# Oracle Advanced Security™

Administrator's Guide

Release 8.1.5

February 1999

Part No. A67766-01

This manual explains how to use the Oracle Advanced Security option to enhance the security of your network.

ORACLE

Oracle Advanced Security Administrator's Guide, Release 8.1.5

Part No. A67766-01



Primary Author:—Richard Smith

Contributing Authors:—Gilbert Gonzalez, Laura Ferrer, Patricia Markee, Kendall Scott, Sandy Venning, Rick Wong
Contributors:—Pierre Baudin, Kristy Browder, Quan Dinh, Pramodini Gattu, Shuvayu Kanjilal, Van Le, Andy Philips, Ramana Rao, Vipin Samar, Debbie Steiner, Juliet Tran, Rick Wessman
Graphic Artist:—Valarie Moore

# Contents

## 2 Configuring Encryption and Checksumming



## 3 Configuring RADIUS Authentication

## 6 Configuring SecurID Authentication

## 7 Configuring Identix Biometric Authentication

## 8 Configuring DCE GSSAPI Authentication



## 9 Configuring SSL Authentication

## 12 Configuring DCE for Oracle DCE Integration



## 13 Configuring Oracle for Oracle DCE Integration

## 14 Connecting to an Oracle Database in DCE



## 15 DCE and Non-DCE Interoperability



## A Encryption and Checksumming Parameters



## B Authentication Parameters

# Send Us Your Comments

**Oracle Advanced Security Administrator's Guide, Release 8.1.5**

**Part No. A67766-01**

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

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

If you find any errors or have any other suggestions for improvement, please indicate the 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- 506-7226. Attn: Server Technologies Documentation Manager
- postal service:

  Oracle Corporation
  Oracle Advanced Security Documentation
  500 Oracle Parkway, Mailstop 4op12
  Redwood Shores, CA 94065
  USA

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

# Preface

The Oracle Advanced Security option provides enhanced security and authentication to your Net8 network, as well as integration with a Distributed Computing Environment (DCE). This Guide provides generic information on all these features of the Oracle Advanced Security option.

This Preface discusses the following topics:

## What This Guide Contains

This guide contains generic information on how to configure your already-existing Net8 network to use the Oracle Advanced Security option. Use it in conjunction with the guide that describes how to install and configure the Oracle Advanced Security option on your particular platform.

You can install and configure the Oracle Advanced Security option with other Oracle networking products and configure everything at once, or you can add the Oracle Advanced Security option to an already existing network.

## How This Manual Is Organized

This manual is divided into two parts:

- Part I: "Oracle Advanced Security Features"
- Part II: "Oracle Advanced Security and Oracle DCE Integration"

Each part describes a different set of Oracle Advanced Security option features.

### Part I: Oracle Advanced Security Features

Chapter 1, "Introduction to Oracle Advanced Security"—This chapter provides an overview of the security and single sign-on features of the Oracle Advanced Security option. These features include:

- network security
- data encryption
- data integrity checking
- token authentication
- single sign-on

> **Note:** These features have been previously packaged as Secure Network Services and Oracle Advanced Networking Option.

This chapter also includes a brief overview of the authentication methods available with this release.

Chapter 2, "Configuring Encryption and Checksumming"—This chapter tells you how to configure encryption and checksumming into your existing Net8 release 8.1.5 network.

Chapter 3, "Configuring RADIUS Authentication"—This chapter tells you how to configure Oracle for use with RADIUS (Remote Authentication Dial-In User Service). It gives an overview of how RADIUS works within an Oracle environment, tells you how to enable RADIUS authentication and accounting, and introduces the challenge-response user interface which third party vendors can customize.

Chapter 4, "Configuring CyberSafe Authentication"—This chapter discusses how to configure Oracle for use with CyberSafe, and provides a brief overview of steps to configure CyberSafe to authenticate Oracle users.

Chapter 5, "Configuring Kerberos Authentication"—This chapter discusses how to configure Oracle for use with MIT Kerberos, and provides a brief overview of steps to configure Kerberos to authenticate Oracle users.

Chapter 6, "Configuring SecurID Authentication"—This chapter discusses how to configure the SecurID authentication adapter in combination with the Oracle server and Oracle clients. It includes system requirements and known limitations. It also contains troubleshooting information if you experience problems while configuring the SecurID authentication adapter.

---

**Note:** For a complete list of Oracle Advanced Security option error messages see the Oracle Network Products Troubleshooting Guide.

---

Chapter 7, "Configuring Identix Biometric Authentication"—This chapter describes how to configure and use the Oracle Biometric authentication adapter, which enables the use of the Identix fingerprint authentication device.

Chapter 8, "Configuring DCE GSSAPI Authentication"—This chapter describes how to configure the Oracle DCE GSSAPI authentication adapter to provide DCE authentication even if you are not using other DCE services in your network.

Chapter 9, "Configuring SSL Authentication"—This chapter discusses the SSL feature of the Oracle Advanced Security option. It explains how to configure SSL and how to use the Oracle Wallet Manager to manage wallets and trustpoints.

Chapter 10, "Choosing and Combining Authentication Methods"—This chapter describes how to use conventional username / password authentication even if you

have configured another authentication service. It also discusses how to configure your network to use one or more authentication services in your network using the Oracle Advanced Security option and how to set up more than one authentication service on a client or on a server.

### Part II: Oracle Advanced Security and Oracle DCE Integration

Chapter 11, "Overview of Oracle DCE Integration"—This chapter provides a brief discussion of OSF's DCE and Oracle's DCE Integration.

Chapter 12, "Configuring DCE for Oracle DCE Integration"—This chapter describes what you need to do to configure DCE to use Oracle DCE Integration. It also describes how to configure the DCE CDS naming adapter.

Chapter 13, "Configuring Oracle for Oracle DCE Integration"—This chapter describes the DCE parameters that you need to add to the SQL*Net or Net8 configuration files to enable clients and servers to access Oracle servers in the DCE environment. It also describes some Oracle Server configuration that you need to perform, such as setting up DCE groups to map to external roles. Additionally, it describes how to configure clients to use the DCE CDS naming adapter.

Chapter 14, "Connecting to an Oracle Database in DCE"—This chapter discusses how to connect to an Oracle database in a DCE environment.

Chapter 15, "DCE and Non-DCE Interoperability"—This chapter discusses how clients outside of DCE can access Oracle databases using another protocol such as TCP/IP.

### Appendices

Appendix A, "Encryption and Checksumming Parameters"—This appendix lists and describes encryption and checksumming configuration parameters of the Oracle Advanced Security option.

Appendix B, "Authentication Parameters"—This appendix lists and describes authentication configuration file parameters of the Oracle Advanced Security option.

Appendix C, "Integrating Authentication Devices Using RADIUS"—This appendix explains how third party vendors of authentication devices can customize this graphical user interface used in RADIUS challenge-response authentication.

## Notational Conventions

The following syntax conventions are used in this guide:

| | |
|---|---|
| *Italic Font* | Italic characters indicate that the parameter, variable, or expression in the command syntax must be replaced by a value that you provide. Italics may also indicate emphasis or the first mention of a technical term. |
| Monospace Font | Monospace font indicates something the computer displays.<br>**Note:** In some cases, brackets surround certain words (for example, `<pin><passcode>`) to more clearly separate words in a command. |
| **Bolded Monospace Font** | Bolded monospace font indicates:<br>■ Terms defined in the Glossary<br>■ Text you need to enter exactly as shown.<br>**Note:** In some cases, angle brackets surround certain words (for example, `<pin><passcode>`) to more clearly separate words in a command. |
| Punctuation | Punctuation other than brackets and vertical bars must be typed as shown. |
| [ ] | Brackets enclose optional items. Do not type the brackets. |
| ( ) | Parentheses enclose all SQL*Net and Net8 Keyword-Value pairs in connect descriptors. They must be entered as part of the connect descriptor, as in `(KEYWORD=value)`. |
| \| | A vertical bar represents a choice of two or more options. You must type one of the options separated by the vertical bar. Do not type the vertical bar. |
| UPPERCASE | Uppercase characters within the text represent command names and parameters. |

## Related Publications

To install and configure Oracle Advanced Security option software on your particular platform, refer to the Oracle platform-specific documentation.

In addition, see the following documents for detailed information about Oracle network products that applies across platforms:

- *Net8 Administrator's Guide*
- *Oracle8i Distributed Database Systems*

For information on roles and privileges, see:

- *Oracle8i Administrator's Guide*

For third-party vendor documentation on security and single sign-on features see:

- *RADIUS Administrator's Guide*
- *Security Dynamics' ACE/Server Installation Manual, release 1.3*
- *Security Dynamics' ACE/Server Version 1.3 Administration Manual*
- *ACE/Server Version 2.0 Client for UNIX*
- *CyberSafe Challenger Release Notes, release 5.2.6*
- *CyberSafe Challenger Administrator's Guide, release 5.2.6*
- *CyberSafe Challenger Navigator Administrator's Guide, release 5.2.6*
- *CyberSafe Challenger UNIX User's Guide, release 5.2.6*
- *CyberSafe Challenger Windows and Windows NT User's Guide, release 5.2.6*

For information on MIT Kerberos see:

- CyberSafe Challenger documentation
- Notes on building and installing Kerberos from Kerberos V5 source distribution
- CNS (Cygnus Network Security) documentation from http://www.cygnus.com/library-dir.html

For additional information about the OSF Distributed Computing Environment (DCE), refer to the following OSF documents published by Prentice Hall, Inc.:

- *OSF DCE User's Guide and Reference*
- *OSF DCE Application Development Guide*
- *OSF DCE Application Development Reference*

- *OSF DCE Administration Guide*
- *OSF DCE Administration Reference*
- *OSF DCE Porting and Testing Guide*
- *Application Environment Specification/Distributed Computing*
- *OSF DCE Technical Supplement*

For information about Identix products, refer to the following Identix documentation.

Client side documentation:

- *Identix TouchNet II User's Guide*

Server side documentation:

- *Identix TouchNet II System Administrator's Guide*

# Part I

# Oracle Advanced Security Features

Part I of this document includes information on how to configure security and authentication into your existing Net8 release 8.1.5 network. Refer also to the port-specific documentation on how to install and configure the Oracle Advanced Security option.

The following chapters of the *Oracle Advanced Security Option Administrator's Guide* provide generic information on the security related features of the Oracle Advanced Security option.



**For information on DCE integration:** See Part II, "Oracle Advanced Security and Oracle DCE Integration"

# 1

# Introduction to Oracle Advanced Security

This chapter introduces the Oracle Advanced Security option **encryption**, **checksumming**, and **authentication** features. These features are available to network products using Net8, including Oracle8*i*, Designer 2000, Developer 2000, and any other Oracle or third-party products that support Net8.

Topics covered in this chapter:

# About the Oracle Advanced Security Option

The Oracle Advanced Security option (formerly Secure Network Services and Oracle Advanced Networking Option) provides a comprehensive suite of security features to protect enterprise networks and securely extend corporate networks to the Internet. The Oracle Advanced Security option provides a single source of integration with network encryption and authentication solutions, single sign-on services, and security protocols. By integrating industry standards, it delivers unparalleled security to the Oracle network and beyond.

## Network Security in a Distributed Environment

Organizations around the world are deploying distributed databases and client/server applications in record numbers, often on a national or global scale, based on Net8 and Oracle8*i*. This proliferation of distributed computing has been matched by an increase in the amount of information that organizations now place on computers. Employee records, financial records, product testing information, and other sensitive or critical data have moved from filing cabinets into file structures. The volume of critical or sensitive information on computers has increased the value of data that may be compromised.

The increased distribution of data in these environments brings with it some serious security threats:

- Data tampering—Distributed environments bring with them the possibility that a malicious third party can execute a computer crime by actually tampering with data as it moves between sites.
- Eavesdropping and data theft—Over the Internet and in Wide Area Network (WAN) environments, both public carriers and private network owners often route portions of their network through insecure land lines, extremely vulnerable microwave and satellite links, or a number of servers, leaving valuable data open to view for any interested party. In Local Area Network (LAN) environments within a building or campus, the potential exists for insiders with access to the physical wiring to view data not intended for them.
- Falsifying user identities—In a distributed environment, it becomes more feasible for a user to falsify an identity to gain access to sensitive and important information. How can you be sure that user Smith connecting to Server A from Client B really is user Smith?

  Moreover, in distributed environments, malefactors may 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 routed instead to a terminal masquerading as Server B.

- Administering too many passwords—In a distributed system, users may need to remember multiple passwords for the different applications and services that they use. For example, a developer may have access to an application in development on a workstation, a production system on a mini-computer, a PC for creating documents, and several mini-computers or workstations for testing, reporting bugs, configuration management, and so on. Administration of all these accounts and passwords is complex and time-consuming.

  Users generally respond to multiple accounts in one of two ways:

  - If they can choose their own passwords, they may standardize them so that they are the same on all machines. This results in a potentially large exposure in the event of a compromised password. Or they may use passwords with slight variations that can be easily guessed from knowing one password.
  - Users with complex passwords may simply write them down or forget them.

  Either strategy severely compromises password secrecy and service availability.

## Features of the Oracle Advanced Security Option

The Oracle Advanced Security option protects against these threats to the security of distributed environments. Specifically, the Oracle Advanced Security option provides the following features, each of which is described in the next few pages.

- Data Integrity—to ensure that data is not modified during transmission
- Data Privacy—to ensure that data is not disclosed during transmission
- Authentication—to ensure that users', hosts', and clients' identities are correctly known, and to provide for single sign-on capability in place of using multiple passwords
- Authorization—to ensure that a user, program, or process receives the appropriate privileges to access an object or set of objects

### Data Integrity

To ensure that data has not been modified, deleted, or replayed during transmission, the Oracle Advanced Security option optionally generates a

cryptographically secure message digest—through cryptographic checksums using the MD5 algorithm—and includes it with each packet sent across the network.

Moreover, the SSL feature of the Oracle Advanced Security option allows the use of the Secure Hash Algorithm (SHA). SHA is slightly slower than MD5, but produces a larger message digest to make it more secure against brute-force collision and inversion attacks.

### Data Privacy

The Oracle Advanced Security option ensures data privacy through both RSA and DES encryption.

- RSA Encryption—An encryption module that uses the RSA Data Security RC4™ encryption algorithm. Using a secret, randomly-generated key for every 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, and 128 bits.

  Since the Oracle Advanced Security option RSA RC4 40-bit implementation meets the U.S. government export guidelines for encryption products, Oracle provides an export version of the media and exports it to all but a few countries, allowing most companies to safeguard their entire worldwide operations with this software.

- DES (Data Encryption Standard) Encryption—The U.S. Data Encryption Standard required for financial and many other institutions. The Oracle Advanced Security option for Domestic Use offers a standard, optimized 56-bit key DES encryption algorithm. Due to current U.S. government export restrictions, standard DES is initially available only to customers located in the U.S.A. and Canada. For customers located outside the U.S.A. and Canada, the Oracle Advanced Security option for Export Use also offers DES40, a version of DES which combines the standard DES encryption algorithm with the international availability of a 40-bit key. Selecting the algorithm to use for network encryption is a user configuration option, allowing varying levels of security and performance for different types of data transfers.

**More Information:** For more information on encryption and checksumming, see Chapter 2, "Configuring Encryption and Checksumming" and Appendix A, "Encryption and Checksumming Parameters".

## Authentication

Establishing user identity is also of primary concern in distributed environments; otherwise, there can be little confidence in limiting privileges by user. The Oracle Advanced Security option release 8.1.5 provides authentication through Oracle authentication adapters that support third-party authentication services such as Kerberos, CyberSafe TrustBroker (a Kerberos-based authentication server), SecurID, Identix TouchNet II, and RADIUS. These adapters are described later in this chapter.

**Centralized Authentication** Many of the Oracle Advanced Security option authentication methods use centralized authentication. This can give you high confidence in the identity of users, clients, and servers in distributed environments. Having a central facility 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 nodes on a network faking their identities.

Centralized authentication can also provide the benefit of single sign-on for users. Single sign-on allows users to access multiple accounts and applications with a single password, eliminates the need for multiple passwords, and simplifies management of user accounts and passwords for system administrators.

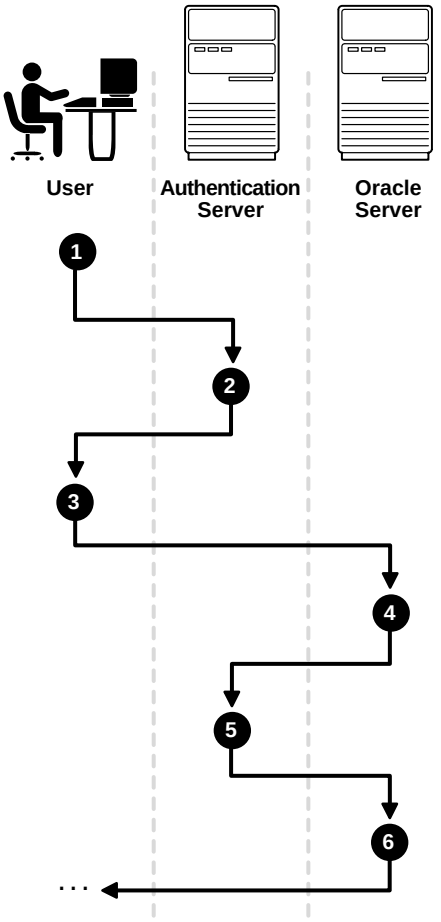
---

**Note:** Oracle Corporation does not provide centralized authentication servers. Rather, it supports only the authentication services provided through other vendors' security services or third-party Kerberos-based servers such as CyberSafe. For a list and brief description of authentication methods supported by the Oracle Advanced Security option, see "Authentication Methods Supported" on page 1-7.

---

**How Centralized Authentication Works** Figure 1–1 illustrates how a centralized network authentication service typically operates.

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




1. A user (client) requests authentication services, providing some identification—such as a token or password—proving that the user is who he or she claims to be.
2. After authenticating the user, the authentication server passes a ticket or credentials back to the client. This ticket may include an expiration time.
3. The client can now take these credentials and pass them to the Oracle server while asking for a service, such as connection to a database.
4. The server, to verify that the credentials are valid, sends them back to the authentication server.
5. If the authentication server accepts the credentials, it notifies the Oracle server.
6. The Oracle server performs the requested task for the user. If the credentials are not accepted, the requested service is denied.

**Authentication Methods Supported** The Oracle Advanced Security option supports the following authentication methods:

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

You can use the SSL feature of the Oracle Advanced Security option to secure communications between any client and any server. Specifically, you can use SSL to authenticate:

- any client or server to one or more Oracle servers
- an Oracle server to any client

You can use SSL features by themselves or in combination with other authentication methods supported by the Oracle Advanced Security option. For example, you can use SSL along with Kerberos, using the encryption provided by SSL in combination with the Kerberos authentication method.

You can configure SSL to require server authentication only, or both client and server authentication.

**RADIUS**—RADIUS (Remote Authentication Dial-In User Service), a client-server security protocol, is most widely known for enabling remote authentication and access. The Oracle Advanced Security option uses this emerging standard in a client-server network environment to enable use of any authentication method that supports the RADIUS protocol. You can use RADIUS with a variety of authentication methods, including token cards and smartcards.

**Kerberos and CyberSafe**—The Oracle Advanced Security option support for Kerberos and CyberSafe provides the benefits of single sign-on and centralized authentication in an Oracle environment. Kerberos is a trusted third-party authentication system that relies on shared secrets. It assumes that the third party is secure. It provides single sign-on capabilities, centralized password storage, database link authentication, and enhanced PC security. It does this through Kerberos authentication and through the CyberSafe TrustBroker, a Kerberos-based authentication server.

---

**Note:** Oracle authentication for Kerberos provides database link authentication (also called "proxy authentication"). CyberSafe and SecurID do *not* provide support for proxy authentication.

---

**Smartcards (RADIUS-Compliant)**—This authentication method uses a hardware device that looks much like a credit card. It has memory and a processor and is read by a smartcard reader located at the client workstation.

Smartcards offer the following benefits:

- Increased security—The smartcard relies on two-factor authentication. The smartcard can be locked, and only the user possessing the card and knowing the correct PIN can unlock it.
- Improved performance—Some sophisticated smartcards contain hardware-based encryption chips that can provide better throughput than software-based implementations. A smartcard can store a user name.
- Accessibility from any workstation—The user logs in simply by inserting the smartcard in a hardware device which reads the card and prompts the user for whatever authentication information the card requires, for example, a PIN. Once the user enters the correct authentication information, the smartcard generates and enters whatever other authentication information is required.

**Token Cards (SecurID and RADIUS-Compliant)**—Token cards can provide improved 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) which the user then types into a token card. The token card provides a response, namely, another number cryptographically-derived from the challenge, which the user then offers to the server.

Token cards provide the following benefits:

- Ease of use—Users need only remember, at most, a personal identification number (PIN) instead of multiple passwords.
- Ease of password management—This is because there is one token card rather than multiple passwords.
- Enhanced password security—To masquerade as a user, a malefactor would have to have the token card as well as the PIN required to operate it.
- Enhanced accountability—Token cards provide a stronger authentication mechanism.

You can use SecurID tokens through either SecurID or through RADIUS.

**Bull ISM**—ISM (Integrated System Management) is an offering of Bull Worldwide Information Systems that provides system administrators with a variety of management tools. This authentication method is available on the AIX platform only. See your AIX-specific documentation for more information.

**Biometric Authentication (Identix)**—Identix Biometric Authentication is used on both the clients and Oracle servers to communicate biometric authentication data between the authentication server and the clients.

### Authorization

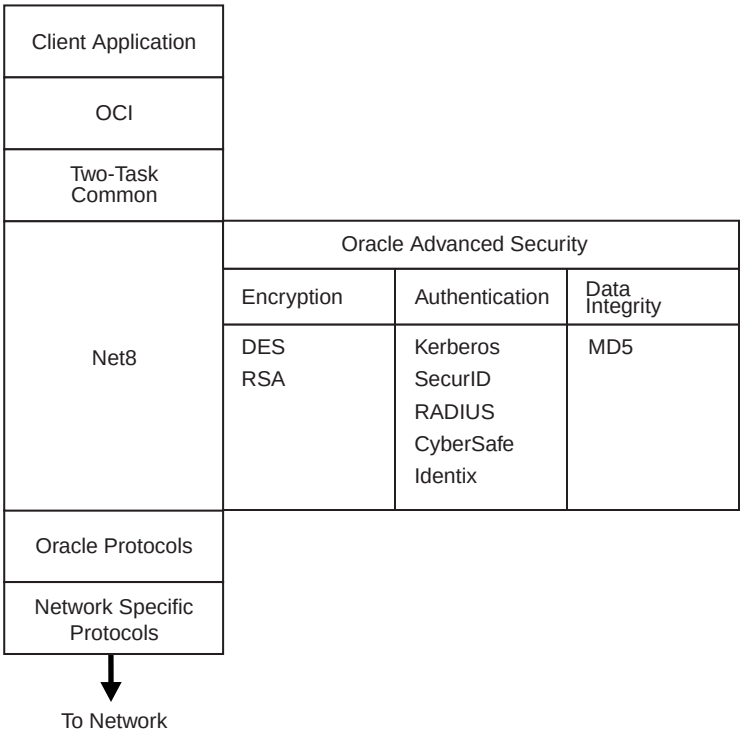
User authorization, already a standard features of Oracle8*i*, is significantly enhanced by using the authentication methods supported by the Oracle Advanced Security option. For example, on certain platforms such as Solaris, the Oracle Advanced Security option supports authorization with DCE.

## Architecture of the Oracle Advanced Security Option

The Oracle Advanced Security option is an add-on product to a standard Net8 Server or Net8 Client. Figure 1–2 shows the location of the Oracle Advanced Security option within a typical stack in an Oracle networking environment.

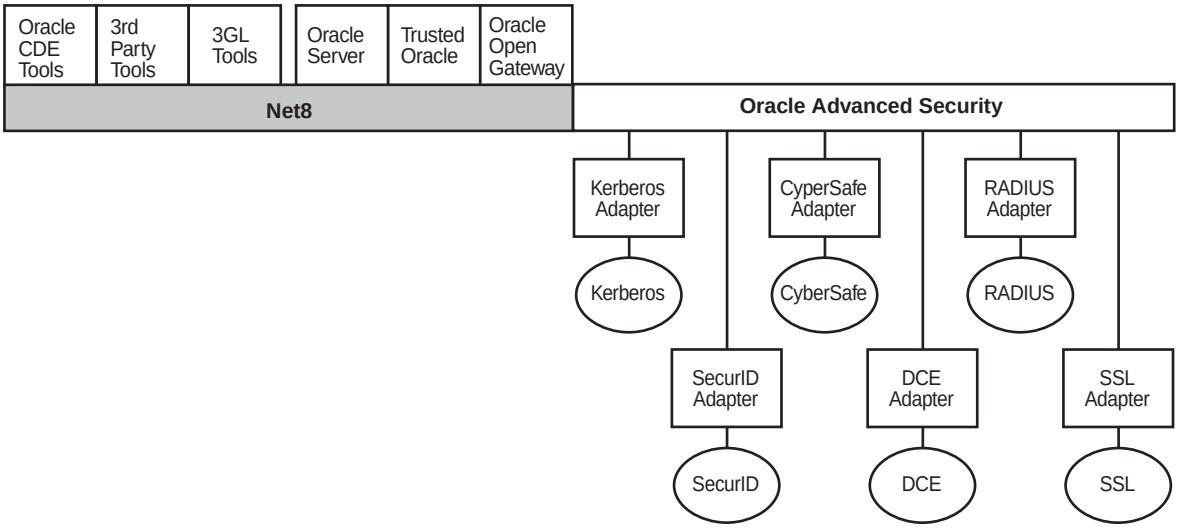
> **More Information:** For more information on stack communications in an Oracle networking environment, see *Net8 Administrator's Guide.*

***Figure 1–2 Oracle Advanced Security in an Oracle Networking Environment***




The Oracle Advanced Security option supports authentication through adapters that are very much like the existing Oracle protocol adapters. As Figure 1–3 shows, authentication adapters integrate below the Net8 interface and allow existing applications to take advantage of new authentication systems transparently, without any changes to the application.

***Figure 1–3 Net8 with Authentication Adapters***




## Secure Data Transfer Across Network Protocol Boundaries

The Oracle Advanced Security option is fully supported by the Oracle Connection Manager, making secure data transfer a reality across network protocol boundaries. Clients using LAN protocols such as NetWare (SPX/IPX), for instance, can now 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

The Oracle Advanced Security option is an add-on product to standard Net8 Server or Net8 Client. It is an extra cost item, and, to be functional, must be purchased on both the client and the server.

The Oracle Advanced Security option release 8.1.5 requires Net8 release 8.1.5.

The Oracle Advanced Security option release 8.1.5 supports Oracle 8*i* Enterprise Edition.

Install the Oracle Advanced Security option on all clients and servers where the Oracle Advanced Security option is required.

> **Note:** The Oracle Advanced Security option release 8.1.5 provides secure communication when used with earlier releases (such as 1.0 and 1.1); however, the security functionality defaults to that provided by the earlier release.

| Authentication Method | System Requirements |
|---|---|
| SSL | A wallet that is compatible with the Oracle Wallet Manager, release 1.3-beta. Wallets created on earlier releases of the Oracle Wallet Manager are not forward compatible. |
| CyberSafe TrustBroker | CyberSafe GSS Runtime Library, version 1.1 or later, installed on both the machine that runs the Oracle client and on the machine that runs the Oracle server.<br>CyberSafe TrustBroker, release 1.2 or later installed on a physically secure machine that will run the authentication server.<br>CyberSafe TrustBroker Client, release 1.2 or later installed on the machine that runs the Oracle client. |
| Kerberos | MIT Kerberos Version 5, release 1.0<br>The Kerberos authentication server must be installed on a physically secure machine. |
| SecurID | ACE/Server 1.2.4 or higher running on the authentication server. |
| Identix Biometric | Identix hardware and driver installed on each Biometric Manager station and client. |
| 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*.<br>If you want to enable challenge-response authentication, you must run RADIUS on a platform which supports the Java Native Interface as specified in release 1.1 of the Java Development Kit from JavaSoft. |

# Oracle Configuration for Network Authentication

This section discusses parameters you set when configuring Oracle for network authentication. Specifically, it discusses 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

> **More Information:** For more specific information on configuring a particular authentication method, see that method's corresponding chapter in this Guide.

## Setting the SQLNET.AUTHENTICATION_SERVICES Parameter in SQLNET.ORA

For clients and servers to be able to use an Oracle authentication method, the following parameter must be in the sqlnet.ora file:

```
SQLNET.AUTHENTICATION_SERVICES=(oracle_authentication_method)
```

For example, the following parameter must be set in the sqlnet.ora files on all clients and servers that use the Kerberos Authentication:

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

## Verifying that REMOTE_OS_AUTHENT Is Not Set to TRUE

> **Attention:** Setting REMOTE_OS_AUTHENT to TRUE may create a security hole, because it allows someone using a non-secure protocol (for example, TCP) to perform an operating system-authorized login (formerly referred to as an OPS$ login).

It is strongly recommended that, when configuring the Oracle authentication methods, you add the following parameter to the initialization file used for the database instance:

```
REMOTE_OS_AUTHENT=FALSE
```

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 will fail, and the connection will be terminated.

If the following parameter is set in the sqlnet.ora file on either the client or server side:

```
SQLNET.AUTHENTICATION_SERVICES=(NONE)
```

the database will attempt to use the provided user name and password to log the user in. However, if REMOTE_OS_AUTHENT is set to FALSE, the connection will fail.

### 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 strongly recommends that you enter a null value for the OS_AUTHENT_PREFIX parameter in the init.ora file used for the database instance:

```
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 could result in previously created externally-identified users not being able to connect to the Oracle server.

---

To create a user, launch SQL*Plus and type:

```
SQL> CREATE USER os_authent_prefix username IDENTIFIED EXTERNALLY;
```

When OS_AUTHENT_PREFIX is set to a null value (""), you would create the user "king" with the following command:

```
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.

**Note:** This applies to creating Oracle users for use with all Oracle authentication methods.

**More Information:** See *Oracle8i Administrator's Guide* and *Oracle8i Distributed Database Systems.*

## Oracle Products Not Yet Supported

The Oracle Advanced Security option requires Net8 to transmit data securely. Accordingly, the Oracle Advanced Security option's authentication features are not currently supported by *some* parts of Oracle Financial, Human Resource, and Manufacturing Applications when they are running on the Windows platform. The portions of these products that use Oracle Display Manager (ODM) cannot yet take advantage of the Oracle Advanced Security option, since ODM does not currently use Net8.

# 2

# Configuring Encryption and Checksumming

This chapter covers the following topics:

# Encryption in the Oracle Advanced Security Option

This section discusses and compares the various encryption algorithms used in both the domestic and the export version of the Oracle Advanced Security option.

## Domestic and Export Versions

Due to export controls placed on encryption technology, the Oracle Advanced Security option is available in a Domestic Version and an Export Version.

| Domestic Version contains: | Export Version contains: |
|---|---|
| Diffie-Hellman key negotiation algorithm | Diffie-Hellman key negotiation algorithm |
| MD5 message digest algorithm | MD5 message digest algorithm |
| The following encryption algorithms (discussed below):<br>■ DES40<br>■ DES<br>■ RC4_40<br>■ RC4_56<br>■ RC4_128 | The following encryption algorithms (discussed below):<br>■ DES40<br>■ RC4_40 |

In certain circumstances, a special license may be obtained to export 56-bit encryption or the entire domestic version. Licenses are generally available to wholly owned subsidiaries of US corporations. Special licenses can be obtained to allow banks to have the export version updated to include DES. Export and import regulations vary from country to country and change from time to time, so it is important to check on current restrictions in your area.

## Encryption Algorithms Supported

This section discusses and compares the following encryption algorithms and their uses.



### DES Algorithm Provides Standards-Based Encryption

The Oracle Advanced Security option for Domestic Use provides the DES (Data Encryption Standard) algorithm for customers with specialized encryption needs. DES has been a U.S. government standard for many years and is sometimes mandated in the financial services industry. In most specialized banking systems today, DES is the algorithm used to protect large international monetary transactions. The Oracle Advanced Security option allows this high-security system to be used to protect any kind of application, without any custom programming.

In a secure cryptosystem, the plaintext (a message that has not been encrypted) can not be recovered from the ciphertext (the encrypted message) except by using the secret decryption key. In a "symmetric cryptosystem", a single key serves as both the encryption and the decryption key. DES is a secret-key, symmetric cryptosystem: both the sender and the receiver must know the same secret key, which is used both to encrypt and decrypt the message. DES is the most well-known and widely-used cryptosystem in the world.

### DES40 Algorithm is Provided for International Use

The DES40 algorithm, available internationally, is a variant of DES in which the secret key is preprocessed to provide 40 effective key bits. It is designed for use by customers outside the USA and Canada who want to use a DES-based encryption algorithm. This feature gives commercial customers a choice in the algorithm they use, regardless of their geographic location.

### RSA RC4 is a Highly Secure, High Speed Algorithm

The RC4 algorithm, developed by RSA Data Security Inc., has quickly become the de-facto international standard for high-speed data encryption. Despite ongoing attempts by cryptographic researchers to "crack" the RC4 algorithm, the only feasible method of breaking its encryption known today remains brute-force, systematic guessing, which is generally infeasible. RC4 is a stream cipher that operates at several times the speed of DES, making it possible to encrypt even large bulk data transfers with minimal performance consequences.

### RC4_56 and RC4_128 Can be Used by Domestic Customers

RC4 is a variable key-length stream cipher. The Oracle Advanced Security option release 8.1.5 for domestic use offers an implementation of RC4 with 56 bit and 128 bit key lengths. This provides strong encryption with no sacrifice in performance when compared to other key lengths of the same algorithm.

### RC4_40 Can be Used by Customers Outside the US and Canada

Oracle has obtained special license to export the RC4 data encryption algorithm with a 40-bit key size to virtually all destinations where other Oracle products are available. This makes it possible for international corporations to safeguard their entire operations with fast, strong cryptography.

### SSL Can Provide Triple-DES

The SSL feature of the Oracle Advanced Security option allows the use of triple-DES. This form of encryption involves encrypting input data three times, and this can occur in a number of ways. A potential drawback of triple-DES, depending on the speed of your communications channel, is that it requires more computing power than normal DES.

> **More Information:** See Chapter 9, "Configuring SSL Authentication"

## Checksumming in the Oracle Advanced Security Option

Encryption of network data provides data privacy, so no unauthorized party is able to view the plaintext data as it passes over the network. The Oracle Advanced Security option also provides protection against two other forms of attack: data modification attack and replay attack.

In a data modification attack, an unauthorized party on the network intercepts data in transit and changes portions of that data before retransmitting it. An example of this would be to change the dollar amount of a banking transaction.

In a replay attack, an entire set of valid data is repeatedly interjected onto the network. An example would be to repeat a valid bank account transfer transaction.

The Oracle Advanced Security option uses a keyed, sequenced implementation of the MD5 message digest algorithm to protect against both of these forms of active attack. This protection is activated independently from the encryption features provided.

## Diffie-Hellman-Based Key Management

The secrecy of encrypted data depends on the existence of a secret key shared between the communicating parties. Providing and maintaining such secret keys is known as "key management." In a multi-user environment, secure key distribution may be difficult; public-key cryptography was invented to solve this problem. The Oracle Advanced Security option uses the public-key based Diffie-Hellman key negotiation algorithm to perform secure key distribution for both encryption and crypto-checksumming.

When encryption is used to protect the security of encrypted data, keys should be changed frequently to minimize the effects of a compromised key. For this reason, the Oracle Advanced Security option key management facility changes the session key with every session.

### Overview of Site-Specific Diffie-Hellman Encryption Enhancement

The Oracle Advanced Security option includes the Diffie-Hellman key negotiation algorithm to choose keys both for encryption and for checksumming.

A key is a secret shared by both sides of the connection and by no one else. Without the key, it is extremely difficult to decrypt an encrypted message or to tamper undetectably with a crypto-checksummed message.

## Overview of Authentication Key Fold-in Encryption Enhancement

The purpose of the Authentication Key Fold-in encryption enhancement is to defeat a possible "person-in-the-middle attack" on the Diffie-Hellman key negotiation. It strengthens the session key significantly by combining a shared secret (which is known only to both 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 itself to the server, there is a shared secret that is only known to both sides. The Oracle Advanced Security option then combines the shared secret and Diffie-Hellman session key to generate a stronger session key that would defeat the person-in-the-middle who has no way of knowing the shared secret.

### Authentication Key Fold-in Feature Requires No Configuration

The authentication key fold-in encryption enhancement feature is included in the Oracle Advanced Security option and requires no configuration by the system or network administrator.

# Configuring Encryption and Checksumming

These configuration instructions assume that your Net8 network software has already been installed and is running.

As a network administrator, you set the encryption and checksumming configuration parameters.

The profile (sqlnet.ora) on clients and servers using encryption and checksumming must contain some or all of the parameters listed below.

## How Encryption and Checksumming are Activated

In any network connection, it is possible that both ends (client and server) may support more than one encryption algorithm and more than one cryptographic checksumming algorithm. When each connection is made, the server decides which algorithm to use, if any, based on the algorithms specified in the sqlnet.ora files.

When the server is trying to find a match between the algorithms it has made available and the algorithms the client has made available, it picks the first algorithm in its own list that also appears in the client’s list. If one side of the connection does not specify a list of algorithms, all the algorithms that are installed on that side are acceptable.

Encryption and checksumming parameters are defined by modifying a sqlnet.ora file for the clients and servers on your network.

> **More Information:** See Appendix A, "Encryption and Checksumming Parameters".

## Negotiating Encryption and Checksumming

To negotiate whether to turn on encryption or checksumming, you can specify four possible values for four of the Oracle Advanced Security option configuration parameters, each of which is described below:

- ACCEPTED
- REJECTED
- REQUESTED
- REQUIRED

The default value for these four parameters is ACCEPTED.

### ACCEPTED

*Turn on the security service if the other side wants it.*

My side of the connection does not desire the security service, but it will be allowed if the other side asks with a setting of REQUIRED or REQUESTED. If the other side is set to REQUIRED or REQUESTED, and an algorithm match is found, the connection will continue without error and with the security service turned on. If the other side is set to REQUIRED and no algorithm match is found, the connection will terminate 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 will continue without error and without the security service enabled.

### REJECTED

*Do not turn on the security service even if the other side wants it.*

My side of the connection specifies that the security service is not allowed. If the other side specifies REQUIRED, the connection will terminate with error message ORA-12650. If the other side is set to REQUESTED, ACCEPTED, or

REJECTED, the connection will continue without error and without the security service enabled.

### REQUESTED

*Turn on the security service if the other side allows it.*

My side of the connection specifies that the security service is desired, but not required. The security service will be active if the other side specifies ACCEPTED, REQUESTED, or REQUIRED. There must be a matching algorithm available on the other side, otherwise the service will not be activated. If the other side specifies REQUIRED and there is no matching algorithm, the connection fails.

### REQUIRED

*Turn on the security service or do not make the connection.*

My side of the connection specifies that the security service must be activated. The connection will fail if the other side specifies REJECTED or if there is no compatible algorithm on the other side.

The following table shows whether or not the security service will be turned on based on a combination of client and server configuration parameters. If either the server or client has specified REQUIRED, lack of a common algorithm will cause the connection to fail. Otherwise, if the service would be on, lack of a common service algorithm will result in the service being turned off.

| Server \ Client | Accepted | Rejected | Requested | Required |
|---|---|---|---|---|
| **Accepted** | OFF | OFF | ON | ON |
| **Rejected** | OFF | OFF | OFF | Connection will fail |
| **Requested** | ON | OFF | ON | ON |
| **Required** | ON | Connection will fail | ON | ON |

## Setting Encryption and Checksumming Parameters

> **More Information:** For a description of each parameter and a sample configuration file using encryption and checksumming, see Appendix A, "Encryption and Checksumming Parameters"
>
> For more detailed configuration information, see the Net8 Assistant HELP system.

You can enter or change encryption and checksumming parameter settings by using any text editor to modify the sqlnet.ora file or by using the Net8 Assistant.

### Using Net8 Assistant

This graphical interface tool makes it easy to set parameters in the sqlnet.ora file and other Oracle8*i* configuration files.

**To launch Net8 Assistant:**

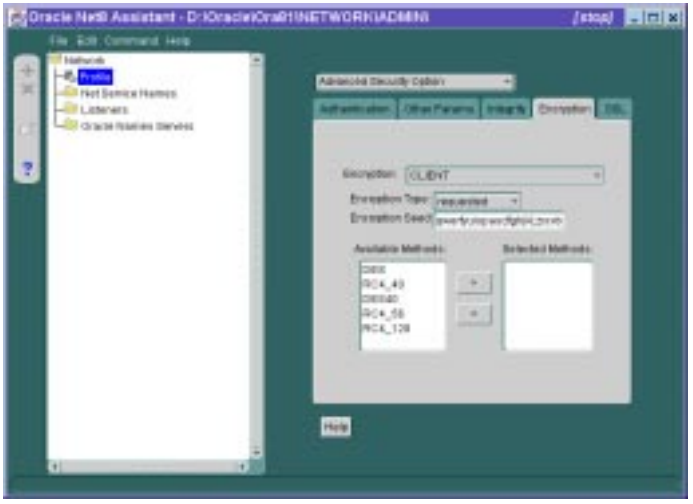
- On UNIX, run the script netasst at $ORACLE_HOME/bin.
- On Windows platforms, click the Start button > Programs > Oracle for Windows NT—*HOME_NAME* > Net8 Assistant.

**To begin configuring the Oracle Advanced Security option using Net8 Assistant:**
In the Net8 Assistant's left pane, click the Profile folder. Then go to the drop down list box at the top of the right pane, and select Advanced Security Option. The tabbed pages for the Oracle Advanced Security option appear.

**To save changes with Net8 Assistant:**
Go to the menu bar and click File > Save Network Configuration.

### Configure encryption on the client and the server

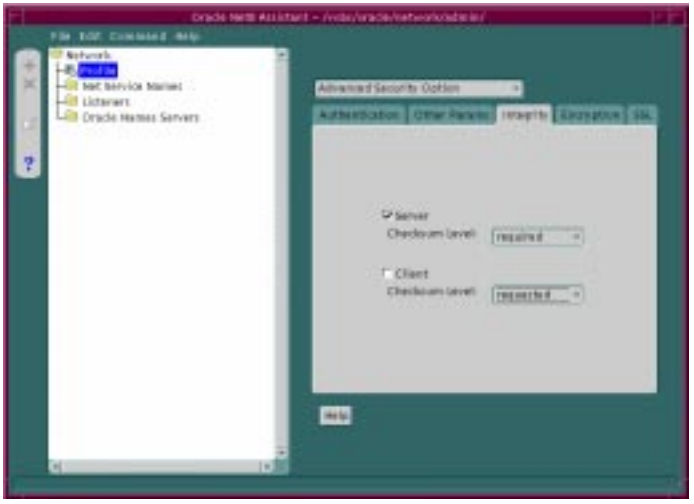
***Figure 2–1 Using Net8 Assistant to Set Encryption***



| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 2–1 on page 2-10.<br>**1.** Select the Encryption tab.<br>**2.** Depending on which machine you are configuring, in the Encryption list, select CLIENT or SERVER.<br>**3.** In the Encryption Type list, select requested, required, accepted, or rejected.<br>**4.** In the Encryption Seed box, type between 10 and 70 random characters.<br>**Note:** The encryption seed for the client should *not* be the same as that for the server.<br>**5.** Select an encryption method in the Available Methods list. Move it to the Selected Methods list by clicking the right arrow button [>]. Repeat for each additional method you want to use. | On the **Server**, set the following parameters:<br>SQLNET.ENCRYPTION_SERVER = [*accepted* \| *rejected* \| *requested* \| *required*]<br>SQLNET.ENCRYPTION_TYPES_SERVER = (*valid_encryption_algorithm [,valid_encryption_algorithm*])<br>SQLNET.CRYPTO_SEED = "*10-70 random characters*"<br>**Note:** The encryption seed for the server should *not* be the same as that for the client.<br>On the **Client**, set the following parameters:<br>SQLNET.ENCRYPTION_CLIENT = [*accepted* \| *rejected* \| *requested* \| *required*]<br>SQLNET.ENCRYPTION_TYPES_CLIENT = (*valid_encryption_algorithm* [,*valid_encryption_algorithm*])<br>SQLNET.CRYPTO_SEED = "*10-70 random characters*"<br>**Note:** The encryption seed for the client should *not* be the same as that for the server.<br>**For valid encryption algorithms:** See "Encryption and Checksumming Parameters" on page A-3. |

## Configure checksumming on the client and the server

***Figure 2–2 Using Net8 Assistant to Set Checksumming***



| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 2–2.<br>1. Select the Integrity tab.<br>2. Select the Server radio button to configure Server checksumming.<br>3. Click the Checksum Level drop-down list box to select one of the following checksum level values: required, requested, accepted, rejected.<br>4. Select the Client radio button to configure Client checksumming.<br>5. Click the Client Checksum Level drop-down list box to select one of the following checksum level values: required, requested, accepted, rejected. | On the **Server**, set the following parameters:<br>SQLNET.CRYPTO_CHECKSUM_SERVER = [*accepted* \| *rejected* \| *requested* \| *required*]<br>SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (*crypto_checksum_algorithm*)<br>On the **Client**, set the following parameter:<br>SQLNET.CRYPTO_CHECKSUM_CLIENT = [*accepted* \| *rejected* \| *requested* \| *required*]<br>SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (*crypto_checksum_algorithm*)<br>**Note:** Currently, the only supported crypto-checksum algorithm choice is RSA Data Security's MD5 algorithm. |

# 3

# Configuring RADIUS Authentication

This chapter tells you how to configure Oracle8*i* for use with RADIUS (Remote Authentication Dial-In User Service).

This chapter covers the following topics:

# RADIUS Overview

RADIUS (Remote Authentication Dial-In User Service) is a client-server security protocol most widely known for enabling remote authentication and access. The Oracle Advanced Security option uses this emerging standard in a client-server network environment.

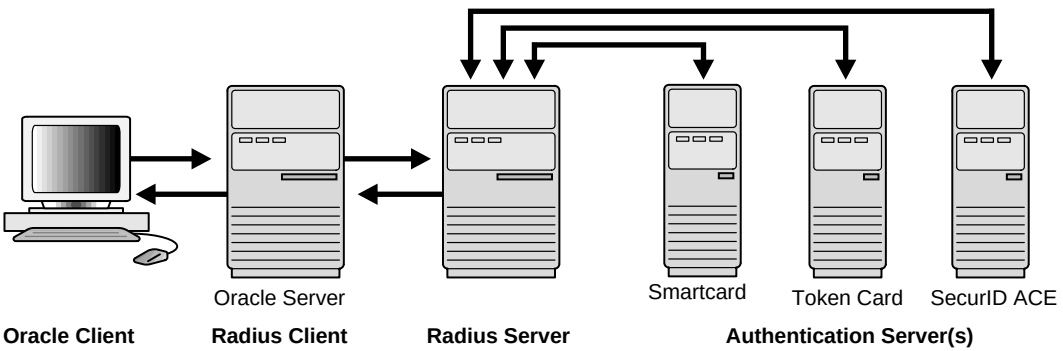
You can enable your network to use any authentication method that supports the RADIUS standard—including token cards and smartcards—simply by installing and configuring the RADIUS adapter. Moreover, when you use RADIUS, you can change your authentication method without modifying either the Oracle client or the Oracle server.

From the user's perspective, the entire authentication process takes place seamlessly and transparently. When the user seeks access to an Oracle server, the Oracle server, acting as the RADIUS client, notifies the RADIUS server. The RADIUS server then:

- looks up the user's security information
- passes authentication and authorization information between the appropriate authentication server(s) and the Oracle server
- logs—by means of the RADIUS accounting feature—such information as when, how often, and for how long the user logged on

## RADIUS in an Oracle Environment

***Figure 3–1 RADIUS in an Oracle Environment***



In an Oracle environment (Figure 3–1), the Oracle server acts as the RADIUS client; it passes information between the Oracle client and the RADIUS server. Similarly, the RADIUS server passes information between the Oracle server and the appropriate authentication server(s). To secure authentication information during transport, RADIUS converts it to a hash value.

The four components— Oracle client, Oracle server / RADIUS client, RADIUS server, and authentication server—can reside on the same machine or on separate machines. When the Oracle client and Oracle server reside on the same machine, they share the same sqlnet.ora file.

> **More Information:** For information about the sqlnet.ora file, see *Net8 Administrator's Guide*

The following table lists each component and the information it stores.

| Component | Stored Information |
|---|---|
| Oracle client | configuration setting for communicating through RADIUS |
| Oracle 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 |
| | an unlimited number of menu files enabling users already authenticated to select different login options without reconnecting |
| Authentication Server(s) | user authentication information such as passcodes and PINs, depending on the authentication method in use |

# RADIUS Authentication Modes

User authentication can take place in either of two ways:

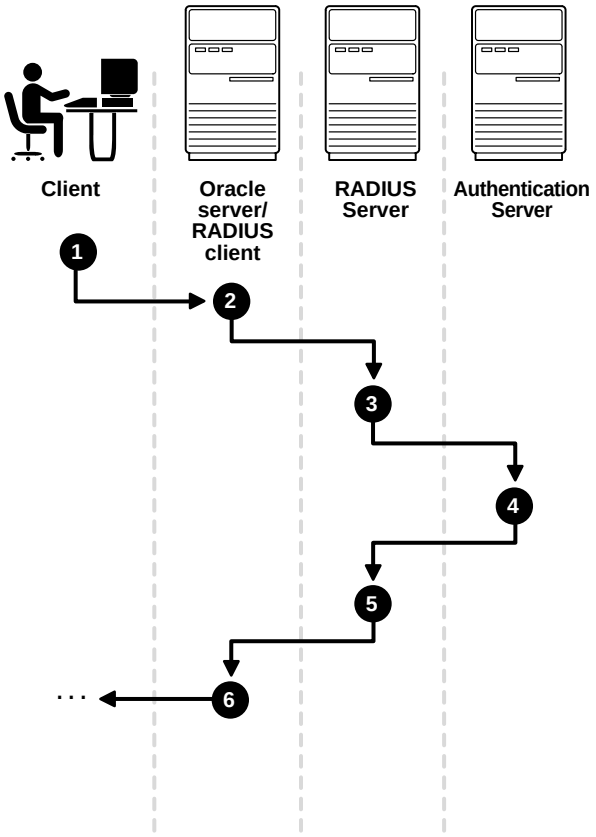
- Synchronous Authentication Mode
- Challenge-Response (Asynchronous) Authentication Mode

## Synchronous Authentication Mode

In the synchronous mode, RADIUS allows you to use various authentication methods, including passwords, SecurID token cards, and smartcards.

Figure 3–2 shows the sequence in which synchronous authentication occurs.

***Figure 3–2 Synchronous Authentication Sequence***




1. A user logs in by entering a connect string, passcode, or other value. The client machine passes this data to the Oracle server.
2. The Oracle 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 (e.g., Smartcard, SecurID ACE, ActivCard) for validation.
4. The authentication server sends back to the RADIUS server either an Access Accept or an Access Reject message.
5. The RADIUS server passes this response to the Oracle server / RADIUS client.
6. The Oracle server / RADIUS client passes the response to the Oracle client.

### Example: Synchronous Authentication with SecurID Token Cards

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

**More Information:** For more information on SecurID token cards, see "Authentication Methods Supported" on page 1-7 and Chapter 6, "Configuring SecurID Authentication". See also documentation provided by your SecurID vendor.

## Challenge-Response (Asynchronous) Authentication Mode

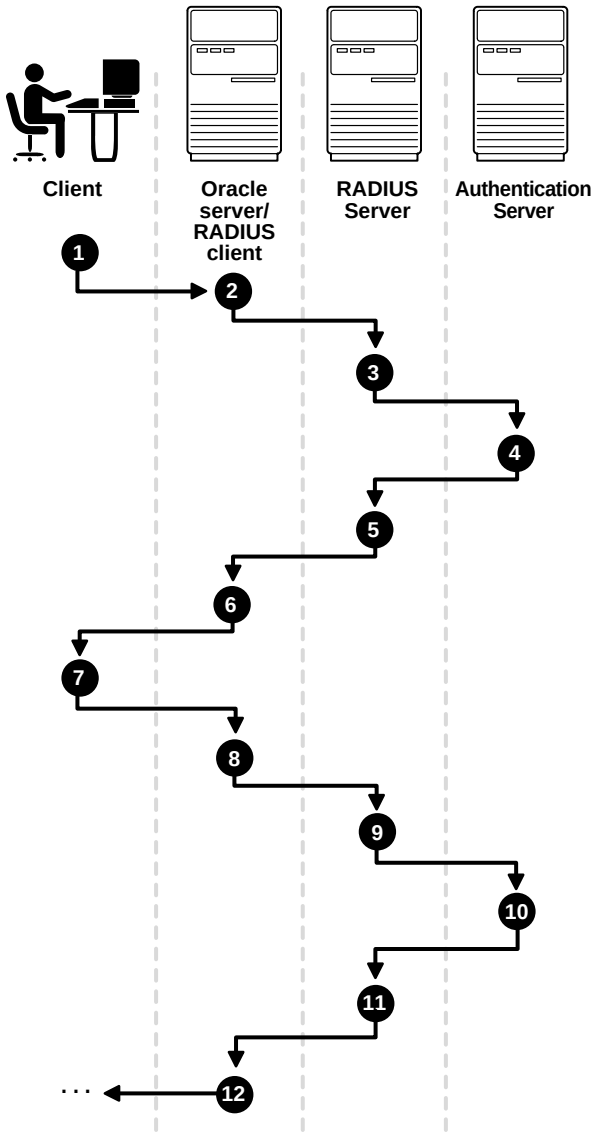
Figure 3–3 shows the sequence in which challenge-response, or asynchronous, authentication occurs.

---

**Note:** When your 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 3–3 Asynchronous Authentication Sequence***




1. A user seeks a connection to the Oracle server. The client machine passes this data to the Oracle server.
2. The Oracle server, acting as the RADIUS client, passes the data from the Oracle client to the RADIUS server.
3. The RADIUS server in turn, passes the data to the appropriate authentication server (e.g., Smartcard, SecurID ACE, ActivCard).
4. The authentication server sends back to the RADIUS server a challenge, for example, a random number.
5. The RADIUS server sends this challenge to the Oracle server / RADIUS client.
6. The Oracle server / RADIUS client, in turn, sends it to the Oracle client. A graphical interface presents the challenge to the user.
7. The user provides a response to the challenge. To formulate a response, the user may, for example, enter the received challenge into the token card. The token card then provides the user with a dynamic password that he or she enters into the graphical interface. The Oracle client passes the user's response to the Oracle server / RADIUS client.
8. The Oracle server / RADIUS client, in turn, 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 back to the RADIUS server either an Access Accept or an Access Reject message.
11. The RADIUS server passes this response to the Oracle server / RADIUS client.
12. The Oracle server / RADIUS client, in turn, passes the response to the Oracle client.

### Example: Asynchronous Authentication with Smartcards

With smartcard authentication, the user logs in by inserting the smartcard—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 this login information contained in the smartcard to the authentication server by way of the Oracle 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 server / RADIUS client—prompting the user for authentication information. That information could be, for example, a PIN as well as additional authentication information contained on the smartcard.

The Oracle client then sends the user's response to the authentication server by way of the Oracle server / RADIUS client 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 server / RADIUS client. 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 server by entering his or her password, the information is passed to the appropriate authentication server by way of the Oracle 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 server / RADIUS client. The user enters that challenge into the token, and the token then 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 server / RADIUS client 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 server / RADIUS client. 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 and Accounting

To enable RADIUS authentication and accounting, you perform the following general tasks, each of which is explained in the next several pages:



## Step 1: Install RADIUS on the Oracle server and the Oracle client

You install the RADIUS adapter along with the Oracle Advanced Security option during a typical installation of Oracle8*i*.

> **More Information:** For information on installing Oracle Advanced Security and the RADIUS adapter, see your platform-specific installation documentation for Oracle8*i*.

## Step 2: Configure RADIUS authentication

This section discusses the following topics.

- Basic RADIUS Configuration on the Oracle Client—The minimum configuration setting for the Oracle client.
- Basic RADIUS Configuration on the Oracle Server—The minimum configuration settings for the Oracle server.
- Configuration of Additional RADIUS Features—Configuration settings for various other features that can enhance your use of RADIUS. You can set parameters for the following:
  - the listening port of the primary RADIUS server
  - the length of time for the Oracle server to wait for responses from the primary RADIUS server
  - the number of times the Oracle server should resend messages to the primary RADIUS server
  - enabling or disabling challenge-response
  - the location of the secret key on the Oracle Server
  - an alternate RADIUS server

Unless otherwise indicated, you perform these configuration tasks by using the Net8 Assistant or by using any text editor to modify the sqlnet.ora file.

### Using Net8 Assistant

This graphical interface tool makes it easy to set parameters in the sqlnet.ora file and other Oracle8*i* configuration files.

**To launch Net8 Assistant:**

- On UNIX, run the script netasst at $ORACLE_HOME/bin.
- On Windows platforms, click the Start button > Programs > Oracle for Windows NT—*HOME_NAME* > Net8 Assistant.

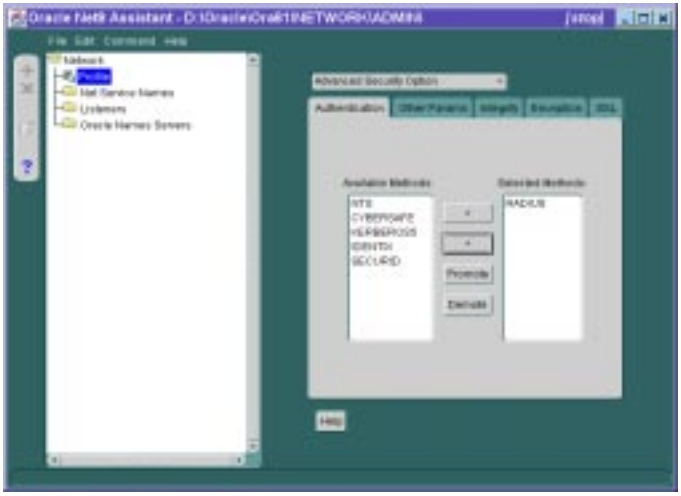
**To begin configuring the Oracle Advanced Security option using Net8 Assistant:**
In the Net8 Assistant's left pane, click the Profile folder. Then go to the drop down list box at the top of the right pane, and select Advanced Security Option. The tabbed pages for the Oracle Advanced Security option appear.

**To save changes with Net8 Assistant:**
Go to the menu bar and click File > Save Network Configuration.

### Basic RADIUS Configuration on the Oracle Client

Set the SQLNET.AUTHENTICATION_SERVICES parameter.

***Figure 3–4 Using Net 8 Assistant to Set the Authentication Services Parameter***



| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 3–4.<br>1. Select the Authentication tab.<br>2. In the Available Methods list, select RADIUS.<br>3. Move RADIUS to the Selected Methods list by clicking the right arrow button [>]. Move any other methods you want to use in the same way.<br>4. Arrange the selected methods in order of desired 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 RADIUS to be the first service used, put it at the top of the list. | Set the following parameter:<br>SQLNET.AUTHENTICATION_SERVICES=(*RADIUS*) |

### Basic RADIUS Configuration on the Oracle Server

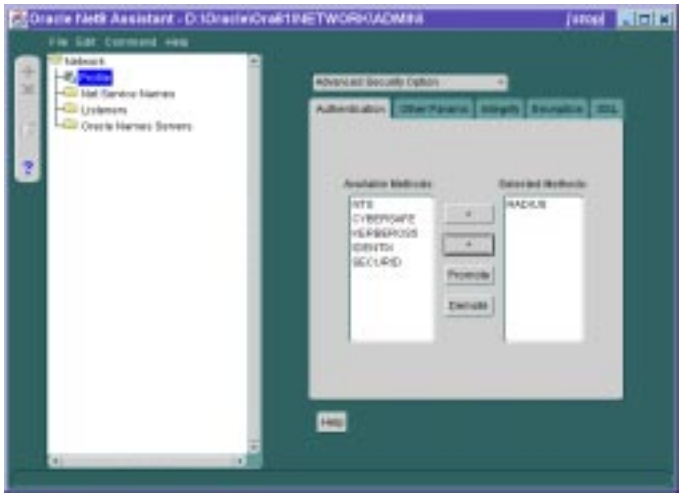
Do the following tasks, each of which is described below.

- Set the authentication services parameter
- Set the primary RADIUS server host name parameter
- Set Oracle server initialization parameters
- Add the classpath parameter in SQLNET.ORA

### Set the authentication services parameter

The SQLNET.AUTHENTICATION_SERVICES parameter sets the authentication method(s) you want to use.

***Figure 3–5 Using Net8 Assistant to Set the Authentication Services Parameter***

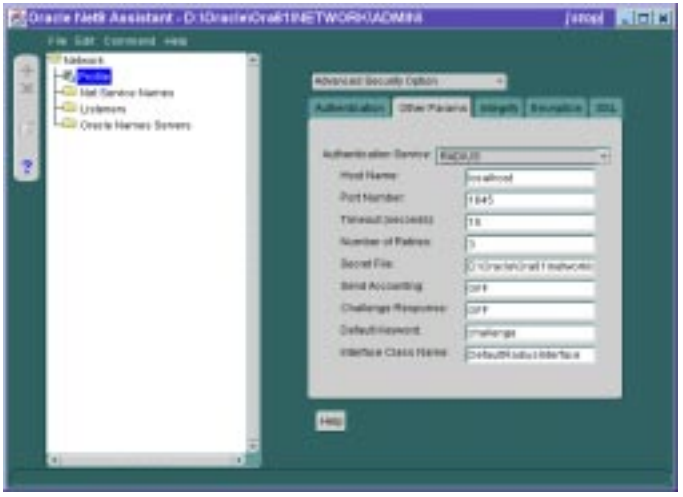


| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 3–5.<br>1. Select the Authentication tab.<br>2. In the Available Methods list, select RADIUS.<br>3. Move RADIUS to the Selected Methods list by clicking the right arrow button [>].<br>4. Arrange the selected methods in order of desired 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 RADIUS to be the first service used, put it at the top of the list. | Set the following parameter:<br>SQLNET.AUTHENTICATION_SERVICES=*RADIUS*) |

### Set the primary RADIUS server host name parameter

The SQLNET.RADIUS_AUTHENTICATION parameter sets the location of the primary RADIUS server. The default is the local host.

***Figure 3–6 Using Net8 Assistant to Set the Primary Radius Server Host Name Parameter***



| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 3–6.<br>1. Click the Other Params tab.<br>2. In the Authentication Service list, select RADIUS<br>3. In the Host Name box, the default is localhost. Accept this default or type the host name of your primary RADIUS server. | Set the following parameter:<br>SQLNET.RADIUS_AUTHENTICATION=<br>(*HOST NAME OR IP ADDRESS OF RADIUS SERVER*) |

### Set Oracle server initialization parameters

Configure the file init<*sid*>.ora which you can find in the directory $ORACLE_BASE\ADMIN\DB_NAME\PFILE. Specify the following values in this file:

```
REMOTE_OS_AUTHENT=FALSE
OS_AUTHENT_PREFIX=""
```

---

**Caution:** Setting REMOTE_OS_AUTHENT to TRUE may create a security hole because it allows someone using a non-secure protocol (for example, TCP) to perform an operating system-authorized login (formerly referred to as an OPS$ login).

---

**More Information:** For information on setting initialization parameters on the Oracle server, see *Oracle8i Reference* and *Oracle8i Administrator's Guide.*

### Add the classpath parameter in SQLNET.ORA

If you 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.

Use a text editor to add the following parameter to the file sqlnet.ora.

```
SQLNET.RADIUS_CLASSPATH=path/netradius.jar:path/ewt-opt-3_1_8_1.zip
```

For example:

```
SQLNET.RADIUS_CLASSPATH=/ohome/network_src/jlib/
                        netradius.jar:/ohome/network_src/jlib
                        /ewt-opt-3_1_8_1.zip
```

### Configuration of Additional RADIUS Features

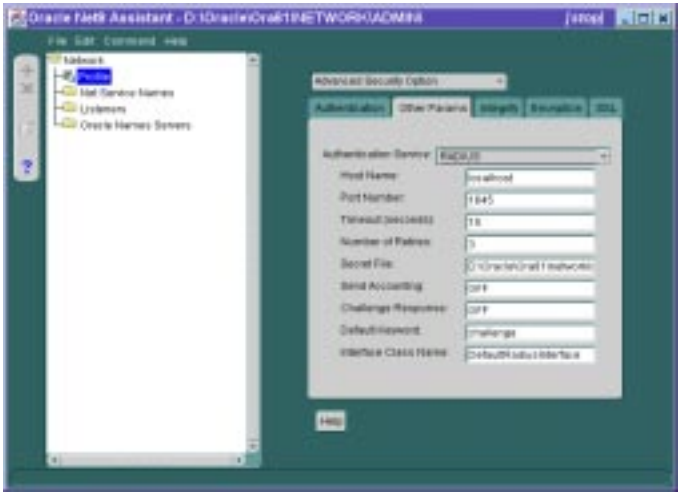
You can make the following additional RADIUS configurations by using the Net8 Assistant, or by modifying the file sqlnet.ora.

- Set the listening port of the primary RADIUS server
- Configure the time for the Oracle server to wait for responses from the primary RADIUS server
- Set the number of times the Oracle server should resend messages to the primary RADIUS server
- Configure challenge-response
- Set the location of the secret key on the Oracle Server
- Set parameters for an alternate RADIUS server

### Set the listening port of the primary RADIUS server

Do this by setting the SQLNET.RADIUS_AUTHENTICATION_PORT parameter. The default is 1645.

***Figure 3–7 Using Net8 Assistant to Set the Listening Port Of the Primary Radius Server***

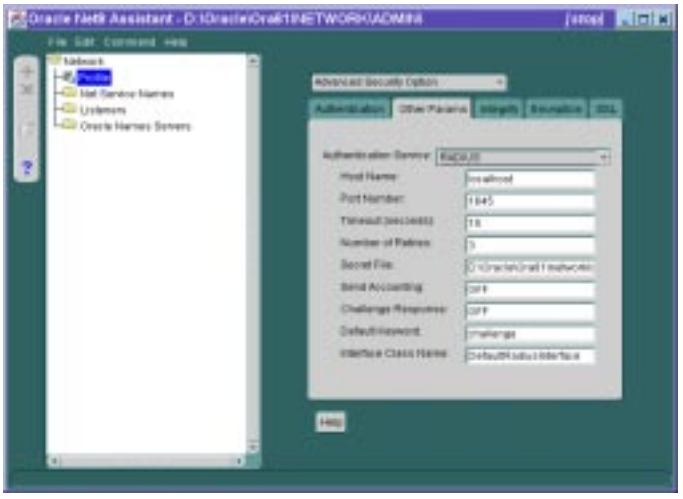


| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 3–7.<br>1. Select the Other Params tab.<br>2. In the Authentication Service list, select RADIUS.<br>3. In the Port Number box, the default is 1645. Accept this default or type the listening port number of your primary RADIUS server. | Set the following parameter:<br>SQLNET.RADIUS_AUTHENTICATION_PORT=(*1645*) |

### Configure the time for the Oracle server to wait for responses from the primary RADIUS server

Do this by setting the SQLNET.RADIUS_AUTHENTICATION_TIMEOUT parameter.

***Figure 3–8 Using Net8 Assistant to Configure the Time for the Oracle Server to Wait for Responses from the Primary Radius Server***

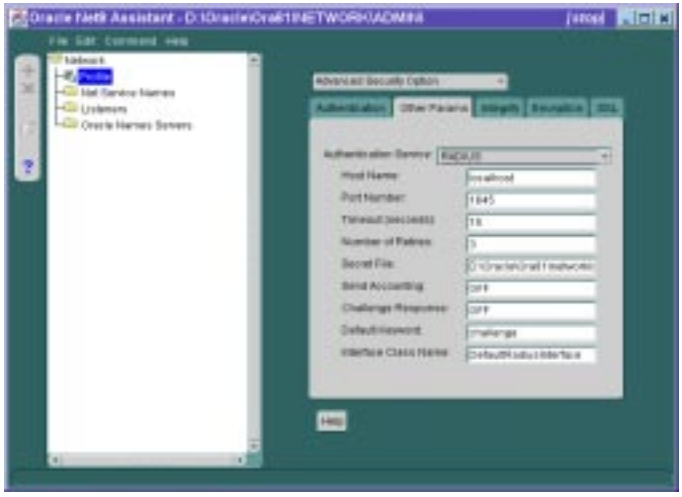


| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 3–8.<br>**1.** Select the Other Params tab.<br>**2.** In the Authentication Service list, select RADIUS<br>**3.** In the Timeout (seconds) box, the default is 15 seconds. Accept this default or type the number of seconds the Oracle server should wait for responses from the primary RADIUS server. | Set the following parameter:<br>SQLNET.RADIUS_AUTHENTICATION_ TIMEOUT=(*NUMBER OF SECONDS TO WAIT FOR RESPONSE*) |

### Set the number of times the Oracle server should resend messages to the primary RADIUS server

Do this by setting the SQLNET.RADIUS_AUTHENTICATION_RETRIES parameter. The default is 3.

***Figure 3–9 Using Net8 Assistant to Set the Number of Times the Oracle Server Should Resend Messages to the Primary Radius Server***



| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 3–9.<br>1. Select the Other Params tab.<br>2. In the Authentication Service list, select RADIUS<br>3. In the Number of Retries box, the default is 3. Accept this default or type the number of times the Oracle server should resend messages to the primary RADIUS server. | Set the following parameter:<br>SQLNET.RADIUS_AUTHENTICATION_RETRIES=(*NUMBER OF TIMES TO RE-SEND TO RADIUS SERVER*) |

**More Information:** For instructions on configuring RADIUS accounting, see "Step 5: Configure RADIUS Accounting" on page 3-26.

### Set the location of the secret key on the Oracle Server

Do this by setting the SQLNET.RADIUS_SECRET parameter.

---

**Note:** This parameter sets the *location* of the secret key; it does not specify the secret key itself.

---

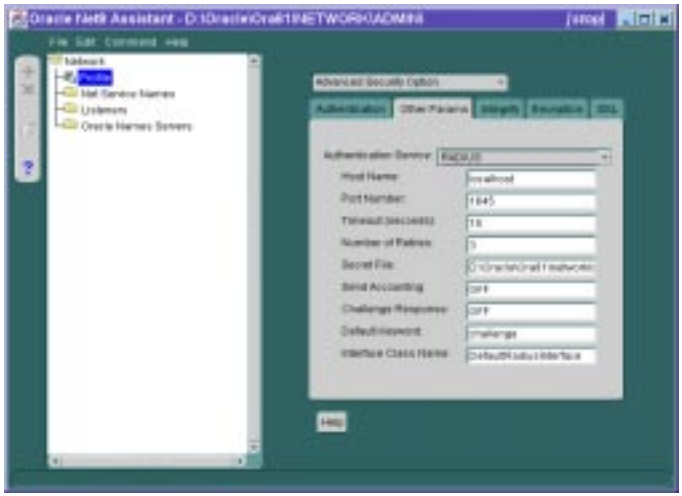
**More Information:** For information on specifying the secret key, see "Step 9: Specify the RADIUS secret key on the Oracle server" on page 3-29.

---

**Note:** For security reasons, Oracle recommends that you change this file to root access only.

---

***Figure 3–10 Using Net8 Assistant to Set the Location Of The Secret Key on the Oracle Server***



| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 3–10.<br>1. Select the Other Params tab.<br>2. In the Authentication Service list, select RADIUS<br>3. In the Secret File box, type the pathname of the secret key file. | Set the following parameter:<br>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 smartcard vendor would customize the Java interface so that the Oracle client reads data, such as a dynamic password, from the smartcard. Then, when the smartcard receives a challenge, it responds by prompting the user for more information, for example, a PIN.

---

**More Information:** For information on how to customize the challenge-response user interface, see Appendix C, "Integrating Authentication Devices Using RADIUS"

To configure challenge-response, do the following tasks, each of which is described below:

- Set the JAVA_HOME environment variable
- Set configuration parameters

**Set the JAVA_HOME environment variable** Set this environment variable to the JRE or JDK location on the system where the Oracle client is to run.

**On UNIX:**

At the command prompt, type the following:

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

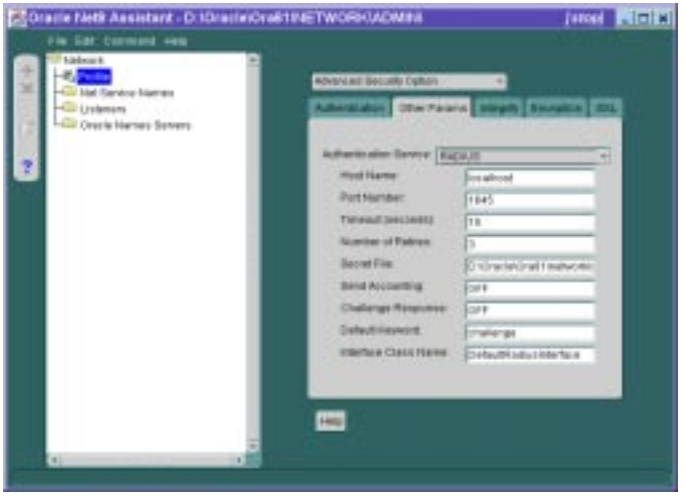
**On Windows NT:**

1. Click the Start button > Settings > ControlPanel > System > Environment.
2. Set the variable JAVA_HOME to: c:\java\jre1.1.7B

**Set configuration parameters** Set the following three parameters in the sqlnet.ora file as described below:

- SQLNET.RADIUS_CHALLENGE_RESPONSE
- SQLNET.RADIUS_CHALLENGE_KEYWORD
- SQLNET.RADIUS_AUTHENTICATION_INTERFACE

***Figure 3–11 Using Net8 Assistant to Configure Challenge-Response***



| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 3–11.<br>1. Select the Other Params tab.<br>2. In the Authentication Service list, select RADIUSI<br>3. In the Challenge Response box, the default is OFF. Accept this default or type ON to enable challenge-response.<br>4. In the Default Keyword[1] box, the default is challenge. Accept this default or type the keyword for requesting a challenge from the RADIUS server.<br>5. In the Interface Class Name box, the default is DefaultRadiusInterface. Accept this default or type the name of the class you have created to handle the challenge-response conversation between the Oracle client and the RADIUS server. | Set the following parameters:<br>SQLNET.RADIUS_CHALLENGE_RESPONSE=([*ON* \| *OFF*])<br>SQLNET.RADIUS_CHALLENGE_KEYWORD=(*KEYWORD*)<br>SQLNET.RADIUS_AUTHENTICATION_INTERFACE=(*package_name, delimited by a slash mark (/) rather than by a period (.) and followed by radius_interface_name*)<br>For example:<br>SQLNET.RADIUS_AUTHENTICATION_INTERFACE=vendor/net/ActivCardRadiusInterface |

[1] 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 allow the user not to use a password to verify his or her 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 on the configuration of the RADIUS server.

### Set parameters for an alternate RADIUS server

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

```
SQLNET.RADIUS_ALTERNATE=(HOSTNAME OR IP ADDRESS OF ALTERNATE RADIUS SERVER)

SQLNET.RADIUS_ALTERNATE_PORT=(1645)

SQLNET.RADIUS_ALTERNATE_TIMEOUT=(NUMBER OF SECONDS TO WAIT FOR RESPONSE)
SQLNET.RADIUS_ALTERNATE_RETRIES=(NUMBER OF TIMES TO RE-SEND TO RADIUS SERVER)
```

## Step 3: Add the RADIUS client name to the RADIUS server database

The RADIUS client is your Oracle server. See Figure 3–1 on page 3-2.

### Adding the RADIUS Client Name to the Livingston RADIUS Server, Version 2.0

---

**Note:** 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, you should consult the documentation for your particular RADIUS server for any unique interoperability requirements.

---

The clients file on the RADIUS server stores each RADIUS client's name or IP address and its shared secret. The pathname for this file is: /etc/raddb/clients.

**To add the RADIUS client name to the Livingston RADIUS Server 2.0 database:**

1. Open the clients file in any text editor. 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 client's name or IP address. In the KEY column, enter the shared secret.

**Note:** The value you enter in the CLIENT NAME column—whether it is the client's name or IP address—depends on your RADIUS server. See your RADIUS documentation.

3. Save and close the clients file.

   **More Information:** See the administration documentation for your RADIUS server.

## Step 4: Create and grant access to a user

1. Create and grant access to a user identified externally on the Oracle server.

   You can do this by launching SQL*Plus and typing the following commands:

```
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 a Windows NT platform, you can do this by using the Security Manager tool of the Oracle Enterprise Manager.

   **More Information:** See *Oracle8i Administrator's Guide* and *Oracle8i Distributed Database Systems.*

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

   **More Information:** See the administration documentation for your RADIUS server.

## Step 5: Configure RADIUS Accounting

RADIUS Accounting logs information about access to the Oracle server and stores it in a file on the RADIUS accounting server. You can use this feature only if both your RADIUS server and authentication server support it

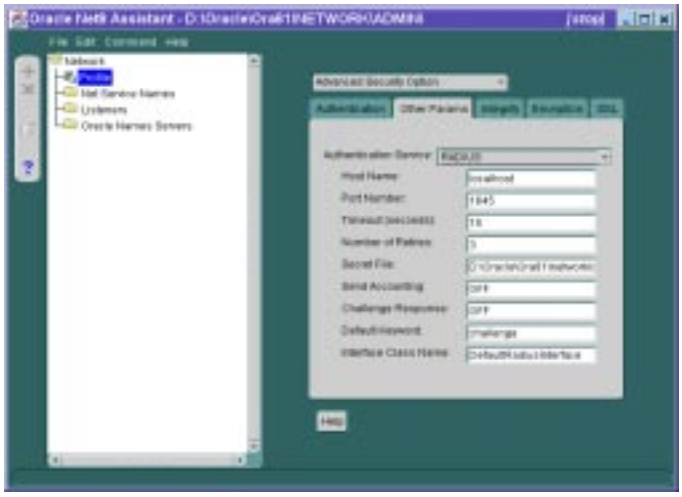
To enable or disable RADIUS accounting, you do the following:

- Set RADIUS Accounting on the Oracle Server
- Configure the RADIUS Accounting Server

### Set RADIUS Accounting on the Oracle Server

Do this by setting the SQLNET.RADIUS_SEND_ACCOUNTING parameter on the Oracle server.

***Figure 3–12 Using Net8 Assistant to Set RADIUS Accounting***



| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 3–12. | Set the following parameter: |
| 1. Select the Other Params tab. | SQLNET.RADIUS_SEND_ACCOUNTING= *ON* |
| 2. In the Authentication Service list, select RADIUSI | |
| 3. In the Send Accounting box, type ON to enable accounting or OFF to disable it. Default is OFF. | |

### 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.

**More Information:** For information on configuring RADIUS accounting, see the administration documentation for your RADIUS server.

## Step 6: Configure the authentication server for use with RADIUS

**More Information:** For instructions on configuring the authentication server, see the documentation for your authentication server. The section "Related Publications" on page xx contains a list of possible resources.

## Step 7: Configure the RADIUS server for use with the authentication server

**More Information:** See the documentation for your RADIUS server.

## Step 8: Create and grant roles

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

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

**To configure roles on the Oracle server:**

1. Use a text editor to set the initialization parameter OS_ROLES in the init.ora file on the Oracle server.
2. Stop and restart the Oracle server.
3. Use the `IDENTIFIED EXTERNALLY` syntax to create on the Oracle server each role you want the RADIUS server to manage.

**More Information:** See *Oracle8i Administrator's Guide.*

**To configure roles on the RADIUS server:**
Create role names with the following format:

```
ORA_DatabaseName.DatabaseDomainName_RoleName
```

| Parameter | Description |
|---|---|
| DatabaseName | The name of the Oracle 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 server belongs. The value is the same as the value of the db_domain initialization. |
| RoleName | The name of the role that you created in the Oracle server. |

For example:

```
ORA_JULIETDB.US.ORACLE.COM_MANAGER
```

**More Information:** See the administration documentation for your RADIUS server.

## Step 9: Specify the RADIUS secret key on the Oracle server

Do this by performing the following tasks:

1. Obtain the RADIUS secret key from your RADIUS server. The administrator of the RADIUS server creates a shared secret key for each RADIUS client, which could be as simple as the text'test123'.
2. On the Oracle server, create a directory $ORACLE_HOME/SECURITY.
3. Create the file radius.key to hold the shared secret from the RADIUS server. Place it in the directory you just created, namely, $ORACLE_HOME/SECURITY.
4. Copy the shared secret key and paste it (and nothing else) into the radius.key file you just created on the Oracle server.

**More Information:** For information on obtaining your secret key, see the administration documentation for your RADIUS server.

Using any text editor, open the file radius.key located in the path $ORACLE_HOME/SECURITY. Enter the RADIUS secret key and save the file.

> **Note:** For security reasons, Oracle recommends that you change this file to root access only.

## Logging in to the Database

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

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

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

If you are using the challenge-response (asynchronous) mode, launch SQL*Plus and, at the prompt, type the following:

```
CONNECT/@database_alias
```

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

# 4

# Configuring CyberSafe Authentication

This chapter contains information on how to configure Oracle for use with CyberSafe, as well as a brief overview of the steps to configure CyberSafe to authenticate Oracle users.

This chapter covers the following topics:

- Enabling CyberSafe Authentication
- Troubleshooting the Configuration of the CyberSafe Authentication Adapter

# Enabling CyberSafe Authentication

You enable CyberSafe authentication by performing the following tasks, each of which is fully described in the next few pages:

**Note: Perform these tasks in the order in which they are listed.**

## Step 1: Install the CyberSafe server

Do this on the machine that will act as the authentication server.

**More Information:** See the CyberSafe documentation listed in the"Related Publications" on page xx in the Preface of this guide.

## Step 2: Install the CyberSafe TrustBroker client

Do this on the machine that runs the Oracle server and the client.

**More Information:** See the CyberSafe documentation listed in "Related Publications" on page xx in the Preface of this guide.

## Step 3: Install the CyberSafe Application Security Toolkit

Do this on the client and on the server.

**More Information:** See the CyberSafe documentation listed in "Related Publications" on page xx in the Preface of this guide.

## Step 4: Configure a service principal for an Oracle server

For the Oracle server to validate the identity of clients, you need to configure a service principal for an Oracle server on the machine running the CyberSafe TrustBroker Master Server. If necessary, you must also configure a realm.

The name of the principal should have the following format:

*kservice/kinstance@REALM*

| | |
|---|---|
| kservice | a string that represents the Oracle service. This may or may not be the same as the database service name |
| kinstance | typically the fully-qualified name of the machine on which Oracle is running |
| REALM | the domain of the server |

**Note:** kservice is case-sensitive, and REALM must always be upper-case.

**Note:** The utility names in this section are actual programs that you run. However, the CyberSafe user name "cyberuser" and realm "SOMECO.COM" are examples only—the actual names will vary.

For example, if kservice is "oracle", and the fully-qualified name of the machine on which Oracle is running is "dbserver.someco.com", and the realm is "SOMECO.COM", the principal name would be:

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

**Note:** It is a common convention to use the DNS domain name as the name of the realm.

Run kdb5_edit as root to create the service principal.

```
# cd /krb5/admin
# ./kdb5_edit
```

To add a principal called "oracle / dbserver.someco.com@SOMECO.COM" to the list of server principals known by CyberSafe, from kdb5_edit type the following:

```
kdb5_edit:  ark oracle/dbserver.someco.com@SOMECO.COM
```

## Step 5: Extract the service table from CyberSafe

You need to extract a service table from CyberSafe and copy it to both the Oracle server and CyberSafe TrustBroker client machines. For example, to extract a service table for dbserver.someco.com, type the following from kdb5_edit:

```
kdb5_edit:  xst dbserver.someco.com oracle
'oracle/dbserver.someco.com@SOMECO.COM' added to keytab
'WRFILE:dbserver.someco.com-new-srvtab'
kdb5_edit:  exit
# /krb5/bin/klist -k -t dbserver.someco.com-new-srvtab
```

**Note:** If you do not enter a REALM (in the example, SOMECO.COM) when using *xst*, kdb5_edit uses the realm of the current host and displays it in the command output, as shown above.

After the service table has been extracted, verify that the new entries are in the table in addition to the old entries. If the new entries are not in the service table, or if you need to add additional new entries, use kdb5_edit to append the additional entries.

At this point, you need to move the CyberSafe service table to the CyberSafe TrustBroker client machine. If the service table is on the same machine as the CyberSafe client, you can simply move it (using a command such as that shown below). If the service table is on a different machine from the CyberSafe TrustBroker client, you must transfer the file with a program like FTP. For example, to move it, type the following:

```
# mv dbserver.someco.com-new-srvtab /krb5/v5srvtab
```

Remember to transfer the file in binary mode when you use FTP.

#### Ensure that the Oracle Server Can Read the Service Table

Make sure that the owner of the Oracle Server executable can read the service table (in the previous example, /krb5/v5srvtab). Set the file owner to the Oracle user or make the file readable by the group to which Oracle belongs. Do not make the file readable to all users, since this would allow a security breach.

## Step 6: Install an Oracle server

Do this on the same machine that is running the CyberSafe TrustBroker client.

**More Information:** See the Oracle8*i* installation documentation for your platform.

## Step 7: Install the Oracle Advanced Security and the CyberSafe adapter

You install the CyberSafe adapter—along with the Oracle Advanced Security option—during a typical installation of Oracle8*i*. Oracle Universal Installer guides you through the entire installation process.

**More Information:** See the Oracle installation documentation for your platform.

## Step 8: Configure Net8 and Oracle on your server and client

**More Information:** See your operating system-specific documentation.

## Step 9: Configure CyberSafe authentication

You must set certain parameters in the Oracle server and client sqlnet.ora files. The next few pages explain the following tasks.

- Configure the authentication service on the client and the server
- Configure CyberSafe authentication service parameters on the client and the server
- Set INIT.ORA Parameter

You can modify the sqlnet.ora file either by using the Net8 Assistant or by using any text editor. The following pages explain both methods. You modify the init.ora file by using a text editor.

**More Information:** See the Net8 Assistant on-line HELP system.

### Using Net8 Assistant

This graphical interface tool makes it easy to set parameters in the sqlnet.ora file and other Oracle8*i* configuration files.

**To launch Net8 Assistant:**

- On UNIX, run the script netasst at $ORACLE_HOME/bin.
- On Windows platforms, click the Start button > Programs > Oracle for Windows NT—*HOME_NAME* > Net8 Assistant.

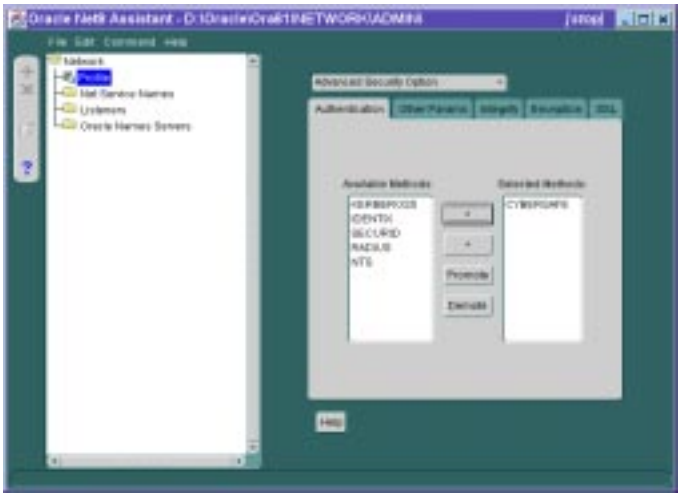
**To begin configuring the Oracle Advanced Security option using Net8 Assistant:**
In the Net8 Assistant's left pane, click the Profile folder. Then go to the drop down list box at the top of the right pane, and select Advanced Security Option. The tabbed pages for the Oracle Advanced Security option appear.

**To save changes with Net8 Assistant:**
Go to the menu bar and click File > Save Network Configuration.

### Configure the authentication service on the client and the server

Do this by setting the SQLNET.AUTHENTICATION_SERVICES parameter.

***Figure 4–1 Using Net8 Assistant to Configure Authentication***

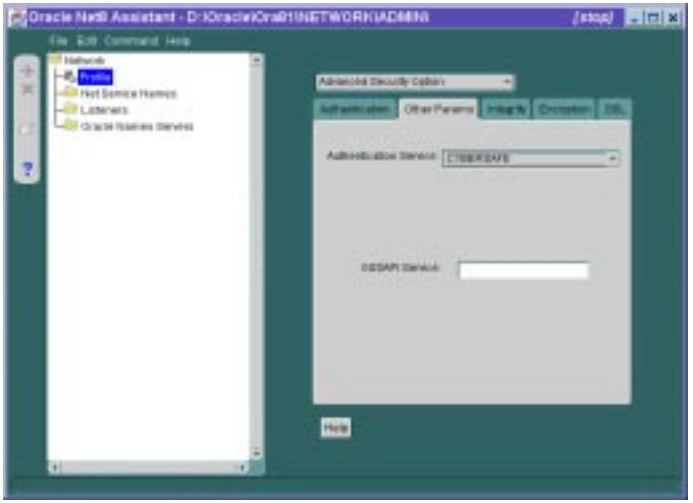


| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 4–1.<br>1. Select the Authentication tab.<br>2. In the Available Methods list, select CyberSafe.<br>3. Click the [>] button to move the service over to the Selected Methods list. Move any other methods you want to use in the same way.<br>4. Arrange the selected methods in order of desired use. Select a method, then click [Promote] or [Demote] to position it in the list. For example, put CyberSafe at the top of the list if you want that service to be the first one used. | Set the following parameter:<br>SQLNET.AUTHENTICATION_SERVICES= (*CYBERSAFE*) |

### Configure CyberSafe authentication service parameters on the client and the server

Do this by setting the SQLNET.AUTHENTICATION_GSSAPI_SERVICE parameter.

***Figure 4–2 Using Net8 Assistant to Configure Authentication Service Parameters***



| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 4–2.<br>1. Select the Other Params tab.<br>2. In the Authentication Service list box, select CYBERSAFE.<br>3. Type the name of the GSSAPI Service in the following format:<br>`oracle/`*`dbserver.someco.com`*<br>*`@SOMECO.COM`* | Set the following parameter:<br>SQLNET.AUTHENTICATION_GSSAPI_SERVICE=*KSERVICE/KINSTANCE@REALM*<br>**Note:** You must insert the principal name, using the format described in "Step 4: Configure a service principal for an Oracle server" on page 4-3. |

### Set INIT.ORA Parameter

Oracle strongly recommends that you add the following parameter to the init<*sid*>.ora file used for the database instance:

```
REMOTE_OS_AUTHENT=FALSE
```

where *sid* is the database system identifier.

---

**Note:** Setting REMOTE_OS_AUTHENT to TRUE may create a security hole because it allows someone using a non-secure protocol (for example, TCP) to perform an operating system-authorized login (formerly referred to as an OPS$ login).

---

Because CyberSafe user names can be long, and Oracle user names are limited to 30 characters, Oracle recommends using the following null value for the value of OS_AUTHENT_PREFIX:

```
OS_AUTHENT_PREFIX=""
```

Restart the Oracle server after modifying the configuration files, so the changes will take effect.

**More Information:** For information on how to restart the Oracle server, see your operating system-specific documentation and *Oracle8i Administrator's Guide.*

## Step 10: Create a CyberSafe User on the authentication server

In order for CyberSafe to authenticate Oracle users, you must create them on the CyberSafe authentication server where the administration tools are installed. The following steps assume that the realm already exists.

**More Information:** For information on creating the realm, see "Related Publications" on page xx in the Preface of this guide.

---

**Note:** The utility names in this section are actual programs that you run. However, the CyberSafe user name "cyberuser" and realm "SOMECO.COM" are examples only; these may vary among systems.

---

Run /krb5/admin/kdb5_edit as root on the authentication server to create the new CyberSafe user, that is, "cyberuser". Type the following:

1. # kdb5_edit
2. kdb5_edit: ank cyberuser
3. Enter password: <password not echoed to screen>
4. Re-enter password for verification: <password...>
5. kdb5_edit: quit

## Step 11: Create an externally authenticated Oracle user on the Oracle server

Run SQL*Plus to create the Oracle user and perform the following commands on the Oracle server machine:

```
SQL> CONNECT INTERNAL;
SQL> CREATE USER "USNERNAME" IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO "USERNAME";
```

In this example, OS_AUTHENT_PREFIX is set to:

```
""
```

---

**Note:** When you create the Oracle user, the name must be in upper case and double-quoted.

---

In the following example, OS_AUTHENT_PREFIX is set to "".

```
SQL> CREATE USER "JDOE" IDENTIFIED EXTERNALLY
SQL> GRANT CREATE SESSION TO "JDOE"
```

**More Information:** See *Oracle8i Administrator's Guide*.

## Step 12: Get the initial ticket for the Kerberos/Oracle user

Before users can connect to the database, they need to run kinit on the clients for an **initial ticket**.

```
% kinit (user name)
Password for CYBERUSER@US.ORACLE.COM:
<password not echoed to screen>
```

### Use klist on the Client to Display Credentials

Users should run klist on the clients to list the tickets currently owned.

```
% klist
```

| Creation Date | Expiration Date | Service |
|---|---|---|
| 11-Aug-95 16:29:51 | 12-Aug-95 00:29:21 | krbtgt/SOMECO.COM@SOMECO.COM |
| 11-Aug-95 16:29:51 | 12-Aug-95 00:29:21 | oracledbserver.someco.com@SOMECO.COM |

## Step 13: Connect to an Oracle server authenticated by CyberSafe

After running *kinit* to get an initial ticket, users can connect to an Oracle server without using a user name or password. Enter a command like the following:

```
%  sqlplus /@net_service_name
```

where *net_service_name* is a Net8 service name.

For example:

```
% sqlplus /@npddoc_db
```

**More Information:** See Chapter 1, "Introduction to Oracle Advanced Security" and *Oracle8i Distributed Database Systems*.

# Troubleshooting the Configuration of the CyberSafe Authentication Adapter

Following are some common configuration problems and tips to help resolve them:

**If you cannot get your ticket-granting ticket using kinit:**

- Make sure the default realm is correct by looking at krb.conf.
- Make sure the TrustBroker Master Server is running on the host specified for the realm.
- Make sure that the Master Server has an entry for your user principal and that the passwords match.
- Make sure 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 server side has a service name that corresponds to a service known to the CyberSafe Master Server.
- Check that the clocks on all the involved machines are within a few minutes of each other.

**If you have a service ticket and you still cannot connect:**

- Check the clocks on the client and server.
- Check that the v5srvtab exists in the correct location and is readable by Oracle.
- Check that the v5srvtab has been generated for the service named in the profile (sqlnet.ora) on the 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 kinit -f.)
- Check the expiration date on the credentials.
- If your credentials have expired, close your connection and run kinit to get a new initial ticket.

# 5

# Configuring Kerberos Authentication

This chapter contains information on how to configure Oracle for use with Kerberos authentication and to configure Kerberos to authenticate Oracle users.

This chapter covers the following topics:

# Enabling Kerberos Authentication

You enable Kerberos authentication by performing the following tasks, each of which is fully described in the next few pages.

Perform the following tasks in the order listed.



## Step 1: Install Kerberos

Do this on the machine that will act as the authentication server

> **More Information:** For information on how to install Kerberos on your machine, see "Related Publications" on page xx in the Preface of this guide.

## Step 2: Configure a service principal for an Oracle server

For the Oracle Server to be able to validate the identity of clients that authenticate themselves using Kerberos, you must first create a service principal for Oracle.

The name of the principal should have the following format:

```
kservice/kinstance@REALM
```

| | |
|---|---|
| kservice | a string that represents the Oracle service. This may or may not be the same as the database service name. It is case-sensitive. |

kinstance typically the fully-qualified name of the machine on which Oracle is running.

REALM the domain of the server. It must always be capitalized.

---

**Note:** The utility names in this section are actual programs that you run. However, the Kerberos user name "krbuser" and realm "SOMECO.COM" are examples only: the actual names may vary among systems.

---

For example, if kservice is oracle, and the fully-qualified name of the machine on which Oracle is running is dbserver.someco.com, and if the realm is SOMECO.COM, the principal name would be:

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

It is a common convention to use the DNS domain name as the name of the realm.

To create the service principal, run kdb5_edit. The following example is UNIX specific.

```
# cd /krb5/admin
# ./kdb5_edit
```

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

```
kdb5_edit:ark oracle/dbserver.someco.com@SOMECO.COM
```

## Step 3: Extract a service table from Kerberos

You now need to extract the service table from Kerberos and copy it to the Oracle server/Kerberos client machine.

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

```
kdb5_edit:  xst dbserver.someco.com oracle
'oracle/dbserver.someco.com@SOMECO.COM' added to keytab
'WRFILE:dbserver.someco.com-new-srvtab'
kdb5_edit:  exit
oklist -k -t dbserver.someco.com-new-srvtab
```

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 kdb5_edit to append the additional entries.

If you do not enter a realm (for example, SOMECO.COM) when using xst, it uses the realm of the current host and displays it in the command output, as shown above.

If the Kerberos service table is on the same machine as the Kerberos client, you can simply move it. If the service table is on a different machine from the Kerberos client, you must transfer the file with a program like binary FTP. The following example is UNIX specific.

```
# mv dbserver.someco.com-new-srvtab /etc/v5srvtab
```

The default name of the service file is /etc/v5srvtab. If a different name is used, then that name should be substituted for the default name.

### Ensure that the Oracle Server Can Read the Service Table

Verify that the owner of the Oracle Server executable can read the service table (in the above example, /etc/v5srvtab). To do that, 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 may allow a security breach.

---

## Step 4: Install an Oracle server and an Oracle client

**More Information:** See your operating system-specific documentation.

## Step 5: Install Net8

Do this on the Oracle server and Oracle client machines.

**More Information:** See the Net8 installation documentation

## Step 6: Configure Net8 and Oracle

Do this on the Oracle server and client.

> **More Information:** See your operating system-specific documentation. See also the *Net8 Administrator's Guide*.

## Step 7: Configure Kerberos authentication

You must set certain parameters in the Oracle server and client sqlnet.ora files. The next few pages explain how to do the following tasks.

- Configure the authentication service on the client and the server
- Configure authentication parameters on the Oracle server and client

> **More Information:** For information on configuring Kerberos authentication with Net8 Assistant, see the Net8 Assistant on-line HELP system.

Unless otherwise indicated, you can configure Kerberos authentication either by using the Net8 Assistant, or by modifying the sqlnet.ora file with any text editor.

### Using Net8 Assistant

This graphical interface tool makes it easy to set parameters in the sqlnet.ora file and other Oracle8*i* configuration files.

**To launch Net8 Assistant:**

- On UNIX, run the script netasst at $ORACLE_HOME/bin.
- On Windows platforms, click the Start button > Programs > Oracle for Windows NT—*HOME_NAME* > Net8 Assistant.

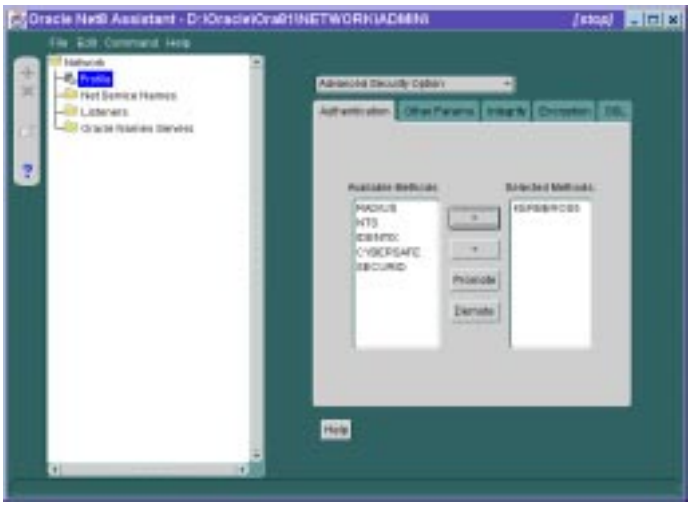
**To begin configuring the Oracle Advanced Security option using Net8 Assistant:**
In the Net8 Assistant's left pane, click the Profile folder. Then go to the drop down list box at the top of the right pane, and select Advanced Security Option. The tabbed pages for the Oracle Advanced Security option appear.

**To save changes with Net8 Assistant:**
Go to the menu bar and click File > Save Network Configuration.

### Configure the authentication service on the client and the server

Do this by setting the SQLNET.AUTHENTICATION_SERVICES parameter.

***Figure 5–1 Using Net8 Assistant to Configure Authentication***

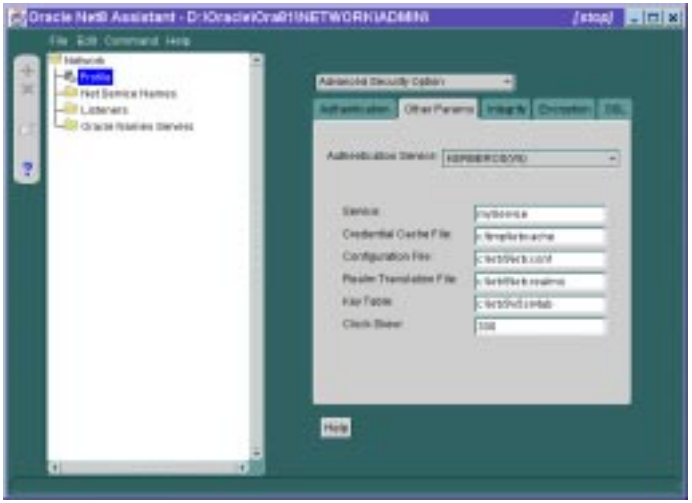


| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 5–1.<br>1. Select the Authentication tab.<br>2. Select KERBEROS5 from the Available Services list.<br>3. Click the [>] button to move the service over to the Selected Services list. Move any other methods you want to use in the same way.<br>4. Arrange the selected services in order of desired use. Click on a service to select it, then click [Promote] or [Demote] to arrange the services in the list. For example, put KERBEROS5 at the top of the list if you want that service to be the first one used. | Set the following parameter:<br>SQLNET.AUTHENTICATION_ SERVICES=(*KERBEROS5*) |

### Configure authentication parameters on the Oracle server and client

Do this by setting the SQLNET.AUTHENTICATION_KERBEROS5_SERVICE parameter. You may also set various optional parameters described in this section.

***Figure 5–2 Using Net8 Assistant to Configure Authentication Parameters***



| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 5–2.<br>1. Select the Other Params tab.<br>2. In the Service text box, type kerberos. **Note:** You *must* provide the value for this parameter. When you do this, the other text boxes are enabled.<br>You *may* provide values for the following parameters:<br>■ Credential Cache File<br>■ Configuration File<br>■ Realm Translation File<br>■ Key Table<br>■ Clock Skew | You *must* set the following parameter:<br>SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=*kservice*<br>**Note:** The above parameter specifies the name of the service Oracle will use to obtain a Kerberos service ticket. You must substitute a value for the kservice part of the service name.<br>**Note also:** The value passed by the parameter SQLNET.AUTHENTICATION_KERBEROS5_SERVICE is case sensitive; it must be lower case.<br>You *may* set the following parameters, each of which is described in the section "Optional SQLNET.ORA Parameters" on page 5-8<br>■ SQLNET.KERBEROS5_CC_NAME<br>■ SQLNET.KERBEROS5_CONF<br>■ SQLNET.KERBEROS5_REALMS<br>■ SQLNET.KERBEROS5_KEYTAB<br>■ SQLNET.KERBEROS5_CLOCKSKEW |

### Set INIT.ORA Parameter

Use a text editor to add the following parameter to the init.ora file used for the database instance:

```
REMOTE_OS_AUTHENT=FALSE
```

---

**Attention:** Setting REMOTE_OS_AUTHENT to TRUE may create a security hole, because it allows someone using a non-secure protocol (for example, TCP) to perform an operating system-authorized login (formerly referred to as an OPS$ login).

---

Because Kerberos user names can be long and Oracle user names are limited to 30 characters, Oracle Corporation strongly recommends that the following null value be used for the value of OS_AUTHENT_PREFIX:

```
OS_AUTHENT_PREFIX=""
```

Setting OS_AUTHENT_PREFIX to a null value overrides the default value of OPS$.

### Optional SQLNET.ORA Parameters

In addition to the above required parameters, you can optionally set the parameters described below on the client or server.

Parameter: SQLNET.KERBEROS5_CC_NAME=*pathname_to_credentials_cache_file*

Description: This parameter specifies the complete pathname to the Kerberos credentials cache (CC) file. The default value is operating system-dependent. For UNIX, it is /tmp/krb5cc_user id. For example:

```
SQLNET.KERBEROS5_CC_NAME=/usr/tmp/krbcache
```

**Note:** You can also set this parameter by using the KRB5CCNAME environment variable.

The value set for the SQLNET.KERBEROS5_CC_NAME parameter in the sqlnet.ora file takes precedence over the value set in the KRB5CCNAME environment variable.

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 server. It is also used by an Oracle server to decide if a credential needs to be stored to protect against a replay attack. The default is 300 seconds. For example:

```
SQLNET.KERBEROS5_CLOCKSKEW=1200
```

Parameter: SQLNET.KERBEROS5_CONF=*pathname_to_Kerberos_configuration_file*

Description: This parameter specifies the complete pathname 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. For example:

```
SQLNET.KERBEROS5_CONF=/krb/krb.conf
```

Parameter: SQLNET.KERBEROS5_KEYTAB=*pathname_to_Kerberos_principal/key_table*

Description: This parameter specifies the complete pathname to the Kerberos principal/secret key mapping file. It is used by the Oracle 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. For example:

```
SQLNET.KERBEROS5_KEYTAB=/etc/v5srvtab
```

Parameter: SQLNET.KERBEROS5_REALMS=*pathname_to_Kerberos_realm_translation_file*

Description: This parameter specifies the complete pathname 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. For example:

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

## Step 8: Create a Kerberos user

To create Oracle users that Kerberos can authenticate, perform the following steps on the Kerberos authentication server where the administration tools are installed.

It is assumed that the realm already exists.

> **More Information:** See "Related Publications" on page xx in the Preface of this guide.

> **Note:** The utility names in this section are actual programs that you run. However, the Kerberos user name "krbuser" and realm "SOMECO.COM" are examples only; these may vary among systems.

Run /krb5/admin/kdb5_edit as root to create the new Kerberos user, for example, "krbuser". The following example is UNIX specific.

```
# ./kdb5_edit
kdb5_edit: ank krbuser
Enter password:  <password not echoed to screen>
Re-enter password for verification:  <password...>
kdb5_edit: quit
```

## Step 9: Create an externally-authenticated Oracle user

Run SQL*Plus on the Oracle server to create the Oracle user that corresponds to the Kerberos user. In the following example, OS_AUTHENT_PREFIX is set to "".

> **Note:** The Oracle user name must be in upper-case and double-quoted. For example, "KRBUSER@SOMECO.COM".

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

## Step 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**. You do this by running the following on the client:

```
okinit (user name)
```

> **More Information:** For information on using okinit, see "Utilities for the Kerberos Authentication Adapter" on page 5-12.

If, when making a database connection, a reference such as

```
sqlplus /@oracle
```

follows a database link, you must use the forwardable flag (-f option). Executing okinit -f enables credentials that can be used across database links. You should be on the Oracle client before running the following commands:

```
% okinit -f
Password for krbuser@SOMECO.COM:<password not echoed to screen>
```

> **More Information:** For information about okinit, see "Use okinit to Obtain the Initial Ticket" on page 5-12.

# Utilities for the Kerberos Authentication Adapter

The following three utilities are shipped with the Oracle Kerberos authentication adapter. You should be on the Oracle client before running these commands.

| Command | Description |
|---|---|
| okinit | Gets an initial ticket |
| oklist | Displays a list of currently-owned tickets |
| okdstry | Removes all tickets from the credentials cache |

These utilities are intended for customers who are running an Oracle client with an Oracle Kerberos authentication adapter installed.

**UNIX Only:** Solaris is shipped with Kerberos version 4. Make sure that the Kerberos version 5 utilities are in your path so that the version 4 utilities are not inadvertently used.

## Use okinit to Obtain the Initial Ticket

okinit obtains and caches Kerberos tickets. okinit is typically used to obtain your 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 following options are available with okinit.

| Option | Description |
|---|---|
| `-f` | Ask for a forwardable ticket-granting ticket. This option is necessary to follow database links. |
| `-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' (months), or 's' (seconds).<br>For example,<br>`okinit -l 2wld6h20m30s`<br>means ask for a ticket-granting ticket that has a lifetime of 2 weeks, 1 day, 6 hours, 20 minutes, and 30 seconds. |

| Option | Description |
|---|---|
| -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. |

## Use oklist to Display Credentials

Users can run oklist to display the list of tickets they hold. The show flag option (-f) displays additional information.

```
% oklist -f
27-Jul-1995 21:57:51   28-Jul-1995 05:58:14
krbtgt/SOMECO.COM@SOMECO.COM
Flags: FI
```

| Option | Description |
|---|---|
| -f | Show flags with credentials. The important ones for Oracle are 'I' (credential is a ticket-granting ticket), 'F' (credential is forwardable), and 'f' (credential is forwarded). |
| -c | Specify an alternative credential cache. For 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. |

## Use okdstry to Remove Credentials from Cache File

Use okdstry to remove credentials from the credentials cache file.

```
$ okdstry -f
```

| Option | Description |
|---|---|
| -f | 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 a profile (sqlnet.ora). |

## Connecting to an Oracle Server Authenticated by Kerberos

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

```
$ sqlplus /@net_service_name
```

where *net_service_name* is a Net8 service name. For example:

```
$ sqlplus /@oracle_dbname
```

> **More Information:** For information on external authentication, see Chapter 1, "Introduction to Oracle Advanced Security" and *Oracle8i Distributed Database Systems.*

# Troubleshooting the Configuration of Kerberos Authentication

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

### If you cannot get your ticket-granting ticket using okinit:

- Make sure the default realm is correct by looking at krb.conf.
- Make sure the KDC is running on the host specified for the realm.
- Make sure that the KDC has an entry for your user principal and that the passwords match.
- Make sure 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 server side has a service name that corresponds to a service known by Kerberos.
- Check that the clocks on all machines involved 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 server.
- Check that the v5srvtab exists in the correct location and is readable by Oracle (remember the sqlnet.ora parameters).
- Check that the v5srvtab has been generated for the service named in the sqlnet.ora file on the 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 okinit -f.)
- Check the expiration date on the credentials.
- If your credentials have expired, you must close your connection and run okinit to get a new initial ticket.

# 6

# Configuring SecurID Authentication

This chapter describes how to configure and use SecurID authentication with the Oracle server and clients. It assumes that you are familiar with the Security Dynamics ACE/Server and that the ACE/Server is installed and running.

> **More Information:** See "Related Publications" on page xx in the Preface of this Guide.

This chapter covers the following topics:

- System Requirements
- Known Limitations
- Enabling SecurID Authentication
- Creating Users for SecurID Authentication
- Troubleshooting the Configuration of SecurID Authentication
- Using SecurID Authentication

## System Requirements

To use SecurID authentication included in the Oracle Advanced Security option release 8.1.5, you need the following:

- Net8
- Oracle 8.0.3 or higher
- ACE/Server 1.2.4 or higher
- The Oracle server machine must be running UDP/IP and TCP/IP protocols, because Oracle needs to communicate with the ACE/Server. Even though the client uses SQL*Net or Net8 to connect to Oracle, Oracle needs UDP to connect to the ACE/Server.

## Known Limitations

Because SecurID card codes can be used only once, SecurID authentication does not support database links, also known as "proxy authentication."

When using SecurID authentication, password encryption is disabled. This means that the SecurID card code (and, if you use standard cards, the PIN), are sent over to the Oracle server in clear text. This could be a security problem. Consequently, Oracle recommends that you turn on the Oracle Advanced Security option encryption, which ensures that the PIN is encrypted when sent to the Oracle server.

> **More Information:** For information on how to turn on encryption, see Chapter 2, "Configuring Encryption and Checksumming".

## Enabling SecurID Authentication

You enable SecurID authentication by performing the following tasks, each of which is fully described in the next few pages:

Step 1: Register Oracle as a SecurID client (ACE/Server Release 1.2.4)

Step 2: Install Oracle Advanced Security

Step 3: Ensure that Oracle can find the correct UDP port (ACE/Server Release 1.2.4)

Step 4: Configure Oracle as a SecurID client

Step 5: Configure SecurID authentication

## Step 1: Register Oracle as a SecurID client (ACE/Server Release 1.2.4)

Register the machine on which the Oracle Server resides as a SecurID client with the ACE server. You can do this with the Security Dynamics tool sdadmin. To create a client, go to the Client menu and choose Create Client (ACE/Server 1.2.4) or Add Client (ACE/Server 2.0).

Refer to the Security Dynamics ACE/Server Instruction manual, version 1.2.4, or to the Security Dynamics ACE/Server version 2.0 Administration manual for more detailed information.

## Step 2: Install Oracle Advanced Security

Install the Oracle Advanced Security option on the Oracle server and Oracle client in a typical installation of Oracle8*i* using the Oracle Installer.

> **More Information:** See your platform-specific installation instructions.

## Step 3: Ensure that Oracle can find the correct UDP port (ACE/Server Release 1.2.4)

First verify that the ACE/Server, the Oracle server, and the Oracle Advanced Security option are installed.

Make sure that the Oracle server can discover what the correct UDP port for contacting the ACE/Server is. These port numbers are typically stored in a file called services. On the UNIX operating system, this file is typically in the /etc directory. If you are using NIS (Network Information Services) as a naming service, make sure that the services map contains the correct entries for SecurID.

---

**Note:** You can verify which port the ACE server is using by running the Security Dynamics tool Kitconts (for ACE/Server 1.2.4) or sdinfo (for ACE/Server 2.0).

---

## Step 4: Configure Oracle as a SecurID client

### Windows NT and Windows 95/98 Platforms

You need the following from your SecurID administrator:

- SDCONF.REC file present in the root drive:\VAR\ACE
- port numbers and service names present in the Windows NT SERVICES file

### UNIX Platform and ACE/Server Release 1.2.4

> **If you are using ACE/Server Release 2.0:** See "UNIX Platform and ACE/Server Release 2.0" on page 6-5.

Install the SecurID configuration files on the Oracle server machine. You can obtain them from any other SecurID client or from the machine that runs the ACE/Server.

These files are typically stored in /var/ace. On the Oracle server machine, create this directory and copy the configuration files to it. At the minimum, you need the file sdconf.rec. The configuration files are used by both Oracle and the standard SecurID tools. Because the SecurID tools run setuid root, there can be a problem with the access permissions on the directory /var/ace and the files in this directory. Make sure that the owner of the Oracle executable (for example, the user "oracle8") is able to read all the files in /var/ace and can create new files in this directory.

---

**Caution:** Do not attempt to overcome this by running Oracle setuid root. It is not necessary, and it is dangerous to do so.

---

There are two methods for reaching this goal without compromising security. Both methods work, but Oracle recommends that you use method #1. Both methods allow you to use Oracle with SecurID authentication and still continue using the other SecurID tools.

### Method #1

The owner of the Oracle executable should also own the /var/ace directory and the files in /var/ace. For example, if the owner of the Oracle executable is the user "oracle8," perform the following steps, as root:

```
# chown oracle8 /var/ace
# chmod 0770 /var/ace
# chown oracle8 /var/ace/*
# chmod 0660 /var/ace/*
```

### Method #2

The other option is to have root own the /var/ace directory and the files in /var/ace, but give the Oracle group read and write access. If the Oracle group is "dba", you need to perform the following steps, as root:

```
# chown root /var/ace
# chmod 0770 /var/ace
# chgrp dba /var/ace
# chown root /var/ace/*
# chmod 0660 /var/ace/*
# chgrp dba /var/ace/*
```

## UNIX Platform and ACE/Server Release 2.0

Note the following:

- The VAR_ACE environment variable is not supported. You have to store the configuration data in the /var/ace directory. If you currently have the ACE configuration data in a different location, you should create a symbolic link using the following command:

  ```
  # ln -s $VAR_ACE /var/ace
  ```

- Oracle needs to be able to read and write the ACE configuration data. This data is stored in the directory /var/ace (or $VAR_ACE if you use the symbolic link shown above).

Whether Oracle can read the configuration data depends on how you installed the ACE client software on the Oracle server. During the installation of the ACE client software, you can specify which administrator should own the configuration files.

---

**Attention:** Whether you use Method 1 or Method 2 below, make sure that you do not install Oracle as root.

---

### Method #1

If root is the owner of the ACE server configuration data files, you will have to change the UNIX file permissions so that the owner of the oracle executable can read and write to these files. For example, the following commands give Oracle access to the files, and all the Security Dynamics tools that run as setuid root will still be able to access the files.

```
# chown oracle8 /var/ace
# chown oracle8 /var/ace/*
# chmod 0770 /var/ace
# chmod 0660 /var/ace/*
```

If the environment variable VAR_ACE is set to a different location than /var/ace, you should instead execute the following commands:

```
# ln -s $VAR_ACE /var/ace
# chown oracle8 $VAR_ACE
# chown oracle8 $VAR_ACE/*
# chmod 0770 $VAR_ACE
# chmod 0660 $VAR_ACE/*
```

### Method #2

If the ACE files are not owned by root, you have two options:

- Install the ACE client or server and Oracle under the same UNIX account. (You have to install the ACE software as root, but you can specify which administrator should own the files. Specify the same user as the owner of the Oracle executable, typically "oracle8").
- Add the owner of the Oracle executable to the ACE administrators' group

---

**Note:** Make sure the owner of the oracle executable remains a member of the DBA group; otherwise you will not be able to control your database.

---

For the change to take effect, do the following:

1. Log out, then log in again as the Oracle owner.
2. Restart your Network listener.
3. Restart your Oracle server.

## Step 5: Configure SecurID authentication

You configure SecurID authentication either by using Net8 Assistant, or by modifying the sqlnet.ora file with any text editor.

### Using Net8 Assistant

This graphical interface tool makes it easy to set parameters in the sqlnet.ora file and other Oracle8*i* configuration files.

**To launch Net8 Assistant:**

- On UNIX, run the script netasst at $ORACLE_HOME/bin.
- On Windows platforms, click the Start button > Programs > Oracle for Windows NT—*HOME_NAME* > Net8 Assistant.

**To begin configuring the Oracle Advanced Security option using Net8 Assistant:**
In the Net8 Assistant's left pane, click the Profile folder. Then go to the drop down list box at the top of the right pane, and select Advanced Security Option. The tabbed pages for the Oracle Advanced Security option appear.

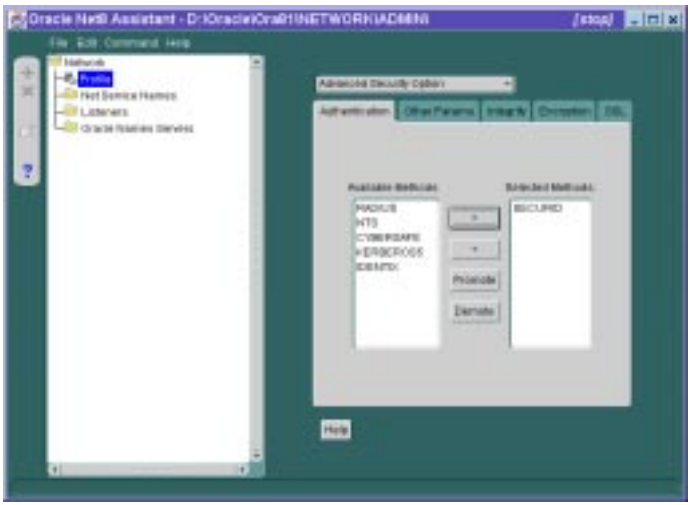
**To save changes with Net8 Assistant:**
Go to the menu bar and click File > Save Network Configuration.

The following pages tell you how to set each parameter.

### Configure an authentication method on the client and the server

Do this by setting the SQLNET.AUTHENTICATION_SERVICES parameter.

***Figure 6–1 Using Net8 Assistant to Configure Authentication***



| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| **1.** Select the Authentication tab.<br>**2.** Select SECURID from the Available Methods list.<br>**3.** Click the [>] button to move the method over to the Selected Methods list. Move any other methods you want to use in the same way.<br>**4.** Arrange the selected methods in order of desired use. To do this, select a method in the list, then click [Promote] or [Demote] to arrange the methods in the list. For example, put SECURID at the top of the list if you want that method to be the first one used. | Set the following parameter:<br>`SQLNET.AUTHENTICATION_SERVICES=`<br>`(`*`SECURID`*`)` |

# Creating Users for SecurID Authentication

You create users for SecurID authentication by performing the following steps:

- Step 1: Assign a card to a person by using the Security Dynamics sdadmin program
- Step 2: Create an Oracle server account for this user
- Step 3: Grant the user database privileges

## Step 1: Assign a card to a person by using the Security Dynamics sdadmin program

When the sdadmin tool asks for a login name when creating a new user, fill in the same name you will use later to create the Oracle user.

> **More Information:** See the Security Dynamics documentation listed in "Related Publications" on page xvi in the Preface of this guide.

If you want the user to be able to specify a new PIN to the card using the Oracle tools, choose the option that allows the user to make up his or her own PIN. If you do not allow this, the user will have to use the Security Dynamics tools to generate a PIN if the card is in new-PIN mode. Activate the user on the Oracle Server. (The Oracle Server should already be registered as a SecurID client.)

## Step 2: Create an Oracle server account for this user

You can do this by using SQL*Plus connected as a user with the create user database role. Use the following syntax to create an account:

```
SQL> CONNECT system/manager
SQL> CREATE USER os_authent_prefix username IDENTIFIED EXTERNALLY
```

The OS_AUTHENT_PREFIX is an Oracle Server initialization parameter (for example, in the file init.ora). The OS_AUTHENT_PREFIX default value is OPS$. The user name should be the same as the name you assigned to the card in step 1 above.

**Note:** Because user names can be long and Oracle user names are limited to 30 characters, Oracle Corporation strongly recommends that OS_AUTHENT_PREFIX be set to a null value:

```
OS_AUTHENT_PREFIX=""
```

At this point, an Oracle user with *username* should not yet exist.

For example, suppose you have assigned a card to the user "king," and that OS_AUTHENT_PREFIX has been set to a null value (""), at this point you should create an Oracle user account using the following syntax:

```
SQL> CREATE USER king IDENTIFIED EXTERNALLY;
```

## Step 3: Grant the user database privileges

You may want to give this user some database privileges. At the minimum, the user should have the "create session" privilege.

```
SQL> GRANT CREATE SESSION TO king;
```

The user king can now connect to Oracle using the appropriate SecurID card.

**More Information:** For information on how to log into an Oracle server after SecurID authentication has been installed and configured, see "Logging in to the Oracle Server" on page 6-13.

# Troubleshooting the Configuration of SecurID Authentication

If you experience problems while configuring SecurID authentication, verify the following:

- The services map should have an entry for the Security Dynamics ACE server. The service name is typically securid, but the SecurID administrator can choose any name.

  Use the SecurID tool kitconts (for ACE/Server 1.2.4) or sdinfo (for ACE/Server 2.0) to verify the name of the authentication service and the port numbers that SecurID is expecting to use. Verify that these port numbers match those in /etc/services, or the services map if you are using NIS.

  **ACE/Server release 1.2.4 only:** Verify that the /var/ace/sdconf.rec file is present on the machine running the Oracle server. Also verify that the permissions on the /var/ace/sdconf.rec file and the directory /var/ace are set so that the Oracle process can read and write in the directory.

  **ACE/Server release 2.0 only:** Make sure the ACE configuration data is in the /var/ace directory. Use of the VAR_ACE environment variable is not supported. Also make sure that the owner of the oracle executable can read and write the files in this directory.

- Check to see if the Oracle server machine is registered as a SecurID client. You can do this by using the Security Dynamics tool sdadmin.

- The user who is trying to connect to Oracle should be activated on the Oracle Server, either as a direct user or as part of a group of users. Verify this using the SecurID tool sdadmin.

- Security Dynamics has developed a few logging facilities that can help you find problems. By using sdadmin, you can see a log of the recent system activities, including failed authentication with the reason for the failure. You can also use sdlogmon to get a similar log listing.

- Turn on tracing by adding the following line to the SQLNET.ORA file on the Oracle side:

  ```
  trace_level_server = admin
  ```

  Turning tracing on at the client side is less informative, because all interaction between the Oracle server and the ACE server happens at the Oracle server side of the SQL*Net or Net8 connection. Be sure to turn off tracing when you have completed your check.

- Make sure that the user has been created in the Oracle database as an externally-identified user with the correct prefix (which defaults to OPS$). When connected as system, enter:

  ```
  SQL> SELECT * FROM all_users;
  ```

  to get a list of all database users.

- When you connect to Oracle as a non-externally identified user, the SecurID log file will indicate a warning. For example, if you connect as 'system' using:

  ```
  sqlplus system/manager@oracle_dbname
  ```

  the SecurID log file displays:

  ```
  03/24/95   10:04  User not on client machinename
  ```

  This is not an error. Since the Oracle client and server negotiated to use SecurID because of the SQLNET.AUTHENTICATION_SERVICES line in SQLNET.ORA, Oracle will contact the ACE/Server to validate 'system'. When validation fails, Oracle will validate the password internally. If the password is valid, you will be able to connect.

  The only way to eliminate the warning message is to disable SecurID authentication. To do so, change the sqlnet.ora file on the Oracle client to:

  ```
  SQLNET.AUTHENTICATION_SERVICES=(NONE)
  ```

  Setting this parameter to this value disables the SecurID authentication adapter. You will no longer be able to connect to Oracle using the SecurID card.

# Using SecurID Authentication

This section describes how to use SecurID authentication with the Oracle client tools. It assumes that you are already familiar with SecurID concepts, and that you have configured Oracle for use with the SecurID authentication.

This section discusses the following topics.

- Logging in to the Oracle Server
- Assigning a New PIN to a SecurID Card
- Logging in When the SecurID Card is in "Next Code" Mode

> **More Information:** See "Step 5: Configure SecurID authentication" on page 6-7. Also refer to "Related Publications" on page xx of the Preface of this Guide.

Before you use SecurID authentication to verify passwords, make sure the following things have been done:

- The SecurID authentication adapter has been installed and linked into the SQL*Net or Net8 configuration.
- Oracle has been configured for use with the ACE/Server (that is, it can act as a SecurID client).
- The client and server have been configured with the necessary parameters so that database passwords can be verified by the central SecurID authentication server.
- Users have been configured for use with the SecurID authentication as described in "Step 5: Configure SecurID authentication" on page 6-7.

## Logging in to the Oracle Server

SecurID authentication allows you to log into the Oracle server with the PASSCODE that is generated by the SecurID card. The PASSCODE replaces the password in the Oracle connect statement.

There are two types of SecurID cards:

- standard (model SD200)
- PINPAD (model SD520)

Depending on the type of card, you type the PIN

- directly onto the card

or

- as part of the Oracle connect statement.

### Using Standard Cards

The standard cards generate and display a PASSCODE. When logging in to Oracle, you need to specify your user name, your PIN and the current PASSCODE, using the following syntax:

```
sqlplus username/<pin><passcode>@net_service_name
```

For example, if the card is assigned to user king, the PIN is "3511," and the card shows the number "698244," this is how you would log into Oracle using SQL*Plus:

```
% sqlplus king/3511698244@oracle_database
```

or,

```
% sqlplus king@oracle_database
% enter password: 3511698244
```

---

**Note:** The Security Dynamics tools support the following characters as delimiters between the PIN and the PASSCODE:

" "  <tab>  \  /  ;  :

You should not use these characters, because Oracle will interpret these characters differently.

---

### Using PINPAD Cards

If you have a PINPAD card, you first have to type in your PIN on the card and generate a new PASSCODE. You would then use this PASSCODE to connect to Oracle using the following syntax:

```
sqlplus username/passcode@net_service_name
```

For example, if the card is assigned to user "king", first generate a PASSCODE by typing the PIN on the PINPAD card. (Refer to the Security Dynamics documentation on how to do this.) For example, if the generated PASSCODE is "698244", to connect to Oracle using SQL*Plus, you would type:

```
% sqlplus king/698244@oracle_dbname
```

## Assigning a New PIN to a SecurID Card

If you are logging in for the first time, or the administrator has put your card in the new-PIN mode, you have to assign a PIN to the card. You can tell that this is the case if, while trying to connect to Oracle, you get the following error message:

```
ORA-12681 "Login failed: the SecurID card does not have a pincode yet"
```

To assign a PIN to a card you connect to the Oracle Server using a special syntax. First, you need to select a PIN, which is typically four to eight digits long. Depending on the type of SecurID card you have, you may be able to use letters as well.

**If you have cleared the old PIN:**
Use the following the syntax while connecting to the Oracle database:

```
sqlplus username/+<new_pin>+<tokencode>@oracle_dbname
```

> **Note:** You must add the two "+" characters in the connect string, because they tell Oracle that this is an attempt to assign a PIN to the card. Also, they separate the new PIN from the passcode.
>
> You must also enclose the PIN / passcode combination in double quotes. Some Oracle tools such as SQL*Plus truncate the password string (PIN / passcode) just before the plus ("+") character. Surrounding the password string (PIN / passcode) in double quotes ("") prevents the password string from being truncated.

For the tokencode, enter the cardcode that is currently displayed on your SecurID card's LCD. If you have a PINPAD card, do not enter the PIN on the card.

For example, if the card is assigned to user "king", your new PIN is "45618", and the SecurID card currently displays number "564728", you would type:

```
% sqlplus king/"+45618+564728"@oracle_dbname
```

**If you have *not* cleared the old PIN:**
Use the following syntax while connecting to the database. Otherwise, the administrator must select the new PIN for you.

```
sqlplus username/+<new_pin>+<old_pin><tokencode>@oracle_dbname
```

For the tokencode, enter the cardcode that is currently displayed on your SecurID card's LCD. If you have a PINPAD card, do not enter the PIN on the card.

If the new PIN is accepted, you will be connected to Oracle. The next time you want to connect to Oracle you should use the procedure described in "Logging into the Oracle Server". If the new PIN is rejected, you will get the following error:

```
ORA-12688 "Login failed: the SecurID server rejected the new pincode"
```

### Possible Reasons Why a PIN Would be Rejected:

- The new PIN is less than 4 or more than 8 characters long.
- The PIN contains invalid characters. Valid characters are numeric digits, and for some SecurID cards, the letters "a" through "z".
- You are not allowed to make up your own PIN. The Security Dynamics ACE/Server can be configured in such a way that you cannot make up your own PIN. If this is the case, you will have to use one of the Security Dynamics tools to generate a new PIN for your card.

## Logging in When the SecurID Card is in "Next Code" Mode

As an additional safety step, the ACE/Server sometimes asks for the next card code, to ensure that the person who is trying to log in actually has the card in his or her possession. This is the case if you get the following error message when you try to log into Oracle:

```
ORA-12682, "Login failed: the SecurID card is in next PRN mode"
```

The next time you want to log in to Oracle, you will have to specify the next two card codes. The syntax you use to log into Oracle depends on the kind of SecurID card you have (Standard versus PINPAD).

### Logging in with a Standard Card

If you have a standard card, specify the following:

1. your PIN
2. the current card code
3. a "+" character and the next card code

Steps 1, 2, and 3 above replace the password. The "+" character is important, because it separates the first card code (passcode) from the second one. Use the following syntax:

```
sqlplus <username>/ "<pincode><passcode>+<next passcode>"@<net_service_name>
```

**Note:** You must enclose the PIN / passcode / next passcode combination in double quotes. Some Oracle tools such as SQL*Plus truncate the password combination just before the plus ("+") character. Surrounding the PIN and passcode in double quotes ("") prevents the password combination from being truncated.

For example, if the card is assigned to user "king", the PIN is "3511", and the card first shows the number "698244" and the next number is "563866", you would type:

```
% sqlplus king/"3511698244+563866"@oracle_database
```

This connects you to the Oracle server and puts the card back into normal mode. The next time you want to log in to the Oracle server, use the procedure described in "Logging in to the Oracle Server" on page 6-13.

### Logging in with a PINPAD Card

If you have a PINPAD card, do the following to log on to the Oracle server:

1. Type in your PIN on the card to generate the first PASSCODE.
2. Clear your card's memory by pressing P, then wait for the next PASSCODE.
3. Log into the Oracle server with these two passcodes, separated by a "+" character. Use the following syntax:

   ```
   sqlplus <username>/ "<first passcode>+<second passcode>"@net_service_name
   ```

For example, if the card is assigned to user "king":

1. Type the PIN on the PINPAD card to generate a passcode: e.g., "231003".
2. Clear the card's memory. The next displayed number might be "831234".
3. To log in, use the following syntax, typing the two passcodes generated in steps 1 and 2:

   ```
   % sqlplus king/"231003+831234"@oracle_dbname
   ```

This connects you to Oracle and puts the card back into normal mode. The next time you want to log in to Oracle, use the procedure described in"Logging in to the Oracle Server" on page 6-13.

# 7

# Configuring Identix Biometric Authentication

This chapter contains information on how to configure Oracle for use with Identix Biometric authentication. It covers the following topics:

# Overview

The Biometric Authentication Service uses the Identix Biometric Authentication Adapter to provide tamper-proof biometric authentication of users using secret-key MD5 hashing, centralized management of biometrically identified users, and centralized management of those database servers that authenticate biometrically identified users.

This section describes how the Biometric Authentication Service works in a client-server environment.

***Figure 7–1 Typical Biometric Authentication Service Configuration***

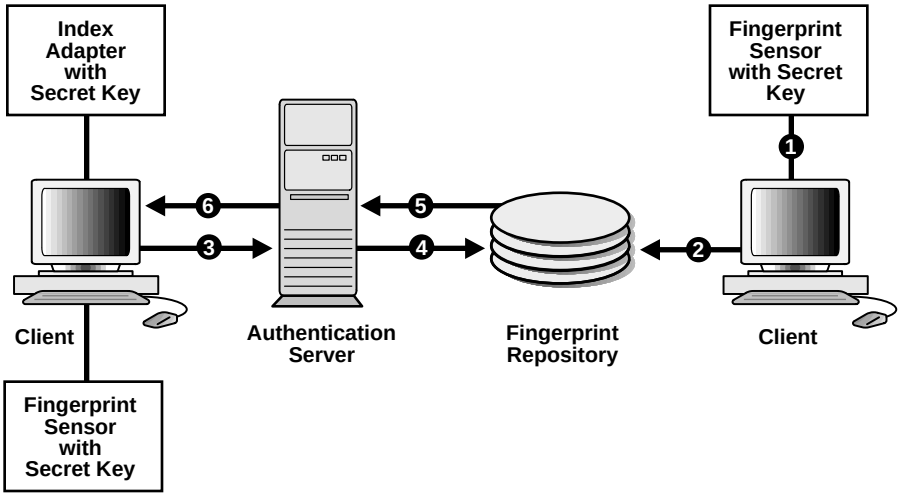



Figure 7–1 presents the components and the configuration of the Biometric Authentication Service.

- The fingerprint repository has one administrator who is responsible for enrolling multiple users' fingerprint templates and defining the DEFAULT policy that will be in force for all databases that subscribe to the fingerprint server for authentication.
- The Fingerprint Security Service Administrator uses a desktop fingerprint scanner to read user fingerprints, convert them to templates, and send them with measured accuracies to the Biometric Authentication Service which stores them in the fingerprint repository: an Oracle database. The measured accuracy of a fingerprint is an estimate of how reliable a comparison can be made

between the stored fingerprint template and the user's fingerprint that is scanned later for authentication. The enrollment quality is expressed as a percent score between 0 and 100. For example, a user may have an enrollment quality of 72%.

- The Fingerprint Security Service Administrator also defines one security policy named DEFAULT for all of the database servers that accept biometrically identified users. The security policy is enforced for all clients serviced by that database server. It contains a secret key and three types of threshold levels for fingerprints: verification, false finger, and high security.
- At the client, before any authentication can occur, the Fingerprint Security Service Administrator stores the secret key in the fingerprint sensor for each client. The secret key stored in the fingerprint sensor will be compared against the secret key stored in the security policy.
- At the client, in response to the user's request for authentication, the database server enforces on the client the set of values that it obtains from the DEFAULT security policy in its fingerprint server. The three threshold levels (values) are:
  - verification threshold
  - false finger threshold
  - high security threshold

  Please refer to the Identix documentation for detailed information on these threshold levels.
- At the client, the biometric authentication service passes the user's fingerprint template and the threshold values to the sensor. The user's fingerprint is then scanned and the verification result, fingerprint template, secret key, and threshold values are used to generate a hash. This hash is sent to the server side adapter for comparison with the hash generated by the server side adapter.

## Architecture of the Biometric Authentication Service

The Biometric Authentication Service consists of the following modules:

- The Biometric Manager, which the administrator uses to enter the security policy and fingerprints. In the remainder of this document, the Biometric Manager will also be referred to as the manager.
- The Biometric Authentication Server (fingerprint repository), which stores the security policies and fingerprint templates, is a specially configured version of a production Oracle Database Server. In the remainder of this document, the

Biometric Authentication Server will also be referred to as the authentication server.

- The Identix authentication adapters are used on both the clients and the database servers to communicate biometric authentication data between the authentication server and the clients in order to authenticate a database user.

Both the manager and the client-side adapter interface with Identix products: TouchNet II Software Libraries, the TouchNet II Hardware Interface, and the TouchNet II Desktop Sensor, TouchNet III software libraries, TouchNet III desktop sensor.

> **More Information:** For a list of Identix documentation that describe these Identix products, see "Related Publications" on page xx in the Preface of this manual.

## Administration Architecture

The Fingerprint Security Server Administrators use the manager to scan user fingerprints, measure the accuracy of the fingerprints, and establish security policies for database servers. The manager sends this information to the authentication server which stores the data in the repository.

The administrator, or someone who can be trusted, uses the Identix TouchNet II or TouchNet III Software to store the secret key on the TouchNet II or TouchNet III device. This key must match the key stored in the DEFAULT security policy before authentication can occur.

## Authentication Architecture

Each user who wants to use the system must place a fingerprint on a TouchNet II or TouchNet II Desktop Sensor. The client-side adapter sends an authentication request to the server-side adapter which uses the previously enrolled fingerprint stored in the authentication server for comparison. For each authentication request from a client, the authentication server retrieves and sends the user's fingerprint and the database server's security policy back to the client-side adapter via the server-side adapter.

The user's authentication request causes the Oracle Advanced Security option Identix authentication adapter (client-side) to send the request to the biometric authentication adapter (server-side), which looks up the user's fingerprint in the authentication server, which returns the stored fingerprint and the associated security policy.

Using threshold level values from the associated security policy, the adapter (client-side) uses the TouchNet II Software Libraries to set threshold values on the TouchNet II Desktop Sensor. It then prompts for the placing of the user's finger on the TouchNet II Desktop Sensor. The adapters on the client and the database server work together to compare the user's fingerprint, the secret key, and the threshold levels against the administrator-entered security policy stored in the authentication server repository. If this data matches, the user is then authenticated.

# Prerequisites

- The Windows NT machine that is to become the manager PC must be running the Oracle Enterprise Manager 2.0 or above.
- Each Windows NT or Windows 95 machine that is to become a client PC must be running Net8.
- The authentication server and each database server must be running Oracle8 release 8.0.3 or higher or Oracle8*i*.
- Before proceeding with the installation of the Oracle Advanced Security option, you must make sure that each Windows NT and Windows 95 client has Net8 connectivity with its associated database server.

## Installing the TouchSAFE II Encrypt Device Driver for Windows NT

The Biometric Manager installation process automatically installs the necessary TouchNet II software and automatically configures the device if requested.

If during the installation of the Biometrics Manager, you chose not to allow the installer to set up your Identix TouchSafe II Device Driver, you can configure it manually as follows.

1. Change directory to $ORACLE_HOME\IDENTIX
   - If you are using the default IO port number 280 and the default Windows NT directory, go to Step 4.
   - If you are not using the default IO port number 280, go to Step 2.
   - If you are not using the default Windows NT directory C:\WINNT35\SYSTEM32\DRIVERS, go to Step 3.
2. Modify the IoPortAddress parameter in ETSIINT.INI to the current TouchSafe II Encrypt I/O port setting. For example:

```
IoPortAddress = REG_DWORD 0x00000360  for I/O port 0x360
```

3. Modify the Windows NT directory setting in ETSIINT.BAT with your Windows NT directory.

   For example:

   ```
   copy etsiint.sys c:\winnt\system32\drivers
   -> copy etsiint.sys path\drivers
   ```

4. Run the batch file ETSIINT.BAT.
5. Use the SetKey utility in the Identix demo program to set a hash key in Hex. Set the key to C001BABY for example (do not use this value!). Make sure the hash key matches exactly the one set in the DEFAULT Security policy.
6. Re-boot the system, and the device driver will start to work.
7. To make sure the device driver is running, check the Device Control Panel after re-boot. The device ETSIINT should be started already.

## Biometric Manager PC

On the manager PC:

1. Install Oracle Enterprise Manager on both the Oracle server and the Oracle client.
2. Install the Identix hardware and the Identix driver firmware and configure the Identix variables and devices.

   **More Information:** See the Identix Readme file

3. Install and test the Identix TouchNet II (Encrypt) 2.0 or TouchNet III.

   **More Information:** See "Installing the TouchSAFE II Encrypt Device Driver for Windows NT" on page 7-5 and your platform-specific installation documentation.

   Follow the instructions in the Identix manual to verify that the module works with the Identix demonstration program. This demonstration program must work on the PC before any other Oracle products can be loaded onto the PC. Refer to the Identix Readme file for additional information.

## Client PC

On each client PC:

1. Install the Identix hardware and the Identix driver firmware and configure the Identix variables and devices. Refer to the Identix Readme file for additional information.
2. Install and test the Identix TouchNet II (Encrypt) 2.0 or TouchNet III. Follow the instructions in the Identix manual to verify that the module works with the Identix demonstration program. This demonstration program must work on the PC before any other Oracle products can be loaded onto the PC.

   **More Information:** See "Installing the TouchSAFE II Encrypt Device Driver for Windows NT" on page 7-5 and your platform-specific installation documentation.

3. Install the Oracle Advanced Security option Identix authentication adapter.

   **More Information:** See your platform-specific documentation. Refer also to the Identix Readme file.

## Database Server

The biometric authentication adapter must be installed on each production database that will use biometric services for its authentication. Install the biometric authentication adapter following the instructions in your platform-specific documentation.

---

**Note:** Do not install the adapter on the database storing the fingerprint repository.

---

## Biometric Authentication Service

The Biometric Authentication Service is the database that houses both the user and fingerprint information. This database can be any Oracle 8.0.3 or later production database. It should be on a secure, trusted system with strict security and access controls. The adapter should not be installed on this database.

# Enabling Biometric Authentication

To configure the Biometric Authentication Service, you perform the following tasks, each of which is described in the next few pages.

- Step 1: Configure the database server that is to become the authentication server
- Step 2: Configure Identix authentication
- Step 3: Establish a net service name for the fingerprint repository server
- Step 4: Verify that the address of the database server is accessible to the client
- Step 5: Configure the manager PC

## Step 1: Configure the database server that is to become the authentication server

1. Connect to the database server as SYSTEM / MANAGER (or whatever your system password is).
2. Test the connection by connecting as:

```
ofm_admin/ofm_admin
```

## Step 2: Configure Identix authentication

To configure Identix authentication you perform the tasks in the following list. Each task is described below.

- Configure the authentication method on the Oracle server and the Oracle client
- Configure the fingerprint server name on the client and the server
- Configure the user name, password, and fingerprint method
- Configure the INIT.ORA file
- Configure the ORACLE.INI file

Unless otherwise indicated, you can configure Identix authentication either by using the Net8 Assistant, or by modifying the file sqlnet.ora with any text editor.

## Using Net8 Assistant

This graphical interface tool makes it easy to set parameters in the sqlnet.ora file and other Oracle8*i* configuration files.

**To launch Net8 Assistant:**

- On UNIX, run the script netasst at $ORACLE_HOME/bin.
- On Windows platforms, click the Start button > Programs > Oracle for Windows NT—*HOME_NAME* > Net8 Assistant.

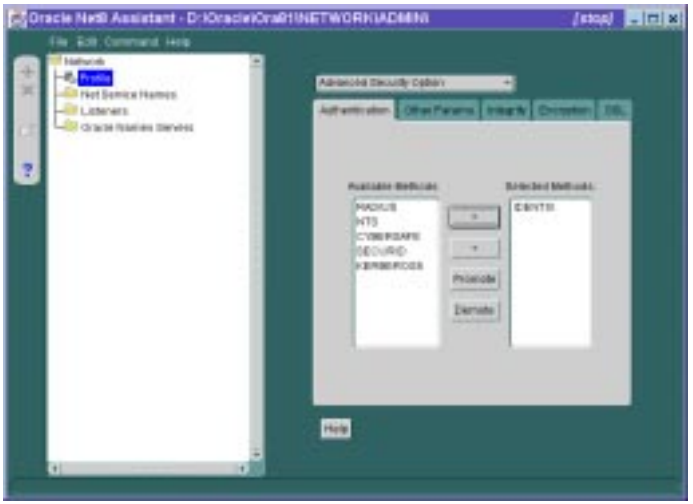
**To begin configuring the Oracle Advanced Security option using Net8 Assistant:**
In the Net8 Assistant's left pane, click the Profile folder. Then go to the drop down list box at the top of the right pane, and select Advanced Security Option. The tabbed pages for the Oracle Advanced Security option appear.

**To save changes with Net8 Assistant:**
Go to the menu bar and click File > Save Network Configuration.

### Configure the authentication method on the Oracle server and the Oracle client

Do this by setting the SQLNET.AUTHENTICATION_SERVICES parameter.

***Figure 7–2 Using Net8 Assistant to Configure Authentication***

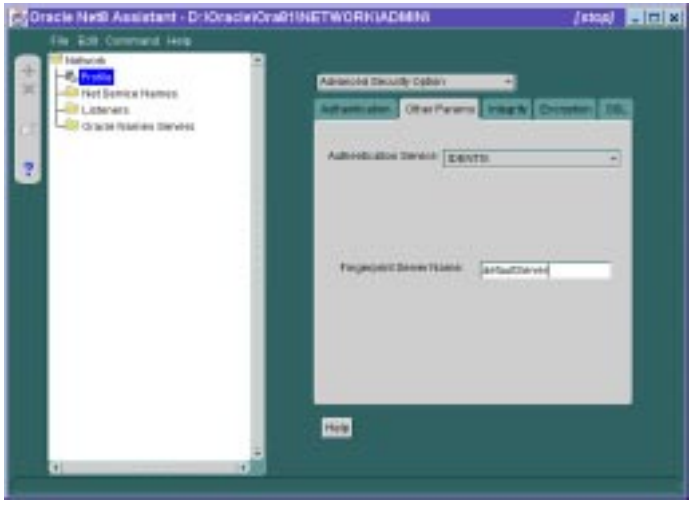


| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 7–2.<br>1. Select the Authentication tab.<br>2. In the Available Methods list, select Identix.<br>3. Click the [>] button to move the method over to the Selected Methods list. Move any other methods you want to use in the same way.<br>4. Arrange the selected methods in order of desired use. To do this, select a method in the list, then click [Promote] or [Demote] to position it in the list. For example, put Identix at the top of the list if you want that method to be the first one used. | Set the following parameter:<br>`SQLNET.AUTHENTICATION_SERVICES=`<br>`(`*`IDENTIX`*`)` |

### Configure the fingerprint server name on the client and the server

Do this by setting the SQLNET.IDENTIX_FINGERPRINT_DATABASE parameter.

***Figure 7–3 Using Net8 Assistant to Configure the Fingerprint Server Name***



| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 7–3.<br>**1.** Select the Other Params tab.<br>**2.** Click the Authentication Service drop-down list box, and select IDENTIX.<br>**3.** In the Fingerprint Server Name box, type the name of the fingerprint server you want to use. | Set the following parameter:<br>`SQLNET.IDENTIX_FINGERPRINT_DATABASE=` *`service_name`*<br>where service_name is the name of your authentication server. |

### Configure the user name, password, and fingerprint method

Use a text editor to set the following parameters in the file sqlnet.ora:

```
sqlnet.identix_fingerprint_database_user= ofm_client
sqlnet.identix_fingerprint_database_password= ofm_client
sqlnet.identix_fingerprint_method= oracle
```

where username is the well-known user name: ofm_client, and `password` is the well-known password: ofm_client

---

**Note:** The samples directory contains a file that shows how to set these parameters.

**Note:** The ofm_client user name and password are set up by running NAUICAT.SQL. You should not change ofm_client.

---

### Configure the INIT.ORA file

Use a text editor to set the following parameters in the initialization file (init.ora):

```
REMOTE_OS_AUTHENT = false
OS_AUTHENT_PREFIX = ""
```

---

**Note:** The local naming configuration file (tnsnames.ora) on the database server should contain the service name of the fingerprint repository. If they are on the same database, use the following with the service name:

```
(security=(authentication_service=none))
```

---

### Configure the ORACLE.INI file

In the Oracle section of the oracle.ini file, use a text editor to specify the USERNAME parameter. This parameter sets the name of the database user with which the client connects to the database.

## Step 3: Establish a net service name for the fingerprint repository server

**More Information:** See the *Net8 Administrator's Guide.*

### Step 4: Verify that the address of the database server is accessible to the client

**More Information:** See the *Net8 Administrator's Guide.*

### Step 5: Configure the manager PC

Configure the manager PC with a net service name to connect to the authentication server.

**More Information:** See *Net8 Administrator's Guide*

## Administering the Biometric Authentication Service

You administer the Biometric Authentication Service by using the Biometric Manager.

**More Information:** See your Identix documentation.

**To create a hashkey on each of the clients:**
Use the Identix Setkey utility to configure a hexadecimal hashkey on each of the clients: e.g., FF30EE. This key must be the same for each client and must match the DEFAULT Policy hashkey. This key can range from one to thirty-two hexadecimal digits.

**To create a user for biometric authentication:**

1. On the client use the Windows NT User Manager to create a user name. (This user name must match the user name used in the next step.)
2. On the database server, restart the database and create an Oracle Server account for the user. Use SQL*Plus if using the Oracle Enterprise Manager or SQL*Plus connected as a user with the create user database role. Use the following syntax to create an account:

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

3. The OS_AUTHENT_PREFIX is an Oracle Server initialization parameter. The default value for OS_AUTHENT_PREFIX is OPS$. The user name in this step should match the user name created at the client. If you reset os_authent_prefix, you must stop and restart your database.

**Note:** Oracle user names are limited to 30 characters and user names can be long, so it is strongly recommended that os_authent_ prefix be set to a null value:

```
OS_AUTHENT_PREFIX=""
```

**Note:** An Oracle user with user name should not yet exist.

## Example

If you create the user "king," and set OS_AUTHENT_PREFIX to a null value (""), you should use SQL*Plus to create an Oracle user account using the following syntax:

```
SQL> CREATE USER king IDENTIFIED EXTERNALLY;
```

At the minimum, you should give the user the "create session" privilege:

```
SQL> GRANT CREATE SESSION TO king;
```

Use the Biometric Manager to enroll the user in the Biometric Authentication Service.

The user "king" can now be biometrically authenticated to Oracle.

**More Information:** For information on creating users identified externally, see *Oracle8i Administrator's Guide* and *Oracle8i Distributed Database Systems.*

For information on logging in to a database server once biometric authentication has been installed and configured, see "Authenticating Users With the Biometric Authentication Service" on page 7-15.

For information on storing the secret key in the client, see your Identix documentation.

# Authenticating Users With the Biometric Authentication Service

Before you authenticate a user, make sure that the Biometric Authentication Service has been installed and configured and the steps in "Administering the Biometric Authentication Service" on page 7-13 have been executed.

**To authenticate users with the Biometric Authentication Service:**

1. Log on as the user name assigned by the database administrator.
2. If you are using TouchNet II, set the system environment variable. The following variable is based on the 10 port setting on your TouchNet II firmware.

   ```
   ETSII_IOPORT = 0X280
   ```

   **Note:** The TouchNet III device does not use the ETSII_IOPORT environment variable. Instead, it uses the file tn3com.ini to set the port and baud rate.

3. Launch SQL*Plus.
4. Type the name of your database server when SQL*Plus displays the prompt:

   ```
   SQL>connect /@net_service_name
   ```

   where, net_service_name is the name of the database server.

5. Wait for the beep that announces the Net8 Native Authentication dialog box.

   **Note:** On some systems the dialog box is displayed behind the current window. The beep alerts you when it is displayed.

6. Click OK in the Net8 Native Authentication dialog box.
7. When a message appears telling you to place your finger on the desktop fingerprint sensor, use the same finger that you and the administrator entered into the authentication server repository.
8. Remove your finger at the prompt. Another prompt tells you whether you have been authenticated.

**If the authentication fails, and the message, "Access Denied," appears:**
Try one of the following recovery methods:

- Restart the authentication process.

  **More Information:** See "Authenticating Users With the Biometric Authentication Service" on page 7-15.

- Have the security administrator lower the threshold value to 80.
- Have the security administrator re-enroll you.

  **More Information:** See the Biometric Manager online Help for task oriented procedures.

# Troubleshooting

Check the following if you encounter any problems while installing or using Biometric Authentication.

- Ensure that the Identix Set Key utility hash key exactly matches the Biometric Manager DEFAULT Policy hash key.
- The NT user name must exactly match the externally defined user name in the database server and the user name used when adding the user with the Biometric Manager.
- Domain naming must be consistent. For example, if the local naming configuration (tnsnames.ora) uses .world as an appendix to the service name, then the profile (sqlnet.ora) must reflect this naming convention for the service name. For example:

```
TNSNAMES.ORA
biometrics.world = (DESCRIPTION =
                     (ADDRESS_LIST =
                      (ADDRESS =
                        ...
SQLNET.ORA
sqlnet.identix_fingerprint_database=biometrics.world
```

- It is possible to use one database for both the biometric authentication service and the production database; however, this is not recommended. If you do this, add the following line of code to the local naming configuration field (TNSNAMES.ORA) on the server and on each PC client.

```
(connect_data =
    (service_name = service_name)
    (security = (Authentication_service = NONE))
```

# 8

# Configuring DCE GSSAPI Authentication

DCE GSSAPI authentication enables you to use DCE authentication even if you do not use other portions of the Oracle DCE Integration product in your environment.

This chapter describes how to configure and use DCE GSSAPI authentication.

---

**Note:** Check your platform-specific installation documentation to be sure that release 8.1.5 of the Oracle Advanced Security option supports Oracle DCE integration for your platform.

---

---

**Note:** If you are already using Oracle DCE Integration, you do not also have to use the DCE GSSAPI authentication adapter. The Oracle DCE Integration product, described in Part II, "Oracle Advanced Security and Oracle DCE Integration", includes DCE authentication.

---

---

**Note:** The instructions in this chapter assume that you are familiar with DCE terminology. For more information about DCE, see:

- Part II,"Oracle Advanced Security and Oracle DCE Integration", in this guide
- your operating system-specific DCE administration guide
- the documentation listed in "Related Publications"in the Preface of this guide

---

# Configuring DCE GSSAPI Authentication

To configure DCE GSSAPI authentication you follow these four general steps, each of which is explained below:



## Step 1: Create the DCE principal

To create the DCE principal used by the Oracle server to validate authentication, type the commands below shown in **bold** typeface. These instructions assume the Oracle Server principal is named "oracle_server". Type the following commands on the database server.

```
% su
  password:    (root password is not echoed)
# dce_login cell_admin cell_admin_password
# rgy_edit
Current site is: registry server at
    /.../cellname/subsys/dce/sec/master
rgy_edit=> do p
Domain changed to: principal
rgy_edit=> add oracle_server
rgy_edit=> do a
Domain changed to: account
rgy_edit=> add oracle_server -g none -o none -pw oracle_server_
password -mp cell_admin_password
rgy_edit=> ktadd -p oracle_server -pw oracle_server_password
rgy_edit=> quit
bye
```

## Step 2: Configure the new DCE principal and turn on DCE GSSAPI authentication

The following instructions assume that the Oracle server principal is named "oracle_server."

Add the following lines to the sqlnet.ora file.

```
SQLNET.AUTHENTICATION_GSSAPI_SERVICE=/.../cellname/oracle_server
SQLNET.AUTHENTICATION_SERVICES=(DCEGSSAPI)
```

> **Note:** The Oracle Server principal name used above must be a fully qualified name, including the cell name.

## Step 3: Set up the account you will use to authenticate to the database

Create the DCE principal used by the Oracle client to connect to the database. The following instructions assume the Oracle client principal is named "oracle".

```
% dce_login cell_admin cell_admin_password
% rgy_edit
Current site is : registry server at /.../cellname/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_client_password
-mp cell_admin_password
rgy_edit=> quit
bye
```

Create the Oracle database user account. The following instructions show how to use the SQL*Plus to do this.

```
SQL> connect internal
Connected
SQL> create user "/.../CELLNAME/ORACLE" identified externally;
Statement processed.
SQL> grant connect to "/.../CELLNAME/ORACLE";
Statement processed.
SQL> exit
```

**Note:** The Oracle client principal name must be a fully qualified principal (including full cell designation), must be in uppercase, and must be enclosed within quotes.

## Step 4: Connect to an Oracle server using DCE GSSAPI authentication

The following instructions assume the Oracle Server principal is "oracle_server", the Oracle client principal is "oracle", and the database service name is "sales".

1. If your DCE authentication is not already encapsulated into the operating system authentication, log in:

   ```
   % dce_login oracle_client_principal oracle_client_password
   ```

   For example:

   ```
   % dce_login oracle oraclnt
   ```

2. Connect to the Oracle database using DCE GSSAPI authentication.

   ```
   % SQLPLUS /@<database_service_name>
   ```

   For example:

   ```
   % SQLPLUS /@sales
   ```

# 9

# Configuring SSL Authentication

This chapter covers the following topics:

- SSL in an Oracle Environment
- SSL beyond an Oracle Environment
- SSL in Combination with Other Authentication Methods
- Issues When Using SSL
- Enabling SSL
- Ongoing Administrative Tasks
- Logging in to the Database

# SSL in an Oracle Environment

SSL (Secure Sockets Layer) is 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).

This section discusses the following topics.



## What You Can Do with the SSL Feature

By including the SSL feature, the Oracle Advanced Security option expands its own support for encryption, and provides public key authentication based on the SSL standard.

You can use the SSL feature of the Oracle Advanced Security option to secure communications between any client and any server. Specifically, you can use SSL to authenticate:

- Any client or server to one or more Oracle servers
- An Oracle server to any client

You can use SSL features by themselves or in combination with other authentication methods supported by the Oracle Advanced Security option. For example, you can use the encryption provided by SSL in combination with the authentication provided by Kerberos.

> **More Information:** For more information on authentication methods, see Chapter 1, "Introduction to Oracle Advanced Security".

You can use SSL in one of three authentication modes. You can require:

- Only the server to authenticate itself to the client
- Both client and server to authenticate themselves to each other
- Neither the client nor the server to authenticate itself to the other

You can also disable SSL authentication and use its encryption feature alone.

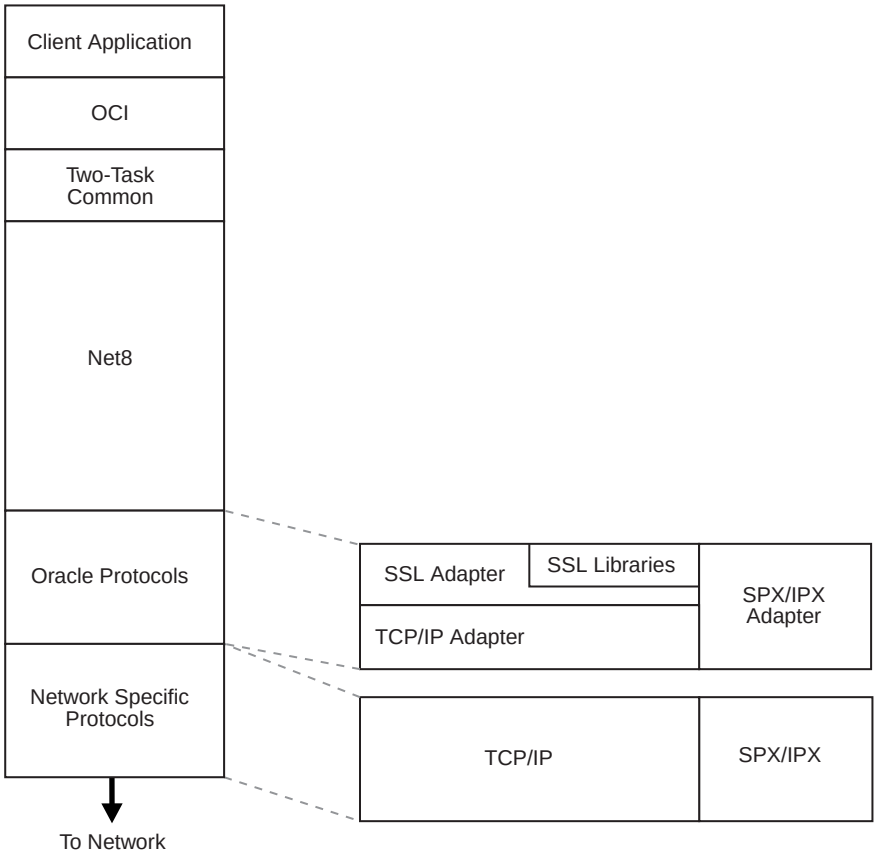
**More Information:** For a full explanation of SSL, see the Internet Engineering Task Force document *The SSL Protocol*, Version 3.0.

For important security concepts and terminology, see the Glossary.

## Architecture of SSL in an Oracle Environment

In an Oracle environment, SSL operates at the Oracle Protocols layer using TCP/IP as illustrated in Figure 9–1.

***Figure 9–1 Architecture of SSL in an Oracle Environment***




## Components of SSL in an Oracle Environment

The components of SSL in an Oracle environment include the following, each of which is described below:

- Certificate
- Certificate Authority (CA)
- Wallet

### Certificate

A certificate ensures that the entity's identity information is correct and that the public key actually belongs to that entity. A certificate is created when an entity's public key is signed by a trusted identity, that is, a certificate authority (CA), described more fully in this section.

A certificate contains the entity's name, public key, serial number, and expiration date. It may contain information about the privileges associated with the certificate. Finally, it contains information about the CA that issued it.

When an entity receives a certificate—either its own certificate from a CA or a certificate from another entity—it verifies that certificate is a **trusted certificate**, that is, that it is issued by a trusted certificate authority. A certificate is valid until it expires.

### Certificate Authority (CA)

A trusted third party that certifies that other entities—users, databases, administrators, clients, servers—are who they say they are. The certificate authority verifies the user's identity and grants a certificate, signing it with the certificate authority's private key.

Different CAs may have different identification requirements when issuing certificates. One certificate authority may want to see a user's driver's license, another may want the certificate request form to be notarized, yet another may want fingerprints of the person requesting a certificate.

The certificate authority publishes its own certificate which includes its public key. Each network entity has a list of such certificates of the CAs it trusts. Before communicating with another entity, a given entity uses this list to verify that the signature on the other entity's certificate is from a trusted CA.

Network entities can obtain their certificates from the same or from different CAs.

---

**Note:** The Oracle Advanced Security option is installed with a set of VeriSign certificates.

For information on adding certificates, see "Step 5: Create a new wallet" on page 9-32.

For information on adding trusted certificates, see "Step 7: Add new trusted certificates" on page 9-39.

---

### Wallet

An abstraction used to store and manage authentication data such as keys, certificates, and trusted certificates which are needed by SSL. In an Oracle environment, each system using SSL has a wallet with an X509 version 3 certificate, private key, and list of trusted certificates.

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

- generate a public-private key pair and create a certificate request for submission to a certificate authority
- install a certificate for the identity
- configure trusted certificates for the identity

---

**Note:** Installation of the Oracle Advanced Security option, release 8.1.5, includes installation of the Oracle Wallet Manager, release 1.3-beta.

---

**More Information:** For information on the Oracle Wallet Manager, see the sections beginning with "Step 4: Start the Oracle Wallet Manager" on page 9-30. See also "Managing Wallets" on page 9-43.

## How SSL Works in an Oracle Environment: The SSL Handshake

**More Information:** For a full explanation of SSL, see the Internet Engineering Task Force document *The SSL Protocol*, Version 3.0.

At the beginning of their communication, the client and server perform an SSL handshake which includes three important tasks:

- The client and server establish which **cipher suite** to use.
- The server sends its certificate to the client. The client verifies that the server's certificate was signed by a trusted CA.

  Similarly, if client authentication is required, the client sends its own certificate to the server. The server verifies that the client's certificate was signed by a trusted CA.
- The client and server exchange key material using public key cryptography, and, from this material, they each generate a session key. All subsequent communication between client and server is encrypted and decrypted by using this set of session keys and the negotiated cipher suite.

In an Oracle environment, the authentication process involves three basic steps:

1. The user initiates a Net8 connection to the server by using SSL.
2. SSL performs the handshake between client and server.
3. If the handshake is successful, the server verifies that the user has the appropriate **authorization** to access the database.

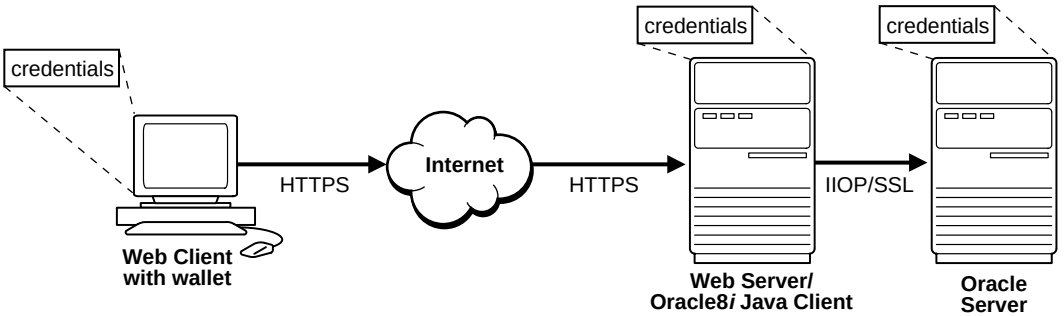
# SSL beyond an Oracle Environment

You can use the SSL feature of the Oracle Advanced Security option to secure connections between non-Oracle clients and Oracle servers. For example, SSL can allow a client outside an Oracle network to access authorized data securely within the Oracle network.

Figure 9–2 offers an example of how you can use SSL to secure connections between Oracle and non-Oracle entities—beginning over the Internet and proceeding to an Oracle server. In this example, a Web server runs as an Oracle8*i* Java client. It receives messages over **HTTPS** (HTTP secured by SSL), and sends **CORBA** requests to the Oracle server via a servlet over **IIOP** / SSL (IIOP secured by SSL.). Note that, in this example, the Web server passes its own—and not the Web client's—certificate to the Oracle server.

**More Information:** For information on using and configuring IIOP / SSL, see *Oracle8i Enterprise JavaBeans and CORBA Developer's Guide.*

***Figure 9–2 Connecting to an Oracle Server over the Internet***




# SSL in Combination with Other Authentication Methods

You can combine the features of SSL with other authentication methods supported by the Oracle Advanced Security option, for example, Kerberos, SecurID, or Identix.

**More Information:** For more information on authentication methods, see Chapter 1, "Introduction to Oracle Advanced Security".

This section discusses the topics in the following list.

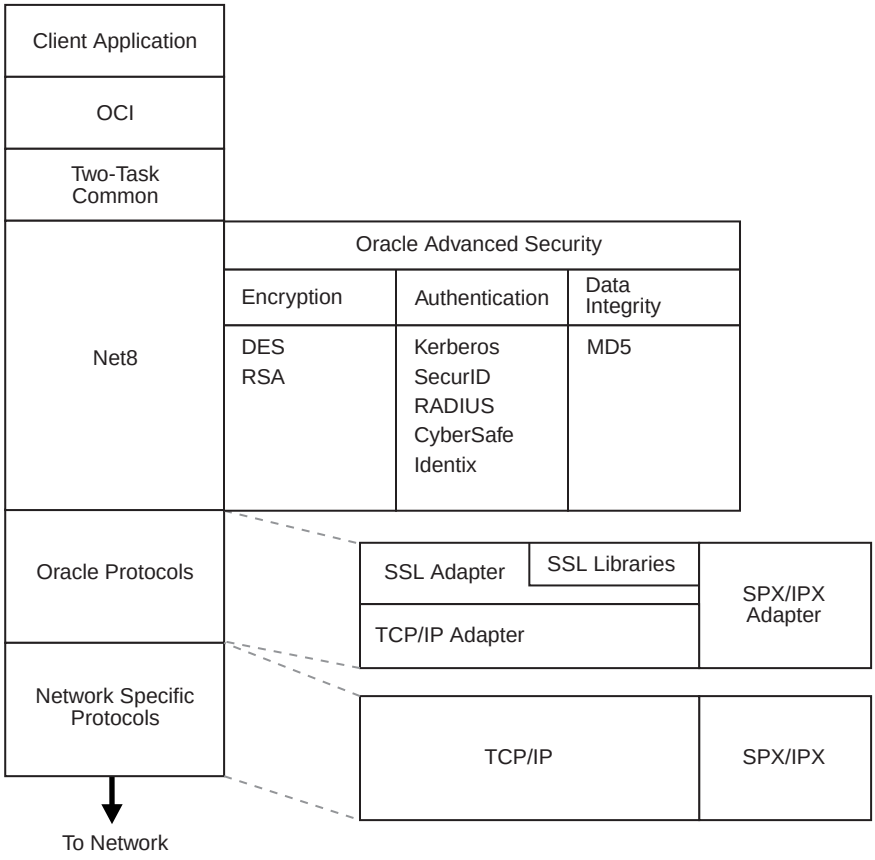
- Architecture of SSL in Combination with Other Authentication Methods
- Example: Using SSL in Combination with Other Authentication Methods

## Architecture of SSL in Combination with Other Authentication Methods

As Figure 9–3 illustrates, the Oracle Advanced Security option operates at the session layer, on top of SSL which uses TCP / IP at the transport layer.

**More Information:** For more information on stack communications in an Oracle networking environment, see *Net8 Administrator's Guide.*

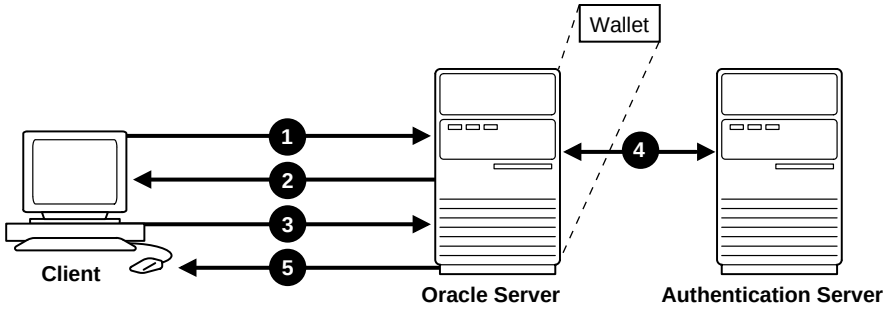
***Figure 9–3 SSL in Relation to the Oracle Advanced Security Option***



## Example: Using SSL in Combination with Other Authentication Methods

Figure 9–4 illustrates one possible scenario when using SSL in combination with another authentication method supported by the Oracle Advanced Security option. In this scenario, server authentication uses SSL, and client authentication uses an authentication method supported by the Oracle Advanced Security option, for example, Kerberos, SecurID, Identix.

***Figure 9–4 Example: SSL in Relation to Other Oracle Advanced Security Authentication Methods***




1. The client seeks to connect to the Oracle 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. See "How SSL Works in an Oracle Environment: The SSL Handshake" on page 9-7.
3. Once the SSL handshake is successfully completed, the user seeks access to the database.
4. The Oracle server exchanges the user's authentication information with the authentication server.
5. Upon validation by the authentication server, the Oracle server grants access and authorization to the user.

**Note:** You can use SSL encryption in combination with another authentication method of the Oracle Advanced Security option. When you do this, you must disable any non-SSL encryption to comply with government regulations prohibiting double encryption. If you do not do this, the connection will fail.

For information on how to disable encryption in the Oracle Advanced Security option, see "Negotiating Encryption and Checksumming" on page 2-7.

You cannot use SSL authentication with the Oracle Advanced Security option encryption.

## Issues When Using SSL

SSL cannot be proxied through traditional application level firewalls (such as the CERN proxy server).

SSL does not provide authorization, that is, the allocation of privileges and roles. Rather, these are provided in Oracle8*i* by the Oracle server.

Because SSL does authentication and encryption, from a performance standpoint it is slower than the standard **Net8** TCP/IP transport.

The SSL feature of the Oracle Advanced Security option does not work with versions of Oracle earlier than Oracle8*i*.

Each SSL authentication mode as described on page 9-2 requires unique configuration settings. These unique settings are explained in the section "Enabling SSL" on page 9-12.

# Enabling SSL

To enable SSL, you perform the general tasks in the following list. Each task is explained more fully in the next several pages.

- Step 1: Install Oracle Advanced Security and the Oracle Wallet Manager
- Step 2: Configure SSL on the client
- Step 3: Configure SSL on the server
- Step 4: Start the Oracle Wallet Manager
- Step 5: Create a new wallet
- Step 6: Install a certificate into the new wallet
- Step 7: Add new trusted certificates
- Step 8: Save changes to your wallet
- Step 9: For single sign-on functionality, create an auto-login wallet
- Step 10: Create a user identified globally through certificates on the Oracle server

## Step 1: Install Oracle Advanced Security and the Oracle Wallet Manager

Do this on both the client and server.

When you install the Oracle Advanced Security option, the Oracle Universal Installer adds both SSL and the Oracle Wallet Manager to your system.

> **More Information:** See the Oracle8*i* installation documentation for your platform.

## Step 2: Configure SSL on the client

To configure SSL on the client, perform the tasks in the following list, each of which is described more fully below.

- If you have not yet configured SSL, specify client configuration
- Set the Oracle wallet location
- Set the SSL cipher suites (optional)
- Set the required SSL version (optional)
- Set SSL as an authentication service (optional)
- Select "TCP/IP with SSL" as the Net Service Name

There are two ways to configure a parameter:

- Static—setting the parameter in sqlnet.ora. You can set the location of the Oracle wallet and modify the default settings of any of the other parameters either by using the Net8 Assistant (Figure 9–5) or by editing the sqlnet.ora file with any text editor.
- Dynamic—setting the parameter in the security subsection of the Net8 address.

> **More Information:** For the dynamic parameter names, see "Parameters for Clients and Servers using SSL" on page B-7.

### Using Net8 Assistant

This graphical interface tool makes it easy to set parameters in the sqlnet.ora file and other Oracle8*i* configuration files.

**To launch Net8 Assistant:**

- On UNIX, run the script netasst at $ORACLE_HOME/bin.
- On Windows platforms, click the Start button > Programs > Oracle for Windows NT—*HOME_NAME* > Net8 Assistant.

**To begin configuring the Oracle Advanced Security option using Net8 Assistant:**
In the Net8 Assistant's left pane, click the Profile folder. Then go to the drop down list box at the top of the right pane, and select Advanced Security Option. The tabbed pages for the Oracle Advanced Security option appear.

**To save changes with Net8 Assistant:**
Go to the menu bar and click File > Save Network Configuration.

### If you have not yet configured SSL, specify client configuration

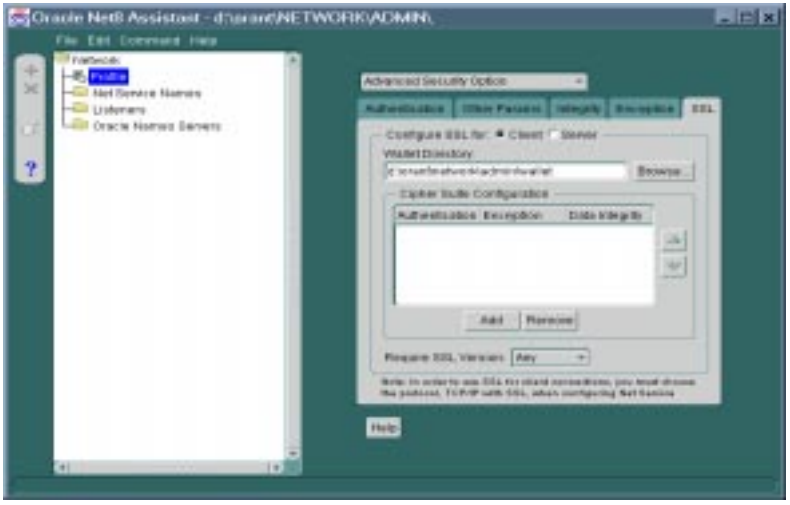
You need to do this only if you are using Net8 Assistant.

***Figure 9–5 Using Net8 Assistant to Specify Client Configuration***


Refer to Figure 9–5.

1. In the right pane, select the SSL tab.
2. On the SSL tabbed page, select the Configure SSL for Client radio button. The SSL tabbed page changes to allow you to make the appropriate configurations for the client (Figure 9–6 on page 9-15).

*Figure 9–6 Net8 Assistant's SSL Tabbed Page for Configuring a Client*




## Set the Oracle wallet location

Do this by setting the The OSS.SOURCE.MY_WALLET parameter. There is no default for this parameter.

| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 9–6.<br>**1.** At the top of the SSL tabbed page, be sure that the Configure SSL for Client radio button is selected.<br>**2.** In the Wallet Directory box, type the directory for the Oracle wallet. To find it by searching your file system, click the Browse button.<br>**Note:** You must enter this same directory later when you come to "Step 5: Create a new wallet" on page 9-32. | Set the following parameter:<br>`oss.source.my_wallet =`<br>`(SOURCE=`<br>`(METHOD=File)`<br>`(METHOD_DATA=`<br>`(DIRECTORY=<`*`your_wallet_location`*`>)`<br>`)`<br>`)` |

### Set the 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.

The SSL_CIPHER_SUITES parameter sets the cipher suites SSL uses.

When you install the Oracle Advanced Security option, several SSL cipher suites are set for you by default. Setting one or more cipher suites yourself overrides the other default cipher suites set during installation. For example, if you use Net8 Assistant 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.
- 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. By contrast, in the case of an Enterprise Java Beans (EJB) user, the server does not require the client to authenticate itself. In this case, you can use Diffie-Hellman anonymous authentication.

Normally, you would prioritize cipher suites starting with the strongest and moving to the weakest.

The following two tables list the available SSL cipher suites supported in both the domestic and export versions of the Oracle Advanced Security option. These cipher suites are set by default when you install Oracle Advanced Security option. These tables also list the authentication, encryption, and data integrity types each cipher suite uses.

*Table 9–1 SSL Cipher Suites in Domestic Version of Oracle Advanced Security*

| Cipher Suite | Authentication | Encryption | Data Integrity |
|---|---|---|---|
| SSL_RSA_WITH_3DES_EDE_CBC_SHA | RSA | 3DES EDE CBC | SHA |
| SSL_RSA_WITH_RC4_128_SHA | RSA | RC4 128 | SHA |
| SSL_RSA_WITH_RC4_128_MD5 | RSA | RC4 128 | MD5 |
| SSL_RSA_WITH_DES_CBC_SHA | RSA | DES CBC | SHA |
| SSL_DH_anon_WITH_3DES_EDE_CBC_SHA | DH anon | 3DES EDE CBC | SHA |
| SSL_DH_anon_WITH_RC4_128_MD5 | DH anon | RC4 128 | MD5 |
| SSL_DH_anon_WITH_DES_CBC_SHA | DH anon | DES CBC | SHA |
| SSL_RSA_EXPORT_WITH_RC4_40_MD5 | RSA | RC4 40 | MD5 |
| SSL_RSA_EXPORT_WITH_DES40_CBC_SHA | RSA | DES40 CBC | SHA |
| SSL_DH_anon_EXPORT_WITH_RC4_40_MD5 | DH anon | RC4 40 | MD5 |
| SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA | DH anon | DES40 CBC | SHA |

*Table 9–2 SSL Cipher Suites in Export Version of Oracle Advanced Security*

| Cipher Suite | Authentication | Encryption | Data Integrity |
|---|---|---|---|
| SSL_RSA_EXPORT_WITH_RC4_40_MD5 | RSA | RC4 40 | MD5 |
| SSL_RSA_EXPORT_WITH_DES40_CBC_SHA | RSA | DES40 CBC | SHA |
| SSL_DH_anon_EXPORT_WITH_RC4_40_MD5 | DH anon | RC4 40 | MD5 |
| SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA | DH anon | DES40 CBC | SHA |

**Note:** If you use SSL in conjunction with another authentication method supported by the Oracle Advanced Security option, you must disable any non-SSL encryption to comply with government regulations prohibiting double encryption. If you do not do this, the connection will fail.

For information on how to disable encryption in the Oracle Advanced Security option, see "Negotiating Encryption and Checksumming" on page 2-7.

| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 9–6 on page 9-15.<br>1. At the top of the SSL tabbed page, be sure that the Configure SSL for Client radio button is selected.<br>2. Click the Add button. A secondary dialog box (Figure 9–7) listing available cipher suites appears.<br>3. To add a suite to your list, select it in the secondary dialog box and click OK. The cipher suite appears in the Cipher Suite Configuration window of the main Net8 Assistant dialog box.<br>4. To prioritize the cipher suites, use the Promote and Demote buttons. The order you set determines the order in which the client is to negotiate with other entities on which cipher suite to use. | Set the following parameter in the server's sqlnet.ora file, listing the cipher suites in priority:<br>`SSL_CIPHER_SUITES=`<br>`(`*`SSL cipher suite1`* `[,`*`SSL cipher suite2`*`])` |

***Figure 9–7 Net8 Assistant Secondary Dialog Box: Selecting a Cipher Suite***

### Set the required SSL version (optional)

Do this by setting the SSL_VERSION parameter. This parameter determines which version of SSL must be running on the machines with which the client is communicating. You can require those machines to use SSL 3.0, or any existing or future versions. The default setting for this parameter in sqlnet.ora is "0"; in Net8 Assistant it is "Any".

| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 9–6 on page 9-15.<br>1. At the top of the SSL tabbed page, be sure that the Configure SSL for Client radio button is selected.<br>2. In the Require SSL Version scroll box the default is Any. Accept this default or select the SSL version you want to enforce. | Set the following parameter:<br>`SSL_VERSION={ 0 \| 3.0 }` |

### Set SSL as an authentication service (optional)

The SQLNET.AUTHENTICATION_SERVICES parameter sets the SSL authentication service.

You must set this parameter only if *both* of the following two conditions apply:

- You want to use SSL authentication in conjunction with another authentication method supported by the Oracle Advanced Security option. For example, 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 or SecurID.

  and

- You are not using Net8 Assistant to make you configurations.

If both of the above conditions apply, add TCPS to this parameter in the sqlnet.ora file by using a text editor. For example:

```
SQLNET.AUTHENTICATION_SERVICES = (BEQ,
                                  TCPS,
                                  identix,
                                  securid)
```

If either or both of the above conditions do *not* apply, you do *not* need to set this parameter.

### Select "TCP/IP with SSL" as the Net Service Name

The client must be configured with the location of the listener. For an SSL connection, the address of the listener must use the TCP/IP with SSL protocol.

> **More Information:** See the on-line help for the Net8 Assistant and the *Net8 Administrator's Guide*.

## Step 3: Configure SSL on the server

During installation, Oracle sets defaults on both the Oracle server and the Oracle client for all SSL parameters except the location of the Oracle wallet. To configure SSL on the server, perform the tasks in the following list, each of which is described below.

- If you have not yet configured SSL, specify server configuration
- Set the Oracle wallet location
- Set the SSL cipher suites (optional)
- Set the required SSL version (optional)
- Set SSL client authentication (optional)
- Set SSL as an authentication service (optional)
- Select "TCP/IP with SSL" as the listening endpoint

As with the Oracle client, there are two ways to configure a parameter on the Oracle server:

- Static—setting the parameter in sqlnet.ora. You can set the location of the Oracle wallet, and modify the default settings of any of the other parameters, either by using the Net8 Assistant or by editing the sqlnet.ora file with any text editor.
- Dynamic—setting the parameter in the security subsection of the Net8 address.

> **More Information:** For the dynamic parameter names, see "Parameters for Clients and Servers using SSL" on page B-7.

### Using Net8 Assistant

This graphical interface tool makes it easy to set parameters in the sqlnet.ora file and other Oracle8*i* configuration files.

**To launch Net8 Assistant:**

- On UNIX, run the script netasst at $ORACLE_HOME/bin.
- On Windows platforms, click the Start button > Programs > Oracle for Windows NT—*HOME_NAME* > Net8 Assistant.

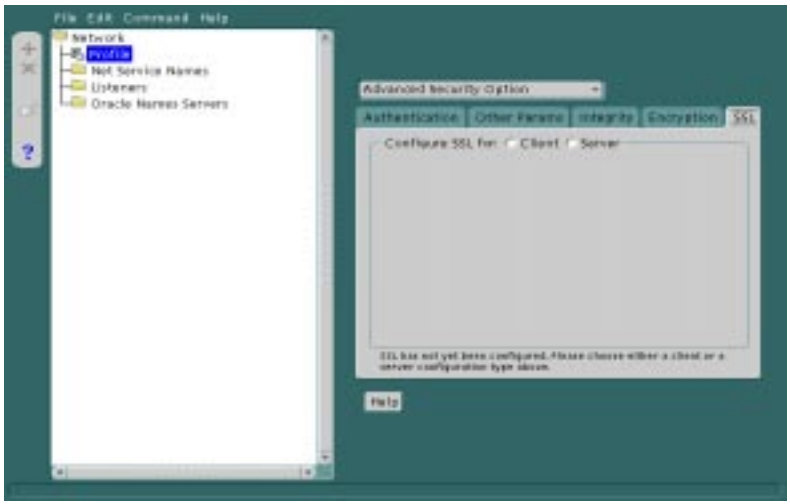
**To begin configuring the Oracle Advanced Security option using Net8 Assistant:**
In the Net8 Assistant's left pane, click the Profile folder. Then go to the drop down list box at the top of the right pane, and select Advanced Security Option. The tabbed pages for the Oracle Advanced Security option appear.

**To save changes with Net8 Assistant:**
Go to the menu bar and click File > Save Network Configuration.

### If you have not yet configured SSL, specify server configuration

You need to do this only if you are using Net8 Assistant.

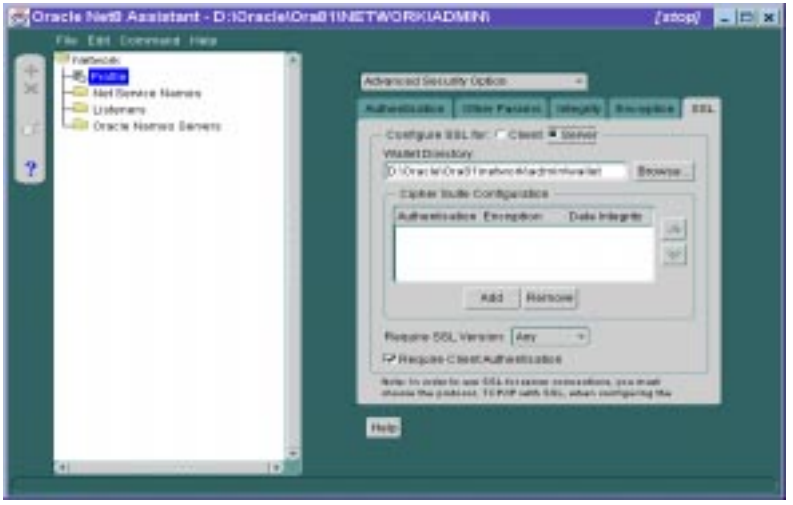
***Figure 9–8 Using Net8 Assistant to Specify Server Configuration***




Refer to Figure 9–8.

1. In the right pane, select the SSL tab.
2. On the SSL tabbed page, select the Configure SSL for Server radio button. The SSL tabbed page changes to allow you to make the appropriate configurations for the server (Figure 9–9 on page 9-23).

***Figure 9–9 Net8 Assistant's SSL Tabbed Page for Configuring a Server***



### Set the Oracle wallet location

Do this by setting the OSS.SOURCE.MY_WALLET parameter. There is no default for this parameter.

---

**Note:** There are two occasions during the client and the server configuration when you set the location of the Oracle wallet. **Be sure to enter the same location on both occasions.**

- On the occasion described in this section, you set the location of the wallet either by using the Net8 Assistant or by modifying the file sqlnet.ora.
- Later, you use the Oracle Wallet Manager as described in "Step 5: Create a new wallet" on page 9-32.

---

| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 9–9 on page 9-23.<br><br>1. At the top of the SSL tabbed page, be sure that the Configure SSL for Server radio button is selected.<br><br>2. In the Wallet Directory box, type the directory for the Oracle wallet. To find it by searching your file system, click the Browse button<br><br>**Note:** You must enter this same directory later when you come to "Step 5: Create a new wallet" on page 9-32. | Set the following parameter:<br><br>`oss.source.my_wallet =`<br>`(SOURCE=`<br>`(METHOD=File)`<br>`(METHOD_DATA=`<br>`(DIRECTORY=your wallet location)`<br>`)`<br>`)` |

### Set the SSL cipher suites (optional)

The SSL_CIPHER_SUITES parameter sets the cipher suites SSL uses.

When you install the Oracle Advanced Security option, several SSL cipher suites are set for you by default. Setting one or more cipher suites yourself overrides the other default cipher suites set during installation. For example, if you use Net8 Assistant to add the cipher suite SSL_RSA_WITH_RC4_128_SHA, all other cipher suites in the default setting are removed.

You can prioritize the cipher suites. When the server negotiates with clients over 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 effect. For example, triple-DES encryption is stronger than DES. See "SSL Can Provide Triple-DES" on page 2-4.
- The impact on performance. For example, Triple-DES encryption is slower than DES.
- Administrative requirements. The cipher suites selected for a server must be compatible with those required by the client. For example, in the case of an Oracle Call Interface (OCI) user, the server requires the client to authenticate itself. In this case, you *cannot* use a cipher suite employing Diffie-Hellman anonymous authentication which disallows the exchange of certificates. By contrast, in the case of an Enterprise Java Beans (EJB) user, the server does not require the client to authenticate itself. In this case, you *can* use Diffie-Hellman anonymous authentication.

> **Note:** In version 8.1.5 of the Oracle Advanced Security option, if you set a cipher suite employing Diffie-Hellman anonymous authentication on the server, you must also set the same cipher suite on the client. Otherwise, the connection will fail.
>
> If you have decided to use a cipher suite employing Diffie-Hellman anonymous, you must set the SSL_CLIENT_AUTHENTICATION parameter to FALSE. See "Set SSL client authentication (optional)" on page 9-28.

Normally, you would prioritize cipher suites starting with the strongest and moving to the weakest.

The following two tables list the available SSL cipher suites supported in both the domestic and export versions of the Oracle Advanced Security option. These tables also list the authentication, encryption, and data integrity types each cipher suite uses.

***Table 9–3 SSL Cipher Suites in Domestic Version of Oracle Advanced Security***

| Cipher Suite | Authentication | Encryption | Data Integrity |
|---|---|---|---|
| SSL_RSA_WITH_3DES_EDE_CBC_SHA | RSA | 3DES EDE CBC | SHA |
| SSL_RSA_WITH_RC4_128_SHA | RSA | RC4 128 | SHA |
| SSL_RSA_WITH_RC4_128_MD5 | RSA | RC4 128 | MD5 |
| SSL_RSA_WITH_DES_CBC_SHA | RSA | DES CBC | SHA |
| SSL_DH_anon_WITH_3DES_EDE_CBC_SHA | DH anon | 3DES EDE CBC | SHA |
| SSL_DH_anon_WITH_RC4_128_MD5 | DH anon | RC4 128 | MD5 |
| SSL_DH_anon_WITH_DES_CBC_SHA | DH anon | DES CBC | SHA |
| SSL_RSA_EXPORT_WITH_RC4_40_MD5 | RSA | RC4 40 | MD5 |
| SSL_RSA_EXPORT_WITH_DES40_CBC_SHA | RSA | DES40 CBC | SHA |
| SSL_DH_anon_EXPORT_WITH_RC4_40_MD5 | DH anon | RC4 40 | MD5 |
| SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA | DH anon | DES40 CBC | SHA |

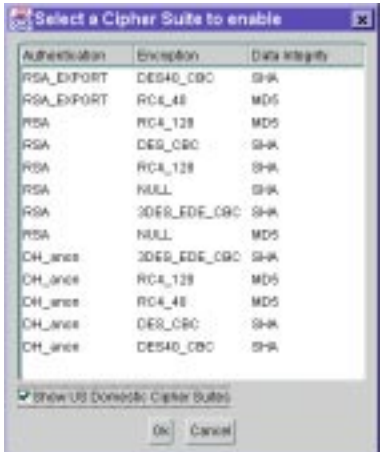
***Table 9–4 SSL Cipher Suites in Export Version of Oracle Advanced Security***

| Cipher Suite | Authentication | Encryption | Data Integrity |
|---|---|---|---|
| SSL_RSA_EXPORT_WITH_RC4_40_MD5 | RSA | RC4 40 | MD5 |
| SSL_RSA_EXPORT_WITH_DES40_CBC_SHA | RSA | DES40 CBC | SHA |
| SSL_DH_anon_EXPORT_WITH_RC4_40_MD5 | DH anon | RC4 40 | MD5 |
| SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA | DH anon | DES40 CBC | SHA |

| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 9–9 on page 9-23.<br>1. At the top of the SSL tabbed page, be sure that the Configure SSL for Server radio button is selected.<br>2. Click the Add button. A secondary dialog box (Figure 9–10) listing available cipher suites appears.<br>3. To add a suite to your list, select it in the secondary dialog box and click OK. The cipher suite appears in the Cipher Suite Configuration window of the main Net8 Assistant dialog box.<br>4. To prioritize the cipher suites, use the Promote and Demote buttons. The order you set determines the order in which the server is to negotiate with other entities on which cipher suite to use. | Set the following parameter in the server's sqlnet.ora file, listing the cipher suites in priority:<br>`SSL_CIPHER_SUITES=`<br>`(SSL cipher suite1 [,SSL cipher suite2])` |

***Figure 9–10 Net8 Assistant Secondary Dialog Box: Selecting a Cipher Suite***

+



### Set the required SSL version (optional)

Do this by setting the SSL_VERSION parameter. This parameter determines which version of SSL must be running on the machines with which the server communicates. You can require those machines to use SSL 3.0 or any existing or future versions.

The default setting for this parameter in sqlnet.ora is "0"; in Net8 Assistant it is "Any." Oracle recommends accepting the default value which allows clients with previous SSL versions to interoperate with servers using later SSL versions.

| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 9–9 on page 9-23.<br>1. At the top of the SSL tabbed page, be sure that the Configure SSL for Server radio button is selected.<br>2. In the Require SSL Version list box the default is Any. Accept this default or select the SSL version you want to enforce. | Set the following parameter:<br>`SSL_VERSION={ 0 \| 3.0 }` |

### Set SSL client authentication (optional)

The SSL_CLIENT_AUTHENTICATION parameter 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 may want to set this parameter to FALSE if you want the client to authenticate itself to the server by using any of the non-SSL authentication methods supported by Oracle Advanced Security option, for example, Kerberos, Identix, etc.

| Use the Net8 Assistant... | ... or modify SQLNET.ORA |
|---|---|
| Refer to Figure 9–9 on page 9-23.<br>1. At the top of the SSL tabbed page, be sure that the Configure SSL for Server radio button is selected.<br>2. By default, the Require Client Authentication check box is selected. Accept this default, or, if you do not want to require client authentication, deselect this check box. | Set the following parameter:<br>`SSL_CLIENT_AUTHENTICATION={ TRUE \| FALSE }` |

### Set SSL as an authentication service (optional)

The SQLNET.AUTHENTICATION_SERVICES parameter sets the SSL authentication service.

You *must* set this parameter only if *both* of the following two conditions apply:

- You want to use SSL authentication in conjunction with another authentication method supported by the Oracle Advanced Security option. For example, 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 or SecurID.

  and

- You are not using Net8 Assistant to make you configurations.

If both of the above conditions apply, add TCPS to this parameter in the sqlnet.ora file by using a text editor. For example:

```
SQLNET.AUTHENTICATION_SERVICES = (BEQ, TCPS,
                                  selected_method_1,
                                 selected_method_2)
```

If either or both of the above conditions do *not* apply, you do *not* need to set this parameter.

### Select "TCP/IP with SSL" as the listening endpoint

To activate SSL for a client connection, you must select the TCP/IP with SSL protocol as the listening endpoint in listener.ora. If you have IIOP clients connecting to the Java option in the database, be sure the port number is 2482.

**More Information:** See the *Net8 Administrator's Guide.*

## Step 4: Start the Oracle Wallet Manager

You use the Oracle Wallet Manager to do the following on both the client and the server:

- create a wallet
- request a certificate from a certificate authority and install it in the wallet
- optionally add new **trusted certificates**
- save a wallet

---

**Note:** If you are not requiring client authentication, you must nevertheless create wallets on both the server and the client, each with a list of trusted certificates.

---

The way you invoke the Oracle Wallet Manager depends on what kind of system you use.

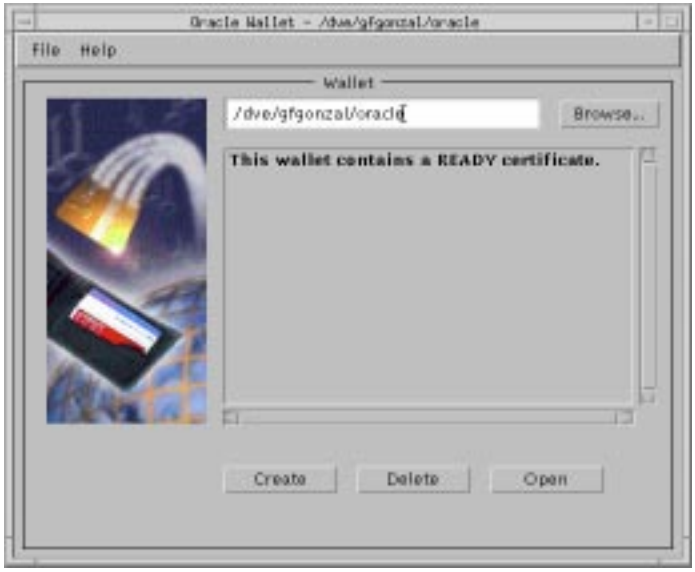
### Windows NT

Start the Oracle Wallet Manager on a Windows NT by clicking Start > Programs > Oracle Wallet Manager > Oracle Wallet Manager. The Oracle Wallet Manager's Oracle Wallet dialog box (Figure 9–11) appears.

### Solaris

Type **`wmtgui`** at the command line to invoke the Oracle Wallet Manager. The Oracle Wallet Manager's Oracle Wallet dialog box (Figure 9–11) appears.

***Figure 9–11 Oracle Wallet Manager***




This dialog box displays the default wallet location, the version of the certificate that is stored in the wallet, and the status of the wallet: EMPTY, REQUESTED, or READY. The following tables describe the fields and buttons in the dialog box.

| Field Name | Description |
|---|---|
| Wallet Location text box | Displays the default wallet file location. Click the Browse... button to locate a wallet at another location. |
| Help > Wallet Information | Displays the version of the Oracle Wallet Manager that you are using and the status of the certificate (if any) installed in the wallet. |

| Button Name | Function |
|---|---|
| Create | Creates a new wallet. |
| Delete | Deletes the wallet displayed in this dialog box. |
| Open | Opens the wallet displayed in this dialog box. |

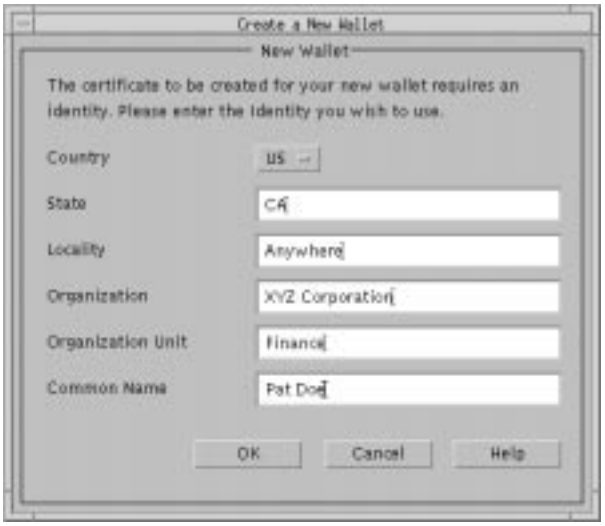
## Step 5: Create a new wallet

Follow the steps below to create a new wallet. The steps assume that you have started the Oracle Wallet Manager and are at the program's initial dialog box (Figure 9–11).

1. Click Create to create a new wallet.

   The New Wallet Identity dialog box (Figure 9–12) appears.

***Figure 9–12 New Wallet Identity***




2. Type the identity you want to use for your certificate, and click OK.

   If you are using single sign-on, make a note of the fields in this dialog box and value you enter in each one. You will need this information later when you create a global user. See "Step 10: Create a user identified globally through certificates on the Oracle server" on page 9-41.

   The New Wallet location dialog box (Figure 9–13) prompts you to choose a directory on your file system in which to store the new wallet.

**Note:** There are two occasions when you set the location of the Oracle wallet. **Be sure to enter the same location on both occasions.**

- On this occasion, you use the Oracle Wallet Manager.
- Earlier in the configuration process, you either used the Net8 Assistant or modified the sqlnet.ora file. Depending on which machine you are now configuring, see either "Step 2: Configure SSL on the client" on page 9-13 or "Step 3: Configure SSL on the server" on page 9-20.

***Figure 9–13 Specify Wallet File Location***


3. Type the file location for the new wallet. Click Browse to find a wallet in another directory. Click Cancel to return to the initial program dialog box, or click Next to continue. The New Wallet password dialog box appears.

***Figure 9–14 Type a Password for the New Wallet***


4. Type your password once in the Enter Password field, and re-type that same password in the Verify Password field.

> **Note:** An Error dialog box appears if you do not type exactly the same password in the Enter Password and Verify Password text boxes. Click OK to return to the New Wallet password dialog box, re-type your password in the Enter Password text box, then verify it by re-typing it in the Verify Password text box.

5. Click Next to continue.

   The New Wallet Random Data dialog box (Figure 9–15) appears.

***Figure 9–15 New Wallet Random Data***


**6.** Type a random string of at least 20 characters in length in the field. This character string will be used to seed the public / private key generation. Click OK.

A New Wallet dialog box (Figure 9–16) informs you that the new wallet will overwrite the wallet, certificate and trusted certificates that already exist at the default file location. This occurs when you already have an existing wallet in the default location.

***Figure 9–16 Create a New Wallet***


**7.** Click OK. The Replace Wallet dialog box (Figure 9–17) appears.

*Figure 9–17 Replace Wallet dialog box*


8. Click the Replace your old wallet radio button to replace the entire wallet including the certificate and trusted certificates, or click the Replace your old wallet, but reuse... radio button to replace the wallet but reuse the old wallet's trusted certificates. Click Cancel to return to the initial program dialog box, or click OK to continue.

   A dialog box (Figure 9–18) displays the location of your certificate request file. This is the file you send to your Certificate Authority.

*Figure 9–18 Certificate Request Location*


9. Click View to view your certificate request, or click Close.

   The Oracle Wallet dialog box (Figure 9–19) appears with a status value of REQUESTED and a certificate value of NONE.

*Figure 9–19 New Wallet with a Requested Certificate*


## Step 6: Install a certificate into the new wallet

Once you send the certificate request to the certificate authority, wait until you receive an e-mail reply containing your signed certificate. Depending upon the Certificate Authority, you may receive a certificate file such as certificate.txt. Proceed to install the certificate into the new wallet using either of the following two options.

- Option 1: Install a Certificate from a File
- Option 2: Install a Certificate from the Body of an E-mail

### Option 1: Install a Certificate from a File

1. Open the file from your certificate authority and locate the certificate text. The certificate content is usually delimited by the words Begin Certificate and End Certificate.
2. Click Install on the Oracle Wallet dialog box (Figure 9–19).

   The Install a new Certificate dialog box (Figure 9–20) appears.
3. Click Browse.

A directory dialog box appears. Use this dialog box to locate the certificate.txt file (it may also have some other name depending upon the Certificate Authority).

4. Select the file name, and click Open. The contents of the certificate file will appear in the dialog box area (Figure 9–20).

***Figure 9–20 Certificate Text Pasted into Window***


5. Click OK.

   You are returned to the Oracle Wallet dialog box. Its status changes to READY.

### Option 2: Install a Certificate from the Body of an E-mail

1. Open the e-mail you received from the certificate authority.
2. Select and copy the certificate text from the body of the e-mail.
3. Click Paste on the Install a new Certificate dialog box. The certificate text will be pasted into this dialog box (Figure 9–20).
4. Click OK.
5. You are returned to the Oracle Wallet dialog box. Its status changes to READY.

## Step 7: Add new trusted certificates

A trusted certificate is a third party identity that is qualified with a level of trust. Trusted certificates are contained within a wallet. The trusted certificate is used when an identity is being validated as the entity it claims to be. Trusted certificates are also referred to as trustpoints.

A default set of trusted certificates from VeriSign is installed in your default wallet when you install the Oracle Wallet Manager. You manage these trusted certificates by using the Oracle Wallet Manager which enables you to add a new trusted certificate, view existing trusted certificate information, and delete a trusted certificate.

If you are using a certificate that is issued by a CA not yet on your list of trusted certificates, you must add that CA to the list. If a CA's certificate is signed by a root CA, you must add to the list the entire **certificate chain**, one certificate at a time.

> **More Information:** For instructions on adding, viewing and managing trusted certificates, see "Managing Trusted Certificates" on page 9-46.

Once the wallet is in the file system, applications can start using SSL, provided each application has been configured to locate the wallet.

## Step 8: Save changes to your wallet

Click File > Save in the Oracle Wallet dialog box to save changes you make to the wallet.

## Step 9: For single sign-on functionality, create an auto-login wallet

If you want to use SSL's single sign-on functionality—as opposed to having users enter passwords each time they open their wallets—you must create an auto-login wallet from the wallet you created earlier in "Step 5: Create a new wallet" on page 9-32. You do this by using the command line version of the Oracle Wallet Manager.

**To create an auto-login wallet:**

1. Be sure that your sqlnet.ora file has the following lines:

```
oss.source.my_wallet =
(SOURCE=
  (METHOD=File)
  (METHOD_DATA=
    (DIRECTORY=your wallet location)
  )
)
```

2. Set the following environment variable:

   `setenv TNS_ADMIN` *`your_sqlnet.ora_file_directory`*

3. Launch the command line version of the Oracle Wallet Manager by typing the following command at the command prompt:

   ```
   owmcmd -f
   ```

   For example:

   ```
   /vobs/oracle/network/bin/owmcmd -f
   ```

   The command line version of the Oracle Wallet Manager prompts you for the user's password.

   Enter the password you entered when you created in Step 5: Create a new wallet on page 9-32

   The Oracle Wallet Manager creates the auto-login wallet, names it cwallet.sso, and places it at the Wallet Resource Locator you specified. If the initial wallet is stored on a directory server rather than on the local machine, the Oracle Wallet Manager downloads it from the directory server, uses it to create an auto-login wallet, and places the latter at the Wallet Resource Locator you specified.

## Step 10: Create a user identified globally through certificates on the Oracle server

If you are using an enterprise directory service, you create global users in each local database by using the Security Manager tool of the Oracle Enterprise Manager, or by typing the following commands:

```
CONNECT system/manager@database_name;
CREATE USER username IDENTIFIED GLOBALLY AS 'external_name'
```

The external_name must match the full distinguished name of the user.

---

**Note:** If you are using a directory server, be sure that the distinguished name in the directory matches that in the Oracle wallet.

---

**About Distinguished Names** A distinguished name consists of up to six fields of information which uniquely identify a user. The fields are:

- Common Name (CN)
- Location (L)
- State (ST)
- Organizational Unit (OU)
- Organization (O)
- Country (C)

The format of a distinguished name begins at the left with the lowest level of granularity:

```
CN=user, L=location, ST=state, OU=unit, O=organization, C=country
```

For example, suppose you have a user with the following attributes:

- Common Name: Tom Jones
- Location: HQ
- State: California
- Organizational Unit: Information Technologies
- Organization: Acme Corporation
- Country: US

The full distinguished name of this user would be:

```
CN=Tom Jones, L=HQ, ST=CA, OU=Information Technologies, O=Acme Corporation,
C=US
```

Thus, the following statement creates a new account for Tom Jones:

```
CREATE USER tjones IDENTIFIED GLOBALLY AS "CN=Tom Jones, L=HQ, ST=CA,
OU=Information Technologies, O=Acme Corporation, C=US"
```

**To obtain the full distinguished name of an Oracle wallet owner:** Refer to the values you entered in the Create a New Wallet dialog box in "Step 5: Create a new wallet" on page 9-32. Start with the field at the bottom of that dialog box, namely, Common Name, and record the value you entered in that field. Then move successively up to the next field, namely, Organizational Unit, and record the value you entered in that field. Then move up to the next field, and so on, until you have recorded the value for each field. Be sure to use the correct format for the distinguished name as described above.

**More Information:** *Oracle8i Administrator's Guide*

# Ongoing Administrative Tasks

Once you have initially configured the SSL feature of the Oracle Advanced Security option, there are various tasks you may need to perform from time to time. This section discusses these tasks in the following categories:

- Managing Wallets
- Managing Trusted Certificates

## Managing Wallets

Use the Oracle Wallet Manager to open, view, or modify an existing wallet or to create a new wallet.

This section discusses the following tasks:

- Opening an Existing Wallet
- Viewing Wallet Contents
- Copying a Wallet to Remote Nodes

> **More Information:** For information on starting the Oracle Wallet Manager, see "Step 4: Start the Oracle Wallet Manager" on page 9-30.
>
> For information on creating a new wallet, see "Step 5: Create a new wallet" on page 9-32.
>
> For information on installing a certificate into a new wallet, see "Step 6: Install a certificate into the new wallet" on page 9-37.

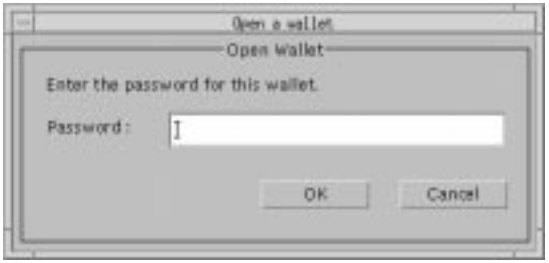
### Opening an Existing Wallet

The Oracle Wallet Manager enables wallet owners to open their default wallets. The default wallet is displayed in the Oracle Wallet Manager Start-up dialog box. Wallet owners must provide a valid **Wallet Resource Locator** (WRL) and the correct password to open the wallet.

1. Click Open on the Oracle Wallet Manager start-up dialog box.

   The Open Wallet Password dialog box (Figure 9–21) appears.

***Figure 9–21 Open the Default Wallet***




2. Type your password. Click Cancel to return to the Oracle Wallet Manager start-up dialog box, or click OK to continue.

   The Oracle Wallet dialog box (Figure 9–22) appears.

---

**Note:** An error dialog box titled "Failed to Open wallet!" will appear if you type an incorrect password. Click OK to return to the Oracle Wallet Manager Start-up dialog box. Check your password and try again.

---

***Figure 9–22 Default Wallet***


## Viewing Wallet Contents

Use the Oracle Wallet dialog box to access functions that allow you to view or modify the wallet's contents. This dialog box contains the following fields and buttons.

| Field Name | Description |
|---|---|
| Status | Displays the status of the wallet. The three values are EMPTY, REQUESTED, and READY. A status of EMPTY means that no certificate is requested or installed in the wallet. A status of REQUESTED indicates that the certificate request for your wallet has been generated. A status of READY means that you have a certificate. |
| Location | The directory in which the wallet is stored. |
| Certificate | The name of the identity for whom this certificate is installed in the wallet. |

| Button Name | Function |
|---|---|
| View | Views the installed certificate. |
| Install | Installs a new certificate into the wallet. |
| Trusted certificates | Views and manages the trusted certificates installed in your wallet. |
| Close | Returns to the Oracle Wallet Manager start-up dialog box. |

### Copying a Wallet to Remote Nodes

If you are using replicated servers, each node must have the same wallet.

## Managing Trusted Certificates

Use the Oracle Wallet Manager to manage the trusted certificates in your wallet. You can add a new trusted certificate, view existing trusted certificate information, and delete a trusted certificate. A default set of four trusted certificates is installed in your default wallet when you install the Oracle Wallet Manager.

This section discusses the following tasks:

- Adding a New Trusted Certificate
- Viewing Existing Trusted Certificate Information
- Deleting a Trusted Certificate
- Saving a Wallet to an Existing WRL (Wallet Resource Locator)

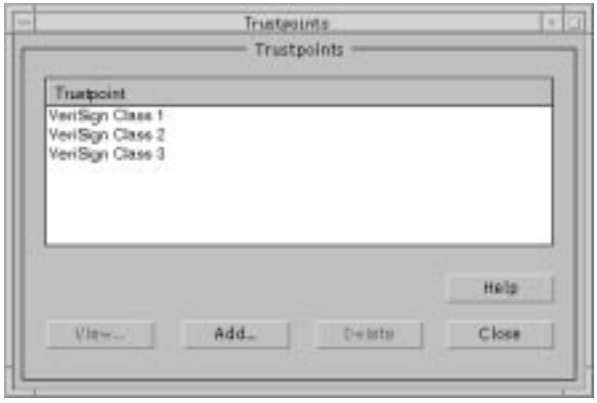
### Adding a New Trusted Certificate

Add a new trusted certificate to your wallet as follows.

1. Click Trustpoints on the Oracle Wallet dialog box (Figure 9–22 on page 9-45).

   The Trustpoints dialog box (Figure 9–23 on page 9-47) appears.

***Figure 9–23 Trusted Certificates***




**2.** Click Add.

The Install a New Trustpoint dialog box (Figure 9–24) appears. This is the dialog box into which you paste the trusted certificate.

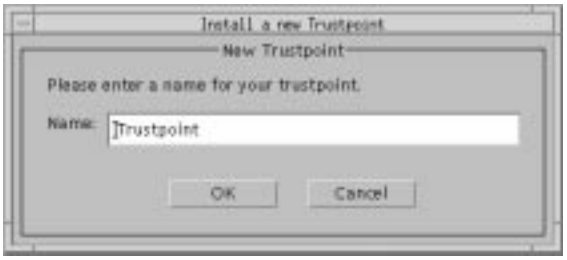
***Figure 9–24 Install a New Trusted Certificate***


**3.** Click Paste. The certificate text appears in the body of the dialog box.

4. Click Next.

   The Trustpoint Name dialog box (Figure 9–25) appears.

***Figure 9–25 Trustpoint Name Dialog Box***




5. Type a name for the trusted certificate alias. This name can be any set of alphanumeric characters, but it cannot contain any spaces.
6. Click Cancel to return to the previous dialog box, or click Next to continue.

   The trusted certificate you created is added to the list of trusted certificates in the Trustpoints dialog box (Figure 9–23 on page 9-47).
7. Click Close, and you are returned to the Oracle Wallet dialog box (Figure 9–22 on page 9-45).

### Viewing Existing Trusted Certificate Information

You can view detailed trusted certificate information from the Trustpoints dialog box as follows.

1. Click the name of the trusted certificate for which you want to view detailed information.
2. Click View.

   The Trustpoint Certificate dialog box (Figure 9–26 on page 9-49) appears.

***Figure 9–26 Trustpoint Certificate***


3. Review the trusted certificate information that was installed into your wallet. This information includes the certificate identity and the certificate issuer.
4. Click Extensions to display the X.509 v3 certificate extension information for your wallet trusted certificate, or click Close to return to the Trustpoints dialog box.

## Deleting a Trusted Certificate

The Oracle Wallet Manager offers you the option of deleting selected trusted certificates in the event that they become compromised. Delete a trusted certificate from the Trustpoints dialog box (Figure 9–23 on page 9-47) as follows.

1. Click the name of the trusted certificate listed in the Trustpoint column to select that trusted certificate.
2. Click Delete.

   A dialog box prompts you with, “Do you really want to delete this trusted certificate?”

3. Click Yes to delete the trusted certificate.

   You are returned to the Trustpoints dialog box, and the deleted trusted certificate is no longer displayed in the trusted certificate list.

   If you click No, you are returned to the Trustpoints dialog box, and the trusted certificate remains displayed in the trusted certificate list.

4. Click Close to return to the Oracle Wallet dialog box.

### Saving a Wallet to an Existing WRL (Wallet Resource Locator)

Click File > Save in the Oracle Wallet dialog box to save changes you make to the wallet.

## Logging in to the Database

If you are using SSL authentication, launch SQL*Plus and, at the prompt, type the following:

```
CONNECT/@database_alias
```

If you are not using SSL authentication, launch SQL*Plus and, at the prompt, type the following:

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

# 10

# Choosing and Combining Authentication Methods

This chapter describes how to use conventional user name / password authentication even if you have configured another authentication method. It also discusses how to configure your network to use one or more authentication methods in your network using the Oracle Advanced Security option and how to set up more than one authentication method on a client or on a server.

This chapter covers the following topics:

# Connecting with User Name and Password

To connect to an Oracle server using a user name and password when an Oracle Advanced Security option authentication method has been configured, disable the latter.

> **More Information:** See "Disabling Oracle Advanced Security Authentication" on page 10-3.

## Using Net8 Assistant

This graphical interface tool makes it easy to set parameters in the sqlnet.ora file and other Oracle8*i* configuration files.

**To launch Net8 Assistant:**

- On UNIX, run the script netasst at $ORACLE_HOME/bin.
- On Windows platforms, click the Start button > Programs > Oracle for Windows NT—*HOME_NAME* > Net8 Assistant.

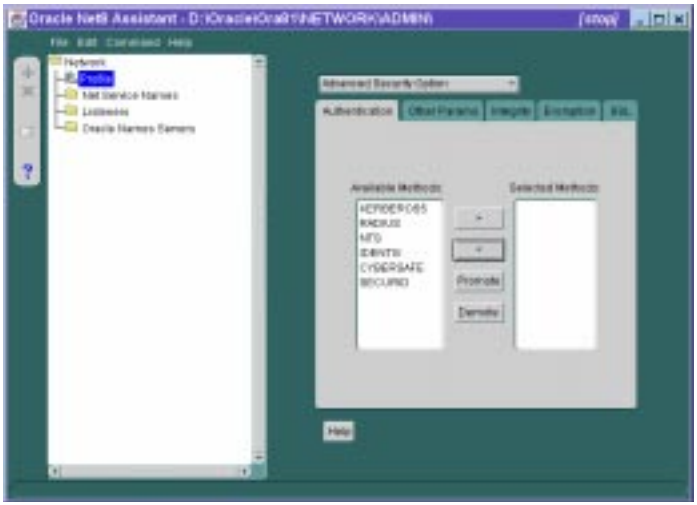
**To begin configuring the Oracle Advanced Security option using Net8 Assistant:**
In the Net8 Assistant's left pane, click the Profile folder. Then go to the drop down list box at the top of the right pane, and select Advanced Security Option. The tabbed pages for the Oracle Advanced Security option appear.

**To save changes with Net8 Assistant:**
Go to the menu bar and click File > Save Network Configuration.

## Disabling Oracle Advanced Security Authentication

Do this either by using the Net8 Assistant, or by using any text editor to modify the file sqlnet.ora.

***Figure 10–1 Using Net8 Assistant to Disable Authentication***



| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 10–1.<br>1. Click the Authentication tab.<br>2. Select a method listed in the Selected Methods area, and transfer it to the Available Methods area by clicking the left arrow [<].<br>3. Repeat until all methods are removed from the Selected Methods area. | Set the following parameter as follows:<br>`SQLNET.AUTHENTICATION_SERVICES = (`*`NONE`*`)` |

A user can now connect to a database using the following user name / password format:

```
% sqlplus username/password@net_service_name
```

For example:

```
% sqlplus scott/tiger@emp
```

## Configuring Oracle For Multiple Authentication Methods

Many networks use more than one authentication method on a single security server. For this reason, the Oracle Advanced Security option allows you to configure your network so that Oracle clients can use a specific authentication method and Oracle servers can accept any method specified.

This section describes how to set up Oracle servers and clients to use multiple authentication methods.

Set up multiple authentication methods on both client and server machines either by using the Net8 Assistant, or by using any text editor to modify the sqlnet.ora file.

The following instructions apply to both clients and servers.

***Figure 10–2 Using Net8 Assistant to Configure Multiple Methods***

| Use the Net8 Assistant... | ...or modify SQLNET.ORA |
|---|---|
| Refer to Figure 10–2.<br>1. Select Authentication tab.<br>2. Select a method listed in the Available Methods list.<br>3. Click the right arrow button [>] to transfer the selected method to the Selected Methods list.<br>4. Repeat until you have added all your required methods to the Selected Methods list.<br>5. Arrange the authentication methods in the order of desired use by selecting a method and clicking either [Promote] or [Demote].<br>Authentication will occur starting with the first method listed at the top of the Selected Methods list. | Set the following parameter:<br>`SQLNET.AUTHENTICATION_SERVICES= (`*`RADIUS`*` \| `*`CYBERSAFE`*` \| `*`KERBEROS`*` \| `*`SECURID`*` \| `*`IDENTIX`*`)`<br>Enter the name of each authentication method until you have added all your required methods.<br>Example:<br>`SQLNET.AUTHENTICATION_SERVICES =(SECURID,CYBERSAFE)` |

# Part II

# Oracle Advanced Security and Oracle DCE Integration

**Note:** Check your platform-specific installation documentation to be sure that release 8.1.5 of the Oracle Advanced Security option supports Oracle DCE integration for your platform.

The following chapters describe Oracle Distributed Computing Environment (DCE) Integration.

# 11

# Overview of Oracle DCE Integration

This chapter briefly describes the Distributed Computing Environment (DCE) and the Oracle DCE Integration product.

> **More Information:** See the list of related books and papers in "Related Publications" on page xx in the Preface of this guide.

This chapter covers the following topics:

- System Requirements
- Backward Compatibility
- Overview of Distributed Computing Environment (DCE)
- Overview of Oracle DCE Integration

# System Requirements

Oracle DCE Integration requires Net8 and Oracle8*i*. It enables Oracle applications and tools to access Oracle8*i* servers in a DCE environment.

---

**Note:** Oracle DCE Integration is based on the Open Software Foundation (OSF) DCE V1.0 and V1.1, and will be compatible with OSF's future DCE releases.

OSF has merged with another standards group, X/OPEN, to form The Open Group. This group will continue to support DCE.

---

# 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, the 2.1.6 clients will not be able to take advantage of external roles.

A client running DCE Integration 2.3.2 or later will *not* be able to 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 2.3.2 or later server in order to connect to a database.

# Overview of Distributed Computing Environment (DCE)

The Distributed Computing Environment (DCE) from the Open Software Foundation (OSF) is a set of integrated network services that work 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.

**More Information:** See "Related Publications" on page xx in the Preface of this guide.

# Overview of Oracle DCE Integration

Oracle DCE Integration enables use of Oracle tools and applications to access Oracle8*i* servers in a DCE environment.

## Components of Oracle DCE Integration

Oracle's DCE Integration product comprises DCE Communication / Security and DCE CDS Native Naming.

### DCE Communication/Security

DCE Communication / Security includes:

- **Authenticated RPC**—Oracle DCE Integration provides authenticated RPC (Remote Procedure Call) as the transport mechanism which 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 referred to as *external authentication* to the database. It is also known as *single sign-on*. 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. For example, users have a range of choices from no protection to full encryption for each connection, with a guarantee that no data has been modified in transit.

> **Note:** For parts of your network that do not use DCE, you may want to use the other security and authentication services included with the Oracle Advanced Security option. These services (formerly included in Secure Network Services) work with SQL*Net release 2.1 and above or with Net8. They provide message integrity and data encryption services in non-DCE environments, allowing administrators to ensure that all network traffic is protected against unauthorized viewing or modification, regardless of the start or end point.

**More Information:** See Part I, "Oracle Advanced Security Features" of this Guide.

### DCE CDS Native Naming

The DCE CDS Native Naming component includes naming and location transparency

DCE Integration registers Oracle8*i* connect descriptors in the DCE Cell Directory Service (CDS), allowing them to 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 Cell Directory Service 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 Cell Directory Service (CDS), Oracle clients can make location-independent connections to Oracle8*i* 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 may be used:

- DCE Global Directory Service (GDS)
- Internet Domain Naming Service (DNS)

**More Information:** For more information about DCE CDS Native Naming, see the following:

- To configure DCE to use CDS naming, see Chapter 12, "Configuring DCE for Oracle DCE Integration".
- To configure Oracle clients and servers to use CDS, see Chapter 13, "Configuring Oracle for Oracle DCE Integration".
- To read about how Oracle Native Naming works with other Oracle name services, refer to the *Net8 Administrator's Guide*.

## Flexible DCE Deployment

The Oracle Advanced Security option 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) described in this part of the Guide.
- You can choose to use only the DCE directory services by using the DCE CDS Native Naming Adapter, along with any conventional protocol adapter, such as TCP/IP. Configuration of the CDS Native Naming adapter is described in Chapter 12, "Configuring DCE for Oracle DCE Integration" and Chapter 13, "Configuring Oracle for Oracle DCE Integration" in this Guide.

  **More Information:** For an overview of how Native Naming works with other Oracle name services, see the *Net8 Administrator's Guide*.

- You can choose to use only DCE authentication services by using the DCE GSSAPI authentication method described in Chapter 8, "Configuring DCE GSSAPI Authentication" of this Guide. This requires OSF DCE 1.1.

## Limitations in This Release

- Only one listener address that uses the DCE protocol is allowed per node.
- Database links must specify a user name and password to connect.
- This release of the DCE Integration adapter does not support the Oracle MultiProtocol Interchange.
- This release does not work with the Oracle Multi-Threaded Server (MTS).

# 12

# Configuring DCE for Oracle DCE Integration

This chapter describes what you need to do to configure DCE to use Oracle DCE Integration after Oracle DCE Integration has been successfully installed.

**More Information:** See the list of books and papers in the "Related Publications" section on page xx in the Preface of this guide.

# Configuring DCE to Use DCE Integration

Following is a list of steps with examples you need to follow to configure DCE to use DCE Integration. The steps assume that a DCE cell has been configured and the machines being used are part of that cell.

As the DCE cell administrator, you need to do the following:

Step 1: Create New Principals and Accounts

Step 2: Install the Key of the Server into a Keytab File

Step 3: Configure DCE CDS for Use by Oracle DCE Integration

## Step 1: Create New Principals and Accounts

First, you need to add server principals using a procedure like the one below:

```
% 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, you just created a DCE principal called "oracle". The principal has a corresponding account with password "*password*". The account does not belong to any DCE group or DCE profile.

You only need to do this once after DCE Integration has been installed. Also, you only need to do this procedure for the Oracle database server, not for the client.

## Step 2: Install the Key of the Server into a Keytab File

In this step by step procedure, you install the key of the server into a keytab file: dcepa.key. This keytab file contains the password of the principal under which the Net8 listener starts. The Net8 listener reads this file to authenticate itself to DCE. You only need to do this once after DCE Integration has been installed. Also, you only need to do this procedure for the Oracle database server, not for the client

---

**Note:** Remember to substitute the correct full pathname for the $ORACLE_HOME variable. If the specified directories do not already exist, you will need to create it before running the command. Type the following to create the directories.

```
mkdir $ORACLE_HOME/dcepa
mkdir $ORACLE_HOME/dcepa/admin
```

---

Run the following command to generate the keytab file.

```
% 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
```

## Step 3: Configure DCE CDS for Use by Oracle DCE Integration

The /.:/subsys/oracle/names directory contains objects that map Net8 service names to connect descriptors, which are used by the CDS naming adapter.

The /.:/subsys/oracle/service_registry directory also contains objects that map the service name in DCE addresses to the network endpoint which is used by both DCE protocol adapter clients and servers.

### Create Oracle Directories in the CDS Namespace

You need to perform the steps in this section after installing the DCE Integration adapter for the first time in a cell.

```
% 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:** Create these directories on all CDS replicas.

### Give Servers Permission to Create Objects in the CDS Namespace

Perform the following steps to add 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
```

### Load Oracle Service Names Into CDS

**More Information:** For instructions on how to configure clients, see "Configuring Clients to Use DCE CDS Naming" on page 13-13.

For information on how to load Oracle service names into CDS, see "Create a TNSNAMES.ORA For Loading Oracle Connect Descriptors into CDS" on page 13-15.

# 13

# Configuring Oracle for Oracle DCE Integration

This chapter discusses how to configure Oracle and Net8 to use Oracle DCE Integration after it has been successfully installed. The following sections describe the parameters you need to configure for servers and clients.

# DCE Address Parameters

DCE addresses in the listener.ora and tnsnames.ora configuration files are defined by DCE parameters. These parameters consist of both mandatory and optional fields, which are described below:

```
ADDRESS=(PROTOCOL=DCE)
        (SERVER_PRINCIPAL=server_name)
        (CELL_NAME=cell_name)
        (SERVICE=dce_service_name))
```

where the components are:

| | |
|---|---|
| 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. |
| 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 below) 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 /.../cell_name/subsys/oracle/service_registry. This service name can fully specify the path in CDS. |

You can specify a service as:

```
SERVICE=/.../cell_name/subsys/oracle/service_registry/dce_service_name
```

or it can be specified as

```
SERVICE=dce_service_name
```

provided that `CELL_NAME=cell_name` is also specified.

A third option is to specify SERVICE=dce_service_name, in which case the cell name defaults to the local cell. However, this third way of specifying service names only works well if you are working within a single cell.

---

**Note:** The dce_service_name in the service field may or may not be the same as the service name used by Net8. The service name used by Net8 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.

**Note:** In this DCE Integration release, the configuration files listener.ora, sqlnet.ora, tnsnames.ora, and protocol.ora are located in the $ORACLE_HOME/network/admin directory. The init<*sid*>.ora file is located in the $ORACLE_HOME/dbs directory.

---

## Configuring the Server

To configure a server for DCE Integration, you need to configure the following Net8 files with DCE address and parameter information as described in "DCE Address Parameters" on page 13-2 and in the following sections.

---

**Note:** Use the Net8 Assistant to create the necessary configuration files. For explanations of the configuration files, refer to the *Net8 Administrator's Guide*

---

Note the following prerequisites:

- Listener configuration file (listener.ora) must be configured with DCE address information for all servers.
- Profile (sqlnet.ora) and protocol.ora need to be configured for servers in distributed systems that will need to make database link connections to other servers.

## LISTENER.ORA Parameters

For a database server to receive connections from Net8 clients in a DCE environment, there must be a Net8 listener active on the server platform. A listener listens for connections on a network address that is defined in the listener configuration file (listener.ora).

The SERVER_PRINCIPAL parameter designates what DCE principal the listener should be running under. In the sample below, the listener is running under principal "oracle".

## Sample DCE Address in LISTENER.ORA

Below 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/oracle7))
```

# Creating and Naming Externally-Authenticated Accounts

To use DCE authentication for logging onto the Oracle database, you need to create database accounts that are "authenticated externally".

> **More Information:** For more information on external authentication, see *Oracle8i Distributed Database Systems*

To enable secure external authentication, do the following:

1. Verify that these lines are in the init<*sid*>.ora file:

   ```
   REMOTE_OS_AUTHENT=FALSE
   OS_AUTHENT_PREFIX=""
   ```

2. Verify that the init<*sid*>.ora file does not have a multi-threaded server (MTS) entry for DCE. For example, an entry such as the following is *not* allowed:

   ```
   mts_dispatchers="dce, 3"
   ```

3. Make sure that you are logged in 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 will allow Oracle access across cell boundaries. The way you define users depends on whether they will be connecting within a single cell, or across cell boundaries.

   **Note:** The privileges shown in the remainder of this section are the minimum privileges necessary. The actual set of privileges needed depends on the instance and / or application.

   If users will be 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;
```

   **Note:** The entire CELL_NAME/SERVER_PRINCIPAL string must be 15 characters or less.

   For example:

```
SQL> CREATE USER "CELL1/ORACLE" IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO "CELL1/ORACLE";
```

   If connecting to the database across multiple cells, specify both the cell_name and the server_principal.

```
SQL> CREATE USER "CELL_NAME/SERVER_PRINCIPAL" IDENTIFIED EXTERNALLY;
SQL> GRANT CREATE SESSION TO "CELL_NAME/SERVER_PRINCIPAL";
```

**Attention:** You must enclose the externally-identified account name in double quotes, 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";
```

**Note:** When using the above format, set the following parameter in protocol.ora to FALSE:

```
dce.local_cell_usernames=false
```

**Note:** References to an Oracle account created in this manner must include the schema / account in the correct format. For example, 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 be:

```
SQL> SELECT * FROM oracle.emp
```

If a user wants to access tables in an another account created for connections across cells, the command might be:

```
SQL> SELECT * FROM "CELL1/ORACLE".emp
```

# Setting up DCE Integration External Roles

This section explains the steps you follow to set up external roles for DCE integration, and how you connect to an Oracle database as SYSOPER or SYSDBA with DCE credentials.

**To set up external roles for DCE Integration:**

1. Set the following parameter in the init<*sid*>.ora file.

   ```
   OS_ROLES=TRUE
   ```

   Then restart the database.

2. Make sure that the DCE groups that map to Oracle roles adhere to the following syntax:

```
ORA_<global_name>_<role>[_[a][d]]
```

where the components are:

| | |
|---|---|
| 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 | Optional character indicating that the user has admin privileges for this role. |
| D | Optional character indicating the role is to be enabled by default at connect time |

---

**Note:** For more details on external roles see the *Oracle8i Administrator's Guide*

---

**3.** DCE authenticate to a user who is a member of a DCE group by performing a `dce_login` and a `klist` command. (Below is some sample output from the `dce_login` and `klist` commands.)

---

**Note:** The DCE group must adhere to the syntax described in step 2.

```
% 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
.
.
.
```

---

4. Connect to the database as usual.

   Following is some sample output showing a listing of 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
```

## Connecting to Oracle Database as SYSDBA or SYSOPER using DCE

To 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 presented in "Setting up DCE Integration External Roles" on page 13-6. 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 explained in "Creating and Naming Externally-Authenticated Accounts" on page 13-4.

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: 1998-12-04-10:28:22
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 1998-12-04-00:28:22 to 1998-12-04-10:28:22
Server: dce-rgy@dce.dlsun685.us.oracle.com
        valid 1998-12-04-00:28:22 to 1998-12-04-10:28:22
Server: dce-ptgt@dce.dlsun685.us.oracle.com
        valid 1998-12-04-00:28:26 to 1998-12-04-02:28:26
Client: dce-ptgt@dce.dlsun685.us.oracle.com     Server:
krbtgt/dce.dlsun685.us.o
racle.com@dce.dlsun685.us.oracle.com
        valid 1998-12-04-00:28:26 to 1998-12-04-02:28:26
Client: dce-ptgt@dce.dlsun685.us.oracle.com     Server:
dce-rgy@dce.dlsun685.us.
oracle.com
        valid 1998-12-04-00:28:27 to 1998-12-04-02:28:26
```

**Note:** list output shows the DCE group membership of Oracle.

5. Connect to the Oracle database as SYSBDA or SYSOPER. For example,

```
SQL> connect /@oradce as SYSDBA
```

# Configuring the Client

To configure a client for DCE Integration, you need to configure the following Net8 files with DCE address and parameter information, as described below:

- 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.

**More Information:** See "Configuring Clients to Use DCE CDS Naming" on page 13-13.

## Parameters in PROTOCOL.ORA

There are four DCE parameters located in protocol.ora. 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 are:

- DCE.AUTHENTICATION=*dce_secret*
- DCE.PROTECTION=*pkt_integ*
- DCE.TNS_ADDRESS_OID=1.3.22.1.5.1
- DCE.LOCAL_CELL_USERNAMES=TRUE

> **Note:** The default for DCE.LOCAL_CELL_USERNAMES is now TRUE. (It was set to FALSE in the DCE Integration 2.1.6 release.)

Configuration parameters are not case-sensitive: you can enter them in either upper-case or lower-case.

> **Note:** If the DCE.AUTHENTICATION entry is not specified, cell-wide default authentication is used.
>
> If the DCE.PROTECTION entry is not specified, cell-wide default protection is used.

DCE.AUTHENTICATION—This parameter is optional. It indicates the authentication value to be used for each DCE RPC. The client's DCE_AUTHENTICATION value must be the same as the server's DEC_AUTHENTICATION value. The choices are:

- *NONE:* No authentication.
- *DCE_SECRET*: DCE shared-secret key authentication (Kerberos).
- *DCE_SECRET*: is the default authentication level.
- *DEFAULT*: The cell default.

> **Note:** It is recommended that DCE_SECRET be used for this parameter.

DCE.PROTECTION—This is an optional field. It specifies the data integrity protection levels for data transmission. The client's DCE_PROTECTION level must be equal to or greater than the server's DCE_PROTECTION level. The choices are:

- *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—This optional parameter enables you to specify an alternative to the default `DCE.TNS_ADDRESS_OID` (shown below):

```
DCE.TNS_ADDRESS_OID=1.3.22.1.x.x
```

> **More Information:** For information on how to determine if you need to include this parameter, and how to specify it, see "Modify the CDS Attributes File and Restart the CDS" on page 13-14.

DCE.LOCAL_CELL_USERNAMES—This optional parameter defines the format used to specify the principal name (`username`) either with or without the cell name.

> **Note:** The choice you make for this parameter should be determined by whether users will be making connections across cells, and if so, whether you have naming conventions that assure that users in different cells do not have duplicate names.

The choices are:

*TRUE:* This is the default. Choose TRUE when using just the SERVER_PRINCIPAL format, without the CELL_NAME. An example of a user specified in this format would be:

```
oracle
```

This choice would be appropriate if users are making connections within a single cell, or if naming conventions in your network assure that users in different cells do not have duplicate names.

*FALSE:* Choose FALSE when using the CELLNAME/SERVER_PRINCIPAL format. An example of a user specified in this format would be:

```
CELL1/ORACLE
```

This choice would be appropriate if users are making connections across cells and there may be users in different cells with identical names.

# Configuring Clients to Use DCE CDS Naming

Clients will typically use CDS to resolve Oracle service names to addresses. Follow the instructions below to configure CDS.

## 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 will use CDS. Also, the CDS namespace must have been configured for use by DCE Integration.

> **More Information:** For instructions on how to install and configure the CDS Naming Adapter, see the DCE Integration installation instructions and "Step 3: Configure DCE CDS for Use by Oracle DCE Integration" on page 12-3.

For example, a service name such as "ORADCE" and its network address can be stored in DCE's CDS.

Typically, users can 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): For example:

```
sqlplus /@ORADCE
```

This example assumes that DCE externally-authenticated accounts are in use.

As an alternative name resolution service, you can use a local naming configuration file (tnsnames.ora) when CDS is inaccessible. To do this, you must locate names and addresses of all Oracle servers in the local naming configuration file (tnsnames.ora).

## Modify the CDS Attributes File and Restart the CDS

On all DCE machines where CDS naming will be 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 with the following format to the /opt/dcelocal/etc/cds_attributes file.

   ```
   1.3.22.1.5.1    TNS_Address    char
   ```

   If the default TNS_Address OID (Object Identifier) value (1.3.22.1.5.1) already exists in the cds_attributes file, then you need to specify a value for the OID that is not already in use.

   **Note:** The first four digits of the TNS_Address attribute value (1.3.22.1.x.y) are fixed under DCE-naming conventions.

   If you are unable to use the default value for the OID, you need to specify the OID in the protocol.ora file on the client.

   If you had to specify a value other than the default (1.3.22.1.5.1), then you need to add the following parameter to the protocol.ora file:

   ```
   DCE.TNS_ADDRESS_OID=1.3.22.1.x.y
   ```

   **Note:** Make sure that the OID value in the cds_attributes file matches the value specified in the DCE.TNS_ADDRESS_OID parameter in the protocol.ora file.

2. Restart the CDS on the machine. (The command to restart CDS may vary from platform to platform. For example, on IBM AIX, you may use `smit` to restart the CDS.) The steps on IBM AIX are as follows:

   1. Type: smit DCE
   2. Choose Restart DCE/CDS Daemons
   3. Select List
   4. Select all CDS daemons available

## Create a TNSNAMES.ORA 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) containing service names (or aliases) and addresses. A sample file is shown below. The local naming configuration file (tnsnames.ora) is used to map service names to addresses for use by Net8.

This section describes the parameters that you need to include in the tnsnames.ora file. tnsnames.ora contains a list of Oracle service names mapped to connect descriptors of destinations or endpoints in the network. The sample DCE address below 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. However, they are often the same.

The keyword value pair PROTOCOL=DCE is mandatory. It appears in the address section of a listener configuration file (listener.ora) and in the address section of a local naming configuration file (tnsnames.ora). It must be the same in both places.

The DCE parameter SERVER_PRINCIPAL is optional in a local naming configuration file (tnsnames.ora).

The DCE parameter SERVICE is mandatory. The value given for the DCE parameter (SERVICE= *dce_service_name*) must be the same in the listener configuration file (listener.ora) and local naming configuration file (tnsnames.ora).

The Oracle parameter SID is mandatory. It identifies the Oracle system ID; each SID value must be unique on a node. This parameter is strictly local and is not used in DCE CDS.

**More Information:** For information on the local naming configuration file (tnsnames.ora), see the *Net8 Administrator's Guide*

## Load Oracle Connect Descriptors into CDS

A separate utility called "tnnfg" is provided with Oracle DCE Integration to load connect descriptors into CDS.

To load the Oracle service names or aliases into CDS, perform the following steps:

```
% dce_login cell_admin
% tnnfg dceload full_pathname_to_TNSNAMES.ORA
% Enter Password:(password will not display)
```

---

**Note:** You must enter the full pathname for the tnsnames.ora file in the previous command.

Also, make sure that the sqlnet.ora file exists in the same directory as the tnsnames.ora file.

---

This procedure loads the service names in tnsnames.ora into DCE's CDS.

---

**Note:** If you configure a new service name and address in tnsnames.ora, tnnfg will add the new service name and address to CDS.

If you change the address for a particular service name, tnnfg will update the address for a particular service name.

---

## Delete or Rename TNSNAMES.ORA File

If you are using SQL*Net 2.2 or earlier, after having loaded the tnsnames.ora file into DCE's CDS, Oracle recommends that you rename it to another name—tnsnames.bak, for example, or delete it. Otherwise, tnsnames.ora may be searched instead of CDS to resolve the service name to an address.

If you are using SQL*Net 2.3 or Net8, you can keep tnsnames.ora available as a backup in case CDS becomes unavailable. To assure that CDS will routinely be searched instead of tnsnames.ora, configure the NAMES.DIRECTORY_PATH parameter in a profile (sqlnet.ora), as described in the next section: "Modify SQLNET.ORA Parameter File to Have Names Resolved in CDS" on page 13-17.

## Modify SQLNET.ORA Parameter File to Have Names Resolved in CDS

The parameters required in a profile (sqlnet.ora) depend upon the version of SQL*Net or Net8 you are using.

### SQL*Net Release 2.3 and Later and Net8

For a client or server to use the DCE CDS Naming, the administrator needs to do the following:

- make sure that the CDS Naming Adapter has been installed on that node
- add the following parameter to the sqlnet.ora file:

```
NAMES.DIRECTORY_PATH=(dce, tnsnames, onames)
```

The first name resolution service listed as a value for this parameter is used. If it is unavailable for some reason, the next name resolution service is used, and so forth.

## Connect to Oracle Servers in DCE

**More Information:** For information on how to connect to Oracle databases in a DCE environment, see Chapter 14, "Connecting to an Oracle Database in DCE".

# 14

# Connecting to an Oracle Database in DCE

This chapter explains how to connect to an Oracle database after having installed Oracle DCE Integration and having configured both DCE and Oracle to use Oracle DCE Integration.

This chapter covers the following topics:

# Starting the Network Listener

To start the Net8 listener, do the following:

1. To start the listener, enter the following commands:

```
% dce_login principal_name password
%  lsnrctl start listener_name
```

   For example, if the listener name is LSNR_DCE in listener.ora, enter the following to start the listener:

```
% dce_login oracle orapwd
% lsnrctl start LSNR_DCE
```

   To make sure the server registered its binding handler with rpcd, enter:

```
% rpccp show mapping
```

   In the computer response, look for the line that includes the dce_service_name that is part of the listener address.

2. To make sure the service has been created, search for the dce_service_name as follows:

```
% cdscp show object "/.:/subsys/oracle/service_registry/dce_service_
name"
```

   For example:

```
% cdscp show object "/.:/subsys/oracle/service_registry/dce_svc"
```

   This shows you the mapping in the CDS namespace that the listener has chosen for the endpoint. For example:

```
              SHOW
            OBJECT    /.../subsys/oracle/service_registry/dce_svc
               AT   1995-05-15-17:10:52
RPC_ClassVersion = 0100
       CDS_CTS = 1995-05-16-00:05:01.221106100/aa-00-04-00-3e-8c
       CDS_UTS = 1995-05-16-00:05:01.443343100/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 Server in the DCE Environment

To connect to an Oracle server in the DCE environment, do one of the following:

1. After externally-identified accounts have been set up, you can take advantage of DCE authentication to log into Oracle without providing any user name / 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 user name 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
   ```

   **More Information:** See *Oracle8i Distributed Database Systems.*

2. From a client, you can still connect with a user name / password:

   ```
   % sqlplus username/password@net_service_name
   ```

   where *net_service_name* is the Net8 service name.

   For example:

   ```
   % sqlplus scott/tiger@ORADCE
   ```

# 15

# DCE and Non-DCE Interoperability

This chapter describes how clients outside DCE can connect to Oracle servers in DCE and how a local naming configuration file (tnsnames.ora) can be used for name lookup when CDS is accessible.

This chapter covers the following topics:

# 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 (see the sample listener.ora file in the next section), non-DCE clients can use normal Oracle and Net8 procedures to connect to an Oracle server in DCE.

---

**Note:** In this case DCE security would not be available to clients. Also, service names would be located and resolved to network addresses in a tnsnames.ora file on the client, not using the CDS name server.

---

**More Information:** For a sample file, see "LISTENER.ORA" on page 15-2. Also see "TNSNAMES.ORA" on page 15-4.

Following are samples of listener.ora and tnsnames.ora files as they would need to be configured if a client from outside of DCE wanted to connect to Oracle database servers in a DCE environment.

# Sample Parameter Files

At least two Oracle parameter files are needed for successful client/server communications. Create and modify these files using you favorite text editor. The files are as follows:

- LISTENER.ORA
- TNSNAMES.ORA

## LISTENER.ORA

This 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 laid out on a separate line, so it is easy to see the file's structure. This is the recommended format. If you must edit a listener.ora file by hand, you do not have to put each element on a separate line. Be careful, though, to include all the appropriate parentheses and to indent if you must continue an element onto 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 may have multiple addresses too. 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 will be better 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/oracle7)
         )
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/oracle7/neatwork/trace
TRACE File_LISTENER=listener.trc
LOG_DIRECTORY_LISTENER=/usr/prod/oracle7/network/log
LOG_FILE_LISTENER=listener.log
```

## TNSNAMES.ORA

This file resides on both the client and the server nodes. It provides a list of the service names and addresses of all services on the network.

The following 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)
          )
          )
```

A user who wished to access the DB1 database would 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. Though the main purpose (in the context of Native Naming adapters) of tnsnames.ora 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 tnsnames.ora for name lookup and resolution, remove (or comment out) the "native name" parameters from sqlnet.ora on the client. To comment out the lines, add a # at the beginning of each line. For example:

```
#native_names.use_native=true
#native_names.directory_path=(dce)
```

## SQL*Net Release 2.3 and Net8

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.

# A

# Encryption and Checksumming Parameters

This appendix lists and describes encryption and checksumming parameters supported in the Oracle Advanced Security option. It also includes an example of a sqlnet.ora file generated after you perform the network configuration described in Chapter 2, "Configuring Encryption and Checksumming".

This appendix covers:

- Sample SQLNET.ORA File
- Encryption and Checksumming 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. This sample sqlnet.ora file includes examples of the Oracle Advanced Security option encryption and checksumming parameters.

```
# SQLNET.ORA Configuration File:/private/users/oracle7/sqlnet.ora
# Generated by Oracle Net8 Assistant

SQLNET_CRYPTO_CHECKSUM_TYPE_SERVER = MD5

OSS.SOURCE.MY_WALLET =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
     (DIRECTORY = /private/users/oracle7/ano814/8.1.4/network/admin/wallet)
    )
  )

SQLNET.AUTHENTICATION_SERVICES= (BEQ, SECURID)

SQLNET.CRYPTO_CHECKSUM_CLIENT = requested

SQLNET.ENCRYPTION_TYPES_SERVER= (RC4_40, DES40)

SQLNET.ENCRYPTION_TYPES_CLIENT= (RC4_40, DES40)

SSL_VERSION = Any

SQLNET_CRYPTO_CHECKSUM_TYPE_CLIENT = MD5

SQLNET.EXPIRE_TIME = 0

SQLNET.ENCRYPTION_SERVER = requested

SQLNET.ENCRYPTION_CLIENT = requested

SQLNET.CRYPTO_CHECKSUM_SERVER = requested

SQLNET.CRYPTO_SEED = qwertyuiopasdfghjkl;zxcvbnm
```

Note the following:

- If you do not specify any values for Server Encryption, Client Encryption, Server Checksum, or Client Checksum, the corresponding configuration parameters will not appear in the sqlnet.ora file. However, the Oracle Advanced Security option defaults the value to ACCEPTED.
- If no encryption or checksumming algorithm is specified on the Server Encryption, Client Encryption, Server Checksum, or Client Checksum pages, the server side of the connection uses the first algorithm in its own list of installed algorithms that also appears in the client's list of installed algorithms.
- Encryption and checksumming function independently of each other; encryption can be activated while checksumming is off, and vice versa.

## Encryption and Checksumming Parameters

There are nine parameters to enable encryption and checksumming. These parameters are described in the following sections.

- Server Encryption Level Setting
- Client Encryption Level Setting
- Server Encryption Selected List
- Client Encryption Selected List
- Server Checksum Level Setting
- Client Checksum Level Setting
- Server Checksum Selected List
- Client Checksum Selected List
- Client Profile Encryption

> **More Information:** See "Negotiating Encryption and Checksumming" on page 2-7.

### Server Encryption Level Setting

Purpose: This parameter specifies the desired behavior when a client (or a server acting as a client) is connecting to this server. The behavior of the server will depend in part on the SQLNET.ENCRYPTION_CLIENT setting at the other end.

Syntax: SQLNET.ENCRYPTION_SERVER = *valid_value*

Possible values: ACCEPTED, REJECTED, REQUESTED, REQUIRED

Default value: ACCEPTED

### Client Encryption Level Setting

Purpose: This parameter specifies the desired behavior when this client (or this server acting as a client) is connecting to a server. The behavior of the client will depend in part on the value set for SQLNET.ENCRYPTION_SERVER at the other end of the connection.

Syntax: SQLNET.ENCRYPTION_CLIENT = *valid_value*

Possible values: ACCEPTED, REJECTED, REQUESTED, REQUIRED

Default value: ACCEPTED

### Server Encryption Selected List

Purpose: This parameter specifies a list of encryption algorithms this server is allowed to use when acting as a server in the order of desired use. Type the most desired algorithm first. This list is used to negotiate a mutually acceptable algorithm with the other end of the connection. Each algorithm will be checked against the list of client algorithm types available until a match is found. If an algorithm that is not installed is specified on this side, the connection will terminate with error message ORA-12650.

Syntax: SQLNET.ENCRYPTION_TYPES_SERVER = (*valid_encryption_algorithm [,valid_encryption_algorithm]*)

Possible values: RC4_40—This is RSA RC4 (40-bit key size) for Domestic & International

RC4_56—This is RSA RC4 (56-bit key size) for Domestic only

RC4_128—This is RSA RC4 (128-bit key size) for Domestic only

DES—This is Standard DES (56-bit key size) for Domestic only

DES40—This is DES40 (40-bit key size) for Domestic & International

Default value: All installed algorithms will be used in a negotiation if no algorithms are defined in the sqlnet.ora file.

Usage Notes: **Domestic version**—If you are using the Domestic version, all five algorithms are installed: RC4_40, RC4_56, RC4_128, DES, and DES40. If no algorithms are specified, the installed algorithms will be used in that order to negotiate a mutually acceptable algorithm with the other end of the connection.

**Export version**— If you are using the Export version, the following algorithms are installed: RC4_40 and DES40. If no algorithms are specified, the installed algorithms will be used in that order to negotiate a mutually acceptable algorithm.

You can specify multiple encryption algorithms, that is, 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*)

### Client Encryption Selected List

Purpose: This parameter specifies a list of encryption algorithms this client (or this server acting as a client) is allowed to use when connecting to a server. This list is used to negotiate a mutually acceptable algorithm with the other end of the connection. The parameters can be listed in any order. If an algorithm that is not installed is specified on this side, the connection will terminate with error message ORA-12650.

| | |
|---|---|
| Syntax: | SQLNET.ENCRYPTION_TYPES_CLIENT = (*valid_encryption_algorithm [,valid_encryption_algorithm]*) |
| Possible values: | RC4_40—This is RSA RC4 (40-bit key size) for Domestic & International<br><br>RC4_56—This is RSA RC4 (56-bit key size) for Domestic only<br><br>RC4_128—This is RSA RC4 (128-bit key size) for Domestic only<br><br>DES—This is Standard DES (56-bit key size) for Domestic only<br><br>DES40—This is DES40 (40-bit key size) for Domestic & International |
| Default value: | All installed algorithms will be used if no algorithms are defined in the sqlnet.ora file. |
| Usage Notes: | **Domestic version**—If you are using the Domestic version, all five algorithms are installed: RC4_40, RC4_56, RC4_128, DES, and DES40. If no algorithms are defined in the sqlnet.ora file, the installed algorithms will be used in that order to negotiate a mutually acceptable algorithm with the other end of the connection.<br><br>**Export version**—If you are using the Export version, the RC4_40 and DES40 algorithms are installed. If no algorithms are defined in the sqlnet.ora file, the installed algorithms will be used in that order to negotiate a mutually acceptable algorithm.<br><br>You can specify multiple encryption algorithms, that is, either a single value or a list of algorithm names. For example, either of the following encryption parameters is acceptable:<br><br>SQLNET.ENCRYPTION_TYPES_CLIENT=(*DES,DES40,RC4_56,RC4_40*)<br><br>SQLNET.ENCRYPTION_TYPES_CLIENT=(*RC4_40*) |

### Server Checksum Level Setting

Purpose: This parameter specifies the desired checksum behavior when a client (or another server acting as a client) is connecting to this server. The resulting behavior will depend in part on the SQLNET.CRYPTO_CHECKSUM_CLIENT setting at the other end.

Syntax: SQLNET.CRYPTO_CHECKSUM_SERVER = valid_value

Possible values: ACCEPTED, REJECTED, REQUESTED, REQUIRED

Default value: ACCEPTED

### Client Checksum Level Setting

Purpose: This parameter specifies the desired checksum behavior when this client (or this server acting as a client) is connecting to a server. The resulting behavior will depend in part on the SQLNET.CRYPTO_CHECKSUM_SERVER setting at the other end of the connection.

Syntax: SQLNET.CRYPTO_CHECKSUM_CLIENT = *valid_value*

Possible values: ACCEPTED, REJECTED, REQUESTED, REQUIRED

Default value: ACCEPTED

### Server Checksum Selected List

Purpose: This parameter specifies a list of the checksumming algorithms this server is allowed to use, in order of desired use with the most desired algorithm first, when acting as a server to a client or another server. This list is used to negotiate a mutually acceptable algorithm with the remote end. Each algorithm will be checked against the list of client algorithm types available until a match is found. The first algorithm match will be the one that is used. If an algorithm is specified that is not installed on this side, the connection will terminate with error message ORA-12650.

Syntax: SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (*crypto_checksum_algorithm*)

Possible values: Currently, the only supported crypto-checksum algorithm choice is RSA Data Security's MD5 algorithm.

Default value: MD5 (currently the only valid value)

### Client Checksum Selected List

Purpose: This parameter specifies a list of checksumming algorithms this client (or this server acting as a client) is allowed to use when connecting to a server. This list is used to negotiate a mutually acceptable algorithm with the remote end. The order in which the algorithms are listed is not important. If an algorithm that is not installed on this side is specified, the connection will terminate with error message ORA-12650.

Syntax: SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (*crypto_checksum_algorithm*)

Possible values: Currently, the only supported crypto-checksum algorithm choice is RSA Data Security's MD5 algorithm.

Default value: MD5 (currently the only valid value)

### Client Profile Encryption

`SQLNET.CRYPTO_SEED = "10-70 random characters"`

The characters that form the value for this parameter are used when generating cryptographic keys. 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 above statement.

---

**Note:** Oracle recommends that you enter as many characters as possible (up to 70) to make the resulting key more random and therefore stronger.

---

This parameter must be present in the sqlnet.ora file whenever encryption or checksumming is turned on.

# B

# Authentication Parameters

This appendix shows some sample configuration files with the necessary profile (sqlnet.ora) and database initialization file (init.ora) authentication parameters when using the CyberSafe, Kerberos, SecurID, RADIUS, or SSL authentication. It includes the following sections:

- Parameters for Clients and Servers using CyberSafe Authentication
- Parameters for Clients and Servers using Kerberos Authentication
- Parameters for Clients and Servers using SecurID Authentication
- Parameters for Clients and Servers using RADIUS Authentication
- Parameters for Clients and Servers using SSL

# Parameters for Clients and Servers using CyberSafe Authentication

Following is a list of parameters to insert into your configuration files for clients and servers using CyberSafe.

## SQLNET.ORA Parameters

```
SQLNET.AUTHENTICATION_SERVICES=(cybersafe)
SQLNET.AUTHENTICATION_GSSAPI_SERVICE=oracle/dbserver.someco.com@SOMECO.COM
```

## INIT.ORA Parameters

```
REMOTE_OS_AUTHENT=FALSE
OS_AUTHENT_PREFIX=""
```

# Parameters for Clients and Servers using Kerberos Authentication

Following is a list of parameters to insert into your configuration files for clients and servers using Kerberos.

## SQLNET.ORA Parameters

```
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_REALMS=/krb5/krb.realms
SQLNET.KERBEROS5_KEYTAB=/krb5/v5srvtab
```

## INIT.ORA Parameters

```
REMOTE_OS_AUTHENT=FALSE
OS_AUTHENT_PREFIX=""
```

# Parameters for Clients and Servers using SecurID Authentication

Following is list of parameters to insert into your configuration files for clients and servers using SecurID.

## SQLNET.ORA Parameters

```
SQLNET.AUTHENTICATION_SERVICES=(securid)
```

## INIT.ORA Parameters

```
REMOTE_OS_AUTHENT=FALSE
OS_AUTHENT_PREFIX=""
```

# Parameters for Clients and Servers using RADIUS Authentication

The following table provides a list of parameters to insert into your configuration files for clients and servers using RADIUS.

## SQLNET.ORA Parameters

### SQLNET.RADIUS_AUTHENTICATION

Description To set 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.

Default `localhost`

### SQLNET.RADIUS_AUTHENTICATION_PORT

Description To set the listening port of the primary RADIUS server.

Default `1645`

### SQLNET.RADIUS_AUTHENTICATION_TIMEOUT

Description To set the time to wait for response.

Default 5

### SQLNET.RADIUS_AUTHENTICATION_RETRIES

Description To set the number of times to re-send.

Default 3

### SQLNET.RADIUS_SEND_ACCOUNTING

Description To set the turn accounting ON / OFF. If you enable accounting, packets will be sent to the active RADIUS server at listening port plus one. Default port is 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.

Default `OFF`

### SQLNET.RADIUS_ALTERNATE

Description To set the location of alternate RADIUS server to be used in case the primary server is unavailable. This feature is set to OFF by default. If you want to set up a second RADIUS server for fault tolerance, you need to specify the host name or the IP address of the host where the second RADIUS server is located.

Default `NONE`

### SQLNET.RADIUS_ALTERNATE_PORT

Description To set the listening port for the alternate RADIUS server.

Default `1645`

### SQLNET.RADIUS_ALTERNATE_TIMEOUT

Description To set the time to wait for response.

Default `5`

### SQLNET.RADIUS_ALTERNATE_RETRIES

Description To set the number of times to re-send messages.

Default `3`

### SQLNET.RADIUS_CHALLENGE_RESPONSE

Description To turn challenge/response support ON/OFF.

Default `OFF`

### SQLNET.RADIUS_CHALLENGE_KEYWORD

Description To set the keyword to request a challenge from the RADIUS server. User types no password on client.

Default `challenge`

### SQLNET.RADIUS_AUTHENTICATION_INTERFACE

Description To set the name of the Java class that contains the graphical user interface when RADIUS is in the challenge-response (asynchronous) mode.

Default `DefaultRadiusInterface`

### SQLNET.RADIUS_CLASSPATH

Description 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.

Default There is no default. You must add this parameter to the sqlnet.ora file.

## INIT.ORA 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 in sqlnet.ora
- Dynamic—the name of the parameter used in the security subsection of the Net8 address.

## Authentication

| | |
|---|---|
| Parameter Name (static): | SQLNET.AUTHENTICATION_SERVICES |
| Parameter Name (dynamic): | AUTHENTICATION |
| Parameter Type: | String LIST |
| Parameter Class: | Static |
| Allowable 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.<br>**Note:** the dynamic version supports only the setting of one type. |
| Existing / New Parameter | Existing |
| Syntax (static): | SQLNET.AUTHENTICATION_SERVICES = (TCPS,<br>selected_method_1,<br>selected_method_2) |
| Example (static): | SQLNET.AUTHENTICATION_SERVICES = (TCPS,<br>cybersafe,<br>securid |
| Syntax (dynamic): | AUTHENTICATION = *string* |
| Example (dynamic): | AUTHENTICATION = (TCPS) |

## Cipher Suites

| | |
|---|---|
| Parameter Name (static): | SSL_CIPHER_SUITES |
| Parameter Name (dynamic): | SSL_CIPHER_SUITES |
| Parameter Type: | String LIST |
| Parameter Class: | Static |
| Allowable Values: | Any known SSL cipher suite |
| Default Value: | No default |
| Description: | To controls what combination of encryption and data integrity is used by SSL. |
| Existing / New Parameter | New |
| Syntax (static): | SSL_CIPHER_SUITES=(*SSL cipher suite1*<br>[, *SSL cipher suite2*, ...<br>*SSL cipher suiteN*]) |
| Example (static): | SSL_CIPHER_SUITES=(SSL_DH_DSS_WITH_DES_CBC_SHA) |
| Syntax (dynamic): | SSL_CIPHER_SUITES=(*SSL cipher suite1*<br>[, *SSL cipher suite2*, ...<br>*SSL cipher suiteN*]) |
| Example (dynamic): | SSL_CIPHER_SUITES=(SSL_DH_DSS_WITH_DES_CBC_SHA) |

### Supported SSL 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_DH_anon_EXPORT_WITH_RC4_40_MD5
- SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA

## SSL Version

| | |
|---|---|
| Parameter Name (static): | SSL_VERSION |
| Parameter Name (dynamic): | SSL_VERSION |
| Parameter Type: | string |
| Parameter Class: | Static |
| Allowable 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 (static): | SSL_VERSION=*version* |
| Example (dynamic): | SSL_VERSION=3.0 |

## SSL Client Authentication

| | |
|---|---|
| Parameter Name (static): | SSL_CLIENT_AUTHENTICATION |
| Parameter Name (dynamic): | SSL_CLIENT_AUTHENTICATION |
| Parameter Type: | Boolean |
| Parameter Class: | Static |
| Allowable 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 |

## Wallet Location

For any application that needs to access a wallet for loading the security credentials into the process space, you must specify the wallet location in the parameter file it reads. The syntax of the parameter for static configuration is as follows:

```
oss.source.my_wallet =
(SOURCE=
  (METHOD=File)
  (METHOD_DATA=
    (DIRECTORY=your wallet location)
      )
    )
```

The dynamic way of specifying this parameter is:

```
MY_WALLET_DIRECTORY = your wallet dir
```

The default wallet location is $HOME/oracle directory.

# C

# Integrating Authentication Devices Using RADIUS

This appendix explains how third party vendors of authentication devices customize the RADIUS challenge-response user interface to fit their particular device.

> **More Information:** See Chapter 3, "Configuring RADIUS Authentication"

This appendix covers the following topics:

- About the RADIUS Challenge-Response User Interface
- Customizing the Challenge-Response User Interface

# 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 smartcard vendor customizes the Oracle client to issue the challenge to the smartcard reader. Then, when the smartcard receives a challenge, it responds by prompting the user for more information, for example, a PIN.

Oracle has developed a Java interface class for this interface. It is a set of methods written in C code using the Java Native Interface as specified in release 1.1 of the Java Development Kit™ from JavaSoft. This code, provided below, is system specific. You can find it in the file OracleRadiusInterface in the following directory: $ORACLE_HOME / network / security / classes.

# Customizing the Challenge-Response User Interface

You customize this interface by creating your own class to handle the challenge-response conversation between the Oracle client and the RADIUS server. You then open your sqlnet.ora file, look up the SQLNET.RADIUS_ AUTHENTICATION_INTERFACE parameter, and replace the name of the class listed there, namely, DefaultRadiusInterface, with the name of the new class you have just created. When you make this change in the sqlnet.ora file, this 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();
  public String getResponse();
  }
```

| 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 relayed as the password from the server. If the user enters a valid password, a challenge may or may not be returned by the server. |
| `radiusChallenge` | presents a request sent from the RADIUS server for the user to enter more information |
| `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. |

# Glossary

### authentication

The process of verifying the identity of a user, device, or other entity in a computer system, often as a prerequisite to allowing access to resources in a system.

### 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.

### certificate

A certificate is created when an entity's public key is signed by a trusted identity, that is, a certificate authority. This 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.

### 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. The important property of a cryptographic checksum is that, without knowing the secret key, a malicious interceptor has only an infinitesimally small chance of being able to construct an altered message with a valid corresponding checksum.

### cipher suite

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

### 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).

### CORBA

Common Object Request Broker Architecture. An architecture that enables pieces of programs, called objects, to communicate with one another regardless of the programming language in which they are written or the operating system on which they are running. CORBA was developed by an industry consortium known as the Object Management Group (OMG).

### cryptography

The act of writing and deciphering secret code resulting in secure messages.

### decryption

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

### DES

The U.S. Data Encryption Standard.

### 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.

### encryption

The process of disguising a message in order to hide its substance.

### HTTP

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.

### 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 kinit program and providing a password.

### integrity

The guarantee that the contents of the message received were not altered from the contents of the original message sent.

### IIOP

Internet Inter-ORB Protocol. A protocol developed by the Object Management Group (OMG) to implement CORBA solutions over the World Wide Web. IIOP enables browsers and servers to exchange integers, arrays, and more complex objects, unlike HTTP, which supports only transmission of text.

### KDC/TGS

Key Distribution Center / Ticket Granting Service. In Kerberos authentication, the KDC maintains a list of user principals and is contacted through the kinit program for the user's **initial ticket**. 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 / TGS is a trusted third party that must run on a secure host. It creates ticket-granting tickets and service tickets. The KDC and TGS are usually the same entity.

### 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.

### kinstance

An instantiation or location of a 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.

### MD5

An algorithm that assures data integrity by generating a unique, 128-bit cryptographic message digest value from the contents of a file. If as little as a single bit value in the file is modified, the MD5 checksum for the file will change. Forgery

of a file in a way that will cause MD5 to generate the same result as that for the original file is considered extremely difficult.

### 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**.

### Net8

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. Net8 supports distributed processing and distributed database capability. Net8 is an "open system" because it is independent of the communication protocol, and users can interface Net8 to many network environments.

### 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.

### principal

A Kerberos object, consisting of *kservice/kinstance@REALM*. See also *kservice*, *kinstance*, and *realm*. A uniquely-identified client or server.

### 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 the recipient's private key.

### public/private key pair

A mathematically related set of two numbers where one is called the private key and the other is called the public key. Public keys are typically made widely

available, while a private key is available only to the owner. Data encrypted with a public key can be decrypted with its associated private key and vice versa. However, data encrypted with a public key cannot be decrypted with the same public key.

### realm

A Kerberos object. A set of clients and servers operating under a single key distribution center / ticket-granting service (KDC / TGS). *kservices* that are in different realms but that have the same name are unique.

### Secure Hash Algorithm

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.

### service

A network resource used by clients; for example, an Oracle database server.

### service name

For Kerberos-based authentication, the **kservice** portion of a service 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.

### session key

A key shared by at least two parties (usually a client and a server).

### server

A provider of a service.

### service principal

See **principal**.

### service ticket

Trusted information used to authenticate the client. A ticket-granting ticket is also known as the initial ticket, is obtained by directly or indirectly running kinit 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.

### SHA

See **Secure Hash Algorithm**.

### smartcard

A plastic card (like a credit card) with an embedded integrated circuit for storing information, including such information as user names and passwords. A smartcard 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.

### ticket

A piece of information that helps identify who the owner is. See **service ticket**.

### 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.

### trusted certificate

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.

### trustpoint

See **trusted certificate**.

### wallet

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

### Wallet Resource Locator

A directory path that provides all the necessary information to locate a particular wallet.

### WRL

See **Wallet Resource Locator**.

### X.509

The public keys can be signed in various data formats. The X.509 format from ISO is one such popular format.

# Index

## D



## E



## F

## G



## H



## I



## K



## L



## M



## N



## O



## P

## R



## S

## T



## U



## V



## W



## X