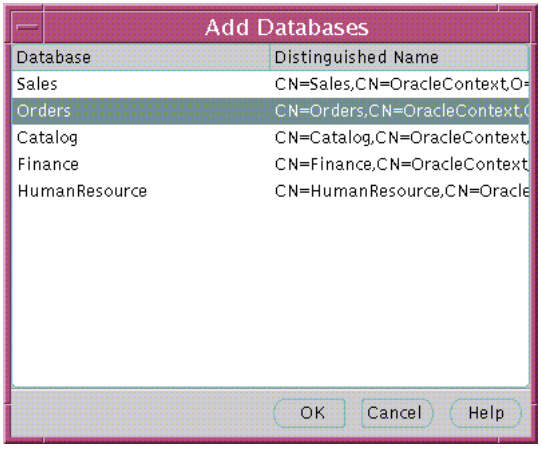
Note: The following restrictions apply to adding databases to an enterprise domain:

- A database must be in an enterprise domain for enterprise users to be able to connect to it.
- You can only add a database to an enterprise domain if both the database and the enterprise domain exist in the same realm.
- A database cannot be added as a member of two different enterprise domains.

---

1. Choose **Add...**. The Add Databases window appears. This window lists all the databases associated with the realm (Figure 13–7):

Figure 13–7 Enterprise Security Manager: Add Databases Window



2. Select a new database to be added to the enterprise domain.
3. Choose **OK**. The selected database is added to the list of databases in the Databases tabbed window (Figure 13–6).
4. Choose **Apply** (Figure 13–6). The new database is added to the enterprise domain.

Managing Database Security Options for an Enterprise Domain

Use the Databases tabbed window (Figure 13–6) to manage database security options applicable to all databases that are members of the enterprise domain.

Database security options are summarized by Table 13–4:

Table 13–4 Enterprise Security Manager Database Security Options

Database Security Option	Description
Enable current user database links	Any database pair can only permit use of *Current User Database Links* if both databases exist in the same enterprise domain where this setting is enabled. By default, current user database links are not enabled.
User authentication	All databases in an enterprise domain allow one, or more, of the following types of authentication for its clients: ■ All (the default setting) Databases can accept all currently available authentication methods for Enterprise User Security. In 10*g* Release 1 (10.1), this includes passwords, SSL by using PKI credentials, or Kerberos credentials. ■ Password ■ SSL (PKI certificates) ■ Kerberos

Managing Enterprise Domain Administrators

An **Enterprise Domain Administrator** is a directory user with privileges to modify the content of that domain. You can use the Administrators tabbed window to manage Enterprise Domain Administrators when an enterprise domain is selected under an realm in the main application tree.

To add a new user to the list of Enterprise Domain Administrators:

1. In the left navigator pane, select the enterprise domain to which you wish to add administrators.
2. In the right pane, select the **Administrators** tab.
3. Choose **Add...**. The Add Users window appears. Use this window to locate and select users for designation as Enterprise Domain Administrators. The new users appear in the Administrators tabbed window.
4. Choose **Apply**. The new Administrators are added to the enterprise domain.

To remove a user from the list of Enterprise Domain Administrators:

1. In the left navigator pane, select the enterprise domain from which you wish to remove administrators.
2. In the right pane, select the **Administrators** tab.
3. Select a user from the list of Administrators.
4. Choose **Remove**. The selected user is removed from the list.
5. Choose **Apply**. The user is removed as an Enterprise Domain Administrator for that domain in the realm.

Managing Enterprise Domain Database Schema Mappings

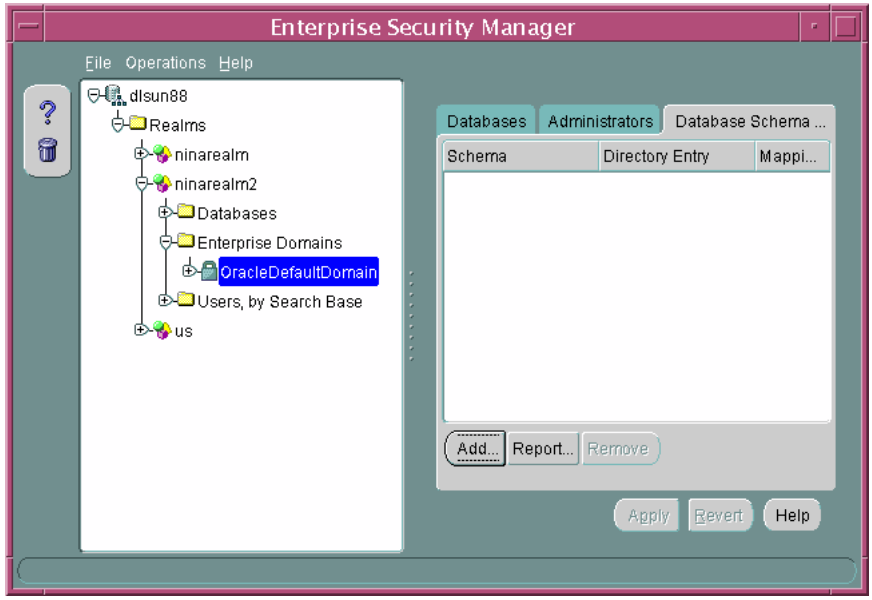
Database **schema mappings**, also referred to as user schema mappings) let databases that are registered in the directory accept connections from users without requiring any dedicated database **schemas** for them. For example, when local user Scott connects to a database, a database schema called Scott must exist—for that logon to be successful. This can be difficult to maintain if there are thousands of users and perhaps hundreds of databases in a very large enterprise.

Users that are defined in an LDAP-compliant directory do not require dedicated schemas on every Oracle9*i* or later database to which they might connect.

A database can use a schema mapping to share one database schema between multiple directory users. The schema mapping is a pair of values: the base in the directory at which users exist, and the name of the database schema they will use.

You can use the Database Schema Mappings tabbed window to manage database schema mappings—when a database is selected under a realm in the main application tree or when a domain is selected. If a domain is selected, these mappings apply to all databases that are members of the enterprise domain. Therefore, each database in the enterprise domain must have a schema of the same name used in the mapping for that mapping to be effective on that database. This window contains a list of database schema names, directory DNs, and mapping types (Figure 13–8):

Figure 13–8 Enterprise Security Manager: Database Schema Mappings Tab

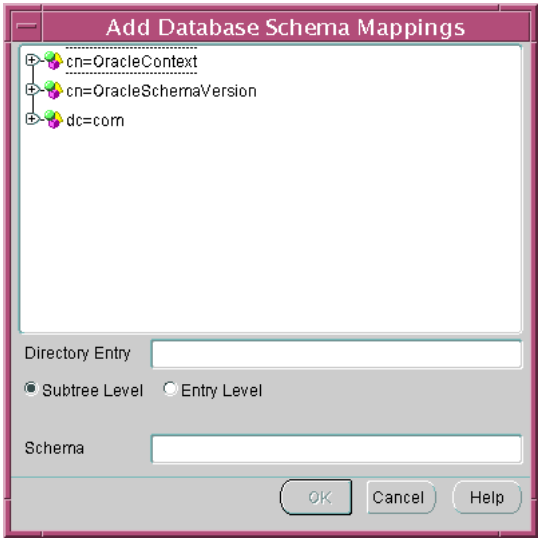


To add a new mapping to the list of database schema mappings in the enterprise domain:

1. In the Database Schema Mapping tabbed window, choose **Add...**.

   The Add Database Schema Mappings window appears (Figure 13–9). Use this window to locate and select a base in the directory and pair it with a database schema name, to make a database schema mapping. There are three components to the window: there is a directory search tree from which to select the user's DN or the base of users, the option to choose either subtree-level or entry-level mapping, and a field in which to enter a schema name.

Figure 13–9 Enterprise Security Manager: Add Database Schema Mappings Window



2. Navigate the directory to select a desired entry as a base for the database schema mapping. This can be any directory entry but should be either the actual user (entry-level) or located above the subtree of users to be mapped (subtree-level). You can also edit the contents of the Directory Entry field in this window to manually define the base.
3. Choose the mapping type: **Subtree Level** or **Entry Level**. Note that subtree-level mapping is usually the most useful.

4. Enter the name of the database schema for which this Mapping will be made into the Schema field, and choose **OK**. This must be a valid name, for a schema that already exists on that database.The new database schema mapping appears in the database schema mappings window (Figure 13–8).
5. Choose **Apply**. The new database schema mapping is added to the selected database or domain in the realm.

To remove a mapping from the list of database schema mappings in an enterprise domain:

1. Select a mapping by selecting from the Database Schema Mapping tabbed window.
2. Choose **Remove**. The selected Mapping is removed from the list.
3. Choose **Apply**. The mapping is removed from the enterprise domain.

Managing Password Accessible Domains

There are three requirements for a database to accept a connection from a password-authenticated user:

- The database must be a member of a domain configured to accept Password authentication (See: Table 13–4 on page 13-19).
- The domain must be a member of a password-accessible domains group, called the **Password-Accessible Domains List**, added by a member of either the OracleContextAdmins or the OracleDBSecurityAdmins directory administrator groups. Domain members (databases) of this list can read the user's password verifier in the directory, while those excluded from this list cannot.
- The user entry must be in a directory subtree of users that has been enabled for Oracle database access.

To configure password accessibility:

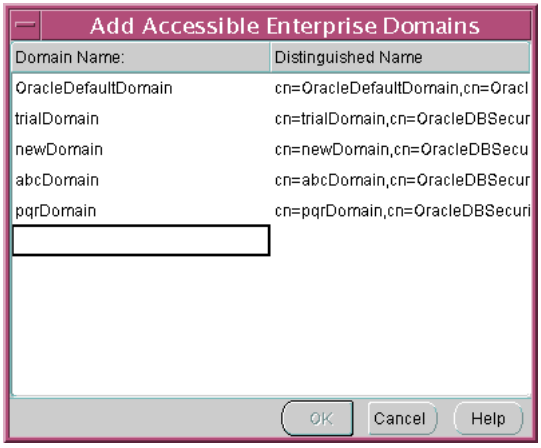
1. Select the enterprise domain in the left navigator pane.
2. Choose the Databases tabbed window and select **Password** or **All Types** from the **User Authentication** methods listed. (See Figure 13–6 on page 13-17)
3. Click **Apply**.

To add a domain to the Password-Accessible Domains List:

1. Select the identity management realm in the left navigator pane.

2. Choose the Accessible Domains tabbed window and click **Add**. The Add Accessible Enterprise Domains dialog box appears. See Figure 13–10 on page 13-24.

Figure 13–10 Enterprise Security Manager: Add Accessible Enterprise Domains Dialog Box



3. Select the **OracleDefaultDomain** from the list of enterprise domains, and click **OK**. The OracleDefaultDomain is added to the password-accessible domains list.

Note:

- By default, the `cn=Users` subtree in an identity management realm has ACLs (access control lists) to enable appropriate database access to user password attributes. If you do not use this subtree to store users, then see *Oracle Internet Directory Administrator's Guide* for information about setting up proper ACLs for another user search base.
- The OracleDefaultDomain is a member of the password-accessible domains list by default, but it can be removed.

To remove an enterprise domain from the password-accessible domains list:

1. Select the identity management realm in the left navigator pane.
2. Choose the Accessible Domains tabbed window and select the enterprise domain that you want to remove from the list.
3. Click **Remove**.

See Also:

- "Defining Database Membership of an Enterprise Domain" on page 13-17
- "Managing Database Security Options for an Enterprise Domain" on page 13-19

Managing Database Administrators

A **Database Administrator** is a directory user that has privileges to modify the database and its subtree in the realm. Database Administrators may be managed by using the Administrators tabbed window when a database is selected under a realm in the main application tree.

To remove a user from the list of Database Administrators:

1. In the Administrators tabbed window, select a user from the list of administrators.
2. Choose **Remove**; the selected user is removed from the list.
3. Choose **Apply**; the user is removed as a Database Administrator for that database.

To add a new user to the list of Database Administrators:

1. In the Administrators tabbed window, choose **Add**; the Add Users window appears. Use this window to locate and select users in the directory.
2. Select a user or users from the directory to be added as a Database Administrator; the new user(s) is displayed in the Administrators tabbed window.
3. Choose **Apply**; the new Administrator(s) is added to the database in the realm.

See Also:

- "Creating New Enterprise Users" on page 13-9
- "Browsing Users in the Directory" on page 13-12

Administering Enterprise Roles

An **enterprise domain** within an identity management realm can contain multiple **enterprise role**s. An enterprise role is a set of Oracle role-based **authorization**s across one or more databases in an enterprise domain.

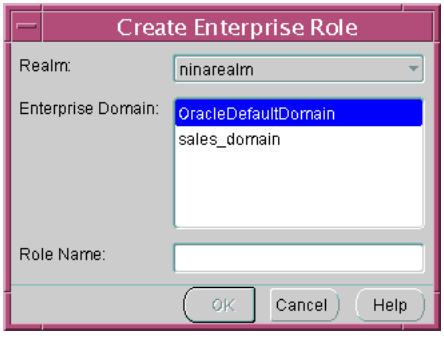
This section describes how to use Enterprise Security Manager to administer enterprise roles in the directory. It contains the following topics:

- Creating a New Enterprise Role
- Assigning Database Global Role Membership to an Enterprise Role
- Granting Enterprise Roles to Users

Creating a New Enterprise Role

You can create an enterprise role in an enterprise domain either from the Operations menu on the Enterprise Security Manager main window (Figure 13–8), or by right-clicking an enterprise domain in the main application tree. In either case, the Create Enterprise Role window appears (Figure 13–11):

Figure 13–11 Enterprise Security Manager: Create Enterprise Role Window



To create a new enterprise role:

1. Choose the target identity management realm from the list. This is the realm containing the target enterprise domain to hold the new enterprise role.

Note: If you invoked the Create Enterprise Role window by right-clicking an enterprise domain, the name of the identity management realm is already selected.

2. Select the appropriate enterprise domain for the new enterprise role, from the Enterprise Domain list.

Note: If you invoked the Create Enterprise Role window by right-clicking an enterprise domain, the name of the enterprise domain is already selected.

3. Enter the name of the new enterprise role in the **Role Name** field.
4. Choose **OK**. The new enterprise role is created in the enterprise domain, and appears on the main application tree.

To remove an enterprise role:

1. Select the target enterprise role from the main application tree (Figure 13–8).
2. Choose **Remove Enterprise Role**, either from the Operations menu or by right-clicking the enterprise domain in the main application tree.
3. Enterprise Security Manager asks you to confirm the removal of the enterprise role. Choose **Yes**.

Assigning Database Global Role Membership to an Enterprise Role

Use the Database Global Roles tabbed window (Figure 13–12) of the Enterprise Security Manager main window to manage database global role membership in an enterprise role. This window lists the names of each **global role** that belongs to the enterprise role, along with the name of the database on which that global role exists.

Figure 13–12 Enterprise Security Manager: Database Global Roles Tab

When populating an enterprise role with different database roles it is only possible to reference roles on databases that are configured to be *global roles* on those databases. A global role on a database is identical to a normal role, except that the **Database Administrator** has defined it to be authorized only through the directory. (Global roles are created with the syntax, `CREATE ROLE <role_name> IDENTIFIED GLOBALLY ' ';`) A Database Administrator cannot locally grant and revoke global roles to users of the database.

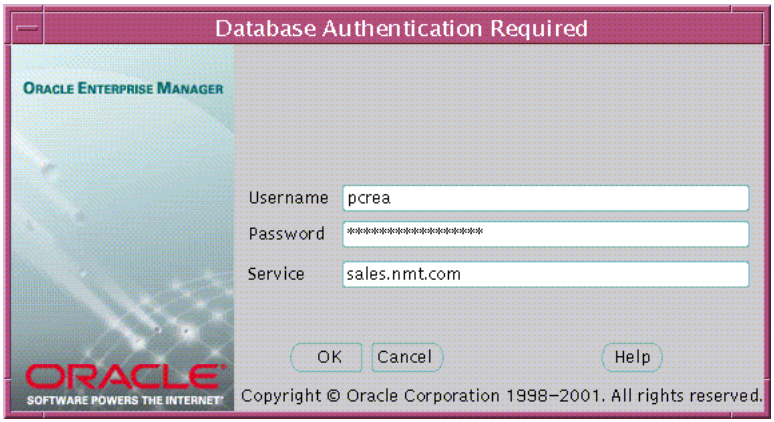
To add a global role to an enterprise role:

1. Choose **Add...** (Figure 13–12). The Add Global Database Roles window appears. This window lists all of the databases in the enterprise domain—from which global roles can be selected to add to an enterprise role.
2. Select a database from which to obtain global roles. A window appears and prompts you for logon details to authenticate to the database (and fetch global roles). Typically, this is a DBA logon to that database.

   Note that the name of the database appears in the Service field by default. You can use this name to connect to the database if your Oracle home has LDAP

enabled as its Oracle Net naming method, or if this name appears as a TNS alias in your local Oracle Net configuration. Otherwise, you can overwrite the content of the Service field with any other TNS alias configured for that database, or by a connect string in the format `<host>:<port>:<oracle sid>`. For example, `cartman:1521:broncos`.

Figure 13–13 Enterprise Security Manager: Database Authentication Required Window



3. Choose **OK**. Enterprise Security Manager connects you to the given database and fetches the list of global roles supported on that database. The list of values, if any, is displayed in the Add Global Database Roles window.
4. Select one or more global roles from the list of returned values and choose **OK**. These global roles appear in the Database Global Roles tabbed window (Figure 13–12).
5. Choose **Apply**. The new global roles are added to the enterprise role in the enterprise domain.

To remove a database global role from an enterprise role:

1. Select a global role from the list in the main application tree, and choose **Remove...**. The global role is removed from the list.
2. Choose **Apply**. The global role is removed from the enterprise role in the enterprise domain.

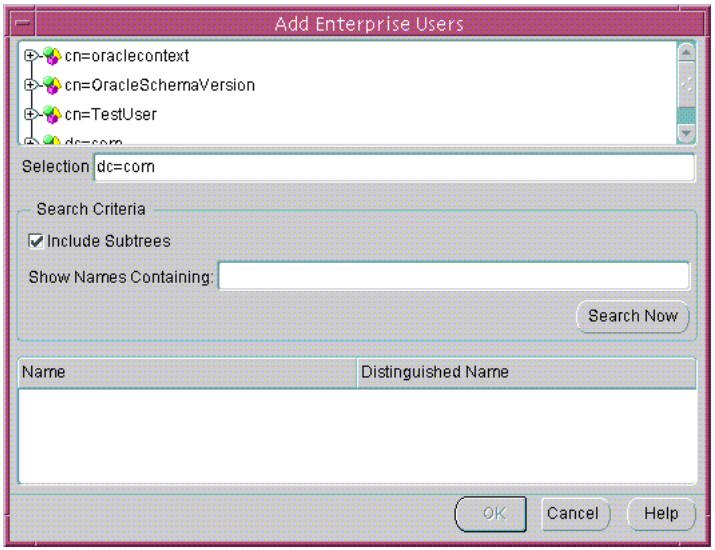
Granting Enterprise Roles to Users

You can grant an enterprise role to users in two ways: you can select a user and add a role (see "Defining an Initial Enterprise Role Assignment" on page 13-11), or you can select a role and add a user. When you grant an enterprise role to a user, it includes all database global roles contained within that enterprise role. Use the Users tabbed window.

To grant an enterprise role to users:

1. Select the role in the navigation tree, and choose **Add...** in the Users tabbed window. The Add Enterprise Users window appears. Use this window to locate and select one or more directory users to add as enterprise role grantees (Figure 13–14):

Figure 13–14 Enterprise Security Manager: Add Enterprise Users Window



2. Select a user or users and click **OK**. The new grantees are added to the list of users who have that enterprise role in the enterprise domain.
3. Choose **Apply**. The user or users are granted the selected enterprise role.

To remove a user from the list of enterprise role grantees:

1. Select a user from the list of grantees in the Users tabbed window.
2. Choose **Remove**. The selected user is removed from the list.
3. Choose **Apply**. The user is removed as a grantee for that enterprise role in the enterprise domain.

Part V

Appendixes

This part contains the following reference appendixes:

- Appendix A, "Data Encryption and Integrity Parameters"
- Appendix B, "Authentication Parameters"
- Appendix C, "Integrating Authentication Devices Using RADIUS"
- Appendix D, "Oracle Advanced Security FIPS 140-1 Settings"
- Appendix E, "orapki Utility"
- Appendix F, "Entrust-Enabled SSL Authentication"
- Appendix G, "Using the User Migration Utility"

A

Data Encryption and Integrity Parameters

This appendix describes **encryption** and data **integrity** parameters supported by Oracle Advanced Security. It also includes an example of a `sqlnet.ora` file generated by performing the network configuration described in Chapter 3, "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients" and Chapter 7, "Configuring Secure Sockets Layer Authentication".

This appendix contains the following topics:

- Sample sqlnet.ora File
- Data Encryption and Integrity Parameters

Sample sqlnet.ora File

This section contains a sample `sqlnet.ora` configuration file for a set of clients with similar characteristics and a set of servers with similar characteristics. The file includes examples of Oracle Advanced Security encryption and data integrity parameters.

Trace File Setup

```
#Trace file setup
trace_level_server=16
trace_level_client=16
trace_directory_server=/orant/network/trace
trace_directory_client=/orant/network/trace
trace_file_client=cli
trace_file_server=srv
trace_unique_client=true
```

Oracle Advanced Security Encryption

```
#ASO Encryption
sqlnet.encryption_server=accepted
sqlnet.encryption_client=requested
sqlnet.encryption_types_server=(RC4_40)
sqlnet.encryption_types_client=(RC4_40)
```

Oracle Advanced Security Integrity

```
#ASO Checksum
sqlnet.crypto_checksum_server=requested
sqlnet.crypto_checksum_client=requested
sqlnet.crypto_checksum_types_server = (MD5)
sqlnet.crypto_checksum_types_client = (MD5)
```

SSL

```
#SSL
WALLET_LOCATION = (SOURCE=
                          (METHOD = FILE)
                          (METHOD_DATA =
                           DIRECTORY=/wallet)

SSL_CIPHER_SUITES=(SSL_DH_anon_WITH_RC4_128_MD5)
SSL_VERSION= 3
SSL_CLIENT_AUTHENTICATION=FALSE
```

Common

```
#Common
automatic_ipc = off
sqlnet.authentication_services = (beq)
names.directory_path = (TNSNAMES)
```

Kerberos

```
#Kerberos
sqlnet.authentication_services = (beq, kerberos5)
sqlnet.authentication_kerberos5_service = oracle
sqlnet.kerberos5_conf= /krb5/krb.conf
sqlnet.kerberos5_keytab= /krb5/v5srvtab
sqlnet.kerberos5_realms= /krb5/krb.realm
sqlnet.kerberos5_cc_name = /krb5/krb5.cc
sqlnet.kerberos5_clockskew=900
sqlnet.kerberos5_conf_mit=false
```

RADIUS

```
#Radius
sqlnet.authentication_services = (beq, RADIUS )
sqlnet.radius_authentication_timeout = (10)
sqlnet.radius_authentication_retries = (2)
sqlnet.radius_authentication_port = (1645)
sqlnet.radius_send_accounting = OFF
sqlnet.radius_secret = /orant/network/admin/radius.key
sqlnet.radius_authentication = radius.us.oracle.com
sqlnet.radius_challenge_response = OFF
sqlnet.radius_challenge_keyword = challenge
sqlnet.radius_challenge_interface =
oracle/net/radius/DefaultRadiusInterface
sqlnet.radius_classpath = /jre1.1/
```

Data Encryption and Integrity Parameters

If you do not specify any values for Server Encryption, Client Encryption, Server Checksum, or Client Checksum, the corresponding configuration parameters do not appear in the `sqlnet.ora` file. However, Oracle Advanced Security defaults to ACCEPTED.

For both data encryption and integrity algorithms, the server selects the first algorithm listed in its `sqlnet.ora` file that matches an algorithm listed in the client `sqlnet.ora` file, or in the client installed list—if the client lists no algorithms in its `sqlnet.ora` file. If there are no entries in the server `sqlnet.ora` file, the server sequentially searches its installed list to match an item on the client side—either in the client `sqlnet.ora` file or in the client installed list. *If no match can be made and one side of the connection REQUIRED the algorithm type (data encryption or integrity), the connection fails.* Otherwise, the connection succeeds with the algorithm type inactive.

Data encryption and integrity algorithms are selected independently of each other; encryption can be activated without integrity, and integrity can be activated without encryption, as shown by Table A–1:

Table A–1 Algorithm Type Selection

Encryption Selected?	Integrity Selected?
Yes	No
Yes	Yes
No	Yes

Table A–1 Algorithm Type Selection

Encryption Selected?	Integrity Selected?
No	No

There are three classes of parameters used to enable data encryption and integrity. The first two classes listed here are required and the third (seeding the random key generator) is optional:

- Encryption and Integrity Parameters
- Seeding the Random Key Generator (Optional)

See Also:

- Chapter 3, "Configuring Network Data Encryption and Integrity for Oracle Servers and Clients"
- "About Activating Encryption and Integrity" on page 3-6

Encryption and Integrity Parameters

The following sections summarize data encryption and integrity parameters:

SQLNET.ENCRYPTION_SERVER

This parameter specifies the desired encryption behavior when a client or a server acting as a client connects to this server. The behavior of the server partially depends on the `SQLNET.ENCRYPTION_CLIENT` setting at the other end of the connection.

Table A–2 SQLNET.ENCRYPTION_SERVER Parameter Attributes

Attribute	Description
Syntax	`SQLNET.ENCRYPTION_SERVER = valid_value`
Valid Values	ACCEPTED, REJECTED, REQUESTED, REQUIRED
Default Setting	ACCEPTED

SQLNET.ENCRYPTION_CLIENT

This parameter specifies the desired encryption behavior when this client or server acting as a client connects to a server. The behavior of the client partially depends

on the value set for SQLNET.ENCRYPTION_SERVER at the other end of the connection.

Table A–3 SQLNET.ENCRYPTION_CLIENT Parameter Attributes

Attribute	Description
Syntax	SQLNET.ENCRYPTION_CLIENT = *valid_value*
Valid Values	ACCEPTED, REJECTED, REQUESTED, REQUIRED
Default Setting	ACCEPTED

SQLNET.CRYPTO_CHECKSUM_SERVER

This parameter specifies the desired data integrity behavior when a client or another server acting as a client connects to this server. The behavior partially depends on the SQLNET.CRYPTO_CHECKSUM_CLIENT setting at the other end of the connection.

Table A–4 SQLNET.CRYPTO_CHECKSUM_SERVER Parameter Attributes

Attribute	Description
Syntax	SQLNET.CRYPTO_CHECKSUM_SERVER = *valid_value*
Valid Values	ACCEPTED, REJECTED, REQUESTED, REQUIRED
Default Setting	ACCEPTED

SQLNET.CRYPTO_CHECKSUM_CLIENT

This parameter specifies the desired data integrity behavior when this client or server acting as a client connects to a server. The behavior partially depends on the SQLNET.CRYPTO_CHECKSUM_SERVER setting at the other end of the connection.

Table A–5 SQLNET.CRYPTO_CHECKSUM_CLIENT Parameter Attributes

Attribute	Description
Syntax	SQLNET.CRYPTO_CHECKSUM_CLIENT = *valid_value*
Valid Values	ACCEPTED, REJECTED, REQUESTED, REQUIRED
Default Setting	ACCEPTED

SQLNET.ENCRYPTION_TYPES_SERVER

This parameter specifies a list of encryption algorithms used by this server, in the order of intended use. This list is used to negotiate a mutually acceptable algorithm with the client end of the connection. Each algorithm is checked against the list of available client algorithm types until a match is found. If an algorithm that is not installed is specified on this side, the connection terminates with error message ORA-12650.

Table A–6 SQLNET.ENCRYPTION_TYPES_SERVER Parameter Attributes

Attribute	Description
Syntax	`SQLNET.ENCRYPTION_TYPES_SERVER = (`*`valid_encryption_algorithm`* `[,`*`valid_encryption_algorithm`*`])`
Valid Values	■ RC4_256: RSA RC4 (256-bit key size). ■ AES256: AES (256-bit key size). ■ AES192: AES (192-bit key size). ■ 3DES168: 3-key Triple-DES (168-bit effective key size). ■ RC4_128: RSA RC4 (128-bit key size). ■ AES128: AES (128-bit key size). ■ 3DES112: 2-key Triple-DES (112-bit effective key size). ■ RC4_56: RSA RC4 (56-bit key size). ■ DES: Standard DES (56-bit key size). ■ RC4_40: RSA RC4 (40-bit key size). ■ DES40: DES40 (40-bit key size).
Default Setting	If no algorithms are defined in the local `sqlnet.ora` file, all installed algorithms are used in a negotiation in the preceding sequence.

Usage Notes	You can specify multiple encryption algorithms—either a single value or a list of algorithm names. For example, either of the following encryption parameters is acceptable:
	`SQLNET.ENCRYPTION_TYPES_SERVER=(RC4_40)`
	`SQLNET.ENCRYPTION_TYPES_SERVER=(DES,RC4_56,RC4_128,DES40)`

SQLNET.ENCRYPTION_TYPES_CLIENT

This parameter specifies a list of encryption algorithms used by this client or server acting as a client. This list is used to negotiate a mutually acceptable algorithm with the other end of the connection. If an algorithm that is not installed is specified on this side, the connection terminates with error message ORA-12650.

Table A–7 SQLNET.ENCRYPTION_TYPES_CLIENT Parameter Attributes

Attribute	Description
Syntax	`SQLNET.ENCRYPTION_TYPES_CLIENT = (`*`valid_encryption_algorithm`* `[,`*`valid_encryption_algorithm`*`])`
Valid Values	■ RC4_256: RSA RC4 (256-bit key size). ■ AES256: AES (256-bit key size). ■ AES192: AES (192-bit key size). ■ 3DES168: 3-key Triple-DES (168-bit effective key size). ■ RC4_128: RSA RC4 (128-bit key size). ■ AES128: AES (128-bit key size). ■ 3DES112: 2-key Triple-DES (112-bit effective key size). ■ RC4_56: RSA RC4 (56-bit key size). ■ DES: Standard DES (56-bit key size). ■ RC4_40: RSA RC4 (40-bit key size). ■ DES40: DES40 (40-bit key size).
Default Setting	If no algorithms are defined in the local `sqlnet.ora` file, all installed algorithms are used in a negotiation.
Usage Notes	You can specify multiple encryption algorithms.

SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER

This parameter specifies a list of data integrity algorithms this server or client to another server uses, in order of intended use. This list is used to negotiate a mutually acceptable algorithm with the other end of the connection. Each algorithm is checked against the list of available client algorithm types until a match is found. If an algorithm is specified that is not installed on this side, the connection terminates with error message ORA-12650.

Table A–8 SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER Parameter Attributes

Attribute	Description
Syntax	SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (*valid_crypto_checksum_algorithm* [,*valid_crypto_checksum_algorithm*])
Valid Values	■ SHA-1: Secure Hash Algorithm ■ MD5: Message Digest 5
Default Setting	If no algorithms are defined in the local sqlnet.ora file, all installed algorithms are used in a negotiation in the preceding sequence.

SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT

This parameter specifies a list of data integrity algorithms this client or server acting as a client uses. This list is used to negotiate a mutually acceptable algorithm with the other end of the connection. If an algorithm that is not installed on this side is specified, the connection terminates with error message ORA-12650.

Table A–9 SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT Parameter Attributes

Attribute	Description
Syntax	SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (*valid_crypto_checksum_algorithm* [,*valid_crypto_checksum_algorithm*])
Valid Values	■ SHA-1: Secure Hash Algorithm ■ MD5: Message Digest 5
Default Setting	If no algorithms are defined in the local sqlnet.ora file, all installed algorithms are used in a negotiation.

Seeding the Random Key Generator (Optional)

Setting this parameter in the sqlnet.ora file as follows is optional:

```
SQLNET.CRYPTO_SEED = "10-70 random characters"
```

The characters that form the value for this parameter can be used to seed the random number generator that is used to generate cryptographic key material. The more random the characters entered into this field are, the stronger the keys are. You set this parameter by entering from 10 to 70 random characters into the preceding statement.

Note: If you use this parameter to seed the random number generator, then Oracle recommends that you enter as many characters as possible, up to 70, to make the resulting key more random and therefore stronger.

If you do not use this parameter, the system uses various sources of random numbers, depending on your operating system, to seed the random number generator.

B

Authentication Parameters

This appendix illustrates some sample configuration files with the profile file (`sqlnet.ora`) and the database initialization file authentication parameters, when using Kerberos, RADIUS, or SSL authentication.

This appendix contains the following topics:

- Parameters for Clients and Servers using Kerberos Authentication
- Parameters for Clients and Servers using RADIUS Authentication
- Parameters for Clients and Servers using SSL

Parameters for Clients and Servers using Kerberos Authentication

Following is a list of parameters to insert into the configuration files for clients and servers using Kerberos.

Table B–1 Kerberos Authentication Parameters

File Name	Configuration Parameters

`sqlnet.ora`	`SQLNET.AUTHENTICATION_SERVICES=(KERBEROS5)`
	`SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=oracle`
	`SQLNET.KERBEROS5_CC_NAME=/usr/tmp/DCE-CC`
	`SQLNET.KERBEROS5_CLOCKSKEW=1200`
	`SQLNET.KERBEROS5_CONF=/krb5/krb.conf`
	`SQLNET.KERBEROS5_CONF_MIT=(FALSE)`
	`SQLNET.KERBEROS5_REALMS=/krb5/krb.realms`
	`SQLNET.KERBEROS5_KEYTAB=/krb5/v5srvtab`

`initialization parameter file`	`REMOTE_OS_AUTHENT=FALSE` `OS_AUTHENT_PREFIX=""`

Parameters for Clients and Servers using RADIUS Authentication

The following sections describe the parameters for RADIUS authentication

- sqlnet.ora File Parameters
- Minimum RADIUS Parameters
- Initialization File Parameters

sqlnet.ora File Parameters

SQLNET.AUTHENTICATION_SERVICES

This parameter configures the client or the server to use the RADIUS adapter. Table B–2 describes this parameter's attributes.

Table B–2 SQLNET.AUTHENTICATION_SERVICES Parameter Attributes

Attribute	Description
Syntax	`SQLNET.AUTHENTICATION_SERVICES=radius`
Default setting	None

SQLNET.RADIUS_AUTHENTICATION

This parameter sets the location of the primary RADIUS server, either host name or dotted decimal format. If the RADIUS server is on a different machine from the Oracle server, you must specify either the host name or the IP address of that machine. Table B–3 describes this parameter's attributes.

Table B–3 SQLNET.RADIUS_AUTHENTICATION Parameter Attributes

Attribute	Description
Syntax	`SQLNET.RADIUS_AUTHENTICATION=`*`RADIUS_server_IP_address`*
Default setting	`localhost`

SQLNET.RADIUS_AUTHENTICATION_PORT

This parameter sets the listening port of the primary RADIUS server. Table B–4 describes this parameter's attributes.

Table B–4 SQLNET.RADIUS_AUTHENTICATION_PORT Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_AUTHENTICATION_PORT=*port_number*
Default setting	1645

SQLNET.RADIUS_AUTHENTICATION_TIMEOUT

This parameter sets the time to wait for response. Table B–5 describes this parameter's attributes.

Table B–5 SQLNET.RADIUS_AUTHENTICATION_TIMEOUT Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_AUTHENTICATION_TIMEOUT=*time_in_seconds*
Default setting	5

SQLNET.RADIUS_AUTHENTICATION_RETRIES

This parameter sets the number of times to re-send. Table B–6 describes this parameter's attributes.

Table B–6 SQLNET.RADIUS_AUTHENTICATION_RETRIES Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_AUTHENTICATION_RETRIES=*n_times_to_resend*
Default setting	3

SQLNET.RADIUS_SEND_ACCOUNTING

This parameter turns accounting on and off. If you enable accounting, packets will be sent to the active RADIUS server at the listening port plus one. By default, packets are sent to port 1646. You need to turn this feature on only when your RADIUS server supports accounting and you want to keep track of the number of times the user is logging on to the system. Table B–7 describes this parameter's attributes.

Table B–7 SQLNET.RADIUS_SEND_ACCOUNTING Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_SEND_ACCOUNTING=*on*
Default setting	off

SQLNET.RADIUS_SECRET

This parameter specifies the file name and location of the RADIUS secret key. Table B–8 describes this parameter's attributes.

Table B–8 SQLNET.RADIUS_SECRET Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_SECRET=*path_to_RADIUS_secret_key*
Default setting	$ORACLE_HOME/network/security/radius.key

SQLNET.RADIUS_ALTERNATE

This parameter sets the location of an alternate RADIUS server to be used in case the primary server becomes unavailable for fault tolerance. Table B–9 describes this parameter's attributes.

Table B–9 SQLNET.RADIUS_ALTERNATE Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_ALTERNATE=*alternate_RADIUS_server_hostname_or_IP_address*
Default setting	off

SQLNET.RADIUS_ALTERNATE_PORT

This parameter sets the listening port for the alternate RADIUS server. Table B–10 describes this parameter's attributes.

Table B–10 SQLNET.RADIUS_ALTERNATE_PORT Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_ALTERNATE_PORT=*alternate_RADIUS_server_listening_port_number*
Default setting	1645

SQLNET.RADIUS_ALTERNATE_TIMEOUT

This parameter sets the time to wait for response for the alternate RADIUS server. Table B–11 describes this parameter's attributes.

Table B–11 SQLNET.RADIUS_ALTERNATE_TIMEOUT Parameter Attributes

Attribute	Description
Syntax	`SQLNET.RADIUS_ALTERNATE_TIMEOUT=`*`time_in_seconds`*
Default setting	`5`

SQLNET.RADIUS_ALTERNATE_RETRIES

This parameter sets the number of times that the alternate RADIUS server re-sends messages. Table B–12 describes this parameter's attributes.

Table B–12 SQLNET.RADIUS_ALTERNATE_RETRIES Parameter Attributes

Attribute	Description
Syntax	`SQLNET.RADIUS_ALTERNATE_RETRIES=`*`n_times_to_resend`*
Default setting	`3`

SQLNET.RADIUS_CHALLENGE_RESPONSE

This parameter turns on or turns off the challenge-response, or asynchronous, mode support. Table B–13 describes this parameter's attributes.

Table B–13 SQLNET.RADIUS_CHALLENGE_RESPONSE Parameter Attributes

Attribute	Description
Syntax	`SQLNET.RADIUS_CHALLENGE_RESPONSE=`*`on`*
Default setting	`off`

SQLNET.RADIUS_CHALLENGE_KEYWORD

This parameter sets the keyword to request a challenge from the RADIUS server. User types no password on the client. Table B–14 describes this parameter's attributes.

Table B–14 SQLNET.RADIUS_CHALLENGE_KEYWORD Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_CHALLENGE_KEYWORD=*keyword*
Default setting	challenge

SQLNET.RADIUS_AUTHENTICATION_INTERFACE

This parameter sets the name of the Java class that contains the graphical user interface when RADIUS is in the challenge-response (asynchronous) mode. Table B–15 describes this parameter's attributes.

Table B–15 SQLNET.RADIUS_AUTHENTICATION_INTERFACE Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_AUTHENTICATION_INTERFACE=*Java_class_name*
Default setting	DefaultRadiusInterface (oracle/net/radius/DefaultRadiusInterface)

SQLNET.RADIUS_CLASSPATH

If you decide to use the challenge-response authentication mode, RADIUS presents the user with a Java-based graphical interface requesting first a password, then additional information—for example, a dynamic password that the user obtains from a token card. Add the SQLNET.RADIUS_CLASSPATH parameter in the sqlnet.ora file to set the path for the Java classes for that graphical interface, and to set the path to the JDK Java libraries. Table B–16 describes this parameter's attributes.

Table B–16 SQLNET.RADIUS_CLASSPATH Parameter Attributes

Attribute	Description
Syntax	SQLNET.RADIUS_CLASSPATH=*path_to_GUI_Java_classes*
Default setting	$ORACLE_HOME/jlib/netradius.jar:$ORACLE_HOME/JRE/lib/sparc/native_threads

Minimum RADIUS Parameters

```
sqlnet.authentication_services = (radius)
sqlnet.authentication = IP-address-of-RADIUS-server
sqlnet.radius_challenge_response = ON
```

Initialization File Parameters

```
REMOTE_OS_AUTHENT=FALSE
OS_AUTHENT_PREFIX=""
```

Parameters for Clients and Servers using SSL

There are two ways to configure a parameter:

- Static: The name of the parameter that exists in the `sqlnet.ora` file.
- Dynamic: The name of the parameter used in the security subsection of the Oracle Net address.

SSL Authentication Parameters

This section describes the static and dynamic parameters for configuring SSL on the server.

Parameter Name (static):	SQLNET.AUTHENTICATION_SERVICES
Parameter Name (dynamic):	AUTHENTICATION
Parameter Type:	String LIST
Parameter Class:	Static
Permitted Values:	Add TCPS to the list of available authentication services.
Default Value:	No default value.
Description:	To control which authentication services a user wants to use. Note: The dynamic version supports only the setting of one type.
Existing/New Parameter	Existing
Syntax (static):	SQLNET.AUTHENTICATION_SERVICES = (TCPS, *selected_method_1*, *selected_method_2*)
Example (static):	SQLNET.AUTHENTICATION_SERVICES = (TCPS, radius)
Syntax (dynamic):	AUTHENTICATION = *string*

Example (dynamic):	AUTHENTICATION = (TCPS)

Cipher Suite Parameters

This section describes the static and dynamic parameters for configuring cipher suites.

Parameter Name (static):	SSL_CIPHER_SUITES
Parameter Name (dynamic):	SSL_CIPHER_SUITES
Parameter Type:	String LIST
Parameter Class:	Static
Permitted Values:	Any known SSL cipher suite
Default Value:	No default
Description:	Controls the combination of encryption and data integrity used by SSL.
Existing/New Parameter	Existing
Syntax (static):	SSL_CIPHER_SUITES=(*SSL_cipher_suite1*[, *SSL_cipher_suite2*, ... *SSL_cipher_suiteN*])
Example (static):	SSL_CIPHER_SUITES=(SSL_DH_DSS_WITH_DES_CBC_ SHA)
Syntax (dynamic):	SSL_CIPHER_SUITES=(*SSL_cipher_suite1* [, *SSL_cipher_suite2*, ...*SSL_cipher_suiteN*])
Example (dynamic):	SSL_CIPHER_SUITES=(SSL_DH_DSS_WITH_DES_CBC_ SHA)

Supported SSL Cipher Suites

Oracle Advanced Security supports the following cipher suites:

- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_RSA_WITH_RC4_128_SHA

- SSL_RSA_WITH_RC4_128_MD5
- SSL_RSA_WITH_DES_CBC_SHA
- SSL_DH_anon_WITH_3DES_EDE_CBC_SHA
- SSL_DH_anon_WITH_RC4_128_MD5
- SSL_DH_anon_WITH_DES_CBC_SHA
- SSL_RSA_EXPORT_WITH_RC4_40_MD5
- SSL_RSA_EXPORT_WITH_DES40_CBC_SHA
- SSL_RSA_WITH_AES_128_CBC_SHA
- SSL_RSA_WITH_AES_256_CBC_SHA

Note that the cipher suites that use Advanced Encryption Standard (AES) work with Transport Layer Security (TLS 1.0) only.

SSL Version Parameters

This section describes the static and dynamic parameters for configuring the version of SSL to be used.

Parameter Name (static): SSL_VERSION

Parameter Name (dynamic): SSL_VERSION

Parameter Type: string

Parameter Class: Static

Permitted Values: Any version which is valid to SSL. (0, 3.0)

Default Value: "0"

Description: To force the version of the SSL connection.

Existing/New Parameter New

Syntax (static): SSL_VERSION=*version*

Example (static): SSL_VERSION=3.0

Syntax (dynamic): SSL_VERSION=*version*

Example (dynamic):	SSL_VERSION=3.0

SSL Client Authentication Parameters

This section describes the static and dynamic parameters for configuring SSL on the client.

Parameter Name (static):	SSL_CLIENT_AUTHENTICATION
Parameter Name (dynamic):	SSL_CLIENT_AUTHENTICATION
Parameter Type:	Boolean
Parameter Class:	Static
Permitted Values:	TRUE / FALSE
Default Value:	TRUE
Description:	To control whether a client, in addition to the server, is authenticated using SSL.
Existing/New Parameter	New
Syntax (static):	SSL_CLIENT_AUTHENTICATION={*TRUE* | *FALSE*}
Example (static):	SSL_CLIENT_AUTHENTICATION=FALSE
Syntax (dynamic):	SSL_CLIENT_AUTHENTICATION={*TRUE* | *FALSE*}
Example (dynamic):	SSL_CLIENT_AUTHENTICATION=FALSE

SSL X.509 Server Match Parameters

This section describes the parameters that are used to validate the identity of a server that the client connects to.

SSL_SERVER_DN_MATCH

Parameter Name	SSL_SERVER_DN_MATCH
Where stored	`sqlnet.ora`

Purpose	Use this parameter to force the server's **distinguished name (DN)** to match its service name. If you force the match verifications, SSL ensures that the certificate is from the server. If you choose to not enforce the match verification, SSL performs the check but permits the connection, regardless if there is a match. *Not forcing the match lets the server potentially fake its identity.*
Values	`yes|on|true`—Specify to enforce a match. If the DN matches the service name, the connection succeeds; otherwise, the connection fails. `no|off|false`—Specify to not enforce a match. If the DN does not match the service name, the connection is successful, but an error is logged to the `sqlnet.log` file.
Default	Oracle8*i*, or later:.FALSE. SSL client (always) checks server DN. If it does not match the service name, the connection succeeds but an error is logged to `sqlnet.log` file.
Usage Notes	Additionally configure the tnsnames.ora parameter `SSL_SERVER_CERT_DN` to enable server DN matching.

SSL_SERVER_CERT_DN

Parameter Name	SSL_SERVER_CERT_DN
Where stored	`tnsnames.ora`—Can be stored on the client, for every server it connects to, OR it can be stored in the LDAP directory, for every server it connects to, updated centrally.
Purpose	This parameter specifies the **distinguished name (DN)** of the server. The client uses this information to obtain the list of DNs it expects for each of the servers—to force the server's DN to match its service name.
Values	Set equal to **distinguished name (DN)** of the server.
Default	n/a
Usage Notes	Additionally configure the sqlnet.ora parameter `SSL_SERVER_DN_MATCH` to enable server DN matching.

Example

```
dbalias=(description=address_
list=(address=(protocol=tcps)(host=hostname)(
port=portnum)))(connect_
data=(sid=Finance))(security=(SSL_SERVER_
DN="CN=Finance,CN=OracleContext,C=US,O=Acme"))
```

Wallet Location

For any application that must access a wallet for loading the security credentials into the process space, you must specify the wallet location parameters defined by Table B–17 in each of the following configuration files:

- `sqlnet.ora`
- `listener.ora`

Table B–17 Wallet Location Parameters

Static Configuration	Dynamic Configuration

`WALLET_LOCATION =`
`(SOURCE=`
`(METHOD=File)`
`(METHOD_DATA=`
`(DIRECTORY=your wallet location)`
`)`
`)`
	`MY_WALLET_DIRECTORY`
	`= your_wallet_dir`

The default wallet location is the `$ORACLE_HOME` directory.

C

Integrating Authentication Devices Using RADIUS

This appendix describes how third party authentication vendors customize the RADIUS challenge-response user interface to fit their particular device.

This appendix contains the following topics:

- About the RADIUS Challenge-Response User Interface
- Customizing the RADIUS Challenge-Response User Interface

See Also: Chapter 5, "Configuring RADIUS Authentication"

About the RADIUS Challenge-Response User Interface

You can set up any authentication device that supports the RADIUS standard to authenticate Oracle users. When your authentication device uses the challenge-response mode, a graphical interface prompts the user first for a password, then for additional information—for example, a dynamic password that the user obtains from a token card. This interface is Java-based to provide optimal platform independence.

Third party vendors of authentication devices must customize this graphical user interface to fit their particular device. For example, a smart card vendor customizes the Oracle client to issue the challenge to the smart card reader. Then, when the smart card receives a challenge, it responds by prompting the user for more information, such as a PIN.

Customizing the RADIUS Challenge-Response User Interface

You can customize this interface by creating your own class to support the functionality described in Table C–1. You can then open the `sqlnet.ora` file, look up the SQLNET.RADIUS_AUTHENTICATION_INTERFACE parameter, and replace the name of the class listed there (`DefaultRadiusInterface`), with the name of the new class you have just created. When you make this change in the `sqlnet.ora` file, the class is loaded on the Oracle client in order to handle the authentication process.

The third party must implement the Oracle RADIUS Interface, which is located in the ORACLE.NET.RADIUS package.

```
public interface OracleRadiusInterface {
  public void radiusRequest();
  public void radiusChallenge(String challenge);
  public String getUserName();
  public String getPassword();
}
```

Table C–1 Server Encryption Level Setting

Parameter	Description
`radiusRequest`	Generally, this prompts the user for a user name and password which will later be retrieved through getUserName and getPassword.
`getUserName`	Extracts the user name the user enters. If this method returns an empty string, it is assumed that the user wants to cancel the operation. The user then receives a message indicating that the authentication attempt failed.
`getPassword`	Extracts the password the user enters. If getUserName returns a valid string, but getPassword returns an empty string, the challenge keyword is replaced as the password by the database. If the user enters a valid password, a challenge may or may not be returned by the RADIUS server.
`radiusChallenge`	Presents a request sent from the RADIUS server for the user to respond to the server's challenge.
`getResponse`	Extracts the response the user enters. If this method returns a valid response, that information then populates the User-Password attribute in the new Access-Request packet. If an empty string is returned, the operation is aborted from both sides by returning the corresponding value.

D

Oracle Advanced Security FIPS 140-1 Settings

Oracle Advanced Security Release 8.1.6 has been validated under **Federal Information Processing Standard (FIPS)** 140-1 at the Level 2 security level. This appendix describes the formal configuration required for Oracle Advanced Security to comply with the FIPS 140-1 standard. Refer to the NIST Cryptographic Modules Validation list at the following Web site address:

`http://csrc.nist.gov/cryptval/140-1/1401val.htm`

This appendix contains the following topics:

- Configuration Parameters
- Post Installation Checks
- Status Information
- Physical Security

Note: The information contained in this appendix should be used with the information provided in Appendix A, "Data Encryption and Integrity Parameters".

Configuration Parameters

This appendix contains information on the Oracle Advanced Security parameters required in the `sqlnet.ora` files that ensure that any connections created between a client and server are encrypted under the control of the server.

Configuration parameters are contained in the `sqlnet.ora` file that is held locally for each of the client and server processes. The protection placed on these files should be equivalent to the level of a DBA.

The following configuration parameters are described in this appendix:

- `ENCRYPTION_SERVER`
- `ENCRYPTION_CLIENT`
- `ENCRYPTION_TYPES_SERVER`
- `CRYPTO_SEED`
- `CRYPTO_SEED_CLIENT`
- `FIPS_140`

Server Encryption Level Setting

The server side of the negotiation notionally controls the connection settings. The following parameter in the server file is mandatory:

```
SQLNET.ENCRYPTION_SERVER=REQUIRED
```

Setting the encryption as REQUIRED on the server side of the connection ensures that a connection is only permitted if encryption is used, irrespective of the parameter value on the client.

Client Encryption Level Setting

The `ENCRYPTION_CLIENT` parameter specifies the connection behavior for the client. One of the following parameter settings in the client file is mandatory:

```
SQLNET.ENCRYPTION_CLIENT=(ACCEPTED|REQUESTED|REQUIRED)
```

A connection to the server is only possible if there is agreement between client and server for the connection encryption. The server has this set to REQUIRED, therefore the client must not reject encryption for a valid connection to be the result. Failure to specify one of these values results in error when attempting to connect to a FIPS 140-1 compliant server.

Server Encryption Selection List

The `ENCRYPTION_TYPES_SERVER` parameter specifies a list of encryption algorithms that the server is permitted to use when acting as a server in the order of required usage.

The specified algorithm must be installed or the connection terminates. For FIPS 140-1 compliance, only DES encryption is permitted and therefore the following parameter setting is mandatory:

```
SQLNET.ENCRYPTION_TYPES_SERVER=(DES|DES40)
```

Client Encryption Selection List

The ENCRYPTION_TYPES_CLIENT parameter specifies the list of encryption algorithms which the client is prepared to use for the connection with the server. In order for a connection to be successful, the algorithm must first be installed and the encryption type must be mutually acceptable to the server.

To create a connection with a server that is configured for FIPS 140-1, the following parameter setting is mandatory:

```
SQLNET.ENCRYPTION_TYPES_CLIENT=(DES|DES40)
```

Cryptographic Seed Value

The CRYPTO_SEED parameter contains characters which are part of the seed for the random number generator. There are no explicit requirements for the value of this parameter within the FIPS 140-1 standard, however it is suggested that a large set of random characters, up to 70, is chosen as follows:

```
SQLNET.CRYPTO_SEED=10_to_70_random_characters
```

FIPS Parameter

The default setting of the FIPS_140 parameter is FALSE. Setting the parameter to TRUE is mandatory for both client and server to ensure Oracle Advanced Security complies with the standards defined in FIPS 140-1 as follows:

```
SQLNET.FIPS_140=TRUE
```

Note: Use a text editor to set the FIPS_140 parameter in the sqlnet.ora file. You cannot use Oracle Net Manager to set this parameter.

Post Installation Checks

After the installation, the following permissions must be verified in the operating system:

- Execute permissions must be set on all Oracle Advanced Security executable files so as to prevent execution of Oracle Advanced Security by users who are unauthorized to do so in accordance with the system security policy.
- Read and write permissions must be set on all executable files so as to prevent accidental or deliberate reading or modification of Oracle Advanced Security files by any user.

To comply with FIPS 140-1 Level 2 requirements, the security policy must include procedures to prevent unauthorized users from reading or modifying executing Oracle Advanced Security processes and the memory they are using in the operating system.

Status Information

Status information for Oracle Advanced Security is available after the connection has been established. The information is contained in the RDBMS virtual table v$session_connect_info.

Running the query `SELECT * from V$SESSION_CONNECT_INFO` displays all of the product banner information for the active connection. Table D–1 shows an example of a connection configuration where both DES encryption and MD5 data integrity is defined:

Table D–1 Sample Output from v$session_connect_info

SID	AUTHENTICATION	OSUSER	NETWORK_SERVICE_BANNER
7	DATABASE	oracle	Oracle Bequeath operating system adapter for Solaris, v8.1.6.0.0
7	DATABASE	oracle	Oracle Advanced Security: encryption service for Solaris
7	DATABASE	oracle	Oracle Advanced Security: DES encryption service adapter
7	DATABASE	oracle	Oracle Advanced Security: crypto-checksumming service
7	DATABASE	oracle	Oracle Advanced Security: MD5 crypto-checksumming service adapter.

Physical Security

To comply with FIPS 140-1 Level 2 requirements, tamper-evident seals must be applied to the cover of each machine—to ensure that removal of the cover is detectable.

E

orapki Utility

The `orapki` utility is provided to manage public key infrastructure (PKI) elements, such as wallets and certificate revocation lists, on the command line so the tasks it performs can be incorporated into scripts. Providing a way to incorporate the management of PKI elements into scripts makes it possible to automate many of the routine tasks of maintaining a PKI.

The following topics are included in this appendix:

orapki Utility Overview

This command line utility can be used to perform the following tasks:

- Creating and viewing 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 certificate revocation lists (CRLs):
  - Renaming CRLs with a hash value for certificate validation
  - Uploading, listing, viewing, and deleting CRLs in Oracle Internet Directory

orapki Utility Syntax

The basic syntax of the `orapki` command line utility is as follows:

```
orapki module command -parameter <value>
```

where *`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 with 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
```

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:

```
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:

```
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`, the command will display the certificate and its expiration date. If you choose `-complete`, it will display additional certificate information, including the serial number and public key.

Managing Oracle Wallets with 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.

- Creating and Viewing Oracle Wallets with orapki
- Adding Certificates and Certificate Requests to Oracle Wallets with orapki
- Exporting Certificates and Certificate Requests from Oracle Wallets with orapki

---

Note: The `-wallet` parameter is manadatory for all `wallet` module commands.

Creating and Viewing Oracle Wallets with orapki

To create an Oracle wallet:

```
orapki wallet create -wallet <wallet_location>
```

This command will prompt you to enter and re-enter a wallet password. It creates a wallet in the location specified for `-wallet`.

To create an Oracle wallet with auto login enabled:

```
orapki wallet create -wallet <wallet_location> -auto_login
```

This command creates a wallet with auto login enabled, or it can also be used to enable auto login on an existing wallet. If the *`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. See "Using Auto Login" on page 8-19 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:

```
orapki wallet display -wallet <wallet_location>
```

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:

```
orapki wallet add -wallet <wallet_location> -dn <user_dn> -keySize
<512|1024|2048>
```

This command adds a certificate request to a wallet for the user with the specified distinguished name (*user_dn*). The request also specifies the requested certificate's key size (512, 1024, or 2048 bits). To sign the request, export it with the export option. See "Exporting Certificates and Certificate Requests from Oracle Wallets with orapki" on page E-6

To add a trusted certificate to an Oracle wallet:

```
orapki wallet add -wallet <wallet_location> -trusted_cert -cert <certificate_
location>
```

This command adds a trusted certificate, at the specified location (`-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

```
orapki wallet add -wallet <wallet_location> -dn <certificate_dn> -keySize
<512|1024|2048> -self_signed -validity <number_of_days>
```

This command creates a new self-signed (root) certificate and adds it to the wallet. The `-validity` parameter (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 (`-keySize`) of 512, 1024, or 2048 bits.

To add a user certificate to an Oracle wallet:

```
orapki wallet add -wallet <wallet_location> -user_cert -cert <certificate_
location>
```

This command adds the user certificate at the location specified with the `-cert` parameter to the Oracle wallet at the `<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 will fail.

Exporting Certificates and Certificate Requests from Oracle Wallets with orapki

To export a certificate from an Oracle wallet:

```
orapki wallet export -wallet <wallet_location> -dn <certificate_dn> -cert
<certificate_filename>
```

This command exports a certificate with the subject's distinguished name (`-dn`) from a wallet to a file that is specified by `-cert`.

To export a certificate request from an Oracle wallet:

```
orapki wallet export -wallet <wallet_location> -dn <certificate_request_dn>
-request <certificate_request_filename>
```

This command exports a certificate request with the subject's distinguished name (`-dn`) from a wallet to a file that is specified by `-request`.

Managing Certificate Revocation Lists (CRLs) with orapki Utility

CRLs must be managed with `orapki`. This utility creates a hashed value of the CRL issuer's name to identify the CRLs location in your system. If you do not use `orapki`, your Oracle server cannot locate CRLs to validate PKI digital certificates. For detailed information about using `orapki` to manage CRLs refer to "Certificate Revocation List Management" on page 7-40.

orapki Utility Commands Summary

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

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 where 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

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

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.

Prerequisites

None

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 certificate authority (CA) who issued the CRL.
- The `-ldap` parameter specifies the hostname and SSL port for the directory where the CRLs are to be deleted. Note that this must be a directory SSL port

with no authentication. See "Uploading CRLs to Oracle Internet Directory" on page 7-42 for more information about this port.

- The `-user` parameter specifies the username 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

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. See "orapki crl list" on page E-10
- 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.
- Choosing 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

Purpose

Use this command to generate a hash value of the certificate revocation list (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 where 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
  - (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

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 hostname and SSL port for the directory server from where you want to list CRLs. Note that this must be a directory SSL port with no authentication. See "Uploading CRLs to Oracle Internet Directory" on page 7-42 for more information about this port.

orapki crl upload

Purpose

Use this command to upload certificate revocation lists (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 where the CRL is located that you are uploading to the directory.
- The `-ldap` parameter specifies the hostname and SSL port for the directory where you are uploading the CRLs. Note that this must be a directory SSL port with no authentication. See "Uploading CRLs to Oracle Internet Directory" on page 7-42 for more information about this port.
- The `-user` parameter specifies the username 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 certificate authority (CA) who issued the CRL. This is an optional parameter. Using it causes the tool to verify the validity of the CRL against the CA's 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 where the CRL is stored in the directory.

orapki wallet add

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. See "orapki wallet export" on page E-13

To add trusted certificates:

```
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:

```
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 it 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

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. See "Using Auto Login" on page 8-19 for details about auto login wallets.

orapki wallet display

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

Purpose

Use this command to export certificate requests and certificates from an Oracle wallet.

Syntax

To export a certificate from an Oracle wallet:

```
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.

F

Entrust-Enabled SSL Authentication

Entrust Authority (formerly known as Entrust/PKI) is a suite of PKI products provided by Entrust, Inc., that provides certificate generation, certificate revocation, and key and certificate management. Oracle Advanced Security is integrated with Entrust Authority so both Entrust and Oracle users can enhance their Oracle environment security.

This appendix contains the following topics:

Benefits of Entrust-Enabled Oracle Advanced Security

Entrust-enabled Oracle Advanced Security provides:

- Enhanced X.509-Based Authentication and Single Sign-On
- Integration with Entrust Authority Key Management
- Integration with Entrust Authority Certificate Revocation

Note:

- Oracle Advanced Security has been certified as *Entrust-Ready* by Entrust, Inc., as of Release 8.1.7.
- See Also: `http://www.entrust.com`

Enhanced X.509-Based Authentication and Single Sign-On

Entrust-enabled Oracle Advanced Security supports the use of Entrust credentials for **X.509**-based authentication and single sign-on. Instead of using an Oracle wallet to hold user PKI credentials, Oracle Advanced Security can access PKI credentials that are created by Entrust Authority and held in an Entrust profile (a `.epf` file). Users who have deployed Entrust software within their enterprise are thus able to use it for authentication and single sign-on to Oracle Database.

Integration with Entrust Authority Key Management

Entrust-enabled Oracle Advanced Security uses the extensive key management and rollover functionality provided by Entrust Authority, which shields users from the complexity of a PKI deployment. For example, users are automatically notified when their certificates are expiring, and certificates are reissued according to preferences that administrators can configure.

Integration with Entrust Authority Certificate Revocation

Entrust provides a certificate authority component, which natively checks certificate revocation status and enables the revocation of certificates.

Users using Entrust credentials for authentication to Oracle are assured that the revocation status of the certificate is checked, and connections are prevented if the certificate is revoked.

Required System Components for Entrust-Enabled Oracle Advanced Security

To implement Entrust-enabled Oracle Advanced Security, the following system components are required:

- Entrust Authority for Oracle
- Entrust Authority Server Login Feature
- Entrust Authority IPSec Negotiator Toolkit

Note: In the following sections, the term **client** refers to a client connecting to an Oracle database, and the term **server** refers to the host on which the Oracle database resides.

Contact your Entrust representative to get these components.

Note: Oracle Advanced Security supports Entrust Authority Security Manager, Entrust Authority Server Login Feature, and Entrust Authority IPSec Negotiator Toolkit versions 6.0 and later.

Contact your Entrust representative for the latest product classification and naming details.

Entrust Authority for Oracle

Entrust Authority for Oracle requires a database for storing information about Entrust users and the infrastructure, and a Lightweight Directory Access Protocol (LDAP)-compliant directory for information such as user names, public certificates, and certificate revocation lists.

Entrust Authority for Oracle is comprised of the following software components:

- Entrust Authority Security Manager
- Entrust Authority Self-Administration Server
- Entrust Entelligence Desktop Manager

Entrust Authority Security Manager

Entrust Authority Security Manager is the centerpiece of Entrust's PKI technology. It performs core certificate authority, certificate, and user management functions, such as creating users and user profiles containing the user's credentials.

Note: Oracle only supports the use of Entrust-enabled Oracle Advanced Security with versions of Entrust Authority Security Manager that run on Oracle Database.

See Also: Chapter 7, "Configuring Secure Sockets Layer Authentication", for information about certificate authorities.

Entrust Authority Security Manager supports unattended login, also called Server Login, which eliminates the need for a **Database Administrator** (DBA) to repeatedly enter a password for the Entrust profile on the server. With unattended login, the DBA need only enter a password once to open the Entrust profile for the server to authenticate itself to multiple incoming connections.

Entrust Authority Self-Administration Server

Entrust Authority Self-Administration Server is the administrator's secure interface to Entrust Authority Security Manager.

Entrust Entelligence Desktop Manager

Entrust Entelligence Desktop Manager provides support for user key management and single sign-on functionality on both clients and server by enabling Oracle Database server process access to incoming SSL connections.

Note: Do not install Entrust Entelligence Desktop Manager on the server computer because it uses unattended login credentials files with `.ual` extensions. See "Configuring Entrust on the Server" on page F-9 for information about creating `.ual` files.

Entrust Authority Server Login Feature

Entrust Authority Server Login Feature is required for single sign-on functionality on servers operating on UNIX platforms.

Entrust Authority Server Login Feature provides single sign-on by enabling Oracle Database server process access to incoming SSL connections. Without this capability, a database administrator or other privileged user would have to enter the password for the Entrust profile on the server for every incoming connection.

Contact your Entrust representative to get Entrust Authority Server Login Feature.

Entrust Authority IPSec Negotiator Toolkit

The Entrust Authority IPSec Negotiator Toolkit is required on both clients and servers for integrating the Oracle Advanced Security SSL stack with Entrust Authority, enabling SSL authentication to use Entrust profiles.

Contact your Entrust representative to get Entrust Authority IPSec Negotiator Toolkit.

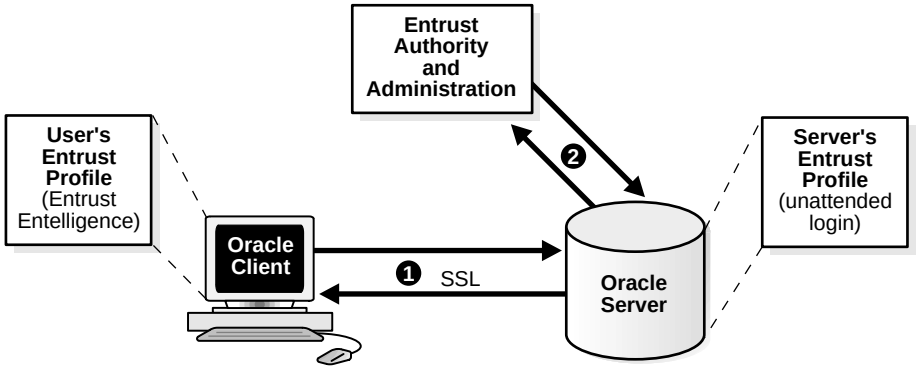
Entrust Authentication Process

Figure F–1 illustrates the following Entrust authentication process:

1. The Entrust user on the Oracle client establishes **a secure connection with the server using SSL and Entrust credentials.**
2. The Oracle SSL adapter on the server communicates with the Entrust Authority to check the certificate revocation status of the Entrust user.

Note: Figure F–1 does not include client and server profiles creation, which is presumed.

Figure F–1 Entrust Authentication Process



See Also: "How SSL Works in an Oracle Environment: The SSL Handshake" on page 7-4

Enabling Entrust Authentication

This section describes the following tasks, which are required to configure Entrust-enabled Oracle Advanced Security SSL authentication:

- Creating Entrust Profiles
- Installing Oracle Advanced Security and Related Products for Entrust-Enabled SSL
- Configuring SSL on the Client and Server for Entrust-Enabled SSL
- Configuring Entrust on the Client
- Configuring Entrust on the Server
- Creating Entrust-Enabled Database Users
- Logging Into the Database Using Entrust-Enabled SSL

Creating Entrust Profiles

This section describes how to create Entrust profiles, which can be created by either administrators or users. On UNIX platforms, administrators create the Entrust profiles for all clients. On Windows platforms, users can create their own Entrust profiles.

Administrator-Created Entrust Profiles

Administrators create Entrust profiles as follows:

1. The Entrust administrator adds the Entrust user using the Entrust Authority Self-Administration Server.

   See Also: The Entrust administration documentation for information about creating Entrust Users

2. The administrator enters the user's name and password.
3. The Entrust Authority creates the profile, or `.epf` file.
4. The administrator securely sends all profile-related files to the user. The preset password can be changed by the user.

User-Created Entrust Profiles

Entrust users create their own Entrust profiles as follows:

1. The Entrust administrator adds the Entrust user using the Entrust Authority Self-Administration Server. In the New User dialog box, the Create Profile option should be deselected.

   See Also: The Entrust administration documentation for information about creating Entrust profiles

2. The user receives a secure e-mail notification from the administrator that contains a reference number, authorization code, and expiration date.
3. The user navigates to the Create Entrust Profiles screen in Entrust Entelligence Desktop Manager as follows:

   Start > Programs > Entrust > Entrust Profiles > Create Entrust Profiles

4. The user enters the reference number, authorization code, and expiration date provided in the e-mail notification, creating a profile, or `.epf` file, and the Entrust initialization file.

Installing Oracle Advanced Security and Related Products for Entrust-Enabled SSL

For Oracle Advanced Security 10*g* Release 1 (10.1), Entrust support installs in Typical mode. A single Oracle installation supports the use of both Oracle Wallets and Entrust profiles.

> **See Also:** Oracle Database operating system-specific installation documentation

Configuring SSL on the Client and Server for Entrust-Enabled SSL

Configure SSL on the client and server.

> **See Also:** Chapter 7, "Configuring Secure Sockets Layer Authentication", for information about configuring SSL on the client and server; skip the section that describes the Oracle wallet location.

Configuring Entrust on the Client

The steps for configuring Entrust on the client vary according to the type of platform:

- Configuring Entrust on a UNIX Client
- Configuring Entrust on a Windows Client

Configuring Entrust on a UNIX Client

If the client resides on a non-Windows platform, perform the following steps:

1. Set the JAVA_HOME variable to the JDK or JRE location.

   For example:

   ```
   >setenv JAVA_HOME $ORACLE_HOME/JRE
   ```

2. Set `WALLET_LOCATION` in the `sqlnet.ora` file.

   For example:

   ```
   WALLET_LOCATION=
       (SOURCE=
           (METHOD=entr)
           (METHOD_DATA =
               (PROFILE=profile_location)
               (INIFILE=initialization_file_location)
   ```

```
        )
    )
```

Configuring Entrust on a Windows Client

If the client resides on a Windows platform, ensure that the Entrust Entelligence Desktop Manager component is installed on the client and perform the following steps to set up the Entrust credentials.

1. Set the `WALLET_LOCATION` parameter in the `sqlnet.ora` file.

   For example:

```
WALLET_LOCATION=
    (SOURCE=
        (METHOD=entr)
        (METHOD_DATA=
            (INIFILE=initialization_file_location)
        )
    )
```

   where *`initialization_file_location`* is the path to the `.ini` file.

2. Choose the Entrust icon on the system tray to open the Entrust_Login dialog box.
3. Log on to Entrust by entering the profile name and password.

Configuring Entrust on the Server

The steps for configuring Entrust on the server vary according to the type of platform:

- Configuring Entrust on a UNIX Server
- Configuring Entrust on a Windows Server

Configuring Entrust on a UNIX Server

If the server is a UNIX platform, ensure that the Entrust/Server Login Toolkit component is installed on the server and perform the following steps:

> **See Also:** "Required System Components for Entrust-Enabled Oracle Advanced Security" on page F-3 for information about downloading the Entrust Server Login toolkit.

1. Stop the Oracle database instance.

2. Set the WALLET_LOCATION parameter in the sqlnet.ora and listener.ora files to specify the paths to the server's profile and the Entrust initialization file:

```
WALLET_LOCATION =
  (SOURCE =
      (METHOD = ENTR)
      (METHOD_DATA =
          (PROFILE = profile_location)
          (INIFILE = initialization_file_location)
      )
  )
```

3. Set the CLASSPATH environment variable to include the following paths:

```
$ORACLE_HOME/JRE/lib/rt.jar
$ORACLE_HOME/JRE/lib/i18n.jar
$ORACLE_HOME/jlib/ewt*.jar
$ORACLE_HOME/jlib/help*.jar
$ORACLE_HOME/jlib/share*.jar
$ORACLE_HOME/jlib/swingall*.jar
$ORACLE_HOME/network/jlib/netentrust.jar
```

4. Enter the etbinder command to create unattended login credentials, or .ual files by using the following steps:

   a. Set the PATH environment variable to include the path to the etbinder command, which is located in the /bin directory where the Server Login Toolkit is installed.

   b. Set the LD_LIBRARY_PATH to include the path to the Entrust libraries.

   c. Set the SSL_ENTRUST_INI environment variable to include the full path to the Entrust initialization file.

   d. Enter the command as follows:

```
etbinder
```

   e. When prompted to enter the location of the profile file, enter the full path name, including the name of the file. Then, when prompted, type in the password.

      A message displays indicating that the credentials file (*filename*.ual) has been created.

Note: Ensure that the listener has a TCPS listening endpoint, then start the listener.

5. Start the Oracle database instance.

Configuring Entrust on a Windows Server

If the server is on a Windows platform, perform the following steps:

See Also: "Required System Components for Entrust-Enabled Oracle Advanced Security" for information about downloading Entrust Entelligence Desktop Manager.

1. Stop the Oracle database instance.
2. Set the `WALLET_LOCATION` parameter in the `sqlnet.ora` and `listener.ora` files to specify the paths to the server's profile and the Entrust initialization file:

```
WALLET_LOCATION =
    (SOURCE =
        (METHOD = ENTR)
        (METHOD_DATA =
            (PROFILE = profile_location)
            (INIFILE = initialization_file_location)
        )
    )
```

3. Run the Entrust binder command to create unattended login credentials, which are files with a `.ual` extension. Ensure that the owner of the `.ual` file is the same as the owner of the Oracle service.

   To run the binder command choose

   Start > Programs > Entrust Toolkit > Server Login > Entrust Binder

   Enter the path to the profile, the password, and the path to the Entrust initialization file. A message informs you that you have successfully created a credential file.

4. Start the Oracle database instance.

> **Note:** For all Windows environments, Oracle Corporation recommends that you do not install Entrust Entelligence Desktop Manager on the server computer.

Creating Entrust-Enabled Database Users

Create global users in the database based on the **distinguished name (DN)** of each Entrust user.

For example:

```
SQL> create user jdoe identified globally as 'cn=jdoe,o=oracle,c=us';
```

where `"cn=jdoe, o=oracle, c=us"` is the Entrust distinguished name of the user.

Logging Into the Database Using Entrust-Enabled SSL

1. Use SQL*Plus to connect to the Oracle instance as follows:

   ```
   sqlplus /@net_service_name
   ```

   where `net_service_name` is the service name of the Oracle instance.

   The Entrust_Login dialog box appears.

2. Enter the path to the profile and the password.
3. If you did not specify a value for the `WALLET_LOCATION` parameter, you are prompted to enter the path to the Entrust initialization file.

> **Note:** Oracle Corporation recommends that the initialization file be specified in the `WALLET_LOCATION` parameter file.

Issues and Restrictions that Apply to Entrust-Enabled SSL

An application must be specifically modified to work with Entrust. If a product is designated as Entrust-ready, then it has been integrated with Entrust by using an Entrust toolkit.

For example, Oracle has modified its SSL libraries to access an Entrust profile instead of an Oracle wallet.

In addition, the following restrictions apply:

- The use of Entrust components for digital signatures in applications based on Oracle is not supported.
- The Entrust-enabled Oracle Advanced Security integration is only supported with versions of Entrust Authority Release 6.0 and later running on Oracle Database.
- The use of earlier releases of Entrust Authority with Entrust-enabled Oracle Advanced Security is not supported.
- Interoperability between Entrust and non-Entrust PKIs is not supported.
- Entrust has certified Oracle Internet Directory version 2.1.1 for Release 8.1.7 and subsequent releases.

Troubleshooting Entrust In Oracle Advanced Security

This section describes how to diagnose errors returned from Entrust to Oracle Advanced Security users.

Note: Entrust returns the following generic error message to Oracle Advanced Security users:

```
ORA-28890 "Entrust Login Failed"
```

This troubleshooting section describes how to get more details about the underlying error, and how to diagnose the problem.

Error Messages Returned When Running Entrust on Any Platform

You may encounter the following error messages regardless of what platform you are running Entrust on.

ORA-28890 Entrust Login Failed

Cause: SQL*Plus login on an Entrust-enabled Oracle client errors out with this generic error message. This error can be caused by a number of problems, including the following causes:

- Entrust / Authority is not online
- Invalid Entrust profile password specified
- Invalid path to the Entrust profile specified

- Invalid Entrust initialization file specified
- Entrust Server Login program has not executed on the server

Action: To get more detail on the Entrust error, turn on tracing for SQL*Plus and the trace output should indicate the Entrust failure code. Enable tracing by specifying the following parameters in the `sqlnet.ora` file:

On the client:

- `TRACE_LEVEL_CLIENT=16`
- `TRACE_DIRECTORY_CLIENT=<`*valid_client_directory_name*`>`
- `TRACE_FILE_CLIENT=client`
- `TRACE_UNIQUE_CLIENT=ON`

On the server:

- `TRACE_LEVEL_SERVER=16`
- `TRACE_DIRECTORY_SERVER=<`*valid_server_directory name*`>`
- `TRACE_FILE_SERVER=server`
- `TRACE_UNIQUE_SERVER=ON`

Search for and locate the string `IKMP` in the generated trace file. Adjacent to this string, error messages are listed that provide details about the problem you are encountering. This detailed error code information is returned by the Entrust API.

Note: The following are examples of valid client directory names for setting the `TRACE_DIRECTORY_CLIENT` or `TRACE_DIRECTORY_SERVER` parameters in the `sqlnet.ora` file:

- (UNIX) `/tmp`
- (Windows) `C:\TEMP`

ORA-28890 Entrust Login Failed
(GUI does not display on the client)

Cause: The `WALLET_LOCATION` parameter does not specify the Entrust initialization file location in the client side `sqlnet.ora` file.

Action: Ensure that the location of the Entrust initialization file is specified in the `WALLET_LOCATION` parameter in the `sqlnet.ora` file on the client.

See Also:

- "Configuring Entrust on a UNIX Client" on page F-8
- "Configuring Entrust on a Windows Client" on page F-9

Error Messages Returned When Running Entrust on Windows Platforms

You may encounter the following error messages if you are running Entrust on a Windows platform.

The software authentication failed. (error code - 162).

Cause: Due to a known FIPS mode incompatibility, Entrust logins may fail and return this error message.

Action: Contact Entrust support to resolve this issue.

Algorithm self-test failed. (error code - 176).

Cause: Due to a known symbol conflict between Entrust and Oracle libraries, Entrust login may fail and return this error message.

Action: Contact Entrust support to resolve this issue.

TNS-12560: TNS protocol adapter error

TNS-00558> Entrust Login Failed

ORACLE SERVER (*host_name*)

This error may occur in the `listener.log` *file on the server when you attempt to log in to Entrust.*

Cause: If you configure the client by making the following recommended changes:

- Remove the `.ual` file
- De-install the Server Login
- Specify the Entrust initialization file location in the `SSL_ENTRUST_INI_FILE` parameter in the client `sqlnet.ora` file

then the server may not be able to authenticate the client when you enter the following command:

```
sqlplus/@net_service_name
```

Action: Perform the following tasks to enable tracing on the server:

1. Choose Control Panel > Services.
2. In the Services dialog box, double click OracleTNSListener and change the Log On As from the System Account to the account that is currently logged in. This enables the server process to read the `.ual` file. Click OK to make the change and you are returned to the Services dialog box.

   In the Services dialog box, make the same changes for OracleService.
3. Make the following changes to the `listener.ora` file:
   - Specify only TCPS as the PROTOCOL in the listener ADDRESS. For example, change all of the PROTOCOL definitions to TCPS as follows:

     ```
     listener_name=
        (DESCRIPTION=
           (ADDRESS=(PROTOCOL=TCPS) (KEY=extproc0))
           (ADDRESS=(PROTOCOL=TCPS) (HOST=sales-pc) (PORT=1521)))
     ```

     Bringing up the listener only using TCPS will show whether there is a problem accessing the Entrust profile when you turn on tracing.
   - Set the SSL_CLIENT_AUTHENTICATION parameter to FALSE as follows:

     ```
     SSL_CLIENT_AUTHENTICATION=FALSE
     ```

   - Turn on tracing by setting the following parameters:

     ```
     TRACE_LEVEL_LISTENER=16
     TRACE_DIRECTORY_LISTENER=C:\temp
     ```

     The trace file is created in the `C:\temp` directory.
4. Make the following changes to the sqlnet.ora file to turn on tracing:

   ```
   TRACE_LEVEL_SERVER=16
   TRACE_DIRECTORY_SERVER=C:\temp
   ```

   The trace file is created in the `C:\temp` directory.
5. Ensure that Entrust Entelligence Desktop Manager is not installed on the server.

Search for and locate the string "`fail`" or "`ntz*`" function calls. Adjacent to these, error messages are listed that provide details about the problem you are encountering.

General Checklist for Running Entrust on Any Platform

The following items apply to all platforms:

1. Confirm that the Entrust Authority is online.
2. Confirm that the `.ual` file is generated. These files are created for unattended login credentials.

   Note: Oracle recommends that you generate an unattended login credential file (`.ual` file) for the server only. If you generate a `.ual` file for the server only, then when users attempt to log in, they are presented a GUI that prompts them for their password and their Entrust profile name. After users supply this information, the connection request is forwarded to the Entrust server, which looks up the revocation file and the `.ual` file to determine the permissions for granting the request.

3. Confirm that the Entrust initialization file contains the following entry in the first section that specifies the Entrust Settings:

   ```
   IdentityLibrary=location
   ```

   The full path to the location of the `libidapi.so` file should be specified in the `IdentityLibrary` parameter. This parameter setting enables generating a `.ual` file on the server.

4. Ensure that all Entrust toolkits, including the Entrust IPSEC Negotiator toolkit and the Server Login toolkit, are the same version so they are compatible.
5. Ensure that you have specified TCP/IP with SSL in the `SQLNET.AUTHENTICATION_SERVICES` parameter in the `sqlnet.ora` file as shown in the following example:

   ```
   SQLNET.AUTHENTICATION_SERVICES=(tcps, authentication_type1, authentication_
      type2)
   ```

Checklist for Entrust Installations on Windows

The following checklist items apply only to Entrust installations on the Windows platform.

1. Ensure that you are logged into Entrust Entelligence Desktop Manager and retry.
2. Choose Windows > Control Panel > Services to confirm that the Entrust Login Interface service has started and is running.
3. Confirm that the Entrust initialization file location is specified in the `SSL_ENTRUST_INI_FILE` parameter of the `sqlnet.ora` file. However, if you choose not to specify the location there, then the Entrust initialization file must reside in `c:\WINNT`.
4. Ensure that you are not running Entrust Entelligence Desktop Manager if your database is running on a Microsoft platform. If this is the case, then only the `.ual` file, which enables unattended login, is required.

   > **See Also:** Step 4 of "Configuring Entrust on a Windows Server" on page F-11 for information about creating a `.ual` file with the Entrust binder command.

5. Confirm that Entrust Authority, as specified in the Entrust Initialization file, is accessible and running.
6. Confirm that the profile password is correctly entered.
7. If an Oracle database server fails to log in to Entrust, confirm that the unattended login credential file (`.ual`) is generated using a valid password. Also, confirm that the versions for Entrust Server Login toolkit and Entrust IPSEC Negotiator toolkit match (that is, that the IPSec Toolkit 6.0 works with Server Login Toolkit 6.0).
8. Ensure that the Entrust initialization file has the following entry in the first section, Entrust Settings:

   ```
   IdentityLibrary = location
   ```

   where *`location`* is the location of `libidapi.so`, including the file name.

G

Using the User Migration Utility

This chapter describes the User Migration Utility, which can be used to perform bulk migrations of database users to an LDAP directory where they are stored and managed as enterprise users. It contains the following topics:

Benefits of Migrating Local or External Users to Enterprise Users

Migrating from a database user model to an enterprise user model provides solutions to administrative, security, and usability challenges in an enterprise environment. In an enterprise user model, all user information is moved to an LDAP directory service.

Enterprise user security provides the ability to easily and securely manage enterprise-wide users by providing the following benefits:

- Centralized storage of user credentials, roles, and privileges in an LDAP version 3-compliant directory server

- Provides the infrastructure to enable single sign-on using X.509v3-compliant certificates, which is typically deployed where end-to-end SSL is required
- Enhanced security

  Because an enterprise user model is easier to manage, security administrators can perform necessary maintenance changes to user information immediately so they have better control over access to critical network resources. In addition, an enterprise user model is easier for users to use because they have fewer passwords to remember so they are less likely to choose easily guessed passwords or write them down where others can copy them.

  See Also: "Introduction to Enterprise User Security" on page 11-2 for detailed conceptual information about enterprise user security.

Introduction to the User Migration Utility

The User Migration Utility is a command-line utility that is used when enterprise user administrators decide to move their users from a local database model to an enterprise user model. This utility makes it easy to migrate thousands of local and external database users to an enterprise user environment in an LDAP directory where they can be managed from a central location. It uses the Oracle JDBC OCI driver to connect to the database.

Enterprise user administrators can select for migration any combination of the following user subsets in a database:

- List of users specified on the command line or in a file
- All external users
- All global users

In addition, enterprise user administrators can specify values for utility parameters that determine how the users are migrated such as

- Where to put the migrated users in the LDAP directory tree
- Map a user with multiple accounts on various databases to a single directory user entry

The following sections explain the migration process and the changes that occur to users' schemas.

Note: After external users are migrated, their external authentication and authorization mechanisms are replaced by directory-based mechanisms. New passwords are randomly generated for migrated users if they are mapped to newly created directory entries.

Bulk User Migration Process Overview

Bulk user migration is a two-phase process. In phase one, you start the migration process by populating user information into an interface database table, where enterprise user administrators can verify that the information is accurate before committing the changes to the database and the directory in phase two. The process is described in the following steps:

- Step 1: Phase One Preparing for the Migration
- Step 2: Verify User Information
- Step 3: Phase Two Completing the Migration

Step 1: Phase One Preparing for the Migration

In the first part of the migration process, the utility checks if the `ORCL_GLOBAL_USR_MIGRATION_DATA` interface table exists in the enterprise user administrator's schema. If it exists, then the administrator can choose to reuse the table (clearing its contents), reuse the table and its contents, or re-create the table. Phase one can be run multiple times, each time adding to the interface table. If the table does not exist, then the utility creates it in the administrator's schema. The interface table is populated with information about the migrating users from the database and the directory. The command line options used determine what information populates this table.

Note: The utility will not create the interface table in the SYS schema.

Step 2: Verify User Information

This is an intermediate step to allow the enterprise user administrator to verify that the user information is correct in the interface table before committing the changes to the database and the directory.

Step 3: Phase Two Completing the Migration

After the interface table user information is checked, then in phase two the utility retrieves the information from the table and updates the directory and the database.

Depending on whether directory entries exist for migrating users, the utility creates random passwords as follows:

- If migrating users are being mapped to newly created directory entries, then the utility generates random passwords, which are used as credentials for both the database and directory.
- If migrating users are being mapped to existing directory entries with unset database passwords, then the utility generates random database passwords only.

In either case, after generating the required random passwords, the utility then stores them in the `DBPASSWORD` and `DIRPASSWORD` interface table columns. The enterprise user administrator can read these passwords from the interface table and inform migrating users.

> **See Also:** "User Migration Utility Parameters" on page G-12 for a list of command line options and their descriptions.

About the ORCL_GLOBAL_USR_MIGRATION_DATA Table

This is the interface table which is populated with information about the migrating users during phase one of the bulk user migration process. The information that populates this table is pulled from the database and checked against existing entries in the directory. If there is corresponding information in the directory, then that is marked in the table for that user. After enterprise user administrators verify the information in this table, changes are made to the directory and the database in phase two.

> **Caution:** The `ORCL_GLOBAL_USR_MIGRATION_DATA` interface table contains very sensitive information. Access to it should be tightly controlled using database privileges.

The table columns are listed in Table G–1.

Table G–1 ORCL_GLOBAL_USR_MIGRATION_DATA Table Schema

Column Name	DataType	Null	Description
USERNAME (Primary Key)	VARCHAR2(30)	NOT NULL	Database user name.
OLD_SCHEMA_TYPE	VARCHAR2(10)	-	Old schema type in the database before migration.
PASSWORD_VERIFIER	VARCHAR2(30)	-	Not used
USERDN	VARCHAR2(4000)	-	Distinguished Name (DN) of the user in the directory (new or existing).
USERDN_EXIST_FLAG	CHAR(1)	-	Flag indicating whether the DN already exists in the directory.
SHARED_SCHEMA	VARCHAR2(30)	-	Shared schema name, if users are to be mapped to a shared schema during phase two.
MAPPING_TYPE	VARCHAR2(10)	-	Mapping type (database or domain).
MAPPING_LEVEL	VARCHAR2(10)	-	Mapping level (entry or subtree).
CASCADE_FLAG	CHAR(1)	-	Cascade flag used when dropping a user (for shared schema mapping only).
DBPASSWORD_EXIST_FLAG	CHAR(1)	-	Flag indicating whether the database password verifier already exists in the directory for this user.
DBPASSWORD	VARCHAR2(30)	-	Randomly generated database password verifiers that are to be stored in the directory.
DIRPASSWORD	VARCHAR2(30)	-	Randomly generated directory password for new entries.
PHASE_COMPLETED	VARCHAR2(10)	-	Information about the phase that has completed successfully.
NEEDS_ATTENTION_FLAG	CHAR(1)	-	Flag indicating whether the row contains abnormalities that require administrator attention.
ATTENTION_DESCRIPTION	VARCHAR2(100)	-	Textual hint for the administrator if the attention flag is set.

Which Interface Table Column Values Can Be Modified between Phase One and Phase Two?

After running phase one of the utility, if necessary, enterprise user administrators can change the interface table columns that are listed in Table G–2.

Table G–2 Interface Table Column Values That Can Be Modified between Phase One and Phase Two

Column Name	Valid Values	Restrictions
USERDN	DN of user	If this value is changed, then the administrator should verify that the USERDN_EXIST_FLAG and the DBPASSWORD_EXIST_FLAG values are set accordingly.
USERDN_EXIST_FLAG	T/F	If the USERDN column value changes, then this column value should also change to reflect the new USERDN status.
DBPASSWORD_EXIST_FLAG	T/F	If the USERDN column value changes, then this column value should also change to reflect whether a database password exists for the new USERDN.
SHARED_SCHEMA	Shared schema name	Specify only if a shared schema exists in the database.
MAPPING_TYPE	DB/DOMAIN	Set this value only if SHARED_SCHEMA is not set to NULL.
MAPPING_LEVEL	ENTRY/SUBTREE	Set this value only if SHARED_SCHEMA is not set to NULL.
CASCADE_FLAG	T/F	Set this value only if SHARED_SCHEMA is not set to NULL. If this column is set to true (T), then the users' schema objects are forcibly deleted. If this column is set to false (F), then the administrator must delete all user schema objects before running phase two.
PHASE_COMPLETED	ZERO/ONE/TWO	If the administrator can resolve the conflicts or ambiguities that are specified with the NEEDS_ATTENTION_FLAG, then this column value can be changed to ONE so phase two can be run with the utility.

Migration Effects on Users' Old Database Schemas

If shared schema mapping is not used, then users retain their old database schemas. If shared schema mapping is used, then users' local schemas are dropped from the database and they are mapped to a shared schema that the enterprise user administrator creates for this purpose before performing the migration. When migrated users own database objects in their old local database schemas, administrators can specify that the schema and objects are not to be dropped by setting the CASCADE parameter to NO. When the CASCADE parameter is set to NO, users who own database objects in their old local schemas do not migrate successfully so their objects are not dropped.

If some users want to retain the objects in their local database schemas and be mapped to a shared schema, then the administrator can manually migrate those objects to the shared schema before performing the bulk user migration. However, when objects are migrated to a shared schema, they are shared among all users who share that new schema.

Table G–3 summarizes the effects of setting the MAPSCHEMA and CASCADE parameters.

Table G–3 Effects of Choosing Shared Schema Mapping with CASCADE Options

MAPSCHEMA Parameter Setting	CASCADE Parameter Setting	User Migration Successful?	User Schema Objects Dropped?
PRIVATE	NO (default setting)	Yes	No
SHARED	NO	Yes[1]	No
SHARED	YES	Yes[2]	Yes

[1] Users migrate successfully only if they do not own objects in their old database schemas; otherwise, they fail.

[2] Users migrate successfully and their old database schemas are dropped.

See Also: "User Migration Utility Parameters" on page G-12 for detailed information about the MAPSCHEMA, CASCADE, and other parameters that can be used with this utility.

Migration Process

Enterprise users, those that are defined and managed in the directory, can be authenticated to the database either with a password or with a certificate. Users that authenticate with a password require an Oracle database password, which is stored in the directory. Users that authenticate with a certificate must have a valid X.509 v3 certificate.

This utility performs the following steps during migration:

1. Selects the users from the database for migration.
2. Creates corresponding user entries or uses existing entries in the directory.
3. Creates new database passwords and copies the corresponding verifiers to the directory for migrating users.
4. Puts the schema mapping information for the migrating users' entries in the directory. (optional)

5. Drops or alters the migrating users' local database schemas. (optional)

Note: In the current release, the utility migrates users with certificate-based authentication and makes them ready for password authentication. Previously SSL-based authenticated users should reset their Oracle database passwords. User wallets are not created as part of this process.

See Also:

- Chapter 8, "Using Oracle Wallet Manager" for information about creating, managing, and using Oracle wallets.

Prerequisites for Performing Migration

The User Migration Utility is automatically installed in the following location when you install Oracle Database Client:

```
$ORACLE_HOME/rdbms/bin/umu
```

The following sections describe what programs must be running and what user privileges are required to successfully migrate users with the User Migration Utility.

Required Database Privileges

To successfully use this utility, enterprise user administrators must have the following database privileges:

- ALTER USER
- DROP USER
- CREATE TABLE
- SELECT_CATALOG_ROLE

These privileges enable the enterprise user administrator to alter users, drop users, look at dictionary views, and create the interface table that is used by this utility.

Required Directory Privileges

In addition to the required database privileges, enterprise user administrators must have the directory privileges which allow them to perform the following tasks:

- Create entries in the directory under the specified user base and Oracle context location
- Browse the user entries under the search bases

Required Setup to Run the User Migration Utility

Perform the following steps before using the User Migration Utility:

1. Ensure that the directory server is running with SSL enabled for no authentication.
2. Ensure that the database server is running with encryption and integrity enabled.
3. Ensure that the database listener has a TCP listening endpoint.
4. Create an identity management realm in the directory, if it does not already exist.
5. Create the parent context for the user entries in the directory, if it does not already exist. The default (and recommended) location is in the `cn=users` subtree in the identity management realm.
6. Set up directory access for the database Oracle home by using Oracle Net Configuration Assistant to create an `ldap.ora` file. Note that the `ldap.ora` file must include the identity management realm DN so the utility can locate the correct administrative context. The utility searches for this file under `$LDAP_ADMIN`, `$ORACLE_HOME/ldap/admin`, `$TNS_ADMIN`, `$ORACLE_HOME/network/admin`, and, finally, the Domain Name System (DNS) server, if you are using DNS discovery. (See *Oracle Internet Directory Administrator's Guide* for information about DNS server discovery.)

Note:

- If you plan to use shared schema mapping when migrating users, then you must create the shared schema before running this utility.
- The same `ldap.ora` file must be used for both phase one and phase two of a user migration.

See Also:

- *Oracle Internet Directory Administrator's Guide*
- Chapter 12, "Enterprise User Security Configuration Tasks and Troubleshooting" for detailed information about setting up enterprise user authentication after the user migration is finished.

User Migration Utility Command Line Syntax

To perform a bulk migration of database users to enterprise users, use the following syntax:

```
umu parameter1 parameter2 ...
```

For parameters that take a single value use the following syntax:

```
keyword=value
```

For parameters that take multiple values, use a colon (:) to separate the values as in the following syntax:

```
keyword=value1:value2:...
```

Example 13–1 shows the syntax used to run the utility through both phases of the bulk user migration process.

Example 13–1 User Migration Utility Command Line Syntax

```
umu PHASE=ONE
    DBADMIN=dba_username:password
    ENTADMIN=enterprise_admin_DN:password
    USERS=[ALL_GLOBAL | ALL_EXTERNAL | LIST | FILE]
    DBLOCATION=database_host:database_port:database_sid
```

```
    DIRLOCATION=ldap_directory_host:ldap_directory_port
    USERSLIST=username1:username2:username3:...
    USERSFILE=filename
    MAPSCHEMA=[PRIVATE | SHARED]:schema_name
    MAPTYPE=[DB | DOMAIN]:[ENTRY | SUBTREE]
    CASCADE=[YES | NO]
    CONTEXT=user_entries_parent_location
    LOGFILE=filename
    PARFILE=filename

umu PHASE=TWO
    DBADMIN=dba_username:password
    ENTADMIN=enterprise_admin_DN:password
    DBLOCATION=database_host:database_port:database_sid
    DIRLOCATION=ldap_directory_host:ldap_directory_port
    LOGFILE=filename
    PARFILE=filename
```

Note: If the enterprise user administrator does not specify the mandatory parameters on the command line, then the utility will prompt the user for those parameters interactively.

See Also:

- "User Migration Utility Parameters" on page G-12 for a complete list of all available parameters and detailed information about them.
- "User Migration Utility Usage Examples" on page G-20 for examples of typical utility uses.

Accessing Help for the User Migration Utility

To display the command-line syntax for using the User Migration Utility, enter the following command at the system prompt:

```
umu HELP=YES
```

While the HELP parameter is set to YES, the utility cannot execute.

User Migration Utility Parameters

The following sections list the available parameter keywords and the values that can be used with them when running this utility. The keywords are not case-sensitive.

Keyword: HELP

Valid Values: YES or NO (These values are not case-sensitive.)

Default Setting: NO

Syntax Examples: HELP=YES

Description: This keyword is used to display help for the utility. YES displays the complete command-line syntax. To execute a command, set the value to NO, or do not specify a value for the parameter to accept the default.

Restrictions: None

Keyword: PHASE

Valid Values: ONE or TWO (These values are not case-sensitive.)

Default Setting: ONE

Syntax Examples: PHASE=ONE

PHASE=TWO

Description: Indicates the phase for the utility. If it is ONE, then the utility populates the interface table with the information specified in the command-line arguments and the existing user entries in the directory. If it is TWO, then the utility uses the information that is available in the interface table and updates the directory and the database.

Restrictions: None

Keyword: DBLOCATION

Valid Values: *host:port:sid*

Default Setting: No default setting.

Syntax Examples: `DBLOCATION=`*my_oracle.us.oracle.com*`:`*7777*`:`*ora902*

Description: Provides the host name, port number, and SID for the database instance.

Restrictions:

- This parameter is mandatory.
- The value for this parameter must be the same for both phase one and phase two.
- The database should be configured for encryption and integrity.

Keyword: DIRLOCATION

Valid Values: *host:port*

Default Setting: This value is automatically populated from the `ldap.ora` file by default.

Syntax Examples: `DIRLOCATION=`*my_oracle.us.oracle.com*`:`*636*

Description: Provides the host name and port number for the directory server where the LDAP server is running on SSL with no authentication.

Restrictions: The value for this parameter must be the same for both phase one and phase two.

Keyword: DBADMIN

Valid Values: *username*`:`*password*

Default Setting: No default setting.

Syntax Examples: `DBADMIN=`*system*`:`*manager*

Description: Username and password for the database administrator with the required privileges for connecting to the database.

Restrictions:

- This parameter is mandatory.
- The `username` value for this parameter must be the same for both phase one and phase two.

Keyword: ENTADMIN

Valid Values: *userDN:password*

Default Setting: No default setting.

Syntax Examples: `ENTADMIN=cn=`*janeadmin*`,dc=`*acme*`,dc=`*com*`:`*welcome*

Description: User Distinguished Name (UserDN) and the directory password for the enterprise directory administrator with the required privileges for logging in to the directory. UserDN can also be specified within double quotation marks ("...").

Restrictions: This parameter is mandatory.

Keyword: USERS

Valid Values: *value1*`:`*value2*`...`

Values can be:

- `ALL_EXTERNAL` to select all external users, including those who use Kerberos and RADIUS authentication
- `ALL_GLOBAL` to select all global users
- `LIST` to specify users on the command line with the "Keyword: USERSLIST"
- `USERSFILE` for selecting users from the file that is specified with the "Keyword: USERSFILE"

This parameter takes multiple values. Separate values with a colon (:).

(These values are not case-sensitive.)

Default Setting: No default setting.

Syntax Examples:

- `USERS=ALL_EXTERNAL:ALL_GLOBAL`

  This usage instructs the utility to migrate all external users and all global users.

- `USERS=ALL_EXTERNAL:FILE`

  This usage instructs the utility to migrate all external users and all users that are specified in the `USERSFILE`.

Description: Specifies which users are to be migrated. If multiple values are specified for this parameter, then the utility uses the union of these sets of users.

Restrictions: This parameter is mandatory for phase one only, and it is ignored in phase two.

Keyword: USERSLIST

Valid Values: *user1*:*user2*:. . .

Separate user names with a colon (:).

Default Setting: No default setting.

Syntax Examples: USERSLIST=*jdoe*:*tchin*:*adesai*

Description: Specifies a list of database users for migration. The users in this list are migrated with other users that are specified with the USERS parameter.

Restrictions: This optional parameter is effective only when LIST is specified with the USERS parameter.

Keyword: USERSFILE

Valid Values: File name and path.

Default Setting: No default setting.

Syntax Examples: USERSFILE=*/home/orahome/userslist/hr_users.txt*

Description: Specifies a file that contains a list of database users (one user listed for each line) for migration. The users in this file are migrated with other users that are specified with the USERS parameter.

Restrictions: This optional parameter is effective only when FILE is specified with the USERS parameter.

Keyword: MAPSCHEMA

Valid Values: *schema_type:schema_name*

Schema type can be:

- PRIVATE

  Retains users' old local schemas. Schema name is ignored when schema type is PRIVATE. No mapping entries are created in the directory.

- SHARED

  Maps users to a shared schema. Mapping entries are created in the directory. Schema name specifies the shared schema name. During shared schema mapping, whether users' local schemas are dropped from the database is determined by the "Keyword: CASCADE" setting.

(These values are not case-sensitive.)

Default Setting: PRIVATE

Syntax Examples: MAPSCHEMA=SHARED:*HR_ALL*

Description: Specifies whether the utility populates the interface table with schema mapping information.

Restrictions:

- See the SHARED option under Valid Values.
- This parameter is only valid for phase one.

Keyword: MAPTYPE

Valid Values: *mapping_type*:*mapping_level*

Mapping type can be:

- DB
- DOMAIN

Mapping level can be:

- ENTRY
- SUBTREE

Separate mapping type from mapping level with a colon (:).

(These values are not case-sensitive.)

Default Setting: DB:ENTRY

Syntax Examples: MAPTYPE=*DOMAIN*:*SUBTREE*

Description: Specifies the type of schema mapping that is to be applied when "Keyword: MAPSCHEMA" is set to SHARED. If DB is specified as the mapping type, then the utility creates a mapping in directory for the database. If DOMAIN is specified as the mapping type, then the utility creates a mapping in the directory for the domain containing the database. For domain mapping, the utility determines the domain that contains the database by an LDAP search in the relevant Oracle context.

Restrictions: This parameter is effective only when MAPSCHEMA is set to SHARED.

See Also: "About Using the SUBTREE Mapping Level Option" on page G-24 for more information about using this mapping level option.

Keyword: CASCADE

Valid Values:

- NO

  When users are mapped to a shared schema, the utility tries to drop their local schemas from the database. If this parameter is set to NO, then users are migrated only if they do not own objects in their local schema. Users who own objects in their old local schemas do not migrate and produce an error message in the migration log file.

- YES

  If this parameter is set to YES, then all users' schema objects are dropped along with their local schemas when they are migrated. Privileges and roles that were previously granted to the users are also revoked.

(These values are not case-sensitive.)

Default Setting: NO

Syntax Examples: CASCADE=YES

Description: Specifies whether a user's local schema is dropped when the user is mapped to a shared schema.

Restrictions: This parameter is effective only when MAPSCHEMA is set to SHARED.

Keyword: CONTEXT

Valid Values: Distinguished Name (DN) of the parent for user entries. This is the same as the user search base or user create base in an Oracle Internet Directory identity management realm.

Parent DN can also be specified within double quotation marks ("...").

Default Setting: This value is automatically populated from the DEFAULT_ADMIN_CONTEXT setting in the ldap.ora file by default. This places new user entries directly under the Oracle Context's parent entry.

In 10*g* Release 1 (10.1), this is not the preferred location for user entries, so do not use the default setting for this parameter unless it is specifically desired. Instead, Oracle recommends that you use "cn=Users, <realm_DN>" as your default. Refer to Figure 11–3, "Related Entries in a Realm Oracle Context" on page 11-16 for a directory information tree diagram that shows an Oracle Context.

Syntax Examples: CONTEXT="c=Users, c=us"

Description: Specifies the DN of the parent entry under which user entries are created in the directory if there is no directory entry that matches the userID for the user.

Restrictions: This parameter is only valid for phase one.

Keyword: LOGFILE

Valid Values: File name and path.

Default Setting: *$ORACLE_HOME*/network/log/umu.log

Syntax Examples: LOGFILE=*home/orahome*/network/log/*filename.log*

Description: Specifies the log file where details about the migration for each user are written.

Restrictions: None

Keyword: PARFILE

Valid Values: File name and path.

Default Setting: No default setting.

Syntax Examples: PARFILE=*home/orahome*/network/*usr/par.txt*

Description: Specifies a text file which contains a list of these parameters that are intended to be used in a user migration. Each parameter must be listed on a separate line in the file. If a parameter is specified in both the parameter file and on the command line, then the one specified on the command line takes precedence.

Restrictions: None

User Migration Utility Usage Examples

The following sections contain examples of the syntax for some typical uses of this utility.

Migrating Users While Retaining Their Own Schemas

To migrate users while retaining their old database schemas, set the MAPSCHEMA parameter to PRIVATE, which is the default setting. For example, to migrate users scott1, scott2, and all external database users, while retaining their old schemas, to the directory at c=Users, c=us with the newly generated database and directory passwords, the syntax shown in Example 13–2 is used.

Example 13–2 Migrating Users with MAPSCHEMA=PRIVATE (Default)

```
umu PHASE=ONE
    DBLOCATION=machine1:1521:ora_sid
    DBADMIN=system:manager
    USERS=ALL_EXTERNAL:LIST
    USERSLIST=scott1:scott2
    DIRLOCATION=machine2:636
    CONTEXT="c=Users,c=us"
    ENTADMIN="cn=janeadmin":welcome

umu PHASE=TWO
    DBLOCATION=machine1:1521:ora_sid
    DBADMIN=system:manager
    DIRLOCATION=machine2:636
    ENTADMIN="cn=janeadmin":welcome
```

After phase one completes successfully, the interface table is populated with the user migration information. Then the enterprise user administrator can review the table to confirm its contents. Because no value was specified for the MAPSCHEMA

parameter, the utility runs phase one using the default value, PRIVATE, so all users' old database schemas and objects are retained.

Migrating Users and Mapping to a Shared Schema

To migrate users and map them to a new shared schema, dropping their old database schemas, set the MAPSCHEMA parameter to SHARED. The shared schema must already exist or the enterprise user administrator must create it before running the utility with this parameter setting. In the following example, users scott1, scott2, and all external database users are migrated to the directory at c=Users, c=us with newly generated database and directory passwords, while mapping all migrated users to a new shared schema in the database.

Use the syntax shown in Example G–1 to run the migration process with MAPSCHEMA set to SHARED.

Example G–1 Migrating Users with MAPSCHEMA=SHARED

```
umu PHASE=ONE
    DBLOCATION=machine1:1521:ora_sid
    DBADMIN=system:manager
    USERS=ALL_EXTERNAL:LIST
    USERSLIST=scott1:scott2
    MAPSCHEMA=SHARED:schema_32
    DIRLOCATION=machine2:636
    CONTEXT="c=Users, c=us"
    ENTADMIN="cn=janeadmin":welcome

umu PHASE=TWO
    DBLOCATION=machine1:1521:ora_sid
    DBADMIN=system:manager
    DIRLOCATION=machine2:636
    ENTADMIN="cn=janeadmin":welcome
```

After phase one completes successfully, the interface table is populated with the user migration information. Then the administrator can review the table to confirm its contents. Users scott1, scott2, and the external users are assigned new randomly generated database and directory passwords. Because no value was specified for the CASCADE parameter, the utility runs phase one using the default value, NO, which means that migrating users who own database objects in their old database schemas will fail and their schemas will not be automatically dropped. To determine which users have failed, review the log file that is located at $ORACLE_HOME/network/log/umu.log by default.

Mapping Users to a Shared Schema Using Different CASCADE Options

The CASCADE parameter setting determines whether users' old database schemas are automatically dropped when mapping to a shared schema during migration. CASCADE can be used only when MAPSCHEMA is set to SHARED.

Mapping Users to a Shared Schema with CASCADE=NO

By default, the CASCADE parameter is set to NO. This setting means that when mapping migrating users to a shared schema, users who own database objects in their old schemas are not migrated. For users who do not own database objects, their old database schemas are automatically dropped and they are mapped to the new shared schema.

> **See Also:** Example G–1 on page G-21 for a syntax example to map users to a shared schema with CASCADE set to NO. Note that because NO is the default setting for CASCADE this parameter does not have to be specified in the utility command syntax.

Mapping Users to a Shared Schema with CASCADE=YES

If it is known that no migrating users own database objects or want to retain the objects that they own in their old database schemas, then setting the CASCADE parameter to YES automatically drops all users' schemas and schema objects and maps them to the new shared schema. Example G–2 shows the syntax to use when setting CASCADE to YES. In this example, users `scott1`, `scott2`, and all external database users are migrated to the directory at `c=Users, c=us`, while mapping all migrating users to a new shared schema in the database.

Example G–2 Migrating Users with Shared Schema Mapping and CASCADE=YES

```
umu PHASE=ONE
    DBLOCATION=machine1:1521:ora_sid
    DBADMIN=system:manager
    USERS=ALL_EXTERNAL:LIST
    USERSLIST=scott1:scott2
    MAPSCHEMA=SHARED:schema_32
    CASCADE=YES
    DIRLOCATION=machine2:636
    CONTEXT="c=Users, c=us"
    ENTADMIN="cn=janeadmin":welcome

umu PHASE=TWO
    DBLOCATION=machine1:1521:ora_sid
```

```
DBADMIN=system:manager
DIRLOCATION=machine2:636
ENTADMIN="cn=janeadmin":welcome
```

After phase one completes successfully, the interface table is populated with the user migration information. Then the administrator can review the table to confirm its contents. Because the CASCADE parameter is set to YES, all migrated users' old database schemas are automatically dropped, including those who own database objects.

Caution: If you set the CASCADE parameter to YES, then Oracle recommends that enterprise user administrators back up the database or take an export dump of the users being migrated before running this utility. Then if migrated users want their old database objects, they can retrieve them from the export dump.

Mapping Users to a Shared Schema Using Different MAPTYPE Options

When MAPSCHEMA is set to SHARED, the mapping type can be set by specifying a value for the MAPTYPE parameter. This parameter takes two values, which are the mapping type and the mapping level.

Mapping type can be set at DB, for database, or DOMAIN, for enterprise domain. When mapping type DB is specified, the mapping is applied only to the database where the shared schema is stored. When DOMAIN is specified as the mapping type, then the mapping is applied to the enterprise domain that contains the database where the shared schema is stored and also applies to all databases in that domain.

Mapping level can be set to ENTRY or SUBTREE. When ENTRY is specified then users are mapped to the shared schema using their full distinguished name (DN). This results in one mapping for each user. When SUBTREE is specified then groups of users who share part of their DNs are mapped together. This results in one mapping for user groups already grouped under some common root in the directory tree. Example G–3 shows the syntax to use when using the MAPTYPE parameter. In this example, users scott1, scott2, and all external database users are migrated to the directory at c=Users, c=us, while mapping all migrated users to a new shared schema in the database. In this example, the mapping will apply to the enterprise domain that contains the database and the mapping will be performed at the entry level, resulting in a mapping for each user.

Example G–3 Migrating Users with Shared Schema Mapping Using the MAPTYPE Parameter

```
umu PHASE=ONE
    DBLOCATION=machine1:1521:ora_sid
    DBADMIN=system:manager
    USERS=ALL_EXTERNAL:LIST
    USERSLIST=scott1:scott2
    MAPSCHEMA=SHARED:schema_32
    MAPTYPE=DOMAIN:ENTRY
    DIRLOCATION=machine2:636
    CONTEXT="c=Users, c=us"
    ENTADMIN="cn=janeadmin":welcome

umu PHASE=TWO
    DBLOCATION=machine1:1521:ora_sid
    DBADMIN=system:manager
    DIRLOCATION=machine2:636
    ENTADMIN="cn=janeadmin":welcome
```

About Using the SUBTREE Mapping Level Option If a user (`scott`, for example) who is being migrated will have future user entries in a subtree under it, then it makes sense to create a subtree level mapping from this user entry (`cn=scott`) to a schema. However, the database does not interpret the user to be in the subtree so the mapping does not apply to `scott` himself. For example, if you are migrating the user `scott` with the DN `cn=scott,o=acme`, and you choose `SUBTREE` as the mapping level when you run the utility, then a new mapping is created from `cn=scott,o=acme` to the shared schema, but the user `scott` is not mapped to that schema. Only new users who are created under the `scott` directory entry are mapped to the shared schema. Consequently, the `SUBTREE` mapping level should only be specified when user directory entries are placed under other user directory entries, which would be an unusual directory configuration.

If you want an arbitrary subtree user to be mapped to a single shared schema with only one mapping entry, then you must use Enterprise Security Manager to create that mapping.

> **See Also:** "Managing Enterprise Domain Database Schema Mappings" on page 13-20 for information about using Enterprise Security Manager.

Migrating Users Using the PARFILE, USERSFILE, and LOGFILE Parameters

It is possible to enter user information and User Migration Utility parameters into a text file and pass the information and parameters to the utility using the PARFILE and USERSFILE parameters. The LOGFILE parameter sets the directory path for the log file where details about the migration for each user are written.

The PARFILE parameter tells the utility where a text file is located that contains the parameters for a bulk user migration. The USERSFILE parameter works like the PARFILE parameter, except it contains database users instead of parameters. The parameters and users lists contain one parameter or user for each line. The LOGFILE parameter tells the utility where to write the system events that occur during a user migration, such as errors. Use the USERSFILE parameter during phase one of the migration process. The PARFILE and LOGFILE parameters can be used in both phases.

Example G–4 shows the syntax for a typical parameter text file to migrate users scott1, scott2, and all external database users, while retaining their old schemas, to the directory at c=Users, c=us. In this example a log of migration events is written to the file errorfile1 in the directory where the utility is run. If another location is desired, then include the path with the file name.

Example G–4 Parameter Text File (par.txt) to Use with the PARFILE Parameter

```
DBLOCATION=machine1:1521:ora_sid
DBADMIN=system:manager
USERS=ALL_EXTERNAL:LIST:FILE
USERSLIST=scott1:scott2
USERSFILE=usrs.txt
DIRLOCATION=machine2:636
CONTEXT="c=Users, c=us"
ENTADMIN="cn=janeadmin":welcome
LOGFILE=errorfile1
```

Example G–5 shows the syntax for a typical users list text file.

Example G–5 Users List Text File (usrs.txt) to Use with the USERSFILE Parameter

```
user1
user2
user3
```

To execute phase one of the migration process with these parameters and users list text files, use the syntax shown in Example G–6.

Example G–6 Migrating Users Using the PARFILE, USERSFILE, and LOGFILE Parameters

```
umu PHASE=ONE
    DBADMIN=system:manager
    PARFILE=par.txt
    LOGFILE=errorfile2
```

Note: Although the `LOGFILE` parameter is specified twice, once in the parameter text file as `errorfile1` (shown in Example G–4) and once on the command line as `errorfile2` (show in Example G–6), command-line parameters take precedence over those specified inside the parameter file. Consequently, in Example G–6 the log file will be written to `errorfile2` because that value is specified on the command line.

Troubleshooting Using the User Migration Utility

Migration failures are reported to the enterprise user administrator with error messages and log messages. The following sections describe common error and log messages and what administrators can do to resolve them.

See Also: "Summary of User Migration Utility Error and Log Messages" on page G-34 for an alphabetical listing of error and log messages and links to where they are described in this section.

Common User Migration Utility Error Messages

When the utility encounters any error while running, it displays an error message and stops running. The following sections describe these messages and explain how to resolve the errors:

- Resolving Error Messages Displayed for Both Phases
- Resolving Error Messages Displayed for Phase One

Resolving Error Messages Displayed for Both Phases

The following error messages may display while the utility is running either phase one or phase two of the migration:

- Attribute value missing : : orclCommonNicknameAttribute

- Database connection failure
- Database error: < database_error_message >
- Database not in any domain : : DB-NAME = < database_name >
- Database not registered with the directory : : DB-NAME = < dbName >
- Directory connection failure
- Directory error : : < directory_error_message >
- Multiple entries found : : uniqueMember = < database_DN >

Attribute value missing : : orclCommonNicknameAttribute

Cause: The nickname attribute is not set in the directory in the root identity management realm.

Action: Use Enterprise Security Manager Console to set the nickname attribute for the identity management realm.

Database connection failure

Cause: The utility was unable to connect to the database.

Action: Perform these steps:

1. Check the database status to determine whether it is configured for encryption and integrity.
2. Check the privileges and credentials of the enterprise user administrator who is running the utility.

Database error: < database_error_message >

Cause: The utility encountered a database error.

Action: Check the database error message details for the database.

See Also: *Oracle Database Error Messages* for information about resolving database error messages.

Database not in any domain : : DB-NAME = < database_name >

Cause: The database is not a member of any enterprise domain.

Action: Use Enterprise Security Manager to add the database to an enterprise domain in the directory.

Database not registered with the directory : : DB-NAME = < dbName >

Cause: There is no entry for the database in the Oracle context that the ldap.ora file points to.

Action: Use Database Configuration Assistant or Enterprise Security Manager to register the database in the directory.

Directory connection failure

Cause: The utility was unable to connect to the directory.

Action: Perform these steps:

1. Check the directory server status to determine whether the directory server port is configured for SSL with no authentication.
2. Check the privileges and credentials of the enterprise user administrator who is running the utility.

Directory error : : < directory_error_message >

Cause: The utility encountered a directory error.

Action: Check the directory error message details for the directory.

> **See Also:** *Oracle Internet Directory Administrator's Guide* for information about resolving error messages for Oracle Internet Directory.

Multiple entries found : : uniqueMember = < database_DN >

Cause: The database belongs to more than one enterprise domain in the directory.

Action: Use Enterprise Security Manager or Oracle Directory Manager to ensure that the database belongs to only one enterprise domain.

Resolving Error Messages Displayed for Phase One

While the utility is running phase one of the migration, syntax or other types of errors may occur. The following error messages may display while the utility is running phase one of the migration:

- Argument missing or duplicated : : < parameter >
- Database object missing : : SHARED-SCHEMA = <shared_schema_name >
- Error reading file : : < file_name > : : < io_error_message >
- Error reading file : : PARFILE = < file_name > : : < io_error_message>

- Getting local host name failed
- Interface table creation in SYS schema not allowed
- Invalid argument or value : : < argument >
- Invalid arguments for the phase
- Invalid value : : < user > [USERSFILE]
- Invalid value : : < user > [USERSFILE] { = = DBADMIN }
- Invalid value : : < user > [USERSLIST]
- Invalid value : : < user > [USERSLIST] { = = DBADMIN }
- Logging failure : : < io_error_message >
- No entry found : : CONTEXT = < context >

Argument missing or duplicated : : < parameter >

Cause: Syntax error. A parameter is missing or has been entered multiple times.

Action: Check the usage syntax.

Database object missing : : SHARED-SCHEMA = <shared_schema_name >

Cause: The shared schema is not present in the database.

Action: Create the shared schema.

Error reading file : : < file_name > : : < io_error_message >

Cause: Syntax error. The utility cannot read the file that contains the users list that is specified in the USERSFILE parameter.

Action: Perform these steps:

1. Check to ensure that the file exists.
2. Check to ensure that the file has the correct permissions so the utility can read it.

Error reading file : : PARFILE = < file_name > : : < io_error_message>

Cause: Syntax error. The utility cannot read the file that contains the list of parameters that is specified in the PARFILE parameter.

Action: Perform these steps:

1. Check to ensure that the file exists.

2. Check to ensure that the file has the correct permissions so the utility can read it.

Getting local host name failed

Cause: Syntax error. The utility is unable to read the local host name for the database location or the directory location.

Action: Explicitly enter the hostname information with the `DBLOCATION` and `DIRLOCATION` parameters.

See Also:

- "Keyword: DBLOCATION" on page G-12
- "Keyword: DIRLOCATION" on page G-13

For information about how to use these parameters.

Interface table creation in SYS schema not allowed

Cause: The interface table cannot be created in the `SYS` schema.

Action: Specify another user in the `DBADMIN` parameter.

See Also: "Keyword: DBADMIN" on page G-13 for information about setting the `DBADMIN` parameter.

Invalid argument or value : : < argument >

Cause: Syntax error. The argument name or value has been entered incorrectly.

Action: Check the usage syntax.

See Also:

- "User Migration Utility Command Line Syntax" on page G-10
- "Accessing Help for the User Migration Utility" on page G-11
- "User Migration Utility Parameters" on page G-12

For information about using the command line syntax for this utility.

Invalid arguments for the phase

Cause: Syntax error. This occurs when you have used a command line argument that is only intended for phase one, but you are running phase two.

Action: Check the usage syntax.

Invalid value : : < user > [USERSFILE]

Cause: Syntax error. The user that is specified in this error message is invalid because they are not a user in the database that is specified in the DBLOCATION parameter.

Action: Remove the invalid user from the file that is specified with the USERSFILE parameter.

Invalid value : : < user > [USERSFILE] { = = DBADMIN }

Cause: Syntax error. The file that is specified in the USERSFILE parameter contains the user who is running the migration utility.

Action: Remove that user from the file.

Invalid value : : < user > [USERSLIST]

Cause: Syntax error. The user that is specified in this error message is invalid because they are not a user in the database that is specified in the DBLOCATION parameter.

Action: Remove the invalid user from the USERSLIST parameter.

Invalid value : : < user > [USERSLIST] { = = DBADMIN }

Cause: Syntax error. The USERSLIST parameter contains the user who is running the migration utility.

Action: Remove that user from the USERSLIST.

Logging failure : : < io_error_message >

Cause: Syntax error. The utility cannot find the log file or it cannot open the file to write to it.

Action: Perform these steps:

1. Check to ensure that the log file exists.
2. Check to ensure that the log file has the correct permissions so the utility can write information to it.

No entry found : : CONTEXT = < context >

Cause: The CONTEXT entry is not present in the directory.

Action: Perform one of the following options:

- Use the directory management tool or the LDAP command line utility to create an entry in the directory for the context value.
- Specify another valid context value.

Resolving Error Messages Displayed for Phase Two

Most of the error messages that you encounter while running this utility occur in phase one. After phase one has completed successfully, and while phase two is running, the following error may occur:

Database object missing : : TABLE = ORCL_GLOBAL_USR_MIGRATION_DATA

Cause: The utility cannot find the interface table.

Action: Perform one of the following options:

- Run phase one of the utility to create the interface table.
- Check to ensure that the user who is specified in the `DBADMIN` parameter is the same user who was specified for that parameter for phase one.

Common User Migration Utility Log Messages

Typically, log messages are written to the log file for each user who is migrated, whether the user was migrated successfully or not. The following sections describe these messages and explain how to resolve the errors:

Common Log Messages for Phase One

While the utility is running phase one of the migration, messages that indicate a user's information has not been successfully populated in the interface table may be written to the log file. After the utility completes phase one, review the log file to check for the following messages:

- Multiple entries found : : < nickname_attribute > = < username >
- No entry found : : < nickname_attribute > = < username > : : Entry found : DN = < dn >

Multiple entries found : : < nickname_attribute > = < username >

Cause: The nickname attribute matches multiple users or the user matches with multiple nickname attributes.

Action: Resolve the multiple matches and run the utility again for the users whose log file entry displayed this message.

No entry found : : < nickname_attribute > = < username > : : Entry found : DN = < dn >

Cause: No entry was found for the nickname matching, but an entry already exists for the DN in the directory.

Action: Specify a different DN for the user.

Common Log Messages for Phase Two

While the utility is running phase two of the migration, messages that indicate a user has not successfully migrated may be written to the log file. After the utility completes phase two, review the log file to check for the following messages:

- Attribute exists : : orclPassword
- Attribute value missing : : orclPassword
- Database object missing : : SHARED-SCHEMA = < shared_schema >
- Entry found : : DN = < user_DN >
- Invalid value : : <interface_table_column_name> = < interface_table_column_value >
- No entry found : : DN = < user_DN >

Attribute exists : : orclPassword

This message typically occurs with the message `Invalid value::<column_name>=<column_value>`.

Cause: The entry already contains a value for the `orclPassword` attribute.

Action: Check the `DBPASSWORD_EXIST_FLAG` column in the interface table for a `T/F` value that correctly reflects whether a database password exists for this user.

Attribute value missing : : orclPassword

This message typically occurs with the message `Invalid value::<column_name>=<column_value>`.

Cause: The `orclPassword` attribute of this user's entry has a null value.

Action: Check the `DBPASSWORD_EXIST_FLAG` column in the interface table for a `T/F` value that correctly reflects whether a database password exists for this user.

Database object missing : : SHARED-SCHEMA = < shared_schema >

Cause: The shared schema that was specified for this user does not exist in the database.

Action: Perform one of the following options:

- Check to ensure that the correct shared schema was specified for this user. If the shared schema name was incorrectly specified, then edit the `SHARED_`

SCHEMA column of the interface table and run phase two of the utility for this user again.

- Create the shared schema in the database and run phase two of the utility for this user again.

Entry found : : DN = < user_DN >

This message typically occurs with the message `Invalid value::<column_name>=<column_value>`.

Cause: An entry already exists for the specified user DN.

Action: Check the `USERDN_EXIST_FLAG` column in the interface table for a `T/F` value that correctly reflects whether a user entry already exists in the directory for this DN.

Invalid value : : <interface_table_column_name> = < interface_table_column_value >

Cause: The value in the interface table column for this user is invalid. Typically, this message is accompanied by additional log messages for this user.

Action: Check to ensure that the correct value has been entered for this user.

No entry found : : DN = < user_DN >

This message typically occurs with the message `Invalid value::<column_name>=<column_value>`.

Cause: The entry for the DN is missing in the directory.

Action: Check the `USERDN_EXIST_FLAG` column in the interface table for a `T/F` value that correctly reflects whether a user entry already exists in the directory for this DN.

Summary of User Migration Utility Error and Log Messages

Table G–4 and Table G–5 list all of the error and log messages in alphabetical order and provides links to the section in this chapter that describes the message and how to resolve it.

Table G–4 Alphabetical Listing of User Migration Utility Error Messages

User Migration Utility Error Message	Phase
Argument missing or duplicated : : < parameter > on page G-29	1
Attribute value missing : : orclCommonNicknameAttribute on page G-27	Both
Database connection failure on page G-27	Both

Table G–4 (Cont.) Alphabetical Listing of User Migration Utility Error Messages

User Migration Utility Error Message	Phase
Database error: < database_error_message > on page G-27	Both
Database not in any domain : : DB-NAME = < database_name > on page G-27	Both
Database not registered with the directory : : DB-NAME = < dbName > on page G-27	Both
Database object missing : : SHARED-SCHEMA = <shared_schema_name > on page G-29	1
Database object missing : : TABLE = ORCL_GLOBAL_USR_MIGRATION_DATA on page G-32	2
Directory connection failure on page G-28	Both
Directory error : : < directory_error_message > on page G-28	Both
Error reading file : : < file_name > : : < io_error_message > on page G-29	1
Error reading file : : PARFILE = < file_name > : : < io_error_message> on page G-29	1
Getting local host name failed on page G-30	1
Interface table creation in SYS schema not allowed on page G-30	1
Invalid argument or value : : < argument > on page G-30	1
Invalid arguments for the phase on page G-30	1
Invalid value : : < user > [USERSFILE] on page G-31	1
Invalid value : : < user > [USERSFILE] { = = DBADMIN } on page G-31	1
Invalid value : : < user > [USERSLIST] on page G-31	1
Invalid value : : < user > [USERSLIST] { = = DBADMIN } on page G-31	1
Logging failure : : < io_error_message > on page G-31	1
Multiple entries found : : uniqueMember = < database_DN > on page G-28	Both
No entry found : : CONTEXT = < context > on page G-31	1

Table G–5 Alphabetical Listing of User Migration Utility Log Messages

User Migration Utility Log Message	Phase
Attribute exists : : orclPassword on page G-33	2
Attribute value missing : : orclPassword on page G-33	2
Database object missing : : SHARED-SCHEMA = < shared_schema > on page G-33	2
Entry found : : DN = < user_DN > on page G-34	2

Table G–5 Alphabetical Listing of User Migration Utility Log Messages

User Migration Utility Log Message	Phase
Invalid value : : <interface_table_column_name> = < interface_table_column_value > on page G-34	2
Multiple entries found : : < nickname_attribute > = < username > on page G-32	1
No entry found : : DN = < user_DN > on page G-34	2
No entry found : : < nickname_attribute > = < username > : : Entry found : DN = < dn > on page G-32	1

Glossary

access control

The ability of a system to grant or limit access to specific data for specific clients or groups of clients.

Access Control Lists (ACLs)

The group of access directives that you define. The directives grant levels of access to specific data for specific clients, or groups of clients, or both.

Advanced Encryption Standard

Advanced Encryption Standard (AES) is a new cryptographic algorithm that has been approved by the National Institute of Standards and Technology as a replacement for DES. The AES standard is available in Federal Information Processing Standards Publication 197. The AES algorithm is a symmetric block cipher that can process data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256 bits.

AES

See **Advanced Encryption Standard**

attribute

An item of information that describes some aspect of an entry in an LDAP directory. An entry comprises a set of attributes, each of which belongs to an **object class**. Moreover, each attribute has both a *type*, which describes the kind of information in the attribute, and a *value*, which contains the actual data.

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.

authentication method

A security method that verifies a user's, client's, or server's identity in distributed environments. Network authentication methods can also provide the benefit of **single sign-on (SSO)** for users. The following authentication methods are supported in Oracle Database when Oracle Advanced Security is installed:

- **Kerberos**
- **RADIUS**
- **Secure Sockets Layer (SSL)**
- **Windows NT native authentication**

authorization

Permission given to a user, program, or process to access an object or set of objects. In Oracle, authorization is done through the role mechanism. A single person or a group of people can be granted a role or a group of roles. A role, in turn, can be granted other roles. The set of privileges available to an authenticated entity.

auto login wallet

An Oracle Wallet Manager feature that enables PKI- or password-based access to services without providing credentials at the time of access. This auto login access stays in effect until the auto login feature is disabled for that 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. Sometimes these are called "SSO wallets" because they provide single sign-on capability.

base

The root of a subtree search in an **LDAP**-compliant directory.

CA

See **certificate authority**

CDS

See **Cell Directory Services (CDS)**

Cell Directory Services (CDS)

An external naming method that enables users to use Oracle tools transparently and applications to access Oracle Database databases in a Distributed Computing Environment (DCE).

certificate

An ITU x.509 v3 standard data structure that securely binds an identify 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.

certificate chain

An ordered list of certificates containing an end-user or subscriber certificate and its certificate authority certificates.

certificate request

A certificate request, which consists of three parts: certification request information, a signature algorithm identifier, and a digital signature on the certification request information. The certification request information consists of the subject's distinguished name, public key, and an optional set of attributes. The attributes may

provide additional information about the subject identity, such as postal address, or a challenge password by which the subject entity may later request certificate revocation. See **PKCS #10**

certificate revocation lists

(CRLs) Signed data structures that contain a list of revoked **certificates**. The authenticity and integrity of the CRL is provided by a digital signature appended to it. Usually, the CRL signer is the same entity that signed the issued certificate.

checksumming

A mechanism that computes a value for a message packet, based on the data it contains, and passes it along with the data to authenticate that the data has not been tampered with. The recipient of the data recomputes the cryptographic checksum and compares it with the cryptographic checksum passed with the data; if they match, it is "probabilistic" proof the data was not tampered with during transmission.

Cipher Block Chaining (CBC)

An encryption method that protects against block replay attacks by making the encryption of a cipher block dependent on all blocks that precede it; it is designed to make unauthorized decryption incrementally more difficult. Oracle Advanced Security employs *outer* cipher block chaining because it is more secure than *inner* cipher block chaining, with no material performance penalty.

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.

cipher suite name

Cipher suites describe the kind of cryptographics protection that is used by connections in a particular session.

ciphertext

Message text that has been encrypted.

cleartext

Unencrypted plain text.

client

A client relies on a service. A client can sometimes be a user, sometimes a process acting on behalf of the user during a database link (sometimes called a proxy).

confidentiality

A function of cryptography. Confidentiality guarantees that only the intended recipient(s) of a message can view the message (decrypt the ciphertext).

connect descriptor

A specially formatted description of the destination for a network connection. A connect descriptor contains destination **service** and network route information. The destination service is indicated by using its service name for Oracle*9i* or Oracle*8i* databases or its Oracle **system identifier (SID)** for Oracle databases version 8.0. The network route provides, at a minimum, the location of the **listener** through use of a network address. See **connect identifier**

connect identifier

A **connect descriptor** or a name that maps to a connect descriptor. A connect identifier can be a **net service name**, database **service name**, or **net service alias**. Users initiate a connect request by passing a username and password along with a connect identifier in a connect string for the service to which they wish to connect:

```
CONNECT username/password@connect_identifier
```

connect string

Information the user passes to a **service** to connect, such as **username**, password and **net service name**. For example:

```
CONNECT username/password@net_service_name
```

credentials

A **username**, password, or certificate used to gain access to the database.

CRL

See **certificate revocation lists**

CRL Distribution Point

(CRL DP) An optional extension specified by the X.509 version 3 certificate standard, which indicates the location of the Partitioned CRL where revocation information for a certificate is stored. Typically, the value in this extension is in the

form of a URL. CRL DPs allow revocation information within a single **certificate authority** domain to be posted in multiple CRLs. CRL DPs subdivide revocation information into more manageable pieces to avoid proliferating voluminous CRLs, thereby providing performance benefits. For example, a CRL DP is specified in the certificate and can point to a file on a Web server from which that certificate's revocation information can be downloaded.

CRL DP

See **CRL Distribution Point**

cryptography

The practice of encoding and decoding data, resulting in secure messages.

data dictionary

A set of read-only tables that provide information about a database.

Data Encryption Standard (DES)

The U.S. data encryption standard.

Database Administrator

(1) A person responsible for operating and maintaining an Oracle Server or a database application. (2) An Oracle username that has been given DBA privileges and can perform database administration functions. Usually the two meanings coincide. Many sites have multiple DBAs.

database alias

See **net service name**

Database Installation Administrator

Also called a database creator. This administrator is in charge of creating new databases. This includes registering each database in the directory using the Database Configuration Assistant. This administrator has create and modify access to database service objects and attributes. This administrator can also modify the Default **domain**.

database link

A network object stored in the local database or in the network definition that identifies a remote database, a communication path to that database, and optionally, a username and password. Once defined, the database link is used to access the remote database.

A public or private database link from one database to another is created on the local database by a DBA or user.

A global database link is created automatically from each database to every other database in a network with Oracle Names. Global database links are stored in the network definition.

database method

See **Oracle database method**

database password verifier

A database password verifier is an irreversible value that is derived from the user's database password. This value is used during password authentication to the database to prove the identity of the connecting user.

Database Security Administrator

The highest level administrator for database enterprise user security. This administrator has permissions on all of the enterprise domains and is responsible for:

- Administering the Oracle DBSecurityAdmins and OracleDBCreators groups.
- Creating new **enterprise domains**.
- Moving databases from one **domain** to another within the enterprise.

DCE

See **Distributed Computing Environment (DCE)**

decryption

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

DES

See **Data Encryption Standard (DES)**

dictionary attack

A common attack on passwords. the attacker creates a dictionary of many possible passwords and their corresponding verifiers. Through some means, the attacker then obtains the verifier corresponding to the target password, and obtains the target password by looking up the verifier in the dictionary.

Diffie-Hellman key negotiation algorithm

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

digital signature

A digital signature is created when a public key algorithm is used to sign the sender's message with the sender's private key. The digital signature assures that the document is authentic, has not been forged by another entity, has not been altered, and cannot be repudiated by the sender.

directory information tree (DIT)

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

directory naming

A **naming method** that resolves a database service, **net service name**, or **net service alias** to a **connect descriptor** stored in a central directory server. A

directory naming context

A subtree which is of significance within a directory server. It is usually the top of some organizational subtree. Some directories only permit one such context which is fixed; others permit none to many to be configured by the directory administrator.

Distributed Computing Environment (DCE)

A set of integrated network services that works across multiple systems to provide a distributed environment. The middleware between distributed applications and the operating system or networkservices; based on a client/server computing model. DCE is supported by the Open Group.

distinguished name (DN)

The unique name of a directory entry. It is comprised of all of the individual names of the parent entries back to the root entry of the directory information tree. See **directory information tree (DIT)**

domain

Any tree or subtree within the **Domain Name System (DNS)** namespace. Domain most commonly refers to a group of computers whose host names share a common suffix, the domain name.

Domain Name System (DNS)

A system for naming computers and network services that is organized into a hierarchy of **domain*s***. DNS is used in TCP/IP networks to locate computers through user-friendly names. DNS resolves a friendly name into an IP address, which is understood by computers.

In **Oracle Net Services**, DNS translates the host name in a TCP/IP address into an IP address.

encrypted text

Text that has been encrypted, using an encryption algorithm; the output stream of an encryption process. On its face, it is not readable or decipherable, without first being subject to **decryption**. Also called **ciphertext**. Encrypted text ultimately originates as **plaintext**.

encryption

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

enterprise domain

A directory construct that consists of a group of databases and **enterprise role**s. A database should only exist in one enterprise domain at any time. Enterprise domains are different from Windows 2000 domains, which are collections of computers that share a common directory database.

Enterprise Domain Administrator

User authorized to manage a specific **enterprise domain**, including the authority to add new enterprise domain administrators.

enterprise role

Access privileges assigned to **enterprise user**s. A set of Oracle role-based **authorization**s across one or more databases in an **enterprise domain**. Enterprise roles are stored in the directory and contain one or more **global role**s.

enterprise user

A user defined and managed in a directory. Each enterprise user has a unique identify across an enterprise.

entry

The building block of a directory, it contains information about an object of interest to directory users.

external authentication

Verification of a user identity by a third party authentication service, such as Kerberos or RADIUS.

file system method

Storing fingerprint templates in files when configuring Identix Biometric authentication. The alternative is to use the **Oracle database method**.

Federal Information Processing Standard (FIPS)

A U.S. government standard that defines security requirements for cryptographic modules—employed within a security system protecting unclassified information within computer and telecommunication systems. Published by the National Institute of Standards and Technology (NIST).

FIPS

See **Federal Information Processing Standard (FIPS)**

forest

A group of one or more Active Directory trees that trust each other. All trees in a forest share a common **schema**, configuration, and global catalog. When a forest contains multiple trees, the trees do not form a contiguous namespace. All trees in a given forest trust each other through transitive bidirectional trust relationships.

forwardable ticket-granting ticket

In Kerberos. A service ticket with the FORWARDABLE flag set. This flag enables authentication forwarding without requiring the user to enter a password again.

GDS

See **Global Directory Service (GDS)**

Global Directory Service (GDS)

GDS is the **DCE** directory service that acts as an agent between **DCE CDS** and any X.500 directory service. Both GDS and **CDS** are obsolete; they are only used by **DCE**.

global role

A role managed in a directory, but its privileges are contained within a single database. A global role is created in a database by using the following syntax:

```
CREATE ROLE <role_name> IDENTIFIED GLOBALLY;
```

grid computing

A computing architecture that coordinates large numbers of servers and storage to act as a single large computer. Oracle Grid Computing creates a flexible, on-demand computing resource for all enterprise computing needs. Applications running on the Oracle 10*g* grid computing infrastructure can take advantage of common infrastructure services for failover, software provisioning, and management. Oracle Grid Computing analyzes demand for resources and adjusts supply accordingly.

HTTP

Hypertext Transfer Protocol: The set of rules for exchanging files (text, graphic images, sound, video, and other multimedia files) on the World Wide Web. Relative to the TCP/IP suite of protocols (which are the basis for information exchange on the Internet), HTTP is an application protocol.

HTTPS

The use of Secure Sockets Layer (SSL) as a sublayer under the regular HTTP application layer.

identity

The combination of the public key and any other public information for an entity. The public information may include user identification data such as, for example, an e-mail address. A user certified as being the entity it claims to be.

identity management

The creation, management, and use of online, or digital, entities. Identity management involves securely managing the full life cycle of a digital identity from creation (provisioning of digital identities) to maintenance (enforcing organizational policies regarding access to electronic resources), and, finally, to termination.

identity management realm

A subtree in Oracle Internet Directory, including not only an **Oracle Context**, but also additional subtrees for users and groups, each of which are protected with access control lists.

initial ticket

In Kerberos authentication, an initial ticket or ticket granting ticket (TGT) identifies the user as having the right to ask for additional service tickets. No tickets can be obtained without an initial ticket. An initial ticket is retrieved by running the `okinit` program and providing a password.

instance

Every running Oracle database is associated with an Oracle instance. When a database is started on a database server (regardless of the type of computer), Oracle allocates a memory area called the **System Global Area (SGA)** and starts an Oracle process. This combination of the SGA and an Oracle process is called an instance. The memory and the process of an instance manage the associated database's data efficiently and serve the one or more users of the database.

integrity

The guarantee that the contents of the message received were not altered from the contents of the original message sent.

java code obfuscation

Java code **obfuscation** is used to protect Java programs from reverse engineering. A special program (an obfuscator) is used to scramble Java symbols found in the code. The process leaves the original program structure intact, letting the program run correctly while changing the names of the classes, methods, and variables in order to hide the intended behavior. Although it is possible to decompile and read non-obfuscated Java code, the obfuscated Java code is sufficiently difficult to decompile to satisfy U.S. government export controls.

Java Database Connectivity (JDBC)

An industry-standard Java interface for connecting to a relational database from a Java program, defined by Sun Microsystems.

JDBC

See **Java Database Connectivity (JDBC)**

KDC

Key Distribution Center. In Kerberos authentication, the KDC maintains a list of user principals and is contacted through the `kinit` (`okinit` is the Oracle version) program for the user's **initial ticket**. Frequently, the KDC and the Ticket Granting Service are combined into the same entity and are simply referred to as the KDC. The Ticket Granting Service maintains a list of service principals and is contacted when a user wants to authenticate to a server providing such a service. The KDC is a trusted third party that must run on a secure host. It creates ticket-granting tickets and service tickets.

Kerberos

A network authentication service developed under Massachusetts Institute of Technology's Project Athena that strengthens security in distributed environments. Kerberos is a trusted third-party authentication system that relies on shared secrets and assumes that the third party is secure. It provides single sign-on capabilities and database link authentication (MIT Kerberos only) for users, provides centralized password storage, and enhances PC security.

key

When encrypting data, a key is a value which determines the ciphertext that a given algorithm will produce from given plaintext. When decrypting data, a key is a value required to correctly decrypt a ciphertext. A ciphertext is decrypted correctly only if the correct key is supplied.

With a symmetric encryption algorithm, the same key is used for both encryption and decryption of the same data. With an asymmetric encryption algorithm (also called a public-key encryption algorithm or public-key cryptosystem), different keys are used for encryption and decryption of the same data.

key pair

A **public key** and its associated **private key**. See **public and private key pair**

keytab file

A Kerberos key table file containing one or more service keys. Hosts or services use *keytab* files in the same way as users use their passwords.

kinstance

An instantiation or location of a Kerberos authenticated service. This is an arbitrary string, but the host machine name for a service is typically specified.

kservice

An arbitrary name of a Kerberos service object.

LDAP

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

ldap.ora file

A file created by Oracle Net Configuration Assistant that contains the following directory server access information:

- Type of directory server
- Location of the directory server
- Default identity management realm or Oracle Context (including ports) that the client or server will use

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.

listener

A process that resides on the server whose responsibility is to listen for incoming client connection requests and manage the traffic to the server.

Every time a client requests a network session with a server, a listener receives the actual request. If the client information matches the listener information, then the listener grants a connection to the server.

listener.ora file

A configuration file for the listener that identifies the:

- Listener name
- Protocol addresses that it is accepting connection requests on
- Services it is listening for

The `listener.ora` file typically resides in `$ORACLE_HOME/network/admin` on UNIX platforms and *`ORACLE_HOME`*`\network\admin` on Windows.

man-in-the-middle

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

MD5

An algorithm that assures data integrity by generating a 128-bit cryptographic message digest value from given data. If as little as a single bit value in the data is modified, the MD5 checksum for the data changes. Forgery of data in a way that will cause MD5 to generate the same result as that for the original data is considered computationally infeasible.

message authentication code

Also known as data authentication code (DAC). A **checksumming** with the addition of a secret key. Only someone with the key can verify the cryptographic checksum.

message digest

See **checksumming**

naming method

The resolution method used by a client application to resolve a **connect identifier** to a **connect descriptor** when attempting to connect to a database service.

National Institute of Standards and Technology (NIST)

An agency within the U.S. Department of Commerce responsible for the development of security standards related to the design, acquisition, and implementation of cryptographic-based security systems within computer and telecommunication systems, operated by a Federal agency or by a contractor of a Federal agency or other organization that processes information on behalf of the Federal Government to accomplish a Federal function.

net service alias

An alternative name for a **directory naming** object in a directory server. A directory server stores net service aliases for any defined **net service name** or database service. A net service alias entry does not have connect descriptor information. Instead, it only references the location of the object for which it is an alias. When a

client requests a directory lookup of a net service alias, the directory determines that the entry is a net service alias and completes the lookup as if it was actually the entry it is referencing.

net service name

The name used by clients to identify a database server. A net service name is mapped to a port number and protocol. Also known as a **connect string**, or **database alias**.

network authentication service

A means for authenticating clients to servers, servers to servers, and users to both clients and servers in distributed environments. A network authentication service is a repository for storing information about users and the services on different servers to which they have access, as well as information about clients and servers on the network. An authentication server can be a physically separate machine, or it can be a facility co-located on another server within the system. To ensure availability, some authentication services may be replicated to avoid a single point of failure.

network listener

A listener on a server that listens for connection requests for one or more databases on one or more protocols. See **listener**

NIST

See **Federal Information Processing Standard (FIPS)**

non-repudiation

Incontestable proof of the origin, delivery, submission, or transmission of a message.

obfuscation

A process by which information is scrambled into a non-readable form, such that it is extremely difficult to de-scramble if the algorithm used for scrambling is not known.

obfuscator

A special program used to obfuscate Java source code. See **obfuscation**

object class

A named group of **attribute**s. When you want to assign attributes to an entry, you do so by assigning to that entry the object classes that hold those attributes. All objects associated with the same object class share the same attributes.

Oracle Context

1. An entry in an LDAP-compliant internet directory called `cn=OracleContext`, under which all Oracle software relevant information is kept, including entries for **Oracle Net Services** directory naming and **enterprise user** security.

There can be one or more Oracle Contexts in a directory. An Oracle Context is usually located in an **identity management realm**.

Oracle database method

Using an Oracle database to store fingerprint templates when configuring Indentix Biometric authentication. The alternative is to use the **file system method**.

Oracle Net Services

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

Oracle PKI certificate usages

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

Password-Accessible Domains List

A group of **enterprise domain**s configured to accept connections from password-authenticated users.

PCMCIA cards

Small credit card-sized computing devices that comply with the Personal Computer Memory Card International Association (PCMCIA) standard. These devices, also called PC cards, are used for adding memory, modems, or as hardware security modules. PCMCIA cards that are used as hardware security modules securely store the private key component of a **public and private key pair** and some also perform the cryptographic operations as well.

peer identity

SSL connect sessions are between a particular client and a particular server. The identity of the peer may have been established as part of session setup. Peers are identified by **X.509 certificate chain**s.

PEM

The Internet Privacy-Enhanced Mail protocols standard, adopted by the Internet Architecture Board to provide secure electronic mail over the Internet. The PEM protocols provide for encryption, authentication, message integrity, and key management. PEM is an inclusive standard, intended to be compatible with a wide range of key-management approaches, including both symmetric and public-key schemes to encrypt data-encrypting keys. The specifications for PEM come from four Internet Engineering Task Force (IETF) documents: RFCs 1421, 1422, 1423, and 1424.

PKCS #10

An RSA Security, Inc., Public-Key Cryptography Standards (PKCS) specification that describes a syntax for certification requests. A certification request consists of a distinguished name, a public key, and optionally a set of attributes, collectively signed by the entity requesting certification. Certification requests are referred to as certificate requests in this manual. See **certificate request**

PKCS #11

An RSA Security, Inc., Public-Key Cryptography Standards (PKCS) specification that defines an application programming interface (API), called Cryptoki, to devices which hold cryptographic information and perform cryptographic operations. See **PCMCIA cards**

PKCS #12

An RSA Security, Inc., Public-Key Cryptography Standards (PKCS) specification that describes a transfer syntax for storing and transferring personal authentication credentials—typically in a format called a **wallet**.

PKI

See **public key infrastructure (PKI)**

plaintext

Message text that has not been encrypted.

principal

A string that uniquely identifies a client or server to which a set of Kerberos credentials is assigned. It generally has three parts: `kservice/kinstance@REALM`. In the case of a user, `kservice` is the username. See also **kservice**, **kinstance**, and **realm**

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 and private key pair**

proxy authentication

A process typically employed in an environment with a middle tier such as a firewall, wherein the end user authenticates to the middle tier, which thence authenticates to the directory on the user's behalf—as its *proxy*. The middle tier logs into the directory as a *proxy user*. A proxy user can switch identities and, once logged into the directory, switch to the end user's identity. It can perform operations on the end user's behalf, using the authorization appropriate to that particular end user.

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 and 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 key infrastructure (PKI)

Information security technology utilizing the principles of public key cryptography. Public key cryptography involves encrypting and decrypting information using a shared public and private key pair. Provides for secure, private communications within a public network.

public and 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 enwrapped with a private key cannot be decrypted with the same private key.

RADIUS

Remote Authentication Dial-In User Service (RADIUS) is a client/server protocol and software that enables remote access servers to communication with a central server to authenticate dial-in users and authorize their access to the requested system or service.

realm

1. Short for **identity management realm**. 2. A Kerberos object. A set of clients and servers operating under a single key distribution center/ticket-granting service (KDC/TGS). Services (see **kservice**) in different realms that share the same name are unique.

realm Oracle Context

An **Oracle Context** that is part of an **identity management realm** in Oracle Internet Directory.

registry

A Windows repository that stores configuration information for a computer.

remote computer

A computer on a network other than the local computer.

root key certificate

See **trusted certificate**

schema

1. Database schema: A named collection of objects, such as tables, **views**, clusters, procedures, packages, **attributes**, **object classes**, and their corresponding matching rules, which are associated with a particular user. 2. LDAP directory schema: The collection of attributes, object classes, and their corresponding matching rules.

schema mapping

See **user-schema mapping**

Secure Hash Algorithm (SHA)

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)

An industry standard protocol designed by Netscape Communications Corporation for securing network connections. SSL provides authentication, encryption, and data integrity using public key infrastructure (PKI).

server

A provider of a service.

service

1. A network resource used by clients; for example, an Oracle database server.

2. An executable process installed in the Windows **registry** and administered by Windows. Once a service is created and started, it can run even when no user is logged on to the computer.

service name

For Kerberos-based authentication, the **kservice** portion of a service principal.

service principal

See **principal**

service table

In Kerberos authentication, a service table is a list of service principals that exist on a *kinstance*. This information must be extracted from Kerberos and copied to the Oracle server machine before Kerberos can be used by Oracle.

service ticket

Trusted information used to authenticate the client. A ticket-granting ticket, which is also known as the initial ticket, is obtained by directly or indirectly running `okinit` and providing a password, and is used by the client to ask for service tickets. A *service ticket* is used by a client to authenticate to a service.

session key

A key shared by at least two parties (usually a client and a server) that is used for data encryption for the duration of a single communication session. Session keys are typically used to encrypt network traffic; a client and a server can negotiate a session key at the beginning of a session, and that key is used to encrypt all network traffic between the parties for that session. If the client and server communicate again in a new session, they negotiate a new session key.

session layer

A network layer that provides the services needed by the presentation layer entities that enable them to organize and synchronize their dialogue and manage their data exchange. This layer establishes, manages, and terminates network sessions between the client and server. An example of a session layer is Network Session.

SHA

See **Secure Hash Algorithm (SHA)**

shared schema

A database or application schema that can be used by multiple enterprise users. Oracle Advanced Security supports the mapping of multiple enterprise users to the same shared schema on a database, which lets an administrator avoid creating an account for each user in every database. Instead, the administrator can create a user in one location, the enterprise directory, and map the user to a shared schema that other enterprise users can also map to. Sometimes called **user/schema separation**.

single key-pair wallet

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

single password authentication

The ability of a user to authenticate with multiple databases by using a single password. In the Oracle Advanced Security implementation, the password is stored in an LDAP-compliant directory and protected with encryption and Access Control Lists.

single sign-on (SSO)

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*. Oracle Advanced Security supports Kerberos, DCE, and SSL-based single sign-on.

smart card

A plastic card (like a credit card) with an embedded integrated circuit for storing information, including such information as user names and passwords, and also for performing computations associated with authentication exchanges. A smart card is read by a hardware device at any client or server.

A smartcard can generate random numbers which can be used as one-time use passwords. In this case, smartcards are synchronized with a service on the server so that the server expects the same password generated by the smart card.

sniffer

Device used to surreptitiously listen to or capture private data traffic from a network.

sqlnet.ora file

A configuration file for the client or server that specifies:

- Client domain to append to unqualified service names or net service names
- Order of naming methods the client should use when resolving a name
- Logging and tracing features to use
- Route of connections
- Preferred Oracle Names servers
- External naming parameters
- Oracle Advanced Security parameters

The `sqlnet.ora` file typically resides in `$ORACLE_HOME/network/admin` on UNIX platforms and *`ORACLE_HOME`*`\network\admin` on Windows platforms.

SSO

See **single sign-on (SSO)**

System Global Area (SGA)

A group of shared memory structures that contain data and control information for an Oracle **instance**.

system identifier (SID)

A unique name for an Oracle **instance**. To switch between Oracle databases, users must specify the desired SID. The SID is included in the `CONNECT DATA` parts of the **connect descriptor** in a **tnsnames.ora** file, and in the definition of the **network listener** in a **listener.ora file**.

ticket

A piece of information that helps identify who the owner is. See **service ticket**.

tnsnames.ora

A file that contains connect descriptors; each **connect descriptor** is mapped to a **net service name**. The file may be maintained centrally or locally, for use by all or individual clients. This file typically resides in the following locations depending on your platform:

- (UNIX) *`ORACLE_HOME`*`/network/admin`
- (Windows) *`ORACLE_BASE`*`\`*`ORACLE_HOME`*`\network\admin`

token card

A device for providing improved ease-of-use for users through several different mechanisms. Some token cards offer one-time passwords that are synchronized with an authentication service. The server can verify the password provided by the token card at any given time by contacting the authentication service. Other token cards operate on a challenge-response basis. In this case, the server offers a challenge (a number) which the user types into the token card. The token card then provides another number (cryptographically-derived from the challenge), which the user then offers to the server.

transport layer

A networking layer that maintains end-to-end reliability through data flow control and error recovery methods. **Oracle Net Services** uses *Oracle protocol supports* for the transport layer.

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 its higher level certificates reverified.

trusted certificate authority

See **certificate authority**

trust point

See **trusted certificate**

username

A name that can connect to and access objects in a database.

user-schema mapping

An **LDAP** directory entry that contains a pair of values: the **base** in the directory at which users exist, and the name of the database schema to which they are mapped. The users referenced in the mapping are connected to the specified schema when they connect to the database. User-schema mapping entries can apply only to one database or they can apply to all databases in a domain. See **shared schema**

user/schema separation

See **shared schema**

user search base

The node in the LDAP directory under which the user resides.

views

Selective presentations of one or more tables (or other views), showing both their structure and their data.

wallet

A wallet is a data structure used to store and manage security credentials for an individual entity. A **Wallet Resource Locator** (WRL) provides all the necessary information to locate the wallet.

wallet obfuscation

Wallet **obfuscation** is used to store and access an Oracle **wallet** without querying the user for a password prior to access (supports **single sign-on (SSO)**).

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.

Windows NT native authentication

An **authentication method** that enables a client single login access to a Windows server and a database running on that server.

WRL

See **Wallet Resource Locator**

X.509

An industry-standard specification for digital **certificate**s.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X