Oracle® Database

Net Services Administrator's Guide

11*g* Release 1 (11.1)

B28316-04

April 2008

ORACLE®

Oracle Database Net Services Administrator's Guide, 11*g* Release 1 (11.1)

B28316-04

Primary Author: Richard Strohm

Contributing Author: Lance Ashdown

Contributors: Robert Achacoso, Santanu Datta, Steve Ding, Bill Hodak, Feroz Khan, Bhaskar Mathur, Scot McKinley, Ed Miner, Sweta Mogra, Srinivas Pamu, Kant Patel, Murali Purayathu, Karthik Rajan, Saravanakumar Ramasubramanian, Sudeep Reguna, Norman Woo

Contents

Preface ix

Audience ix
Documentation Accessibility x
Related Documentation xi
Conventions xi

What's New in Oracle Net Services? xiii

Oracle Database 11*g* New Features in Oracle Net Services xiii

Part I Getting Started with Oracle Net Services

1 Introduction to Oracle Net Services

About Oracle Net Services 1-1
Connectivity 1-1
Manageability 1-5
Internet and Intranet Scalability 1-7
Performance Between the Middle Tier and Oracle Database 1-11
Network Security 1-12
A Suite of Networking Components 1-13
Oracle Net 1-13
Oracle Net Listener 1-14
Oracle Connection Manager 1-15
Networking Tools 1-15
Oracle Advanced Security 1-16

2 Quick Start to Oracle Net Connections

Prerequisites to Establishing Connectivity 2-1
Task 1: Confirm Network Availability 2-1
Task 2: Start Oracle Net Listener and the Oracle Database Server 2-2
Task 3: Configure the Client for Connection to a Database 2-3
Task 4: Connect to the Database 2-6

3 Connectivity Concepts

Database Service and Database Instance Identification 3-1

Database Services ... 3-1
Database Instances ... 3-3
Service Accessibility ... 3-5
Protocol Address ... 3-5
Connect Data ... 3-6
Enhanced Service Accessibility with Multiple Listeners ... 3-7
Connect-Time Failover ... 3-7
Transparent Application Failover ... 3-7
Client Load Balancing ... 3-8
Connection Load Balancing ... 3-8
Service Handlers ... 3-8
Dispatchers ... 3-8
Dedicated Server Processes ... 3-10
Database Resident Connection Pooling ... 3-11
Naming ... 3-12

4 Configuration Management Concepts

Configuration Models ... 4-1
Localized Configuration File Support ... 4-1
Directory Server Support ... 4-3
Directory Naming Overview ... 4-3
Naming Configuration Storage in a Directory Server ... 4-4
Net Service Alias Entries ... 4-5
Directory Entries ... 4-6
Adding or Modifying Entries in the Directory Server ... 4-7
Client Connections Using Directory Naming ... 4-9
Oracle Net Configuration and Directory Server Design ... 4-10
Limitations of Directory Naming Support with Microsoft Active Directory ... 4-14

5 Architecture of Oracle Net Services

Oracle Net Stack Communication Architecture ... 5-1
Stack Communication for Client/Server Application Connections ... 5-1
Stack Communication for Java Application Connections ... 5-5
Stack Communication for Web Client Connections ... 5-6
Listener Architecture ... 5-7
Database Server Process Architecture ... 5-9
Shared Server Processes ... 5-9
Dedicated Server Processes ... 5-10
Oracle Connection Manager Architecture ... 5-11
A Complete Architecture ... 5-12

6 Configuration and Administration Tools Overview

User Interface Tools ... 6-1
Oracle Enterprise Manager ... 6-1
Oracle Net Manager ... 6-2
Selecting When to Use Oracle Enterprise Manager and Oracle Net Manager ... 6-5

Oracle Net Configuration Assistant 6-6
Oracle Net Control Utilities 6-7
Listener Control Utility 6-7
Oracle Connection Manager Control Utility 6-8
Duties of a Network Administrator 6-8

Part II Configuration and Administration of Oracle Net Services

7 Planning the Network

Deploying a Network Inside an Organization 7-1
Scalability 7-2
Availability 7-4
Naming Methods 7-5
JDBC Drivers 7-5
Security 7-5
Tuning and Performance 7-5
Deploying a Network for the Internet 7-6
Scalability 7-7
Availability 7-7
Naming Methods 7-7
JDBC Drivers 7-8
Security 7-8
Naming Considerations 7-8
Performance Considerations 7-9
Listener Queue Size 7-9
Session Data Unit Size for Data Transfer Optimization 7-10
Persistent Buffer Flushing for TCP/IP 7-10
Planning Summary 7-10

8 Configuring Naming Methods

Naming Method Configuration Overview 8-1
About Connect Descriptors 8-1
Naming Methods 8-2
Using the Easy Connect Naming Method 8-3
Configuring the Local Naming Method 8-8
Configuring the tnsnames.ora File During Installation 8-8
Configuring the tnsnames.ora File After Installation 8-9
Configuring the Directory Naming Method 8-14
Directory Naming Method Configuration Steps 8-14
Administering the OracleNetAdmins Group 8-19
Exporting Local Naming Entries to a Directory Naming Server 8-21
Creating Multiple Default Contexts in a Directory Naming Server 8-24
Exporting Directory Naming Entries to a tnsnames.ora File 8-24
Configuring External Naming Methods 8-25
Network Information Service 8-25
Distributed Computing Environment (DCE) Cell Directory Service (CDS) 8-28

9 Configuring Profiles

Profile Configuration Overview 9-1
Profile Configuration During Installation 9-1
Configuring Client Attributes for Names Resolution 9-2
Configuring a Default Domain for Clients 9-2
Prioritizing Naming Methods 9-3
Routing Connection Requests 9-4
Configuring Database Access Control 9-4
Configuring Advanced Profile Information 9-5
Configuring External Naming Methods 9-8
Configuring Oracle Advanced Security 9-8

10 Configuring and Administering Oracle Net Listener

Oracle Net Listener Configuration Overview 10-1
Oracle Net Listener Configuration During Installation 10-2
Customizing Oracle Net Listener Configuration 10-3
Configuring Listening Protocol Addresses 10-3
Configuring Access to Oracle JServer 10-4
Handling Large Volumes of Concurrent Connection Requests 10-5
Configuring Static Service Information 10-6
Default Oracle Net Listener Administration 10-7
Configuring Service Registration 10-10
Configuring Service Registration 10-10
Registering Information with the Default, Local Listener 10-10
Registering Information with a Nondefault Listener 10-11
Registering Information with a Remote Listener 10-12
Configuring a Naming Method 10-14
Listener Administration 10-14
Starting and Stopping a Listener 10-14
Determining the Current Status of a Listener 10-15
Monitoring Services of a Listener 10-17
Monitoring Listener Log Files 10-19

11 Configuring and Administering Oracle Connection Manager

Oracle Connection Manager Configuration Overview 11-1
Configuring Oracle Connection Manager 11-2
Configuring the Oracle Connection Manager Computer 11-2
Configuring Clients for Oracle Connection Manager 11-4
Configuring the Oracle Database Server for Oracle Connection Manager 11-5
Enabling Oracle Connection Manager Features 11-6
Enabling Session Multiplexing 11-7
Enabling Access Control 11-7
Migrating cman.ora from Oracle9*i* to Oracle Database 11*g* 11-8

12 Configuring Dispatchers

Configuring Dispatchers 12-1

Enabling Connection Pooling 12-2
Enabling Session Multiplexing 12-3
Grouping Services by Dispatcher 12-3
Configuring Clients for Environments Using Both Dedicated Server and Shared Server 12-4

13 Enabling Advanced Features of Oracle Net Services

Configuring Advanced Network Address and Connect Data Information 13-1
Creating a List of Listener Protocol Addresses 13-1
Configuring Address List Parameters 13-3
Configuring Advanced Connect Data Parameters 13-5
Configuring Connection Load Balancing 13-7
Example: Connection Pool Load Balancing for Shared Server Configuration 13-8
Example: Connection Pool Load Balancing for Dedicated Server Configuration 13-10
Configuring Transparent Application Failover 13-13
About TAF 13-13
What TAF Restores 13-14
TAF Database Configurations 13-15
FAILOVER_MODE Parameters 13-15
TAF Implementation 13-16
TAF Verification 13-18
Specifying the Instance Role for Primary and Secondary Instance Configurations 13-18
Configuring Connections to Non-Oracle Database Services 13-20
Default Configuration for External Procedures 13-21
Configuring Oracle Net Services for Oracle Heterogeneous Services 13-26
Configuring Oracle Net Services for an Oracle Rdb Database 13-27

14 Optimizing Performance

Configuring Session Data Unit 14-1
Configuring I/O Buffer Space 14-3
Configuring SDP Protocol Support for Infiniband Network Communication to the Database Server 14-6
Configuring the Listener and the Oracle Database To Limit Resource Consumption By Unauthorized Users 14-8

Part III Testing and Troubleshooting Oracle Net Services

15 Establishing a Network and Testing the Connection

Connecting to a Database 15-1
Starting Oracle Net Services Components 15-1
Entering a Connect String 15-4
Initiating Connections 15-6
Testing the Network 15-8
Testing a Listener 15-8
Testing Oracle Connection Manager 15-8
Testing Configuration on the Database Server 15-8
Testing Network Connectivity from the Client 15-9

16 Troubleshooting Oracle Net Services

Diagnosing Oracle Net Services 16-1
Automatic Diagnostic Repository 16-2
ADRCI: ADR Command Interpreter 16-6
Server Diagnostics 16-7
Client Diagnostics 16-8
Resolving the Most Common Error Messages for Oracle Net Services 16-10
Troubleshooting Directory Naming Errors 16-16
Troubleshooting Tips from the Field for Oracle Net Services 16-17
Questions to Ask When Troubleshooting Oracle Net Services 16-17
Troubleshooting the TNS-12154 Error 16-18
Problem Description for TNS-12154 16-18
Troubleshooting TNS-12154 on UNIX 16-18
Troubleshooting Network Problems Using Log and Trace Files 16-20
Logging Error Information for Oracle Net Services 16-20
Oracle Net Error Stacks 16-20
Oracle Net Services Log File Names 16-22
Setting Logging Parameters 16-22
Setting Logging Settings During Runtime of Control Utilities 16-25
Using Log Files 16-25
Analyzing Listener Log Files 16-26
Analyzing Oracle Connection Manager Logs 16-30
Tracing Error Information for Oracle Net Services 16-32
Oracle Net Services Trace File Names 16-33
Setting Tracing Parameters 16-33
Setting Tracing Settings During Runtime of Control Utilities 16-39
Evaluating Oracle Net Services Traces 16-40
Using the Trace Assistant to Examine Trace Files 16-44
Contacting Oracle Support Services 16-58

Glossary

Index

Preface

The *Oracle Database Net Services Administrator's Guide* provides the information you need to understand and use Oracle Net Services and its related applications and components.

This document describes the features of Oracle Database 11*g* software that apply to the UNIX, Linux, Windows 2000, Windows XP, and Windows Server 2003 operating systems.

This preface contains these topics:

- Audience
- Documentation Accessibility
- Related Documentation
- Conventions

Audience

Oracle Database Net Services Administrator's Guide is intended for the following kinds of readers:

- Network administrators
- Directory server administrators
- Database administrators
- Decision makers

This guide is especially targeted for network administrators who are responsible for ensuring connectivity. For network administrators, Oracle Corporation recommends:

- For a conceptual understanding of Oracle Net Services, read all of Part I, "Getting Started with Oracle Net Services"
- For essential configuration instructions, read Chapter 3, "Connectivity Concepts" in Part I and all of Part II, "Configuration and Administration of Oracle Net Services"
- For troubleshooting, read Part III, "Testing and Troubleshooting Oracle Net Services"

For directory administrators, Oracle Corporation recommends:

- To understand how Oracle Net Services uses a directory server, read Chapter 3, "Connectivity Concepts" in Part I

- For instructions for configuring naming information in a directory server, as well as exporting existing naming data to a directory server, read Chapter 8, "Configuring Naming Methods" in Part II

For database administrators, Oracle Corporation recommends:

- To gain an understanding of the big networking picture, read Chapter 1, "Introduction to Oracle Net Services" and Chapter 2, "Quick Start to Oracle Net Connections"
- To get an overview of networking tools, read Chapter 5, "Architecture of Oracle Net Services"
- To understand how to configure Oracle database server features that require listener and shared server configuration, read Chapter 7, "Planning the Network", Chapter 10, "Configuring and Administering Oracle Net Listener", Chapter 12, "Configuring Dispatchers", and Chapter 14, "Optimizing Performance"

For decision makers, Oracle Corporation recommends reading Chapter 1, "Introduction to Oracle Net Services", Chapter 2, "Quick Start to Oracle Net Connections", Chapter 4, "Configuration Management Concepts", and Chapter 7, "Planning the Network" to gain an understanding of how Oracle Net Services fits into the overall network architecture and for explaining the basics of Oracle Net Services.

Oracle Corporation recommends that all readers skim Part I, to ensure that they have the background required to benefit from the rest of the guide.

Documentation Accessibility

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

Accessibility of Code Examples in Documentation

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

Accessibility of Links to External Web Sites in Documentation

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

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, 7 days a week. For TTY support, call 800.446.2398. Outside the United States, call +1.407.458.2479.

Related Documentation

For more information, see these Oracle resources:

- *Oracle Database Net Services Reference*
- Oracle Database 11*g* documentation set

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

Printed documentation is available for sale in the Oracle Store at

`http://oraclestore.oracle.com`

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

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

If you already have a user name and password for Oracle Technology Network, you can go directly to the documentation section of the Oracle Technology Network Web site at

`http://www.oracle.com/technology/documentation`

For additional information about the **OSI**, see:

`http://www.ietf.org`

Oracle error message documentation is only available in HTML. If you only have access to the Oracle Documentation CD, you can browse the error messages by range. Once you find the specific range, use your browser's "find in page" feature to locate the specific message. When connected to the Internet, you can search for a specific error message using the error message search feature of the Oracle online documentation.

Conventions

The following text conventions are used in this document:

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

What's New in Oracle Net Services?

This preface describes the new networking features of Oracle Database 11*g* and provides pointers to additional information.

Oracle Database 11*g* New Features in Oracle Net Services

The new features for Oracle Net Services in Oracle Database 11*g* include:

- Enhanced Network Administration Security
  - Non-Anonymous LDAP Access for Net Naming

    Database administrators can now restrict access to a service by associating an access control list (ACL) with it. This feature can be used with installations that require an extremely high level of security.

  See "Configuring the Directory Naming Method" on page 8-14 and "Configuring Database Access Control" on page 9-4.
- Performance Improvements
  - New Oracle Net fastpath for the common usage scenarios significantly improves Oracle Net performance and is enabled by default in Oracle Database 11*g*. Users do not need to perform any configuration for this feature.
  - Efficient network support for bulk data transfers (for example, LOBs). This feature relaxes the notion of session data unit (SDU) and optimizes large data transfers over an Oracle Net connection by using new paradigms. See "Session Data Unit (SDU) Size" on page 7-6.
  - PHP Scalability

    This feature adds Oracle Net support for efficient event-dispatch mechanisms on platforms that support them. This is internally enabled for PHP usage scenarios and users do not need to perform any configuration for enabling this feature.
- Fast Reconnects for High Availability (HA)
  - This feature adds a mechanism for efficient dead node detection for connect time failover. Configurable timeouts have been implemented at various levels.
- Support for Database Resident Connection Pooling
  - This feature enables you to share connections or sessions between multiple middle-tier processes

See Also:

Oracle Database Concepts for more information about database resident connection pooling

"Database Resident Connection Pooling" on page 3-11

- Enhancements to the Easy Connect Naming Method

  For TCP/IP environments, you can simplify client configuration by using the easy connect naming method. The easy connect naming method simplifies network management by allowing clients to connect to Oracle Database services without first configuring net service names. Instead, clients make connections with the host name and optional port and service name of the database. See "Using the Easy Connect Naming Method" on page 8-3.

Part I

Getting Started with Oracle Net Services

Part I provides an overview of Oracle Net Services concepts, products, and tools.

This part contains the following chapters:

- Chapter 1, "Introduction to Oracle Net Services"
- Chapter 2, "Quick Start to Oracle Net Connections"
- Chapter 3, "Connectivity Concepts"
- Chapter 4, "Configuration Management Concepts"
- Chapter 5, "Architecture of Oracle Net Services"
- Chapter 6, "Configuration and Administration Tools Overview"

1

Introduction to Oracle Net Services

This chapter describes the basic elements of Oracle Net Services architecture and the Oracle Net foundation layer.

This chapter contains the following topics:

- About Oracle Net Services
- A Suite of Networking Components

About Oracle Net Services

Oracle Net Services provides enterprise wide connectivity solutions in distributed, heterogeneous computing environments. Oracle Net Services ease the complexities of network configuration and management, maximize performance, and improve network diagnostic capabilities.

This section introduces the basic networking concepts involved in a typical network configuration. The topics include:

- Connectivity
- Manageability
- Internet and Intranet Scalability
- Performance Between the Middle Tier and Oracle Database
- Network Security

Connectivity

Oracle Net, a component of Oracle Net Services, enables a network session from a client application to an Oracle Database server. Once a network session is established, Oracle Net acts as the data courier for both the client application and the database server. It is responsible for establishing and maintaining the connection between the client application and database server, as well as exchanging messages between them. Oracle Net is able to perform these jobs because it is located on each computer in the network.

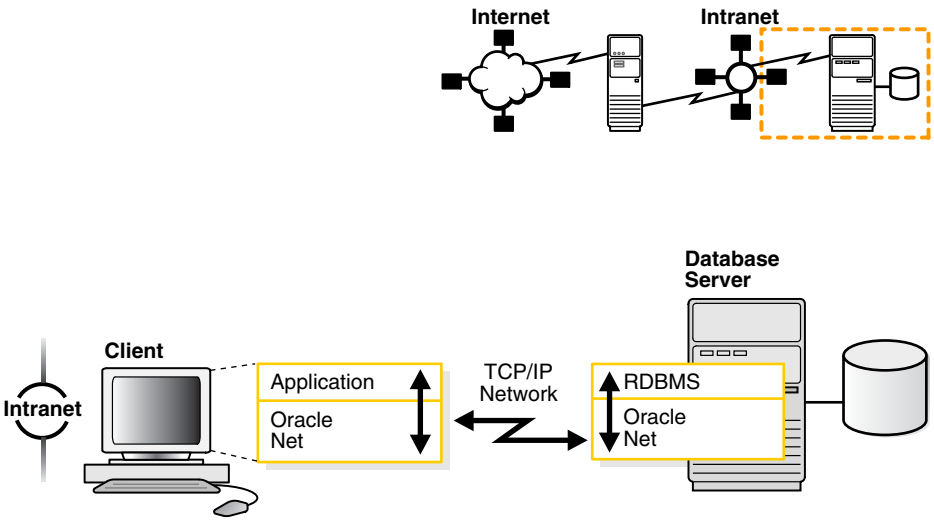
This section discusses the following connectivity topics:

- Client/Server Application Connections
- Web Client Application Connections
- Web Client Connections Without an Application Web Server

Client/Server Application Connections

Oracle Net enables connections from traditional client/server applications to Oracle Database servers. Figure 1–1 shows how Oracle Net enables a network connection between a client and a database server. Oracle Net is a software component that resides on both the client and the database server. Oracle Net is layered on top of a network **Oracle protocol support**—rules that determine how applications access the network and how data is subdivided into packets for transmission across the network. In this illustration, Oracle Net communicates with the **TCP/IP protocol** to enable computer-level connectivity and data transfer between the client and the database server.

Figure 1–1 Client/Server Application Connection



Specifically, Oracle Net is comprised of the **Oracle Net foundation layer**, which establishes and maintains connections, and **Oracle protocol support**, which maps the foundation layer's technology to industry-standard protocols.

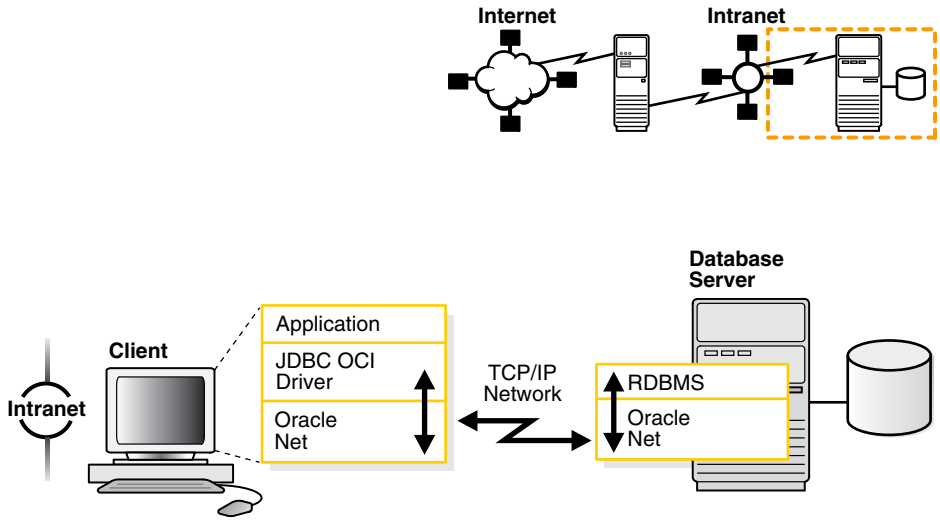
Java Client Application Connections Java client applications access an Oracle Database through a **Java Database Connectivity (JDBC) Driver**, a standard Java interface for connecting from Java to a relational database. Oracle offers the following drivers:

- **JDBC OCI Driver** for client side use with an Oracle client installation
- **JDBC Thin Driver**, a pure Java driver for client side use without an Oracle installation, particularly with applets

These drivers use Oracle Net to enable connectivity between a client application and an Oracle Database.

Figure 1–2 shows a Java client application using a JDBC OCI driver and an Oracle Database server. The Java client application makes calls to the JDBC OCI driver, which in turn translates the JDBC calls directly into the Oracle Net layer. The client then uses Oracle Net to communicate with an Oracle Database that is also configured with Oracle Net.

Figure 1–2 Java Application Connection



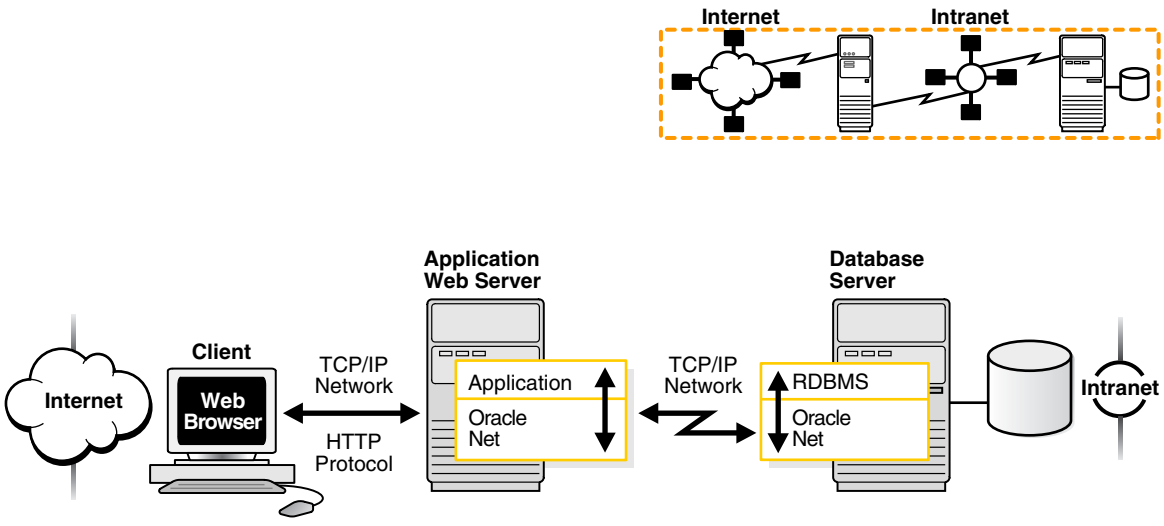
See Also: *Oracle Database JDBC Developer's Guide and Reference*.

Web Client Application Connections

Internet connections from client Web browsers to an Oracle Database server are similar to client/server applications, except that the connection request goes over an application Web server.

Figure 1–3 shows the basic architecture for Web client connections, including a client Web browser, an application Web server, and an Oracle Database server. The browser on the client communicates with the **HTTP protocol** to a Web server to make a connection request. The Web server sends the request to an application where it is processed. The application then uses Oracle Net to communicate with an Oracle Database server that also is configured with Oracle Net.

Figure 1–3 Web Client Connections through Application Web Server



The basic components have the following characteristics:

- Hypertext Transport Protocol (HTTP)

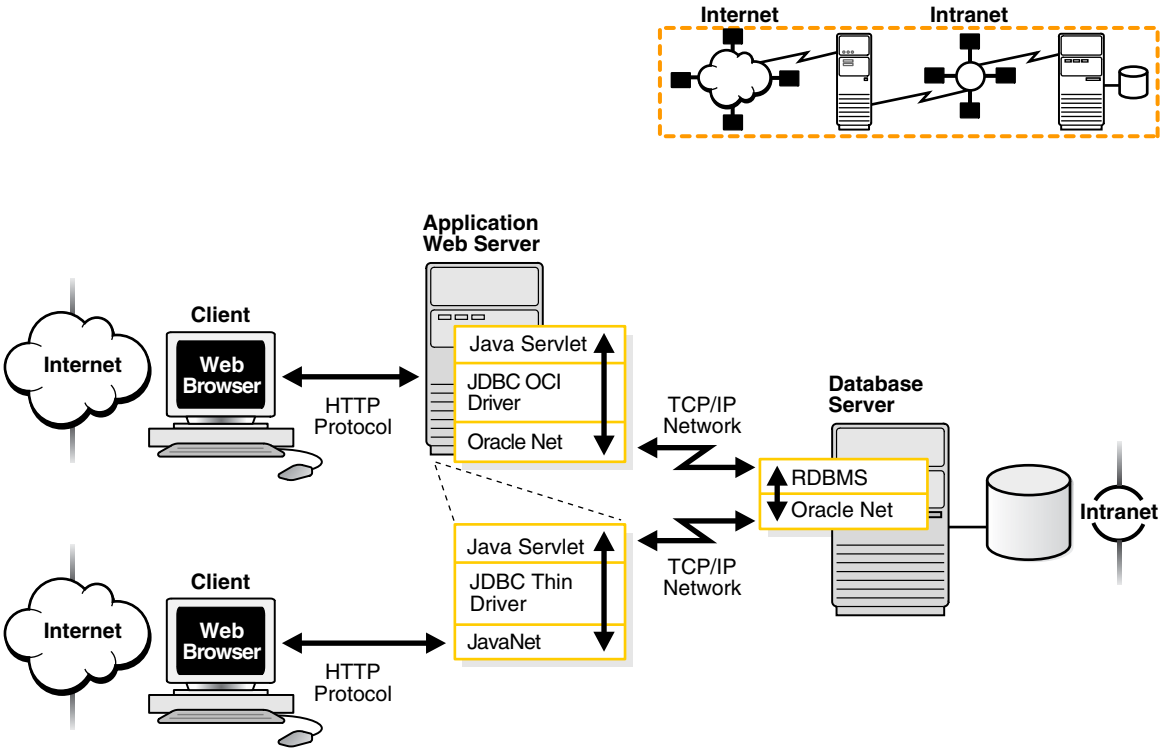
HTTP provides the language that enables Web browsers and application Web servers to communicate.

- Application Web Server

  An application Web server manages data for a Web site, controls access to that data, and responds to requests from Web browsers. The application on the Web server communicates with the database and performs the job requested by the Web server.

Web Client Connections Through Java Application Web Server An application Web server can host Java applications and servlets, as shown in Figure 1–4. Web browsers make a connection request by communicating through HTTP to an application Web server. The application Web server sends the request to an application or a servlet, which in turn uses a JDBC OCI or a JDBC Thin driver to process the request. The driver then uses Oracle Net to communicate with an Oracle Database server that also is configured with Oracle Net.

Figure 1–4 Web Client Connections Through Java Application Web Server



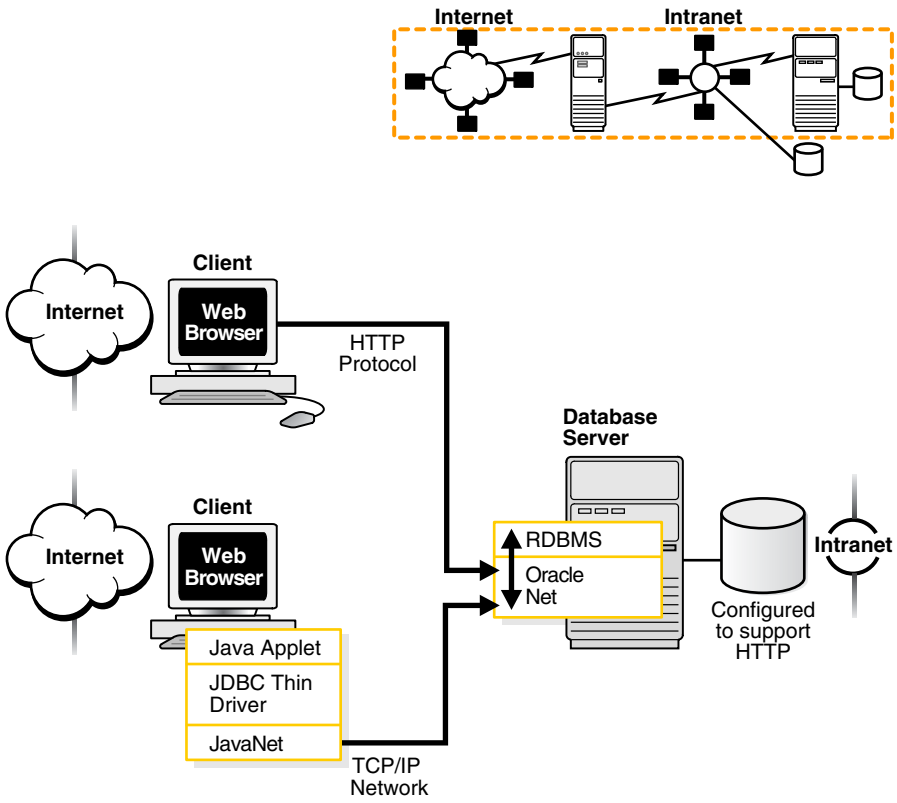
Web Client Connections Without an Application Web Server

Web clients that do not require an application Web server to access applications can access the Oracle Database directly, for example, by using a Java applet. In addition to regular connections, the database can be configured to accept HTTP protocol, **FTP protocol**, or **WebDAV protocol** connections. These protocols are used for connections to **Oracle XML DB** in the Oracle Database instance.

Figure 1–5 shows two different Web clients. The first Web client makes an HTTP connection to the database. The second Web client uses a Web browser with a JDBC Thin driver, which in turn uses a Java version of Oracle Net called JavaNet to communicate with the Oracle Database server that is configured with Oracle Net.

See Also: *Oracle XML DB Developer's Guide*

Figure 1–5 Web Client Connection Scenarios



Manageability

Oracle Net Services offer a number of manageability features that enable you to easily configure and manage networking components. These features are described in the following topics:

- Location Transparency
- Centralized Configuration and Management
- Quick Installation and Configuration

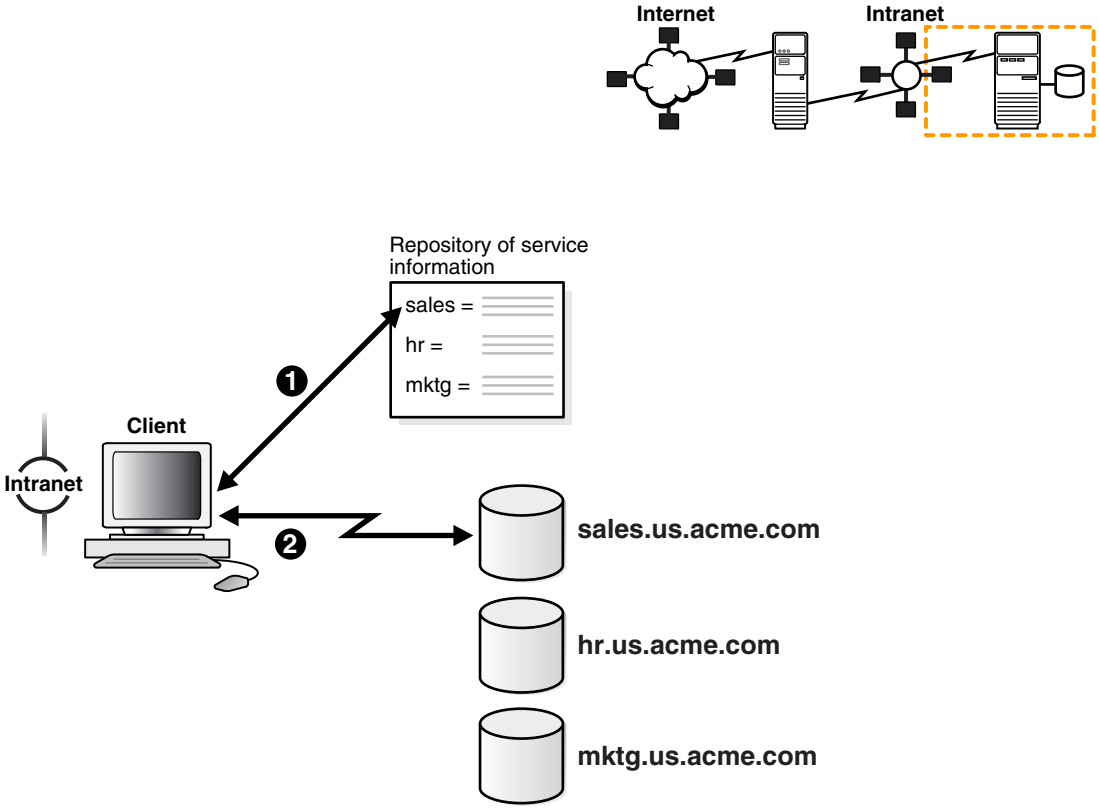
Location Transparency

A company can have several databases, each representing a specific type of service for various client applications. For example, a company may have three databases, which it uses for sales, human resources, and marketing applications. Each database is represented by one or more services. A **service** is identified by a **service name**, for example, `sales.us.acme.com`. A client uses this service name to identify the database it needs to access. The information about the database service and its location in the network is transparent to the client because the information needed for a connection is stored in a repository.

For example, in Figure 1–6, a company has three databases that clients can access. Each database has a distinct service name: `sales.us.acme.com`, `hr.us.acme.com`, and `mktg.us.acme.com`.

1. The client uses the repository to find the information it needs for `sales.us.acme.com`.
2. Once the client has the information it needs, it connects to the database.

Figure 1–6 Service Information Repository



The repository is represented by one or more **naming method**s. Oracle Net Services offer several types of naming methods that support localized configuration on each client, or centralized configuration that can be accessed by all clients in the network. Easy-to-use graphical user interfaces enable you to manage data stored in the naming methods.

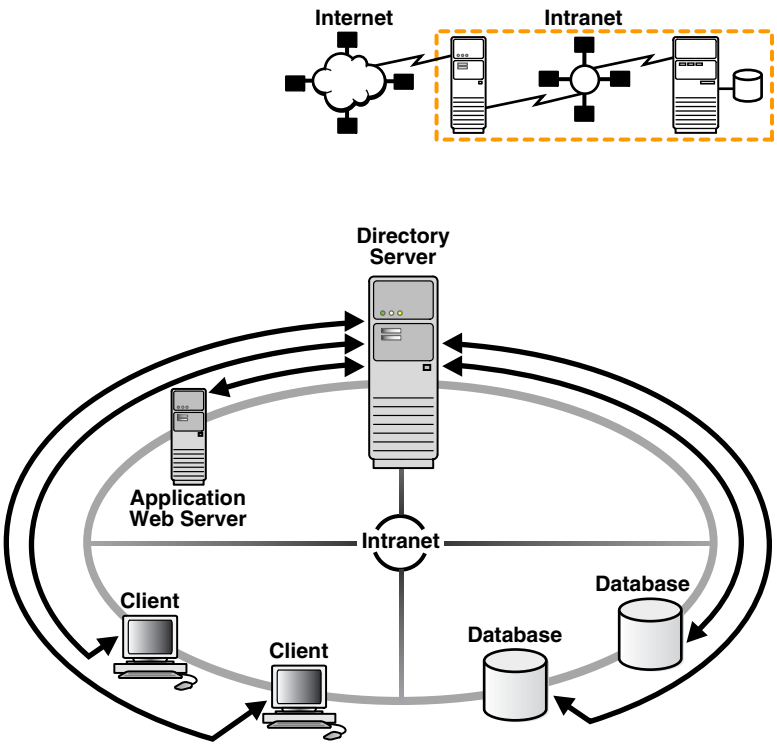
Centralized Configuration and Management

To manage large networking environments, administrators have to be able to easily access a centralized repository to specify and modify the network configuration. For this reason, the Oracle Net Services configuration can be stored in a LDAP-compliant directory server.

Support of LDAP-compliant directory servers provides a centralized vehicle for managing and configuring a distributed Oracle network. The directory can act as a central repository for all information on database network components, user and corporate policies, and user authentication and security, thus replacing client-side and server-side localized configuration files.

All computers on the heterogeneous network can refer to the directory for information. Figure 1–7 shows clients, other servers (such as application Web servers) and Oracle Database servers connecting to a centralized directory server.

Figure 1–7 Centralized Storage of Network Configuration with a Directory Server



See Also: "Directory Server Support" on page 4-3 for an in-depth overview of directory server concepts

Quick Installation and Configuration

Oracle Net Services install quickly and easily. Networking elements for the Oracle Database server and clients are preconfigured for most environments. Information about an Oracle Database service is populated in one or more naming methods. As a result, clients and servers are ready to immediately connect when installed, giving users the benefits of distributed computing.

Internet and Intranet Scalability

Oracle Net provides scalability features that enable you to maximize system resources and improve performance.

Shared Server

The Oracle Database **shared server** architecture increases the scalability of applications and the number of clients that can be simultaneously connected to the database. The shared server architecture also enables existing applications to scale up without making any changes to the application itself.

When using shared server, clients do not communicate directly with a database's **server process**—a database process that handles a client's requests on behalf of a database. Instead, client requests are routed to one or more **dispatcher**s. The dispatchers place the client requests on a common queue. An idle **shared server** from the shared pool of server processes picks up and processes a request from the queue. This means a small pool of server processes can serve a large number of clients.

Figure 1–8 and Figure 1–9 show the basic difference between the shared server connection model and the traditional **dedicated server** connection model. In the shared server model, a dispatcher can support multiple client connections concurrently. In the dedicated server model, there is one server process for each client. Each time a connection request is received, a server process is started and dedicated to that connection until completed. This introduces a processing delay.

Shared server is ideal in configurations with a large number of connections because it reduces the server memory requirements. Shared server is well suited for both Internet and intranet environments.

Figure 1–8 Dedicated Server Architecture

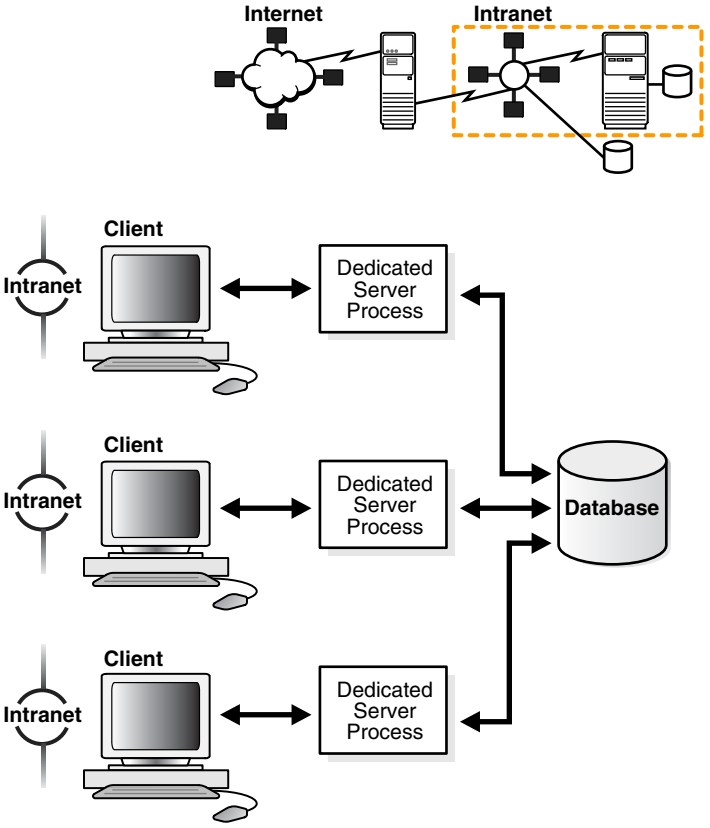
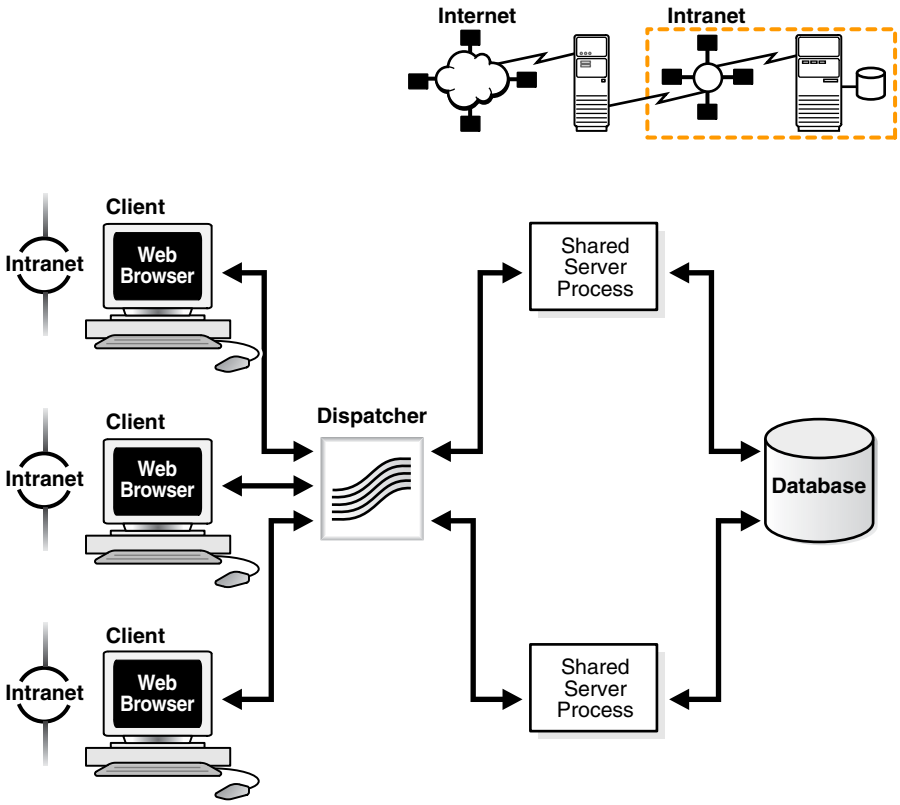


Figure 1–9 Shared Server Architecture



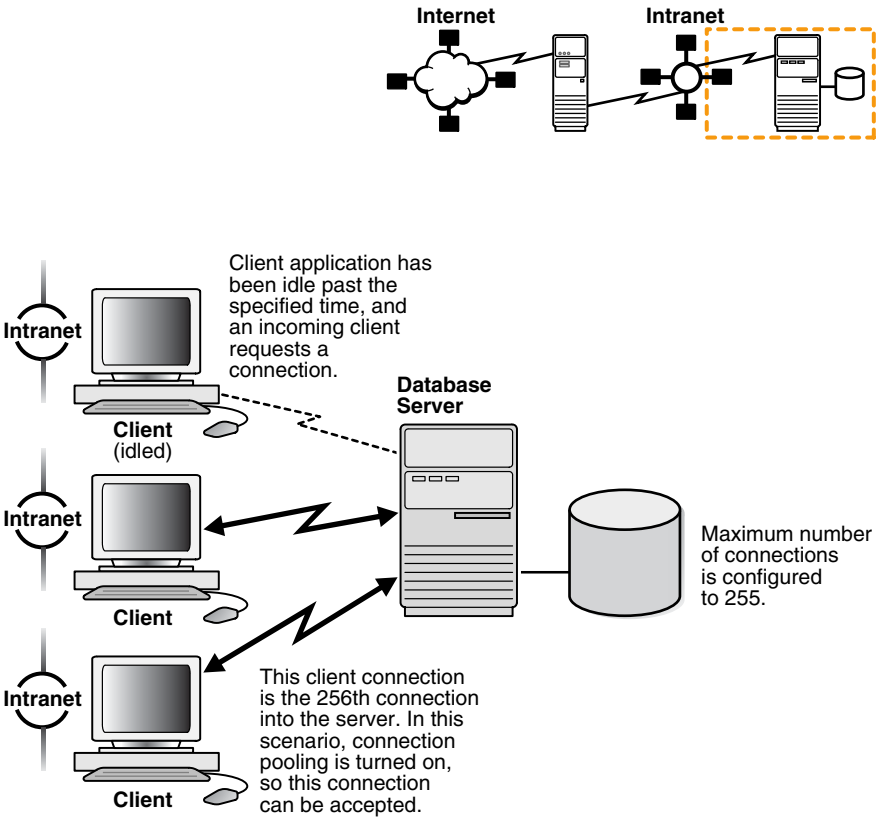
Utilization of server resources can be further enhanced with Oracle Net Services features that are configurable through shared server. These features are discussed in the following sections:

- Connection Pooling
- Session Multiplexing

Connection Pooling When thousands of clients are running interactive Web applications, many of these sessions may be idle at a given time. The **connection pooling** feature enables the database server to timeout an idle session and use the connection to service an active session. The idle logical session remains open, and the physical connection is automatically reestablished when the next request comes from that session. Therefore, Web applications can allow larger numbers of concurrent users to be accommodated with existing hardware.

Figure 1–10 shows how connection pooling works. In this example, the Oracle Database server has been configured with 255 connections. One of the clients has been idle past a specified amount of time. Connection pooling makes this connection available to an incoming client connection, which is the 256th connection. When the idle client has more work to do, the connection is reestablished for that client with another client's idle connection.

Figure 1–10 Connection Pooling

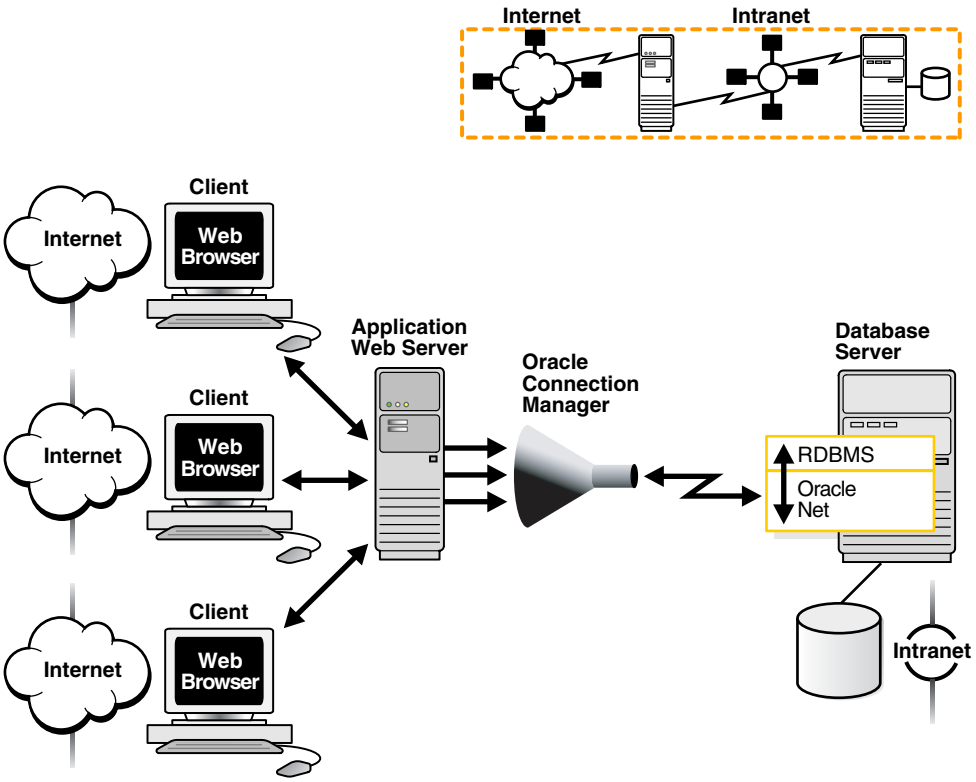


Session Multiplexing **Oracle Connection Manager**, an Oracle Net Services component, enables multiple client network sessions to be multiplexed, or funneled, through a single network connection to a database.

The **session multiplexing** feature reduces the demand on resources needed to maintain multiple network sessions between two processes by enabling the server to use fewer network connection endpoints for incoming requests. In this way the total number of network sessions that a server can handle is increased. One Oracle Connection Manager with multiple gateways enables thousands of concurrent users to connect to a server.

Figure 1–11 shows how session multiplexing can be used in a Web architecture. When Oracle Connection Manager is run on the same computer as an application Web server, the application Web server can route multiple client sessions through Oracle Connection Manager to ensure that those sessions have continuous access to an Oracle Database server. This functionality is especially useful for Web applications where session availability and response time are major concerns.

Figure 1–11 Session Multiplexing



Performance Between the Middle Tier and Oracle Database

Oracle Net Services provides support for Infiniband high-speed networks. InfiniBand is a high-bandwidth I/O architecture designed to increase communication speed between CPUs, server-side devices, and network subsystems. Specifically, Oracle Net Services provides support for the **SDP protocol**. SDP is an industry-standard wire protocol intended for use between Infiniband network peers.

SDP reduces the overhead of TCP/IP by eliminating intermediate replication of data and transferring most of the messaging burden away from the CPU and onto the network hardware. The result is a low-**latency**, increased bandwidth, high-throughput connection that reduces the amount of CPU cycles dedicated to network processing.

The communication between clients, including Oracle Application Server (OracleAS) or any other third-party middle-tier client, and an Oracle Database 11*g* database can take advantage of high-speed interconnect benefits. OracleAS installs with Oracle TCP/IP support.

A driver installed on the OracleAS servers transparently converts TCP/IP support to SDP support. The SDP requests are then sent to an Infiniband switch that processes and forwards the requests from the OracleAS servers to the database server. The SDP requests are then sent to an Infiniband switch that processes and forwards the requests from the OracleAS servers to the database server.

See Also: "Configuring SDP Protocol Support for Infiniband Network Communication to the Database Server" on page 14-6

Network Security

Data access and secure transfer of data are important considerations when deploying Oracle. Granting and denying access to a database is crucial for a secure network environment. Oracle Net Services enable database access control using features described in the following topics:

- Firewall Access Control
- Protocol Access Control

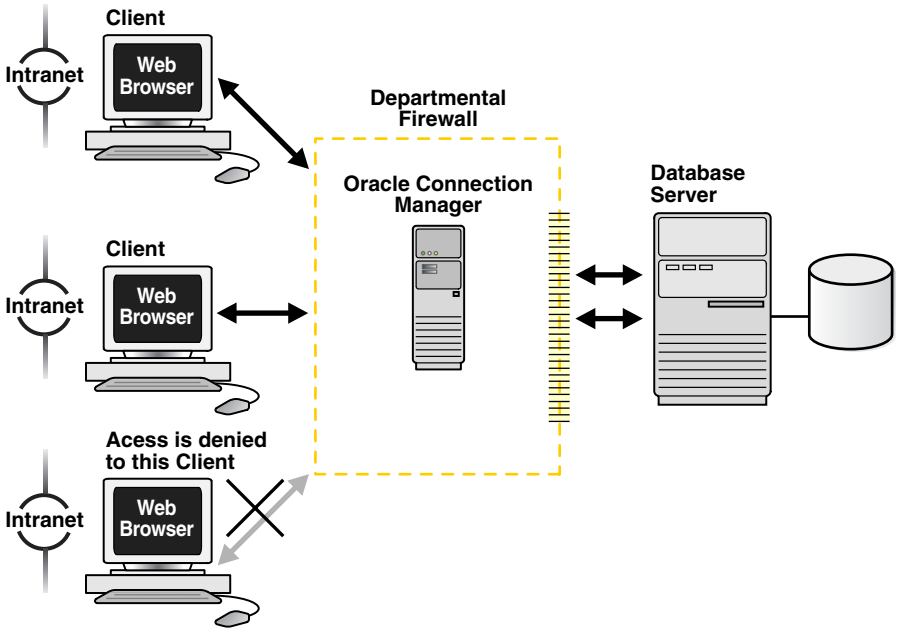
Firewall Access Control

Oracle Connection Manager can be configured to grant or deny client access to a particular database service or a computer. By specifying filtering rules, you can allow or restrict specific client access to a server, based on the following criteria:

- Source host names or IP addresses for clients
- Destination host names or IP addresses for servers
- Destination database service names
- Client use of **Oracle Advanced Security**

Figure 1–12 shows an Oracle Connection Manager positioned between three Web clients and an Oracle Database server. Oracle Connection Manager is configured to allow access to the first two Web clients and to deny access to the third. In order for this configuration to work, clients require the JDBC Thin driver.

Figure 1–12 Intranet Network Access Control with Oracle Connection Manager

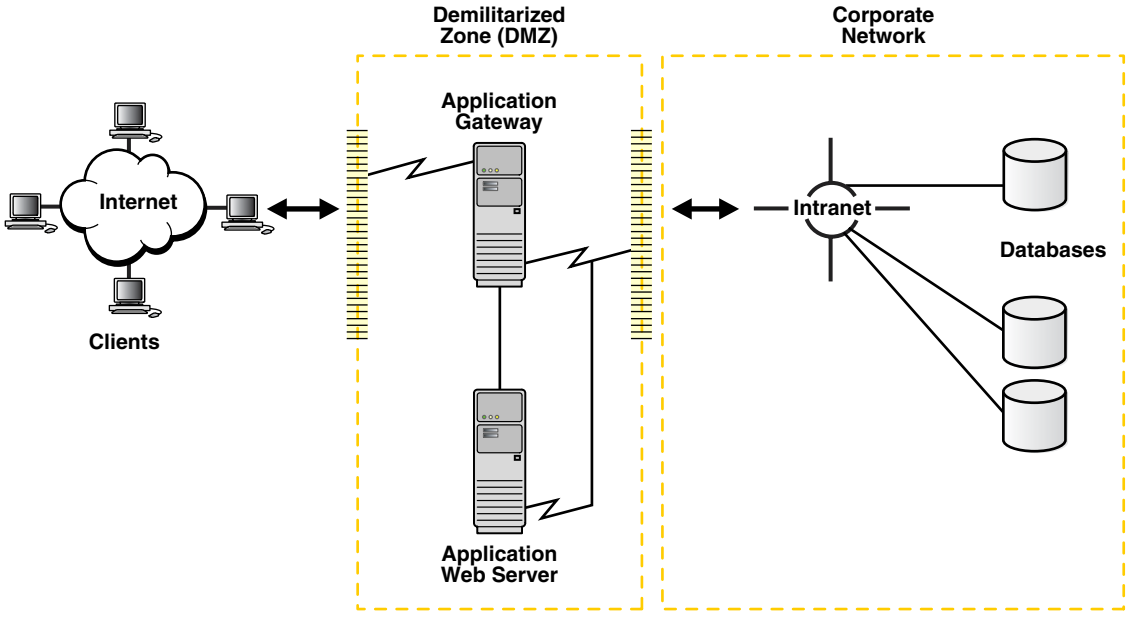


Although Oracle Connection Manager cannot currently be integrated with third-party firewall products, vendors can package it with their own products in a way that enables this product mix to serve as an application gateway.

Figure 1–13 shows an application gateway controlling traffic between internal and external networks and providing a single checkpoint for access control and auditing. As a result, unauthorized Internet hosts cannot directly access the database inside a

corporation, but authorized users can still use Internet services outside the corporate network. This capability is critical in Internet environments to restrict remote access to sensitive data.

Figure 1–13 Internet Network Access Control with an Application Gateway



Protocol Access Control

The database server can be configured with access control parameters in the `sqlnet.ora` configuration file. These parameters specify whether clients are allowed or denied access based on the protocol.

A Suite of Networking Components

The connectivity, manageability, scalability, and security features described in this chapter are provided by the following components:

- Oracle Net
- Oracle Net Listener
- Oracle Connection Manager
- Networking Tools
- Oracle Advanced Security

Oracle Net

Oracle Net is a software layer that resides on the client and the Oracle Database server. It is responsible for establishing and maintaining the connection between the client application and server, as well as exchanging messages between them, using industry-standard protocols. Oracle Net is comprised of two software components:

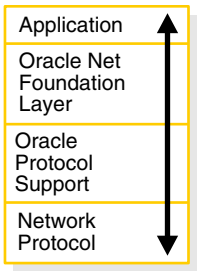
- Oracle Net Foundation Layer
- Oracle Protocol Support

Oracle Net Foundation Layer

On the client side, applications communicate with Oracle Net foundation layer to establish and maintain connections. The Oracle Net foundation layer uses Oracle protocol support that communicates with an industry-standard network protocol, such as TCP/IP, to communicate with the Oracle Database server.

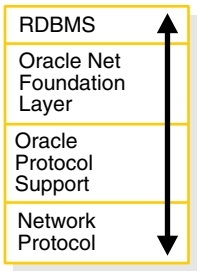
Figure 1–14 illustrates the communication stack on the client.

Figure 1–14 Oracle Net on the Client



The Oracle Database server side is similar to the client side as illustrated in Figure 1–15. A network protocol sends client request information to an Oracle protocol support layer, which then sends information to the Oracle Net foundation layer. The Oracle Net foundation layer then communicates with the Oracle Database server to process the client request.

Figure 1–15 Oracle Net on the Server



Oracle Protocol Support

The Oracle Net foundation layer uses Oracle protocol support to communicate with the following industry-standard network protocols:

- TCP/IP
- TCP/IP with SSL
- Named Pipes
- SDP

Oracle protocol support maps Oracle Net foundation layer functionality to industry-standard protocols used in client/server connections.

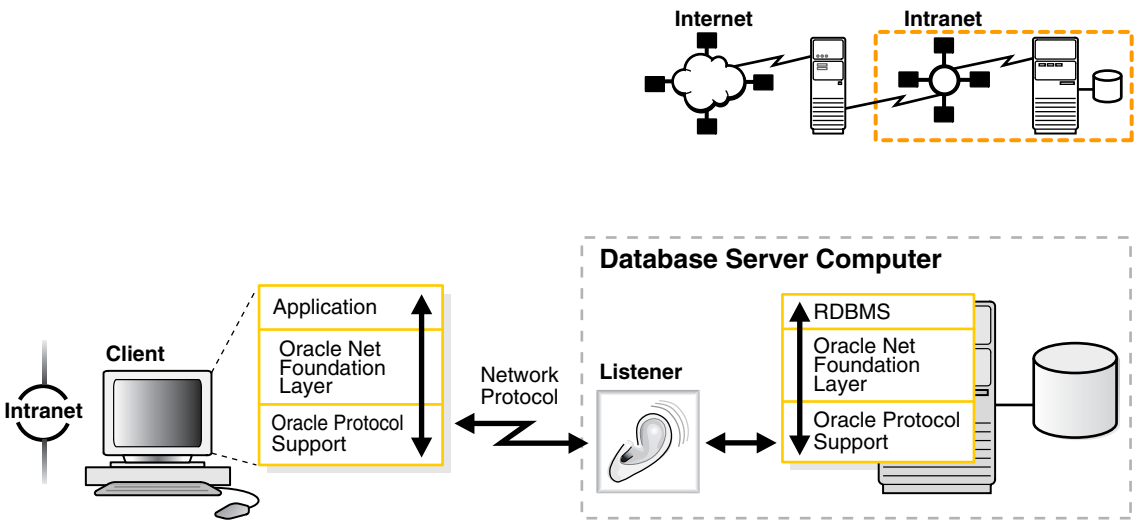
Oracle Net Listener

The one operation unique to the Oracle Database server side is the act of receiving the initial connection through **Oracle Net Listener**. The Oracle Net Listener, referred to in this document as the **listener**, brokers a client request, handing off the request to the server. The listener is configured with a protocol address. Clients configured with the

same protocol address can send connection requests to the listener. When a connection is established, the client and Oracle server communicate directly with one another.

Figure 1–16 shows the listener accepting a connection request from a client and forwarding that request to an Oracle server.

Figure 1–16 Listener in a Connection Request



See Also: Chapter 10, "Configuring and Administering Oracle Net Listener" for more information about the listener

Oracle Connection Manager

Oracle Connection Manager is a software component that resides on its own computer, separate from a client or an Oracle Database server. It proxies and screens requests for the database server. In addition, it multiplexes database sessions.

In its session multiplexing role, Oracle Connection Manager funnels multiple sessions through a single transport protocol connection to a particular destination. In this way, Oracle Connection Manager reduces the demand on resources needed to maintain multiple sessions between two processes by enabling the Oracle Database server to use fewer connection end points for incoming requests.

As an access control filter, Oracle Connection Manager controls access to Oracle databases.

See Also:

- "Session Multiplexing" on page 1-10
- "Firewall Access Control" on page 1-12 for a description of filtering

Networking Tools

Oracle Net Services provides user interface tools and command-line utilities that enable you to easily configure, manage, and monitor the network.

- **Oracle Net Configuration Assistant** enables you to configure listeners and naming methods.

- **Oracle Enterprise Manager** combines configuration functionality across multiple file systems, along with listener administrative control to provide an integrated environment for configuring and managing Oracle Net Services.
- **Oracle Net Manager** provides configuration functionality for an Oracle home on a local client or server host. With Oracle Enterprise Manager (Enterprise Manager) or Oracle Net Manager, you can fine-tune the listener and naming method configuration created with Oracle Net Configuration Assistant. In addition, Enterprise Manager and Oracle Net Manager offer built-in wizards and utilities that enable to you to test connectivity, migrate data from one naming method to another, and create additional network components.
- Command-line control utilities enable you to configure, administer, and monitor network components, including listeners and Oracle Connection Managers.

See Also: Chapter 6, "Configuration and Administration Tools Overview"

Oracle Advanced Security

Oracle Advanced Security is a separately licensable product that provides a comprehensive suite of security features for the Oracle environment. This suite of security features protects enterprise networks and securely extends corporate networks to the Internet. It provides a single source of integration with network encryption and authentication solutions, single sign-on services, and security protocols. Oracle Advanced Security integrates industry standards and delivers unparalleled security to the Oracle network and other networks.

See Also: *Oracle Database Advanced Security Administrator's Guide*

2

Quick Start to Oracle Net Connections

This chapter is designed to help novice users set up and test a simple but common configuration—one between a client application and a database over a TCP/IP network.

This chapter contains these topics:

- Prerequisites to Establishing Connectivity
- Task 1: Confirm Network Availability
- Task 2: Start Oracle Net Listener and the Oracle Database Server
- Task 3: Configure the Client for Connection to a Database
- Task 4: Connect to the Database

Prerequisites to Establishing Connectivity

The tasks in this quick start guide show a TCP/IP connection between a client computer and a database server. The following about the database server and client computers is assumed:

- Database Server Computer
  - It is running on the same network as the client.
  - An Oracle database is installed.
  - TCP/IP protocol support is installed.
  - A listener is configured.
- Client Computer
  - It is running on the same network as the database server.
  - Oracle Client is installed.
  - TCP/IP protocol support is installed.

Task 1: Confirm Network Availability

Before using Oracle Net to connect a client computer to a database server, confirm that the client computer can successfully communicate with the database server computer. Evaluating network connectivity can eliminate network-based errors.

To confirm network connectivity:

1. Confirm that the database server computer can communicate with itself with a **loopback test**.

   A loopback test is a connection from the database server back to itself. Many network protocols provide a means of testing network connections. The utility PING can be used for TCP/IP network.

   In a TCP/IP network, each computer has a unique **IP address**. A name resolution service, such as **Domain Name System (DNS)**, can be used to map the IP address of a computer with its host name. If a name resolution service is not used, then the mapping is typically stored in a centrally maintained file called `hosts`. This file is located in the `/etc` directory on UNIX and the `\winnt` directory on Windows. For example, an entry for a database server computer named `sales-server` may look like the following:

   ```
   #IP address of server     host name       alias
   144.25.186.203            sales-server    sales.us.acme.com
   ```

   To use PING, enter the following at the command line:

   ```
   ping database_server_host
   ```

   The *`database_server_host`* is the host name of the database server computer. For example:

   ```
   ping sales-server
   ```

   If the loopback was unsuccessful, try using the IP address of the database server. For example:

   ```
   ping 144.25.186.203
   ```

2. Verify the client computer can successfully communicate with the database server computer.

   This varies according to the network protocol. For TCP/IP, you can use PING, FTP or TELNET utilities. If the client computer cannot reach the server, verify that the network cabling and network interface cards are correctly connected. Contact your network administrator to correct these problems.

Task 2: Start Oracle Net Listener and the Oracle Database Server

Oracle Net Listener and the Oracle Database server must be running in order for the database server to receive connections.

1. Start the listener with the Listener Control utility. From the command line, enter:

   ```
   lsnrctl
   LSNRCTL> START [listener_name]
   ```

   where *`listener_name`* is the name of the listener defined in the `listener.ora` file. It is not necessary to identify the listener if you are using the default, named `LISTENER`.

   A status message indicating that the listener has started successfully displays.

2. Start the database:

   a. Start SQL*Plus without connecting to the database:

      ```
      sqlplus /nolog
      ```

b. Connect to the database as SYSDBA:

```
SQL> CONNECT username as sysdba
```

For example, `SYSTEM` is a SYSDBA user. You will be prompted to enter a password.

Note: For simplicity in demonstrating this feature, this example does not perform the password management techniques that a deployed system normally uses. In a production environment, follow the Oracle Database password management guidelines, and disable any sample accounts. See *Oracle Database Security Guide* for password management guidelines and other security recommendations.

---

c. Enter the `STARTUP` command, specifying the database name and full path of the parameter file:

```
SQL> STARTUP database_name pfile=file
```

If you do not specify the `PFILE` option, the Oracle database uses the standard initialization parameter file located in the `$ORACLE_HOME/dbs` directory on UNIX platforms, and `%ORACLE_HOME%\database` directory on Windows. If you do not specify a database name, then the database uses the value of the `DB_NAME` parameter specified in the initialization parameter file.

See Also: *Oracle Database Administrator's Guide* for further information about starting the database

3. Confirm that database **service registration** with the listener has completed. From the Listener Control utility, enter:

```
LSNRCTL> SERVICES [listener_name]
```

The `SERVICES` command lists the services supported by the database, along with at lease one available **service handler**.

See Also: "Monitoring Services of a Listener" on page 10-17

Task 3: Configure the Client for Connection to a Database

Once network connectivity has been verified, you can use easy connect naming to connect to the database.

NOTE: Oracle Database 11*g* does not support the use of Oracle Names. Neither Oracle Database 11*g* clients nor Oracle Databases can use Oracle Names, including by LDAP proxy, to resolve naming. Oracle8*i* and Oracle9*i* clients can still use Oracle Names to resolve naming for an Oracle Database 11*g* database; however, customers are strongly recommended to migrate to LDAP to take advantage of the new features of Oracle Database 11*g*.

The easy connect naming method can eliminate the need for service name lookup in the `tnsnames.ora` files for TCP/IP environments. This naming method provides out-of-the-box TCP/IP connectivity to databases. It extends the functionality of the

host naming method by enabling clients to connect to a database server with an optional port and service name in addition to the host name of the database.

```
CONNECT username/password@host[:port][/service_name][:server][/instance_name]
```

> **Note:** In Oracle Call Interface documentation, *server* is referred to as *connect_type*.

If you have performed Oracle Database server install in **Typical** mode, the default service name used by the `oracle` instance is `ORCL`, and the following easy connect syntax can be used to connect to that instance:

```
CONNECT username@host/ORCL
Enter password: password
```

> **See Also:** "Using the Easy Connect Naming Method" on page 8-3 for more information on using this method

Alternate Connection using Oracle Net Configuration Assistant

If you do not wish to use the easy connect naming method, you can use **Oracle Net Configuration Assistant** to create a **net service name**, a simple name for the database service. The net service name resolves to the **connect descriptor**, that is, the network address of the database and the name of the database service. The client will use the net service name to connect to the database.

The following example shows the net service name `sales` mapped to a connect descriptor for a database called `sales.us.acme.com`. A client can use `sales` mapped to connect to `sales.us.acme.com`.

```
sales=
 (DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
   (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)))
```

To configure a client with a net service name:

1. Start Oracle Net Configuration Assistant.

   > **See Also:** "Oracle Net Configuration Assistant" on page 6-6

   The Welcome page appears.
2. Select **Local Net Service Name Configuration**.
3. Click **Next**.

   The Net Service Name Configuration page appears.
4. Click **Add**, and then click **Next**.

   The Net Service Name Configuration, Database Version page appears:

1. If the destination service is an Oracle9*i* or Oracle8*i* database, then select **Oracle8*i* or later database or service**. If destination service is an Oracle8 release 8.0 database, then select **Oracle8 release 8.0 database or service**.
2. Click **Next**.

The Net Service Name Configuration, Service Name appears.

3. Enter the name that identifies the database.

   The **service name** is defined during database creation. If you are unsure what the service name is, ask the Database Administrator who created the database.

4. Click **Next**.

   The Net Service Name Configuration, Select Protocols page appears.

5. Select the protocol on which the listener is configured to listen. Note that this protocol must also be installed on the client. The listener is configured to listen on TCP/IP by default.

6. Click **Next**.

   The page appropriate for the selected protocol appears.

7. Enter the appropriate protocol parameters for the selected protocol in the fields provided, and then click **Next**.

   The Net Service Name Configuration, Test page appears.

8. Click **Yes perform a test**.

   During a test, Oracle Net Configuration Assistant contacts the remote database service, establishes a connection, and ends contact.

   A successful test results in the following message:

   ```
   Connecting...Test successful.
   ```

   If the test fails, it can be because the:

   - Default username (`scott`) and password (`tiger`) are not valid
   - Protocol address information does not match the listener information
   - The listener is not running
   - Destination database service is down

   Depending on your diagnosis of the problem, perform one of the following tasks:

   - Click **Change Login** to change the username and password for the connection.
   - Click **Back** to review protocol address information.
   - Start the listener or Oracle Database on the server, as described in "Task 2: Start Oracle Net Listener and the Oracle Database Server" on page 2-2.

9. Click **Next**.

   The Net Service Name Configuration, Net Service Name page appears.

10. Accept the default net service name or enter another net service name in the Net Service Name field. The name you enter should be unique to the client.

11. Click **Next**.

    The Net Service Name Configuration, Another Net Service Name page appears.

12. Click **No**, and then click **Next**.

    The Net Service Name Configuration, Configuration Done page appears.

13. Click **Next**, and then click **Finish**.

Task 4: Connect to the Database

From the client computer, connect to the database server as follows.

1. Start SQL*Plus:

   ```
   sqlplus
   ```

2. Connect to the database as follows:

   ```
   CONNECT username@net_service_name
   Enter password: password
   ```

   where *username* and *password* are the database user and password, and *net_service_name* is the net service name that you created in "Task 3: Configure the Client for Connection to a Database" on page 2-3.

3

Connectivity Concepts

This chapter explains how databases are identified and how clients access them.

This chapter includes the following topics:

- Database Service and Database Instance Identification
- Service Accessibility
- Enhanced Service Accessibility with Multiple Listeners
- Service Handlers
- Naming

> **See Also:** Chapter 1, "Introduction to Oracle Net Services" for an introductory level overview of networking concepts

Database Service and Database Instance Identification

This section includes the following topics:

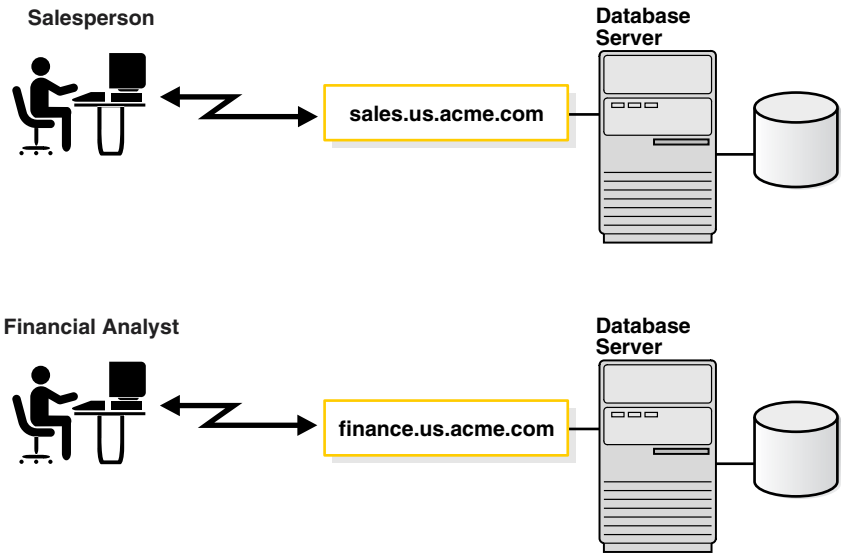
- Database Services
- Database Instances

Database Services

An Oracle database is represented to clients as a **service**, which means that the database performs work on behalf of clients. A database can have one or more services associated with it.

Figure 3–1 shows two databases, each with its own database service for intranet clients. One service, `sales.us.acme.com`, enables salespersons to access the sales database. Another service, `finance.us.acme.com`, enables financial analysts to access the finance database.

Figure 3–1 One Service for Each Database



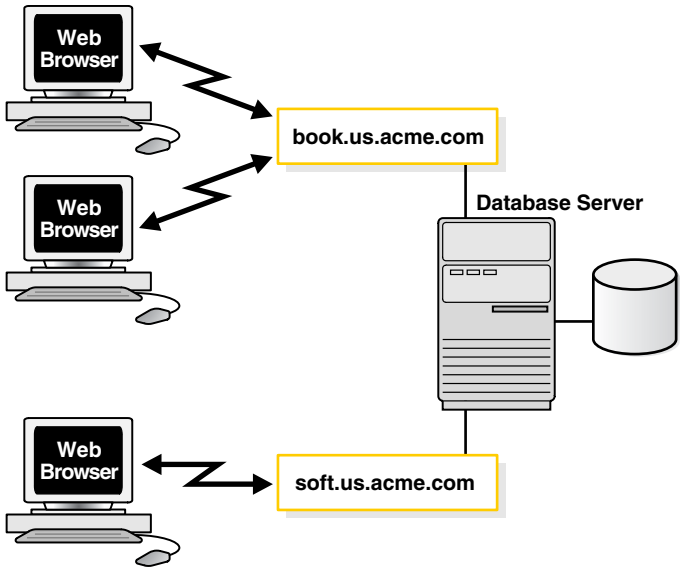
The sales and finance databases are each identified by a **service name**, `sales.us.acme.com` and `finance.us.acme.com`. The service name is specified by the `SERVICE_NAMES` initialization parameter in the **server parameter file**.

The service name defaults to the **global database name**, a name comprising the database name (`DB_NAME` initialization parameter) and domain name (`DB_DOMAIN` initialization parameter). In the case of `sales.us.acme.com`, `sales` is the database name and `us.acme.com` is the domain name.

Note: You can change the value of `SERVICE_NAMES` parameter dynamically with the SQL statement `ALTER SYSTEM` when the database is running.

A database can have multiple services associated with it. Figure 3–2 shows one database that has two different services for Web clients. One service, `book.us.acme.com`, is dedicated to clients making book purchases. The other service, `soft.us.acme.com`, is dedicated to clients making software purchases.

Figure 3–2 Multiple Services Associated with One Database



Associating multiple services with one database enables the following functionality:

- A single database can be identified in a number of different ways by different clients.
- A database administrator can limit or reserve system resources. This level of control enables better allocation of resources to clients requesting one of these services.

> **See Also:** *Oracle Database SQL Language Reference* for more information about the `ALTER SYSTEM` statement and the *Oracle Database Reference* for further information about the `SERVICE_NAMES` parameter

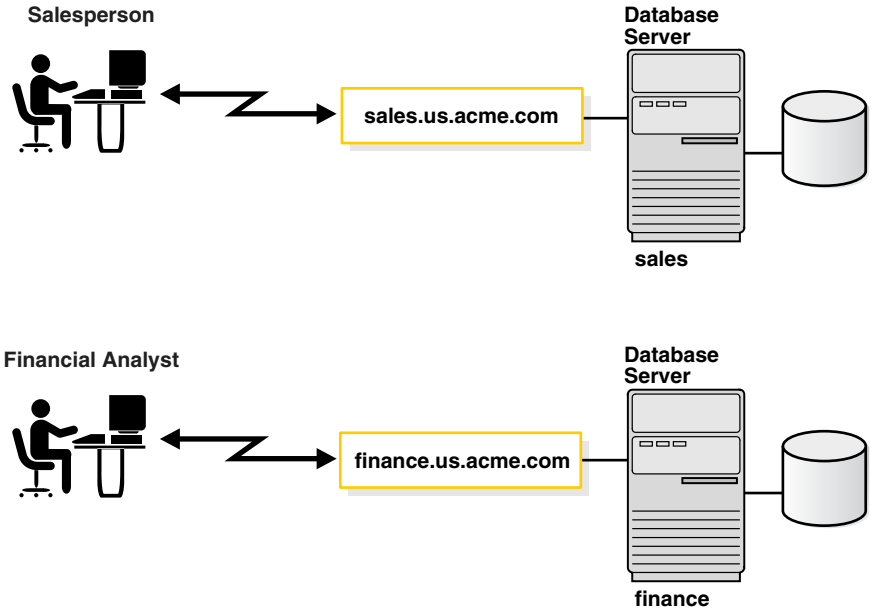
Database Instances

A database has at least one **instance**. An instance is comprised of a memory area called the **System Global Area (SGA)** and Oracle background processes. The memory and processes of an instance efficiently manage the associated database's data and serve the database users.

> **Note:** An instance also manages other services, such as **Oracle XML DB**.

Figure 3–3 shows two database instances, `sales` and `finance`, associated with their respective databases.

Figure 3–3 One Instance for Each Database

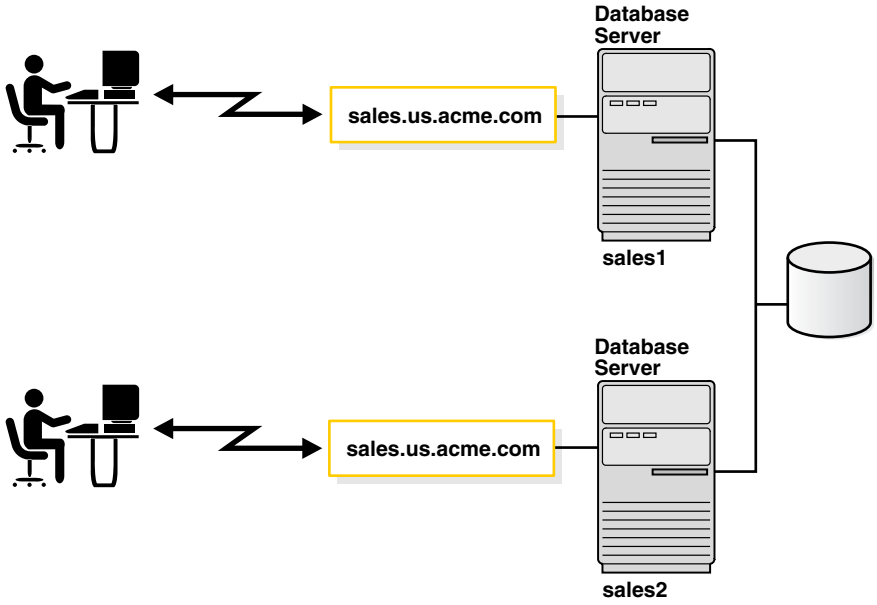


Like services, instances are identified by an **instance name**, `sales` and `finance` in this example. The instance name is specified by the `INSTANCE_NAME` initialization parameter. The instance name defaults to the **Oracle System Identifier (SID)** of the database instance.

Some hardware architectures allow multiple computers to share access to data, software, or peripheral devices. **Oracle Real Application Clusters** can take advantage of such architecture by running multiple instances on different computers that share a single physical database.

Figure 3–4 shows an Oracle Real Application Clusters configuration. In this example, two instances, `sales1` and `sales2`, are associated with one database service, `sales.us.acme.com`.

Figure 3–4 Multiple Instances Associated with a Database



Service Accessibility

To connect to a database service, clients use a **connect descriptor** that provides the location of the database and the name of the database service. The following example shows a connect descriptor that enables clients to connect to a database service called `sales.us.acme.com`.

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
  (CONNECT_DATA=
    (SERVICE_NAME=sales.us.acme.com)))
```

Protocol Address

The address portion of the connect descriptor is the protocol address of the **listener**. To connect to a database service, clients first contact a listener process that typically resides on the database server. The listener receives incoming client connection requests and hands these requests to the database server. After the connection is established, the client and database server communicate directly.

Much like a business address, the listener is configured to accept requests from clients at a **protocol address**. This address defines the protocol the listener is listening on and any other protocol specific information. For example, the listener could be configured to listen at the following protocol address:

```
(DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521)))
```

The preceding example shows a TCP/IP protocol address that specifies the host of the listener and a port number. Clients configured with this same protocol address can send connection requests to this listener.

Connect Data

The connect descriptor also specifies the database service name with which clients seek to establish a connection. The listener knows which services for which it can handle connection requests, because an Oracle database dynamically registers this information with the listener. This process of registration is called **service registration**. Registration also provides the listener with information about the database instances and the **service handler**s available for each instance. Service handlers act as connection points to an Oracle database server. A service handler can be a **dispatcher** or a **dedicated server**.

Instance Name

If connecting to a specific instance of the database is required, clients can also specify the `INSTANCE_NAME` of a particular instance in the connect descriptor. This feature can be useful if you have an Oracle Real Application Clusters configuration. For example, the following connect descriptor specifies an instance name of `sales1` that is associated with `sales.us.acme.com`.

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
  (CONNECT_DATA=
    (SERVICE_NAME=sales.us.acme.com)
    (INSTANCE_NAME=sales1))
```

Service Handlers

Alternatively, clients that always want to use a particular service handler type can use a connect descriptor that specifies the service handler type. In the following example, a connect descriptor is configured to use a dispatcher for a shared server configuration, as indicated by `(SERVER=shared)`.

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
  (CONNECT_DATA=
    (SERVICE_NAME=sales.us.acme.com)
     (SERVER=shared))
```

If you want the client to use a dedicated server, you can specify `(SERVER=dedicated)` in place of `(SERVER=shared)`. If the `SERVER` parameter is not set, shared server configuration is assumed. However, the client will use a dedicated server if no dispatchers are available.

If database resident connection pooling is enabled on the server, then you can specify `(SERVER=pooled)` to get a connection from the pool. If database resident connection pooling is not enabled on the server, then the client request is rejected.

When the listener receives the client request, it selects one of the service handlers that were previously registered. Depending on the type of handler selected, the communication protocol used, and the operating system of the database server, the listener performs one of the following actions:

- Hands the connect request directly off to a dispatcher.
- Sends a redirect message back to the client with the location of the dispatcher or dedicated server process. The client then connects directly to the dispatcher or dedicated server process.
- Spawns a dedicated server process and passes the client connection to the dedicated server process.

After the listener has completed the connection operation for the client, the client communicates with the Oracle database without the listener's involvement. The listener resumes listening for incoming network sessions.

See Also:

- "Service Handlers" on page 3-8 for a description of these service handler types
- "Listener Architecture" on page 5-7 for a discussion of how the listener works with service handlers
- "Database Resident Connection Pooling" on page 3-11
- *Oracle Call Interface Programmer's Guide* and *Oracle Database Administrator's Guide* for more information about enabling and configuring database resident connection pooling

Enhanced Service Accessibility with Multiple Listeners

For some configurations, such as Oracle Real Application Clusters, multiple listeners on multiple nodes can be configured to handle client connection requests for the same database service. In the following example, `sales.us.acme.com` can connect to `sales.us.acme.com` using listeners on either `sales1-server` or `sales2-server`.

```
(DESCRIPTION=
  (ADDRESS_LIST=
    (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521))
    (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521)))
  (CONNECT_DATA=
    (SERVICE_NAME=sales.us.acme.com)))
```

A multiple-listener configuration also enables you to leverage the following failover and load balancing features, either singly or in combination with each other:

- Connect-Time Failover
- Transparent Application Failover
- Client Load Balancing
- Connection Load Balancing

Connect-Time Failover

The **connect-time failover** enables clients to connect to another listener if the initial connection to the first listener fails. The number of listener protocol addresses determines how many listeners are tried. Without connect-time failover, Oracle Net attempts a connection with only one listener.

Transparent Application Failover

The **Transparent Application Failover (TAF)** feature is a runtime failover for High Availability environments, such as Oracle Real Application Clusters. TAF fails over and reestablishes application-to-service connections. It enables client applications to automatically reconnect to the database if the connection fails and, optionally, resume a `SELECT` statement that was in progress. The reconnection happens automatically from within the **Oracle Call Interface (OCI)** library.

Client Load Balancing

The **client load balancing** feature enables clients to randomize connection requests among the listeners. Oracle Net progresses through the list of protocol addresses in a random sequence, balancing the load on the various listeners. Without client load balancing, Oracle Net progresses through the list of protocol addresses sequentially until one succeeds.

Connection Load Balancing

The **connection load balancing** feature improves connection performance by balancing the number of active connections among multiple dispatchers. In a single-instance environment, the listener selects the least loaded dispatcher to handle the incoming client requests. In an Oracle Real Application Clusters environment, connection load balancing also has the capability to balance the number of active connections among multiple instances.

Due to dynamic service registration, a listener is always aware of all instances and dispatchers regardless of their location. Depending on the load information, a listener decides which instance and, if shared server is configured, to which dispatcher to send the incoming client request.

In a shared server configuration, a listener selects a dispatcher in the following order:

1. Least-loaded node
2. Least-loaded instance
3. Least-loaded dispatcher for that instance

In a dedicated server configuration, a listener selects an instance in the following order:

1. Least loaded node
2. Least loaded instance

If a database service has multiple instances on multiple nodes, the listener chooses the least loaded instance on the least loaded node. If shared server is configured, then the least loaded dispatcher of the selected instance is chosen.

Service Handlers

This section includes the following topics:

- Dispatchers
- Dedicated Server Processes
- Database Resident Connection Pooling

Dispatchers

The shared server architecture uses a dispatcher process to direct client connections to a common request queue. An idle shared server process from a shared pool of server processes picks up a request from the common queue. This approach enables a small pool of server processes to serve a large number of clients. A significant advantage of the shared server model over the dedicated server model is reduced system resources, enabling support of an increased number of users.

The listener uses the dispatcher as a type of service handler to which it can direct client requests. When a client request arrives, the listener performs one of the following actions:

- Hands the connection request directly to a dispatcher.
- Issues a redirect message to the client, containing the protocol address of a dispatcher. The client then terminates the network session to the listener and establishes a network session to the dispatcher, using the network address provided in the redirect message.

The listener uses direct hand off whenever possible. Redirect messages are used, for example, when dispatchers are remote to the listener.

Figure 3–5 shows the listener handing a connection request directly off to a dispatcher.

1. The listener receives a client connection request.
2. The listener hands the connect request directly to the dispatcher.
3. The client is now connected to the dispatcher.

Figure 3–5 Direct Hand-Off to a Dispatcher

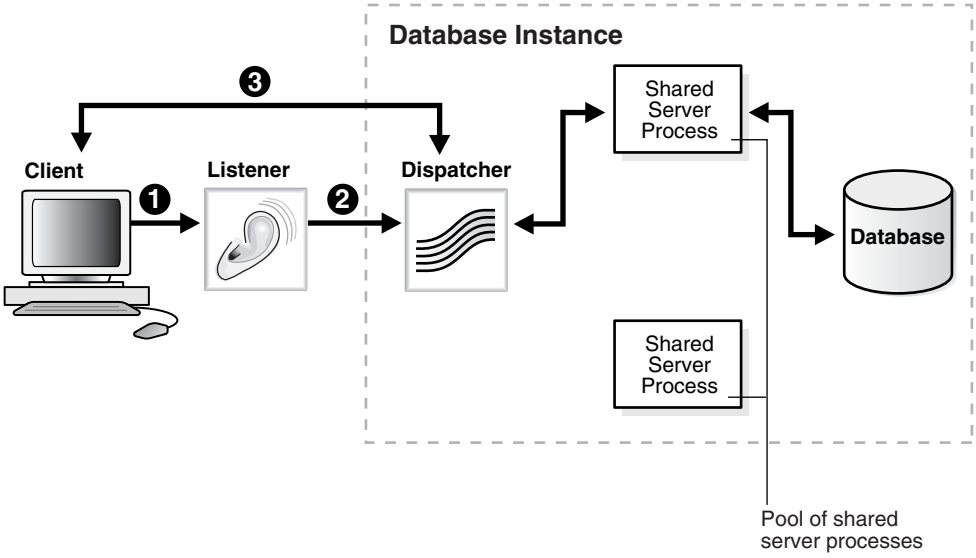
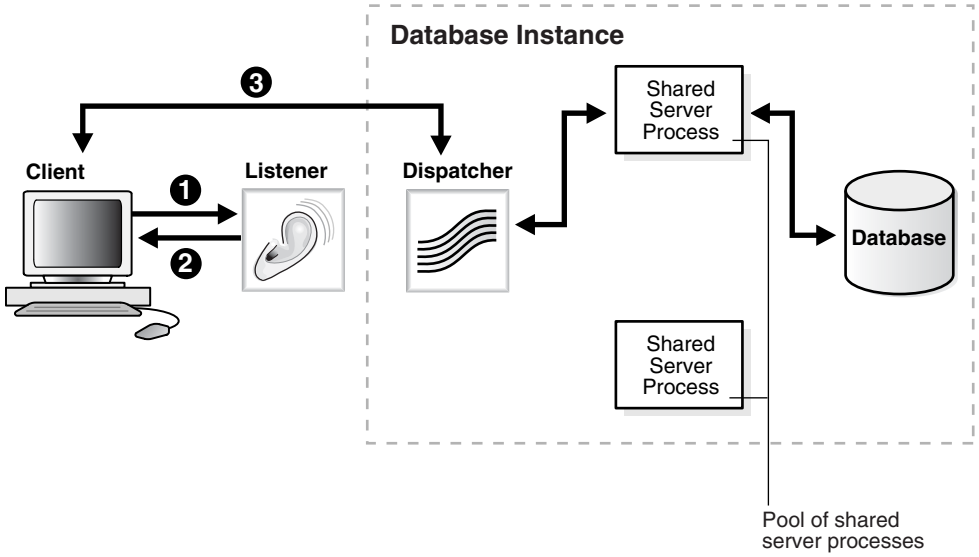


Figure 3–6 shows the role of a dispatcher in a redirected connection.

1. The listener receives a client connection request.
2. The listener provides the location of the dispatcher to the client in a redirect message.
3. The client connects directly to the dispatcher.

Figure 3–6 Redirected Connection to a Dispatcher



Dedicated Server Processes

In a dedicated server configuration, the listener starts a separate dedicated server process for each incoming client connection request dedicated to servicing the client. After the session is complete, the dedicated server process terminates. Because a dedicated server process has to be started for each connection, this configuration may require more system resources than shared server configurations.

A dedicated server process is a type of service handler that the listener starts when it receives a client request. To complete a client/server connection establishment, one of the following actions occurs:

- The dedicated server inherits the connection request from the listener.
- The dedicated server informs the listener of its listening protocol address. The listener passes the protocol address to the client in a redirect message and terminates the connection. The client connects to the dedicated server directly using the protocol address.

> **Note:** One of the options is selected based on the operating system and the transport protocol.

If the client and database exist on the same computer, a client connection can be passed directly to a dedicated server process without going through the listener. The application initiating the session spawns a dedicated server process for the connection request. This happens automatically if the application that is used to start the database is on the same computer as the database.

> **Note:** In order for remote clients to connect to dedicated servers, the listener and the database instance must be running on the same computer.

Figure 3–7 shows the listener passing a client connection request to a dedicated server process.

1. The listener receives a client connection request.
2. The listener starts a dedicated server process, and the dedicated server inherits the connection request from the listener.
3. The client is now connected directly to the dedicated server.

Figure 3–7 Connection to a Dedicated Server Process

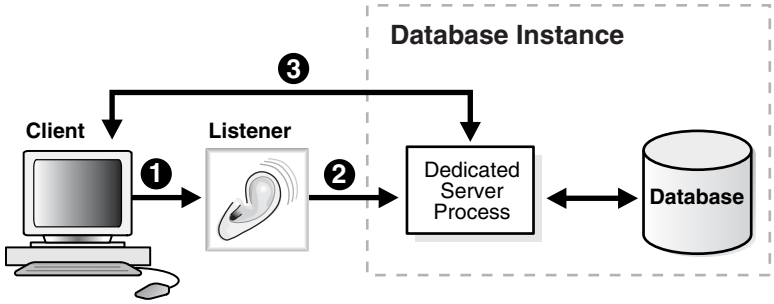
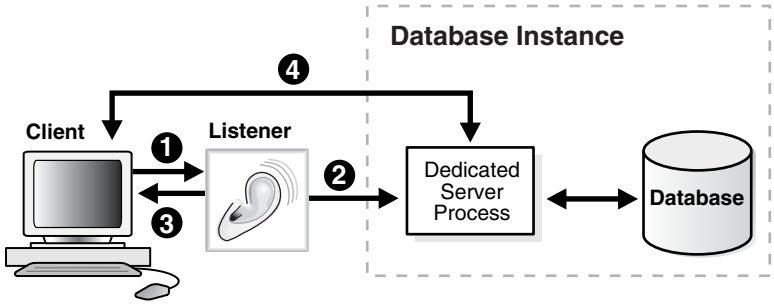


Figure 3–8 shows the role of a dedicated server in a redirected connection.

1. The listener receives a client connection request.
2. The listener starts a dedicated server process.
3. The listener provides the location of the dedicated server process to the client in a redirect message.
4. The client connects directly to the dedicated server.

Figure 3–8 Redirected Connection to a Dedicated Server Process



Database Resident Connection Pooling

Applications may be deployed in the following ways:

- As multiple processes
- On multiple hosts
- As multiple processes on multiple hosts

Database resident connection pooling provides pooling for **dedicated connection**s across client applications and processes. This feature is useful for applications that must maintain persistent connections to the database and optimize server resources (such as memory).

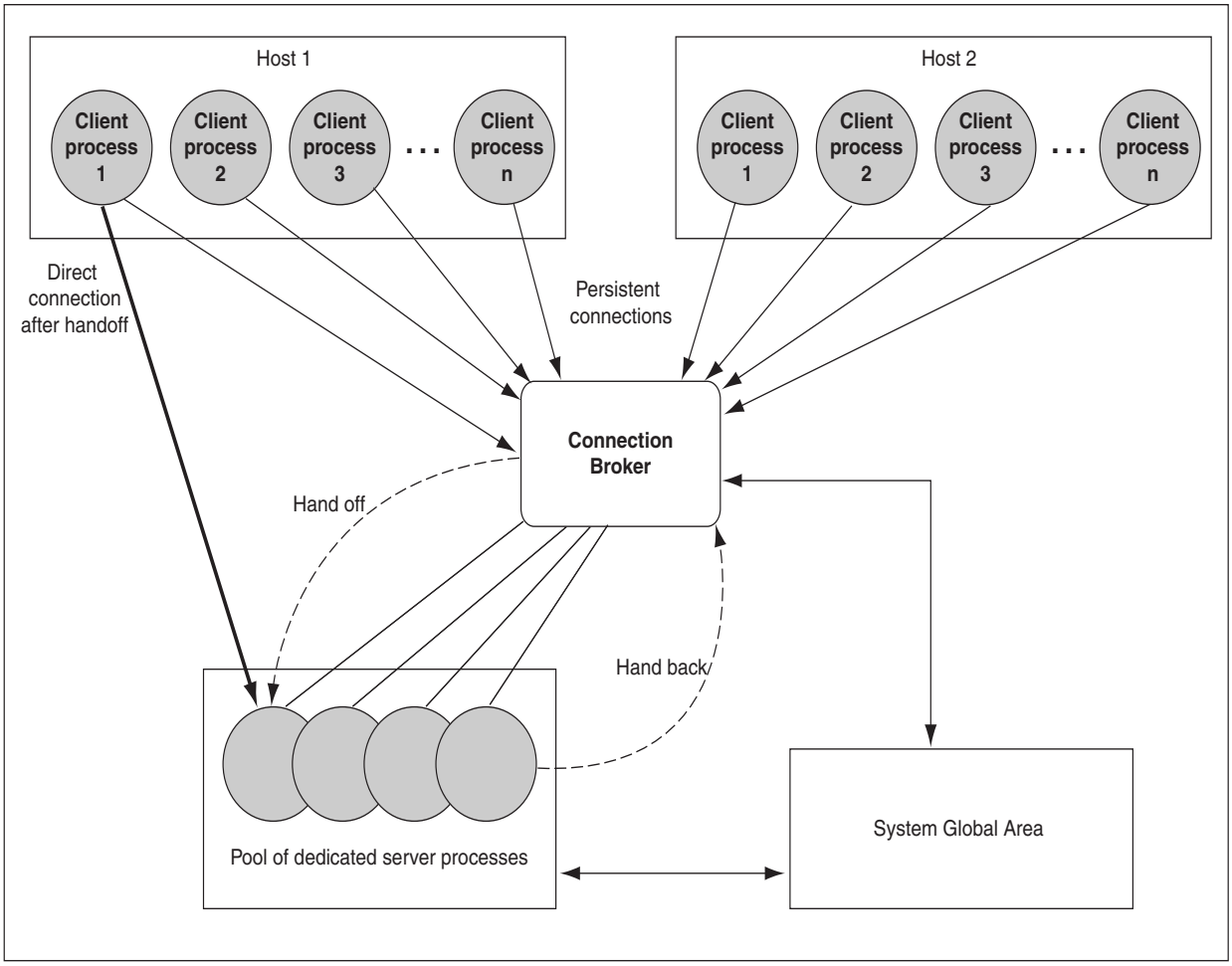
Clients obtaining connections out of the database resident connection pool are persistently connected to a background process—the connection broker—instead of to

the dedicated servers. The connection broker implements the pool functionality and performs the multiplexing of inbound connections from the clients to a pool of dedicated servers with sessions.

When a client needs to perform database work, the connection broker picks up a dedicated server from the pool and assigns it to the client. Subsequently, the client is directly connected to the dedicated server until the request is served. After the server finishes processing the client request, the server goes back into the pool and the connection from the client is restored to the connection broker.

Figure 3–9 graphically illustrates the process.

Figure 3–9 Pool of Dedicated Server Processes Handling Connections Through the Connection Broker Process



Naming

Users initiate a connection request by providing a **connect string**. A connect string includes a user name and password, along with a **connect identifier**. A connect identifier can be the connect descriptor itself or a name that resolves to a connect descriptor.

One of the most common connect identifiers is a **net service name**, a simple name for a service. The following example of a `CONNECT` command uses a connect string that uses a complete connect descriptor as the connect identifier:

```
SQL> CONNECT hr@(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)
(HOST=sales-server1)(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)))

Enter password: password
```

The following example of a `CONNECT` command uses a connect string that uses net service name `sales` as the connect identifier:

```
SQL> CONNECT hr@sales

Enter password: password
```

When net service name `sales` is used, connection processing takes place by first mapping sales to the connect descriptor. This mapped information is stored in one or more repositories of information that are accessed with **naming method**s.

The process for establishing a client session with the aid of a naming method is as follows:

1. The client initiates a connect request by providing a connect identifier.
2. The connect identifier is resolved to a connect descriptor by a naming method. This information is returned to the client.
3. The client makes the connection request to the address provided in the connect descriptor.
4. A listener receives the request and directs it to the appropriate database server.
5. The connection is accepted by the database server.

Oracle Net provides support for following naming methods:

- Local Naming
- Directory Naming
- Easy Connect Naming
- External Naming

Note: Besides connect descriptors, you can use naming methods to map a name to a protocol address or protocol address list.

Local Naming

The **local naming** method stores net service names and their connect descriptors in a localized configuration file named `tnsnames.ora`.

See Also: "Configuring the Local Naming Method" on page 8-8

Directory Naming

The **directory naming** method stores connect identifiers in a centralized LDAP-compliant **directory server** to access a database service.

See Also: "Configuring the Directory Naming Method" on page 8-14

Easy Connect Naming

The easy connect naming method enables clients to connect to an Oracle database server by using a TCP/IP connect string consisting of a host name and optional port and service name:

```
CONNECT username@[//]host[:port][/service_name][:server][/instance_name]
```

The easy naming method requires no configuration.

See Also: "Using the Easy Connect Naming Method" on page 8-3

External Naming

The **external naming** method stores net service names in a supported non-Oracle naming service. These supported third-party services include:

- **Network Information Service (NIS)** External Naming
- Distributed Computing Environment (DCE) **Cell Directory Services (CDS)**

See Also: "Configuring External Naming Methods" on page 8-25

4

Configuration Management Concepts

This chapter describes how configuration information for Oracle Net Services can be stored in localized configuration files or centralized in a directory server.

The topics covered include:

- Configuration Models
- Localized Configuration File Support
- Directory Server Support

Configuration Models

Configuration information can be stored in a localized configuration file or a centralized repository, as described in the Table 4–1.

Table 4–1 Oracle Net Configuration Models

Network Configuration Model	Description
Localized management	Network address information stored in `tnsnames.ora` files on each computer in the network.
Centralized management	Network address information is stored in centralized directory services, including a LDAP-compliant **directory server**.

Localized Configuration File Support

Depending on the configuration model used, network computers can be configured with the files described in Table 4–2.

Table 4–2 Oracle Net Configuration Files

Configuration File	Description
`cman.ora`	Located on the computer where Oracle Connection Manager runs, this configuration file includes: ■ A listening endpoint ■ Access control rule list ■ Parameter list Each Oracle Connection Manager configuration is encapsulated within a single NV string, which consists of the components just described.

Table 4–2 (Cont.) Oracle Net Configuration Files

Configuration File	Description
`listener.ora`	Located on the database server, this configuration file for the **listener** may include: ■ Protocol addresses it is accepting connection requests on ■ Database and nondatabase services it is listening for ■ Control parameters used by the listener
`sqlnet.ora`	Located on client and database server computer, this file may include: ■ Client domain to append to unqualified service names or net service names ■ Order of naming methods the client should use when resolving a name ■ Logging and tracing features to use ■ Route of connections ■ External naming parameters ■ Oracle Advanced Security parameters ■ Database access control parameters
`tnsnames.ora`	Located primarily on the clients, this file contains **net service name**s mapped to **connect descriptor**s. This file is used for the **local naming** method.

Configuration files are typically created in `$ORACLE_HOME/network/admin` on UNIX operating systems and `%ORACLE_HOME%\network\admin` on Windows operating systems. However, configuration files can be created in a variety of places, because Oracle Net searches for the configuration files in a variety of places.

The search order for `sqlnet.ora` is as follows:

1. The directory specified by the `TNS_ADMIN` environment variable
2. The `$ORACLE_HOME/network/admin` directory on UNIX operating systems and the `%ORACLE_HOME%\network\admin` directory on Windows operating systems

The search order for `cman.ora`, `listener.ora`, and `tnsnames.ora` is as follows:

1. The directory specified by the `TNS_ADMIN` environment variable
2. On UNIX operating systems, the global configuration directory

   For example, on the Solaris Operating System, this directory is `/var/opt/oracle`.
3. The `$ORACLE_HOME/network/admin` directory on UNIX operating systems and the `%ORACLE_HOME%\network\admin` directory on Windows operating systems.

Note: On Windows, `TNS_ADMIN` is used if it is set in the environment of the process; if `TNS_ADMIN` is not defined in the environment, or if the process is a service or other such process which does not have an environment, Windows scans the Registry for `TNS_ADMIN`.

See Also: Oracle operating system-specific documentation

Directory Server Support

Today, network information is stored in multiple systems and in multiple directory formats. With new requirements for Internet computing and new e-business technologies, a common repository infrastructure is needed as a foundation for management and configuration of all data and resources. This kind of infrastructure reduces the cost of managing and configuring resources in a network.

Support of **Oracle Internet Directory** provides a centralized vehicle for managing and configuring a distributed Oracle network. The directory server can replace client-side and server-side localized `tnsnames.ora` files.

This section contains these topics:

- Directory Naming Overview
- Naming Configuration Storage in a Directory Server
- Net Service Alias Entries
- Directory Entries
- Adding or Modifying Entries in the Directory Server
- Client Connections Using Directory Naming
- Oracle Net Configuration and Directory Server Design
- Limitations of Directory Naming Support with Microsoft Active Directory

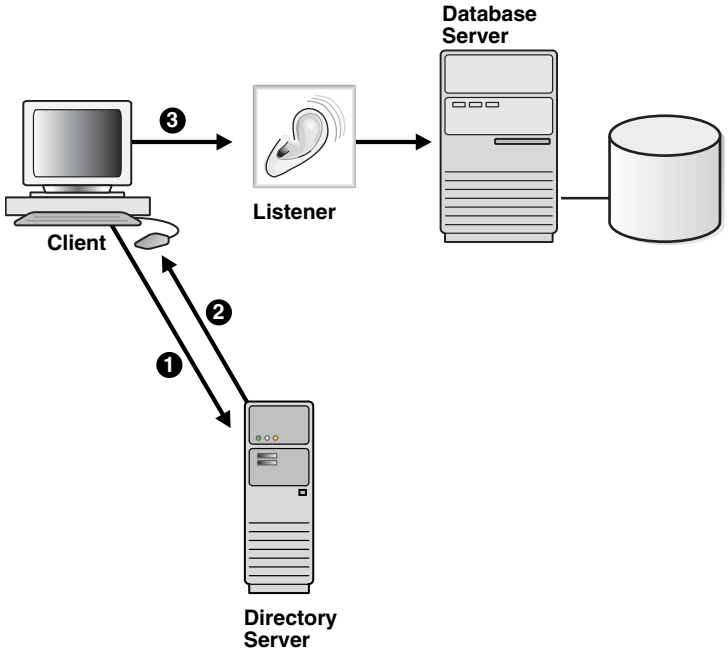
Directory Naming Overview

Oracle Net Services use a centralized directory server as one of the primary methods for storage of **connect identifier**s. Clients can use the connect identifiers in their connect string. The directory server resolves the connect identifier to a connect descriptor that is passed back to the client. This feature is called **directory naming**

Figure 4–1 shows a client resolving a connect identifier through a directory server.

1. The client contacts the directory server to resolve a connect identifier to a connect descriptor.
2. The directory server resolves the connect identifier and retrieves the connect descriptor for the client.
3. The client sends the connection request to the listener, using the connect descriptor.

Figure 4–1 Client Using a Directory Server to Resolve a Connect Identifier

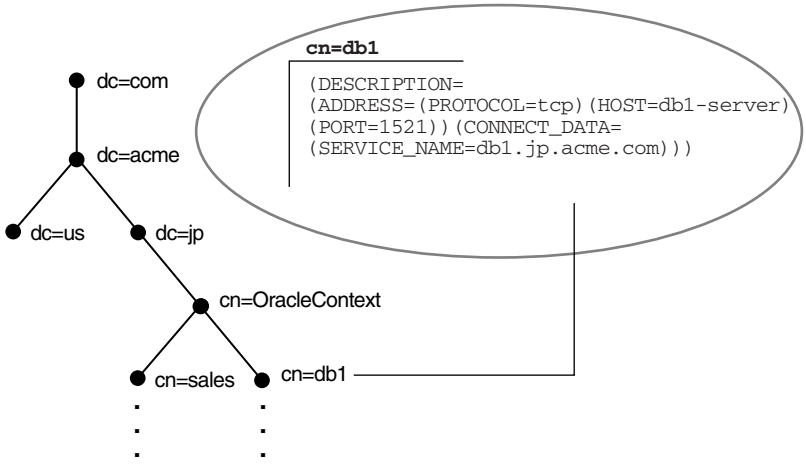


Note: **Java Database Connectivity (JDBC) Drivers** support directory naming. See the *Oracle Database JDBC Developer's Guide and Reference* for further information.

Naming Configuration Storage in a Directory Server

Directory servers store information in a tree structure called a **directory information tree (DIT)**. Each node in the tree is called an **entry**. Oracle Net Services makes use of both the tree structure and specific entries in the tree. For example, consider Figure 4–2.

Figure 4–2 Database Service and Net Service Entries in a DIT



The `cn=sales` and `cn=db1` entries represent a net service name and a database service, respectively. Additional entries under `cn=sales` and `cn=db1` contain the connect descriptor information. These entries are not represented in the graphic. The

cn=sales and cn=db1 entries enable clients to connect to the database using connect strings CONNECT *username*@sales and CONNECT *username*@db1.

Each entry is uniquely identified by a **distinguished name (DN)**. The DN tells you exactly where the entry resides in the directory server's hierarchy. The DN for db1 is dn:cn=db1,cn=OracleContext,dc=jp,dc=acme,dc=com, and the DN for sales is dn:cn=sales,cn=OracleContext,dc=jp,dc=acme,dc=com. Note that the format of a DN places the lowest component of the DIT to the left, then moves progressively up the DIT. Each DN is made up of a sequence of **relative distinguished names (RDNs)**, much the way a directory path contains a sequence of directories. In the entry for db1, the RDN is cn=db1. An entry is made up of a set of **attribute**s. For example, in cn=db1, cn is one of the entry's attributes. The attribute, along with its value, uniquely identifies the entry.

Notice that db1 and sales reside under cn=OracleContext. This entry is a special RDN called an **Oracle Context**. The entries under the Oracle Context support various directory-enabled features, including directory naming.

During directory usage configuration, you establish a default Oracle Context. Clients use this Oracle Context as the default location to look up connect identifiers in the directory server. With Oracle Internet Directory, an Oracle Context located at the root of the DIT, with DN of dn:cn=OracleContext, points to a default Oracle Context in an **identity management realm**. An identity management realm is a collection of identities governed by the same administrative policies. This Oracle Context is referred to as a **realm Oracle Context**. Unless configured to use another Oracle Context, clients use this realm-specific Oracle Context as the default Oracle Context.

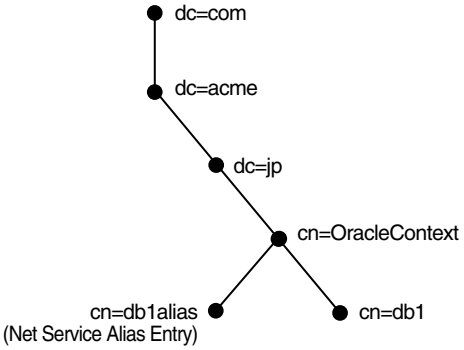
The default Oracle Context affects the connect string. For example, if a client needs to access the db1 and sales entry frequently, a reasonable default Oracle Context would be dc=jp,dc=acme,dc=com. cn=OracleContext does not have to be explicitly specified in the connect string. If a client's directory entry does not match the directory entry where the service is located, then the client must specify an entry's absolute name in the connect string, as described in "Client Connections Using Directory Naming" on page 4-9.

> **See Also:** *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory* for further information about an identity management realm

Net Service Alias Entries

In addition to database service and net service name entries, directory naming enables you to create **net service alias** entries. A net service alias is an alternative name for a net service name or database service. A net service alias entry does not have connect descriptor information. Instead, it only references the location of the entry for which it is an alias. When a client requests a directory lookup of a net service alias, the directory determines that the entry is a net service alias and completes the lookup as if it is the referenced entry. For example, in Figure 4–3, a net service alias of db1alias is created for a database service of db1. When db1alias is used to connect to a database service, as in CONNECT *username*@db1alias, it will actually resolve to and use the connect descriptor information for db1.

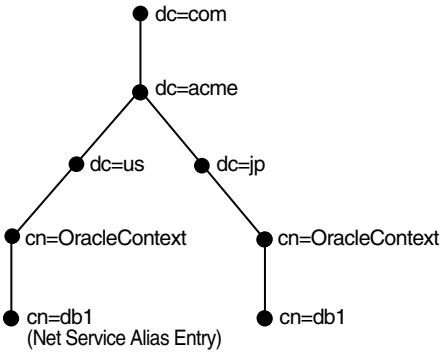
Figure 4–3 Net Service Alias db1alias in a Directory Server



There are several uses for using net service aliases. As shown in Figure 4–3, a net service alias can be useful as a way for clients to refer to a net service name by another name. Another use is to have a net service alias in one Oracle Context for a database service or net service name in a different Oracle Context. This enables a database service or net service name to be defined once in the directory server, but referred to by clients that use other Oracle Contexts.

In Figure 4–4, database service `db1` resides in `dc=jp,dc=acme,dc=com`. A net service alias named `db1` is created in `dc=us,dc=acme,dc=com`. This enables clients in both Japan and the United States to use the connect string `CONNECT` *`username`*`@db1` as opposed to clients in the United States needing to specify `CONNECT` *`username`*`@db1.jp.acme.com`.

Figure 4–4 Net Service Alias db1 in a Directory Server



Directory Entries

DITs are commonly structured using:

- A Domain Name Space (DNS) structure
- A geographical and organization structure

Other structures are also permitted, but Oracle Corporation provides support for these structures.

Figure 4–5 shows a DIT structured according to DNS domain components.

Figure 4–5 Domain Component DIT

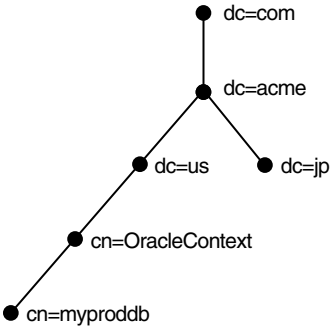
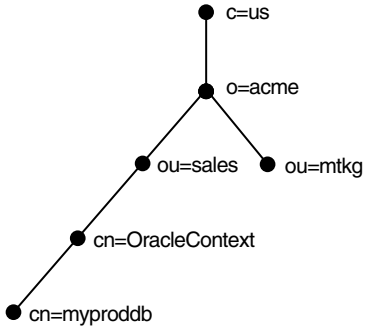


Figure 4–6 shows a DIT structured according to country, organization, and organizational units. This structure is commonly referred to as an X.500 DIT.

Figure 4–6 X.500 Style DIT

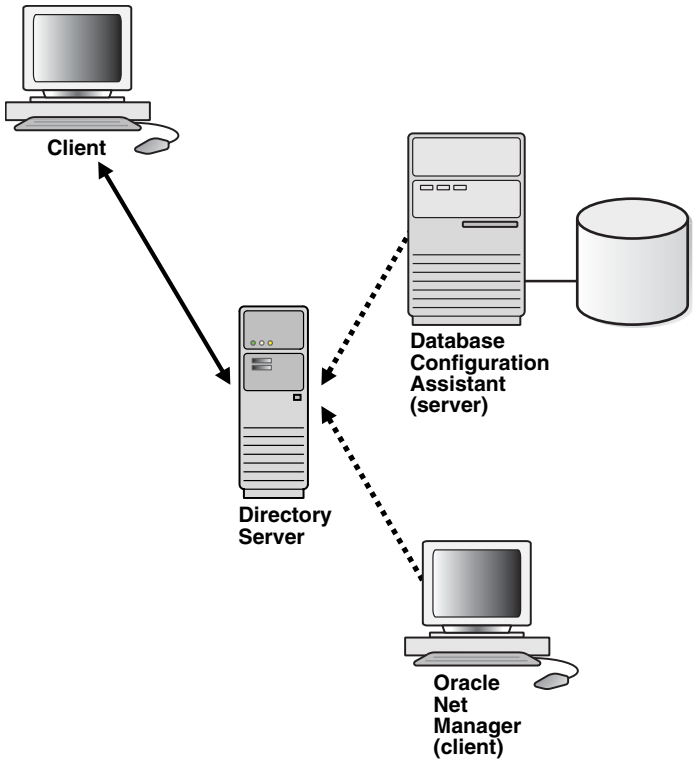


Adding or Modifying Entries in the Directory Server

Database Configuration Assistant creates database service entries during or after some modes of installation. You can then use **Oracle Enterprise Manager** or **Oracle Net Manager** to modify the Oracle Net attributes of the database service entries. You can also use these tools to create net service name and net service alias entries.

Figure 4–7 shows how the tools interface with the directory server.

Figure 4–7 Creating Entries in the Directory Server with Applications



Note: Oracle Enterprise Manager is supported but not represented in this figure.

Clients that are configured for directory naming, as described in "Client Connections Using Directory Naming" on page 4-9, can connect to a database using entries created by these configuration tools.

To use these configuration tools to add entries, a DIT structure containing a **root Oracle Context** and identity management realm must exist. The directory administrator creates this structure with Oracle Internet Directory Configuration Assistant. For some deployments, the directory administrator may need to create additional Oracle Contexts.

To create directory naming entries with the Oracle Enterprise Manager or Oracle Net Manager, you must be a member of the following groups:

- `OracleDBCreators` group (`cn=OracleDBCreators,cn=OracleContext...`) or the `OracleContextAdmins` group (`cn=OracleContextAdmins,cn=Groups,cn=OracleContext...`) to create a database service entry with Database Configuration Assistant
- `OracleNetAdmins` group (`cn=OracleNetAdmins,cn=OracleContext...`) or the `OracleContextAdmins` group to create net service names or net service aliases with Oracle Net Manager

The directory user that created the Oracle Context is automatically added to these groups. Other users can be added to these groups by the directory administrator.

The `OracleContextAdmins` group is a super-user group for the Oracle Context. Members of the `OracleContextAdmins` group can add all supported types of entries to the Oracle Context.

See Also:

- "Configuring the Directory Naming Method" on page 8-14 for further information about using Oracle Net Manager
- *Oracle Database Administrator's Guide* for further information about how to register a database service with Database Configuration Assistant

Client Connections Using Directory Naming

Most clients only need to perform name lookups in the directory server. To perform a lookup, the directory server must allow anonymous authentication. Directory servers usually do this by default.

To look up entries, a client must be able to find the directory server in which that entry resides. Clients locate a directory in one of two ways:

- Dynamically, by using DNS. In this case, the directory server location information is stored and managed in a central domain name server, and the client, at request processing time, retrieves this information from the DNS server dynamically.
- Statically, in a directory server usage file (`ldap.ora`) created by Oracle Internet Directory Configuration Assistant and stored on the client host

Once a directory is found, clients are directed to the realm Oracle Context from the root Oracle Context.

In the same way they might use other naming methods, clients make connections to a database using connect identifiers. A connect identifier can be a database service, net service name, or net service alias. These can be referred to by their common names, or they can require additional directory location information. The default Oracle Context determines how the connect identifier must be specified.

An entry may be identified in one of two ways:

- Using the Entry's Relative Name
- Using the Entry's Absolute Name

Note: The **JDBC OCI Driver** supports both relative and absolute naming. The **JDBC Thin Driver** supports absolute naming only when the complete DN is used. See the *Oracle Database JDBC Developer's Guide and Reference* for further information.

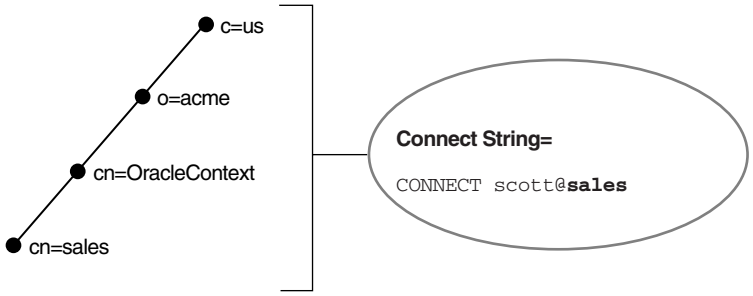
See Also: *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory* for further information about clients locate a directory

Using the Entry's Relative Name

In the following example, an entry is identified by its relative name, and the service can be referred to by its common name. A relative name can be used if the entry is in the same Oracle Context that was configured to be the default Oracle Context for the client's Oracle home.

Consider a directory server that contains an entry for a database called `sales` with a DN of `dn:cn=sales,cn=OracleContext,o=acme,c=us`, as shown in Figure 4–8. If the client is configured with a default realm Oracle Context of `cn=OracleContext,o=acme,c=us`, then the connect identifier can simply be `sales`.

Figure 4–8 Relative Naming



Using the Entry's Absolute Name

Consider the same directory structure as shown Figure 4–8, but with the client's Oracle home configured with a default realm Oracle Context of `cn=OracleContext,o=acme,c=jp`.

Because the client is configured with a default Oracle Context that does not match the location of `sales` in the directory server, a connect string that uses `sales` does not work. Instead, the client must specifically identify the location of `sales`, which can be done in one of two ways:

- The entry's complete DN can be used in the connect string, for example:

  ```
  CONNECT username@"cn=sales,cn=OracleContext,o=acme,c=us"
  Enter password: password
  ```

  Many applications do not support the use of a DN.

- The entry can be referred to by a fully-qualified name, a name that includes the name of the object and its location in the directory server, for example:

  ```
  CONNECT username@sales.acme.us
  Enter password: password
  ```

> **Note:** JDBC Thin drivers support absolute naming only when the complete DN is used.

See Also: "Absolute Name Specification for Directory Naming" on page 15-5 for further information about absolute names

Oracle Net Configuration and Directory Server Design

If you are responsible for designing directory servers for directory naming, consider the following issues:

- Performance
- Security
- Schema

Performance

Connect identifiers are stored in a directory server for all clients to access. Depending on the number of clients, there can be a significant load on a directory server.

During a connect identifier lookup, a name is searched under a specific Oracle Context. Because of the scope of the lookup, you probably want users to experience relatively quick performance so that the database connect time is not affected. Users may begin to notice slow connect times if lookups takes more than one second.

You can resolve performance problems by changing the network topology or implementing replication.

> **See Also:** Directory server vendor documentation for details on resolving performance issues

Security

Because administrative clients can create and modify entries in the directory server, security is essential. This section covers the following security-related topics:

- Authentication Methods
- Access Control Lists

Authentication Methods Clients use two different methods of authentication depending upon the task that is to be performed.

- Clients that perform lookups for information in the directory server typically use anonymous authentication.

  In Oracle Database 11*g*, it is possible to configure clients to authenticate their LDAP bind during name lookup. Sites that need to either protect their net service data or disable anonymous binds to the directory, or both, must configure their clients to use wallets in order to authenticate during name lookup. These clients require setting the following two new parameters in the `sqlnet.ora` file:

  ```
  names.ldap_authenticate_bind = TRUE
  wallet_location = location_value
  ```

- Clients that add or modify entries in a directory must authenticate with the directory server. Database Configuration Assistant or Oracle Net Manager may be used to add or modify the entries. Only authenticated users with proper privileges can modify entries. Use one of the following authentication methods:

  – Simple Authentication

    The client identifies itself to the directory server by means of a DN and a password, which are sent in the clear over the network. The server verifies that the DN and password sent by the client match the DN and password stored in the directory server.

  – Strong Authentication

    Directories provide strong authentication by using public-key encryption available with Secure Sockets Layer (SSL). In public-key encryption, the sender of a message encrypts the message with the public key of the recipient. Upon delivery, the recipient decrypts the message using the recipient's private key.

  – Configuration on Systems that Require Authenticated Name Lookups

Access Control Lists Authentication is used with **access control lists (ACLs)** to determine whether clients can read, modify, or add information in the directory server.

> **See Also:** Documentation from your directory server vendor for information about how to set ACLs on directory entry

ACLs specify the following:

- The entries that the user can access
- The authentication method used to access the entry
- The access rights, or what the user can do with the object (read/write)

ACLs are established for a group of users. During Oracle Context creation, the `OracleDBCreators`, `OracleNetAdmins`, and `OracleContextAdmins` groups are created.

The user who creates the Oracle Context with Oracle Net Configuration Assistant is automatically added as the first member of these groups.

Table 4–3 describes ACL requirements for these groups and anonymous users and their relation to Oracle Net entries in the directory server.

Table 4–3 LDAP Directory User Groups

Group	ACL Requirements
Anonymous users	All Oracle Net attributes and objects in the directory server have read access for the anonymous user. Read access of these objects for anonymous is also applied to the Oracle Context. This enables anonymous users to browse directory naming entries contained within the `cn=OracleContext` RDN. This does not include objects used for enterprise user security. Oracle Net Configuration Assistant sets up this access right during client installation.
`OracleContextAdmins` group users	Members of `OracleContextAdmins` (`cn=OracleContextAdmins,cn=Groups,cn=OracleContext,...`) have create, modify, and read access to all directory naming objects. Oracle Net Configuration Assistant establishes these access rights for this group during Oracle Context creation. In addition to the Oracle Context creator, other users can be added to this group by the directory administrator with Oracle Enterprise Manager.
`OracleDBCreators` group users	Members of `OracleDBCreators` (`cn=OracleDBCreators,cn=OracleContext,...`) have create and read access to database service objects and attributes. Oracle Net Configuration Assistant establishes these access rights for this group during Oracle Context creation. In addition to the Oracle Context creator, other users can be added to this group by the directory administrator with Oracle Enterprise Manager. **See Also:** *Oracle Database Advanced Security Administrator's Guide* for further information about the `OracleDBCreators` group

Table 4–3 (Cont.) LDAP Directory User Groups

Group	ACL Requirements
`OracleNetAdmins` group users	Members of `OracleNetAdmins` (`cn=OracleOracleNetAdmins,cn=OracleContext,...`) have create, modify, and read access to directory naming objects and attributes. Oracle Net Configuration Assistant establishes these access rights for this group during Oracle Context creation. In addition to the Oracle Context creator, other users can be added to this group by the directory administrator. **See Also:** "Administering the OracleNetAdmins Group" on page 8-19 for information on adding users to the `OracleNetAdmins` group

In situations where a high degree of security is desired for lookup or read-access to Oracle Net Services name and related data, administrators can define additional read-access control for some or all of this data. Such ACL definitions can prevent *anonymous* users from reading the Oracle Net Services data. If read-access to Oracle Net Services data is restricted, clients must use authenticated binds in order to do name lookup.

> **See Also:** "Authentication Methods" on page 4-11 for information about configuring clients to use authenticated binds for name lookup

There are currently no predefined groups or procedures in Oracle's configuration tools for defining read-access restrictions on this data, so administrators must use standard object management tools from their directory system in order to manually create any necessary groups and ACLs.

ACLs can be added to Oracle Net Services objects using `ldapmodify` and an LDIF-format file. The following is a simple example that restricts all access by a given user, `cn=user1`:

```
dn: cn=emn10,cn=oraclecontext,dc=stiger,dc=com
replace: orclentrylevelaci
orclentrylevelaci: access to attr=(*)
 by dn="cn=user1" (noread,nosearch,nowrite,nocompare)
```

This example illustrates the basic form of an ACL for a single object but this approach is not necessarily the best way to define access. Because the access definitions for objects are complex and may involve security properties which are inherited from parent nodes in the DIT, Oracle recommends that administrators refer to the relevant tools and documentation for the directory system they are using, and formulate or integrate access management for Oracle Net Services objects into a directory-wide policy and security implementation.

For Oracle Internet Directory directories, `oidadmin` has functionality to create users, groups, and also define ACLs for objects and general directory security.

> **See Also:**
>
> - *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory* for more information
> - Microsoft Active Directory documentation for Microsoft-specific information

Schema

Directories must be populated with the correct version of the Oracle schema before Oracle Contexts or a database service or net service name entry can be created. The Oracle schema defines the type of objects, called **object classes**, that can be stored in the directory server and their attributes. Table 4–4 lists the object classes for database service, net service name, and net service alias entries.

Table 4–4 Oracle Net Services LDAP Main Object Classes

Object Class	Description
`orclDbServer`	Defines the attributes for database service entries
`orclNetService`	Defines the attributes for net service name entries
`orclNetServiceAlias`	Defines the attributes for net service alias entries

Table 4–5 lists the object classes used by `orclDbServer`, `orclNetService`, and `orclNetServiceAlias`.

Table 4–5 Oracle Net Services LDAP Derived Object Classes

Object Class	Description
`orclNetAddress`	Defines a listener protocol address
`orclNetAddressList`	Defines a list of addresses
`orclNetDescription`	Specifies a connect descriptor containing the protocol address of the database and the connect information to the service
`orclNetDescriptionList`	Defines a list of connect descriptors

These object classes use attributes that specify the contents of connect descriptors.

> **See Also:** *Oracle Database Net Services Reference* for further information about these object classes and their attributes

Limitations of Directory Naming Support with Microsoft Active Directory

In addition to Oracle Internet Directory, directory naming support is also provided with **Microsoft Active Directory**. Note the following limitations:

- Oracle provides support for Microsoft Active Directory only on Windows operating systems. Therefore, client computers and the database server must also run on Windows operating systems to access or create entries in Microsoft Active Directory.
- The following features are not supported by Microsoft Active Directory:
  - Multiple Oracle Contexts

    Microsoft Active Directory can support only one Oracle Context.
  - Net service aliases

    You cannot create net service aliases in Microsoft Active Directory. However, you can create net service names.
  - Automatic client discovery of directory servers for clients

    You must statically configure directory server usage on the clients. The Oracle Internet Directory Configuration will not provide directory server usage for Microsoft Active Directory. You must use **Oracle Net Configuration Assistant**.

5

Architecture of Oracle Net Services

This chapter describes the architecture of **Oracle Net**, the **listener**, **shared server**, **dedicated server**, and **Oracle Connection Manager**.

This chapter contains these topics:

- Oracle Net Stack Communication Architecture
- Listener Architecture
- Database Server Process Architecture
- Oracle Connection Manager Architecture
- A Complete Architecture

> **See Also:** Chapter 1, "Introduction to Oracle Net Services" for an introductory level overview of Oracle Net architecture

Oracle Net Stack Communication Architecture

The primary function of Oracle Net is to establish and maintain connections between a client application and an Oracle database server. Oracle Net is comprised of several communication layers that enable clients and database servers to share, modify, and manipulate data.

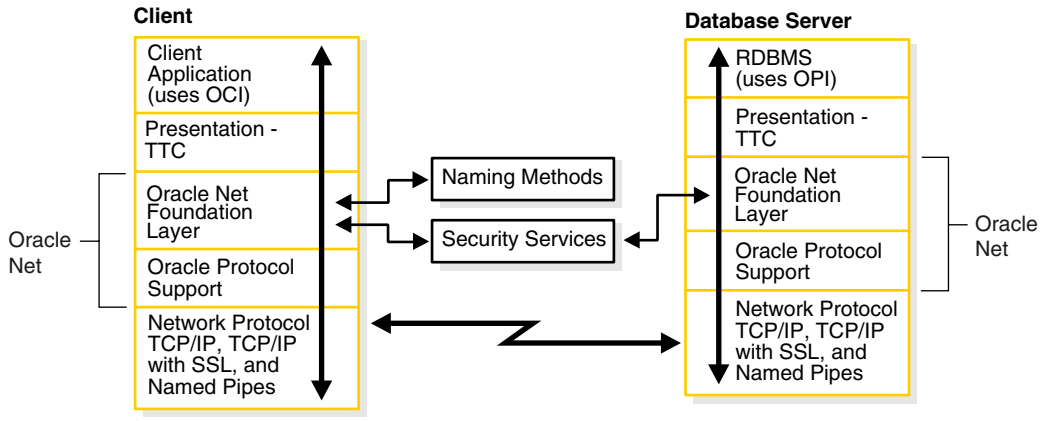
This section contains these topics:

- Stack Communication for Client/Server Application Connections
- Stack Communication for Java Application Connections
- Stack Communication for Web Client Connections

Stack Communication for Client/Server Application Connections

Figure 5–1 illustrates the various layers on the client and on the database server after a connection has been established.

Figure 5–1 Layers Used in a Client/Server Application Connection



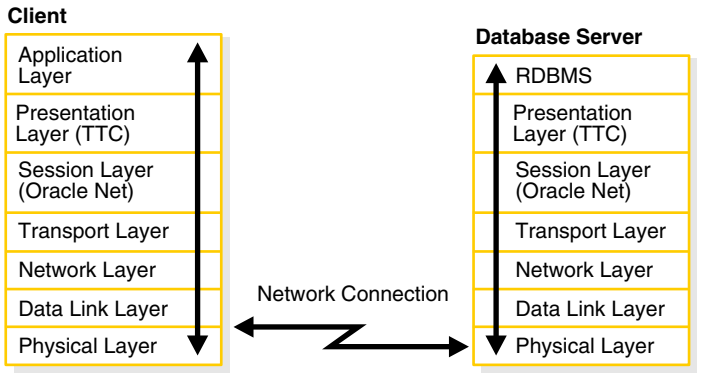
Note: The SDP protocol is supported but is not represented in this figure.

This communication architecture is based on the **Open Systems Interconnection (OSI)** model. In the OSI model, communication between separate computers occurs in a stack-like fashion with information passing from one node to the other through several layers of code, including:

1. Physical layer
2. Data link layer
3. Network layer
4. Transport layer
5. Session layer
6. Presentation layer
7. Application layer

Figure 5–2 shows how Oracle Net software—Oracle Net Foundation layer and Oracle Protocol Support—fits into the session layer of the OSI model.

Figure 5–2 OSI Communication Layers



See Also: The following URL, for information about the OSI stack:

`http://www.ietf.org`

As shown in Figure 5–1 on page 5-2, the client/server stack includes the following:

- Client Application
- Presentation
- Oracle Net Foundation Layer
- Oracle Protocol Support
- Network Protocol
- RDBMS

Client Application

During a session with the database, the client uses **Oracle Call Interface (OCI)** to interact with the database server. OCI is a software component that provides an interface between the client application and the SQL language the database server understands.

See Also: *Oracle Call Interface Programmer's Guide*

Presentation

Character set differences can occur if the client and database server are running on different operating systems. The presentation layer resolves any differences. It is optimized for each connection to perform conversion only when required.

The presentation layer used by client/server applications is **Two-Task Common (TTC)**. TTC provides character set and data type conversion between different character sets or formats on the client and database server.

At the time of initial connection, TTC is responsible for evaluating differences in internal data and character set representations and determining whether conversions are required for the two computers to communicate.

Oracle Net Foundation Layer

The Oracle Net foundation layer is responsible for establishing and maintaining the connection between the client application and database server, as well as exchanging messages between them. The Oracle Net foundation layer is able to perform these tasks because of a technology called **Transparent Network Substrate (TNS)**. TNS provides a single, common interface for all industry-standard protocols. In other words, TNS enables peer-to-peer application connectivity, where two or more computers (called **node**s when they are employed in a networking environment) can communicate with each other directly, without the need for any intermediary devices.

On the client side, the Oracle Net foundation layer receives client application requests and resolves all generic computer-level connectivity issues, such as:

- The location of the database server or destination
- How many protocols are involved in the connection
- How to handle interrupts between client and database server based on the capabilities of each

On the server side, the Oracle Net foundation layer performs the same tasks as it does on the client side and also works with the listener to receive incoming connection requests.

In addition to establishing and maintaining connections, the Oracle Net foundation layer communicates with naming methods to resolve names and uses security services to ensure secure connections.

Oracle Protocol Support

Positioned between the Oracle Net foundation layer and the network protocol layer, the Oracle protocol support layer is responsible for mapping TNS functionality to industry-standard protocols used in the client/server connection. This layer supports the following network protocols:

- TCP/IP
- TCP/IP with SSL
- Named Pipes
- SDP

Network Protocol

All Oracle software in the client/server connection process requires an existing network protocol stack to establish the computer-level connection between the two computers for the transport layer. The network protocol is responsible for transporting data from the client computer to the database server computer, at which point the data is passed to the server-side Oracle protocol support layer.

TCP/IP Protocol The Transmission Control Protocol/Internet Protocol (TCP/IP) is the de facto standard communication protocol used for client/server conversation over a network.

TCP/IP with SSL Protocol The TCP/IP with **Secure Sockets Layer (SSL)** protocol enables an Oracle application on a client to communicate with remote Oracle databases through TCP/IP and SSL. **Oracle Advanced Security** is required in order to use TCP/IP with SSL.

SSL stores authentication data, such as certificates and private keys, in an Oracle Wallet. When the client initiates a connection to the database server, SSL performs a handshake between the two using the certificate. During the handshake, the following processes occur:

- The client and database server negotiate a cipher suite—a set of authentication, encryption, and data integrity types—to apply to the messages they exchange.
- Depending on its configuration, the database server sends its certificate to the client in a message encrypted with the client's public key. The database server may also send a request for the client's certificate in the same message. The client decrypts this message by using its own private key, then verifies that the database server's certificate bears the certificate authority's signature.
- If required, the client may send the user's certificate to the database server. The certificate ensures that the user's information is correct and that the public key actually belongs to that user.

The database server checks the user's certificate to verify that it bears the certificate authority's signature.

See Also: *Oracle Database Advanced Security Administrator's Guide*

Named Pipes Protocol The **Named Pipes protocol** is a high-level interface providing interprocess communications between clients and database servers using distributed applications. One sever-side process creates the pipe, and the other client-side process opens it by name. What one side writes, the other can read, and vice versa. Named Pipes is specifically designed for PC LAN environments.

Named Pipes enables client/server connection over a network using Named Pipes. This combination of Oracle products enables an Oracle application on a client to communicate with remote Oracle databases through Named Pipes (if the Oracle database is running on a host system that supports network communication using Named Pipes).

SDP The Sockets Directory Protocol (SDP) is an industry-standard wire protocol between Infiniband network peers. When SDP is used over an Infiniband network, it reduces the overhead of TCP/IP by eliminating intermediate replication of data and transferring most of the messaging burden away from the CPU and onto the network hardware.

RDBMS

Information passed from a client application across a network protocol is received by a similar communications stack on the database server side. The process flow on the database server side is the reverse of the process flow on the client side, with information ascending through the communication layers.

Instead of OCI, the database server uses **Oracle Program Interface (OPI)**. For each statement sent from OCI, OPI provides a response. For example, an OCI request to fetch 25 rows would elicit an OPI response to return the 25 rows once they have been fetched.

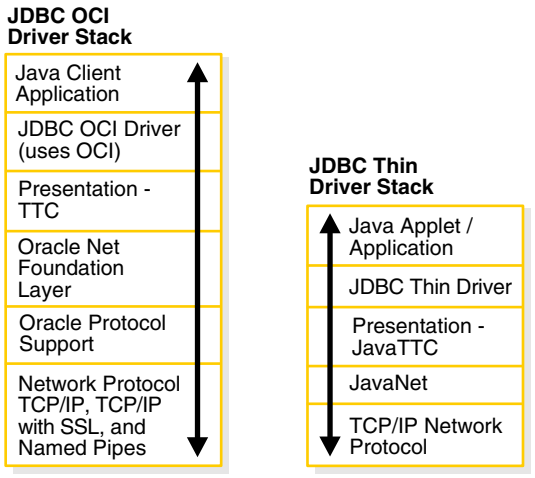
Stack Communication for Java Application Connections

The Oracle **Java Database Connectivity (JDBC) Driver**s provide Java applications access to an Oracle database. Oracle offers two JDBC drivers.

- **JDBC OCI Driver** driver is a level 2 JDBC driver which is used by client/server Java applications. The JDBC OCI driver converts JDBC invocations to calls to OCI which are then sent over Oracle Net to the Oracle database server.
- **JDBC Thin Driver** is a level 4 driver which is used by Java applets. The JDBC Thin driver establishes a direct connection to the Oracle database server over Java sockets. Access to the database is assisted with a lightweight implementation of TTC and Oracle Net.

Figure 5–3 shows the stack communication layers used by JDBC drivers.

Figure 5–3 Layers Used for Java-Client Applications



Note: The SDP protocol is supported but is not represented in this figure.

The JDBC OCI driver uses a communication stack similar to a standard client/server communication stack. The JDBC Thin driver uses a Java implementation of the Oracle Net foundation layer called JavaNet and a Java implementation of TTC called JavaTTC.

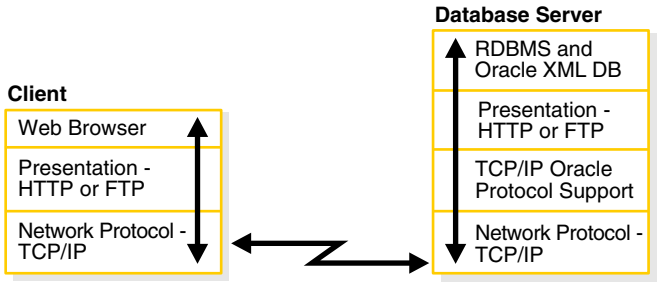
See Also: *Oracle Database JDBC Developer's Guide and Reference*

Stack Communication for Web Client Connections

In addition to the TTC presentation, the Oracle database server supports many other presentations that can be used for Web clients accessing features inside the database. The listener facilitates this by supporting any presentation requested by the database.

For example, Figure 5–4 shows the stack communication layers used in an HTTP or FTP connection to **Oracle XML DB** in the Oracle database instance. WebDAV connections use the same stack communication layers as HTTP and FTP.

Figure 5–4 Layers Used in Web Client Connections

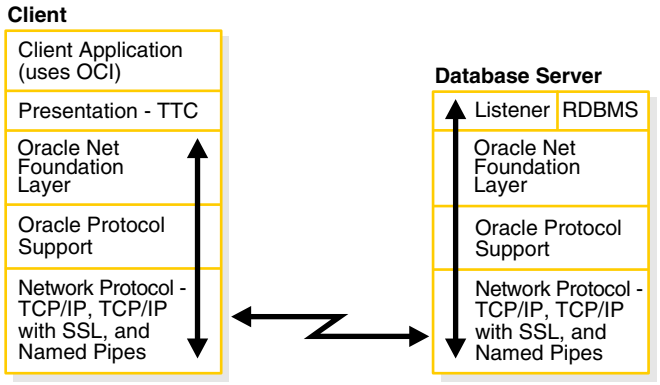


See Also: *Oracle XML DB Developer's Guide*

Listener Architecture

The database server receives an initial connection from a client application through the listener. The listener is an application positioned on top of the Oracle Net foundation layer. Figure 5–5 illustrates the various layers on the client and database server during an initial connection.

Figure 5–5 Layers Used in an Initial Connection



Note: The SDP protocol is supported but is not represented in Figure 5–5.

The listener brokers client requests, handing off the requests to the Oracle database server. Every time a client requests a network session with a database server, a listener receives the initial request.

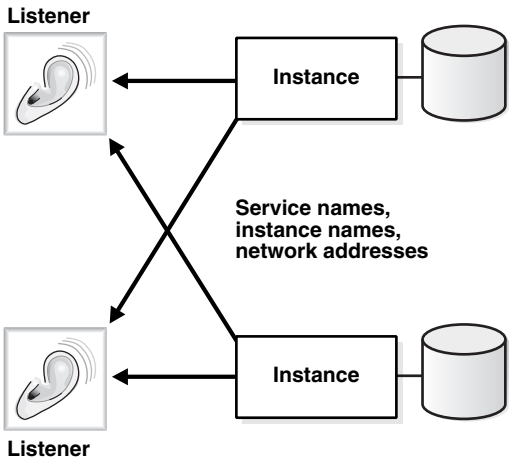
Each listener is configured with one or more protocol addresses that specify its listening endpoints. Clients configured with a **protocol address** can send connection requests to the listener.

Once a client request has reached the listener, the listener selects an appropriate **service handler** to service the client's request and forwards the client's request to it. The listener determines if a database service and its service handlers are available through **service registration**. During service registration, the **PMON process**—an instance background process—provides the listener with information about the following:

- Names of the database services provided by the database
- Name of the **instance** associated with the services and its current and maximum load
- Service handlers (**dispatcher**s and dedicated servers) available for the instance, including their type, protocol addresses, and current and maximum load

This information enables the listener to direct a client's request appropriately Figure 5–6 shows instances registering information with listeners. Note that it does not represent all the information that can be registered.

Figure 5–6 Service Registration



Optionally, listening endpoints—port numbers—can be dynamically registered with the listener. For example, with Oracle XML DB, HTTP, FTP, and WebDAV listening endpoints are registered with the listener.

If the listener is not running when an instance starts, PMON is not able to register the service information. PMON attempts to connect to the listener periodically, however, it may take up to 60 seconds before PMON registers with the listener after it has been started. To initiate service registration immediately after the listener is started, use the SQL statement `ALTER SYSTEM REGISTER`. This is especially useful in high-availability configurations.

If a listener receives an incoming request before the respective instance has been registered, the listener rejects the request.

If an instance is in restricted mode, then PMON instructs the listener to block all connections to the instance. Clients attempting to connect receive one of the following errors:

- `ORA-12526: TNS:listener: all appropriate instances are in restricted mode`
- `ORA-12527: TNS:listener: all appropriate instances are in restricted mode or blocking new connections`
- `ORA-12528: TNS:listener: all appropriate instances are blocking new connections`

> **See Also:** *Oracle Database Error Messages* for further information about these error messages

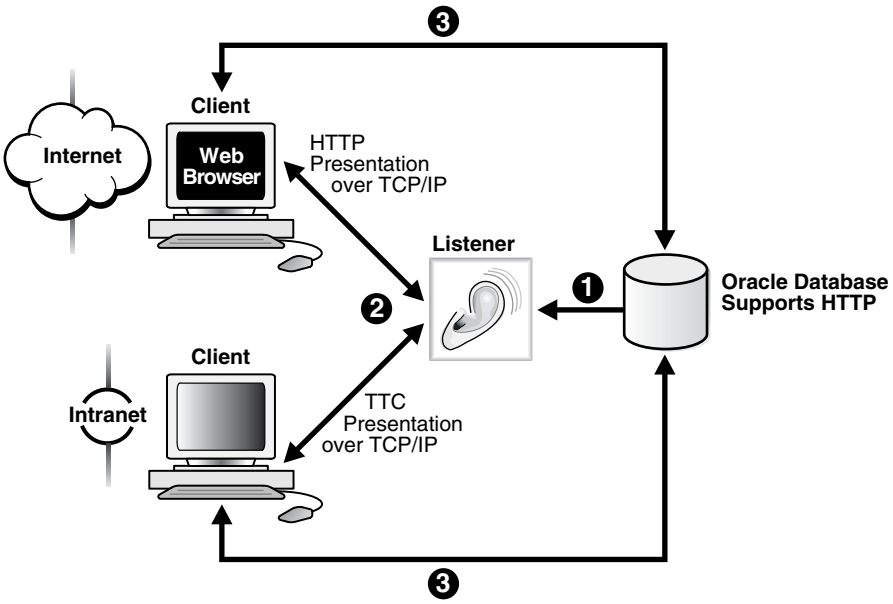
Figure 5–7 on page 5-9 shows the role of a listener during connection establishment with a browser making an HTTP connection and a client making a database connection:

1. The database registers information about the services, instances, and service handlers with the listener.
2. The client makes an initial connection with the listener.
3. The listener parses the client request and forwards it to the service handler for the database service requested.

See Also:

- *Oracle Database SQL Language Reference* for further information about the `ALTER SYSTEM REGISTER` statement
- *Oracle XML DB Developer's Guide* for information about dynamically registering HTTP, FTP, and WebDAV listening endpoints

Figure 5–7 Listener Architecture



Database Server Process Architecture

Based on the service handler type registered with the listener, the listener forwards requests to either a shared server or dedicated server process.

This section includes the following topics:

- Shared Server Processes
- Dedicated Server Processes

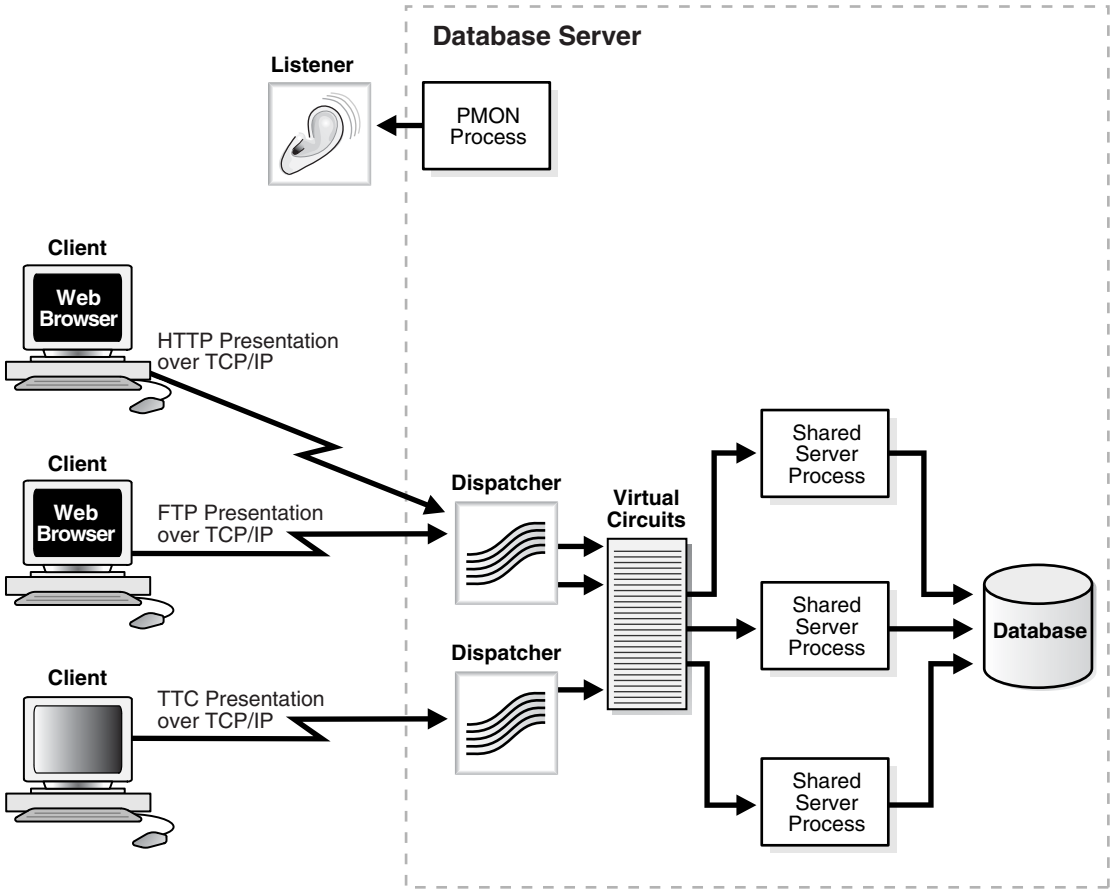
Shared Server Processes

Shared server processes are used in the shared server architecture, as shown in Figure 5–8 on page 5-10. With shared server architectures, client processes ultimately connect to a dispatcher. The PMON process registers the location and load of the dispatchers with the listener, enabling the listener to forward requests to the least loaded dispatcher.

A dispatcher can support multiple client connections concurrently. Each client connection is bound to a **virtual circuit**. A virtual circuit is a piece of shared memory used by the dispatcher for client database connection requests and replies. The dispatcher places a virtual circuit on a common queue when a request arrives. An idle shared server picks up the virtual circuit from the common queue, services the request, and relinquishes the virtual circuit before attempting to retrieve another virtual circuit

from the common queue. This approach enables a small pool of server processes to serve a large number of clients.

Figure 5–8 Shared Server Architecture



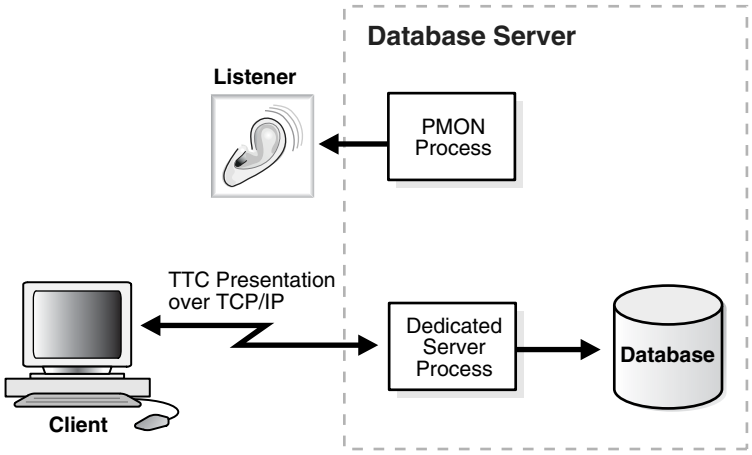
Dedicated Server Processes

Figure 5–9 depicts a dedicated server architecture. With a dedicated server architecture, each client process connects to a dedicated server process. The server process is not shared by any other client.

PMON registers information about dedicated server processes with the listener. This enables the listener to start up a dedicated server process when a client request arrives and forward the request to it.

Note: Dedicated server architectures do not support HTTP, FTP, or WebDAV clients. Only database clients are supported.

Figure 5–9 Dedicated Server Architecture



Oracle Connection Manager Architecture

Oracle Connection Manager is a router through which a client connection request is sent either to its next hop or directly to the database server. Clients who route connection requests through an Oracle Connection Manager can take advantage of the session multiplexing and access control features configured on that Oracle Connection Manager.

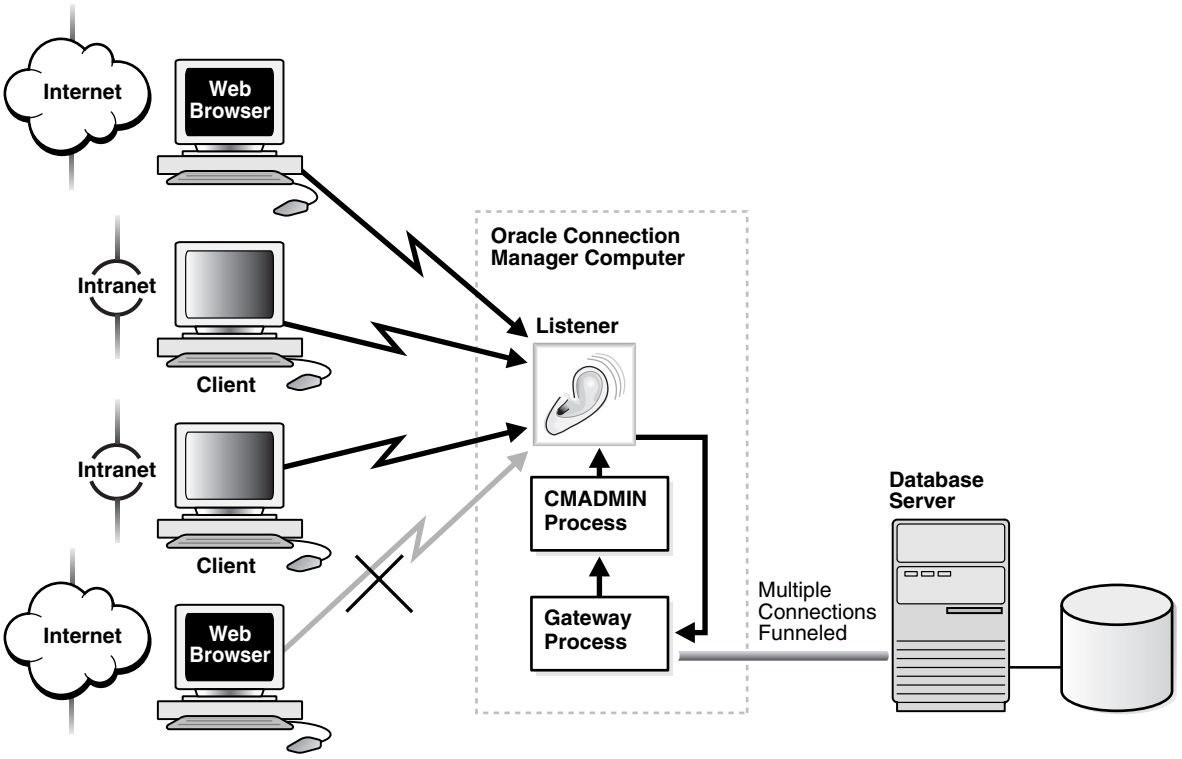
Oracle Connection Manager consists of three components:

- **listener**
- **CMGW (Connection Manager gateway)**
- **CMADMIN (Connection Manager Administration)**

The listener receives client connections and evaluates against a set of rules whether to deny or allow access. If it allows access, the listener forwards a request to a gateway process, selecting the one with the fewest connections. The `CMGW` process, in turn, forwards the request to another Oracle Connection Manager or directly to the database server, relaying data until the connection terminates. If a connection to the server already exists, the gateway multiplexes, or funnels, its connections through the existing connection. `CMADMIN` monitors the health of the gateway processes and the listener, shutting down or starting up processes as needed. In addition, it registers the location and load of the gateway processes with the listener, and it answers requests from the Oracle Connection Manager Control utility.

In Figure 5–10, the listener screens connection requests. A gateway process registers with the `CMADMIN` process. And the `CMADMIN` process registers with the listener. Finally, the listener forwards the connection requests to the gateway process. Notice that the listener has denied access to the fourth client. After receiving the three valid client connections, the gateway process multiplexes them through a single network protocol connection to the database.

Figure 5–10 Oracle Connection Manager Architecture

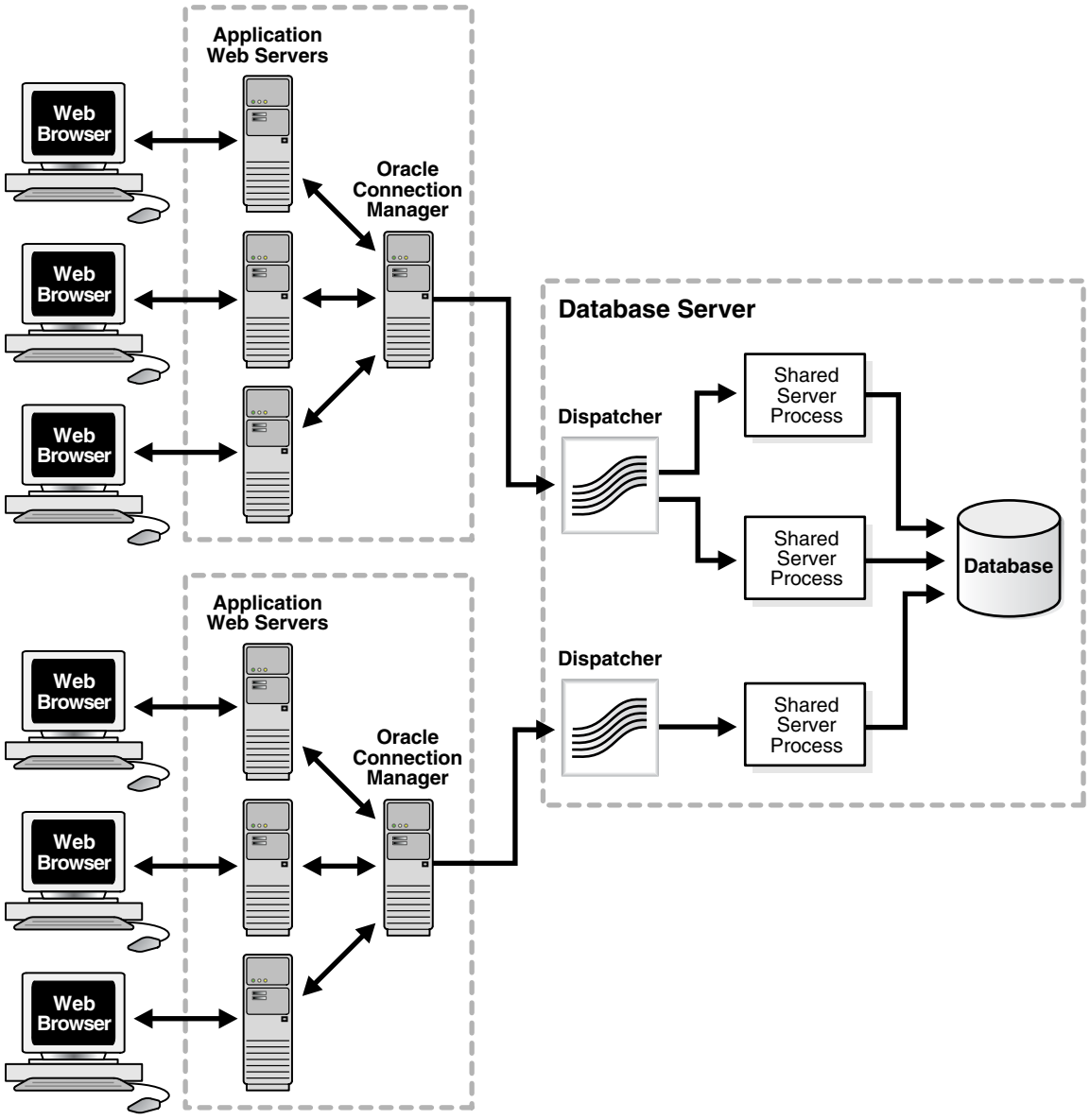


A Complete Architecture

Oracle Net provides an architectural solution that allows for greater scalability in Internet and intranet environments.

Figure 5–11 shows how multiple connections to an Oracle database server are made more scalable with Oracle Connection Manager and a shared server architecture. Oracle Connection Manager is used to offload some of the network I/O of the application Web servers, and shared server is used to serve more concurrent users.

Figure 5–11 Scalable Architectural Solutions



6

Configuration and Administration Tools Overview

This chapter introduces the various administration tools of Oracle Net Services. It discusses the main administration applications, **Oracle Enterprise Manager** and **Oracle Net Manager**. It also introduces the command-line control utilities.

This chapter contains these topics:

- User Interface Tools
- Oracle Net Control Utilities
- Duties of a Network Administrator

User Interface Tools

Oracle Net Services provides the following tools to help you perform configuration and administrative tasks:

- Oracle Enterprise Manager
- Oracle Net Manager
- Selecting When to Use Oracle Enterprise Manager and Oracle Net Manager
- Oracle Net Configuration Assistant

Oracle Enterprise Manager

Oracle Enterprise Manager enables you to configure Oracle Net Services for any Oracle home across multiple file systems. It also provides common administration functions for listeners. Oracle Enterprise Manager provides an integrated environment for configuring and managing Oracle Net Services.

You can use Oracle Enterprise Manager to configure and administer the following from multiple Oracle homes:

- Listeners: Configure listeners to receive client connections.
- Naming: Define simple names, **connect identifier**s, and map them to **connect descriptor**s to identify the network location and identification of a service. Oracle Net Manager supports configuration of connect descriptors in local `tnsnames.ora` files or a centralized directory service.
- File Location: Change the file location of the Oracle home.

See Also: Oracle Enterprise Manager documentation set and online Help for information about using Oracle Enterprise Manager

Accessing the Net Services Administration Page

To access the Net Services Administration page:

1. From the Login to Database page, enter the database credentials, and then click **Login**.

   The Database page appears.

2. In the **General** section, click the listener.

   The Listener Home page appears.

3. In the **Related Links** section, click Net Services Administration.

   The Net Services Administration page appears.

4. Select an Oracle home location for network configuration files.

5. From the **Administer** list, select one of the following options:

   - Listeners
   - Directory Naming
   - Local Naming
   - File Location

6. Click **Go**.

   If you selected Listeners, Directory Naming, or Local Naming, then the Net Services Administration: Host Login page appears.

7. Enter the username and password credentials for the computer, and then click **Login**.

   If you selected Directory Naming, then the Net Services Administration: Directory Server Login page appears.

8. In the **User DN** field, enter an authorized user that has access to the directory.

9. If you want to add or modify entries, then enter a directory administrator user that has been given add and modify privileges in the form of a distinguished name. `c=US,o=ACME,ou=admin,cn=scott`, for example, is the DN for an administrator named Scott.

10. In the **Password** field, enter the password for the user.

11. Click **Login**.

Oracle Net Manager

Oracle Net Manager enables you to configure Oracle Net Services for an Oracle home on a local client or server host.

You can use Oracle Net Manager to configure the following network components:

- Naming: Define simple names, **connect identifier**s, and map them to **connect descriptor**s to identify the network location and identification of a service. Oracle Net Manager supports configuration of connect descriptors in local `tnsnames.ora` files or a centralized directory service.

- Naming Methods: Configure the different ways in which connect identifiers are resolved into connect descriptors.
- Profiles: Configure preferences for enabling and configuring Oracle Net features on the client or server.
- Listeners: Create and configure listeners to receive client connections.

This section introduces you to the features of Oracle Net Manager. However, the primary documentation for using Oracle Net Manager is the accompanying online Help. This section contains these topics:

- Starting Oracle Net Manager
- Navigating Oracle Net Manager
- Oracle Net Manager Wizards

Starting Oracle Net Manager

You can start Oracle Net Manager using the Oracle Enterprise Manager Console or as an independent application.

To start Oracle Net Manager from the Oracle Enterprise Manager console, on the Oracle Enterprise Manager console, choose **Tools** > **Service Management** > **Oracle Net Manager**.

To start Oracle Net Manager as standalone application:

- On UNIX, run `netmgr` from `$ORACLE_HOME/bin`
- On Windows, choose **Start** > **Programs** > **Oracle - HOME_NAME** >**Configuration and Migration Tools** > **Net Manager.**

Navigating Oracle Net Manager

The Oracle Net Manager interface includes a toolbar and various menu options, as well as property sheets for configuring network components.

The navigator pane provides a tree view of network objects and the objects they contain, organized in hierarchies of folders. You can expand and contract the folders to monitor or manage objects such as connect identifiers, listeners, and profiles. Right-click an object to make changes to it.

Table 6–1 lists the main folders in the navigator pane.

Table 6–1 Oracle Net Manager Navigator Pane Folders

Option	Description
Local	Displays networking elements configured in local configuration files: ▪ Net service names in the `tnsnames.ora` file ▪ Listeners in the `listener.ora` file ▪ Profile in the `sqlnet.ora` file
Directory	Displays connect identifiers configured in a directory server

Oracle Net Manager Wizards

The Oracle Net Manager wizards provide step-by-step guidance for tasks that require many steps. The wizards simplify complex tasks by guiding you through the task in manageable steps. The wizards are not intended to provide all configuration options. Once you have completed a task with a wizard, use other components of Oracle Net Manager to modify the configuration.

The following topics describe the Oracle Net Manager wizards:

- Net Service Name Wizard
- Directory Server Migration Wizard

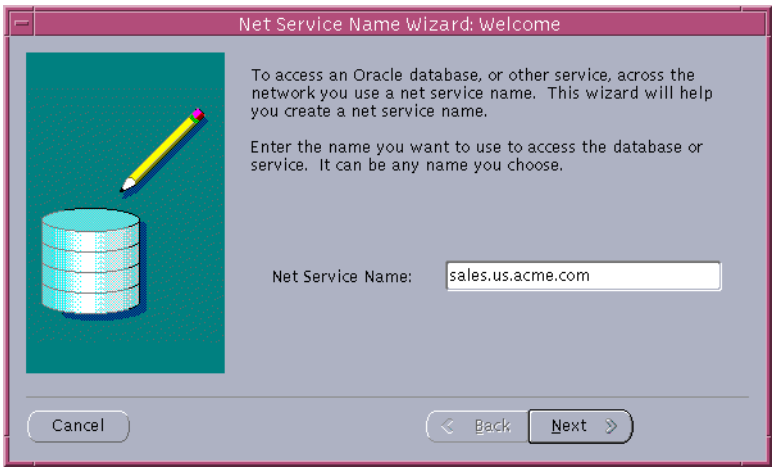
Net Service Name Wizard

The Net Service Name Wizard guides you through creating a basic net service name in a directory server or a `tnsnames.ora` file.

To start the Net Service Name Wizard to create net service names:

1. In the navigator pane, choose **Directory** or **Local** > **Service Naming**.
2. Choose plus (+) from the toolbar, or choose **Edit** > **Create** from the menu bar.

Figure 6–1 Opening Page of the Net Service Name Wizard



See Also: Oracle Net Manager online Help for detailed information about using the Net Service Name Wizard to create a net service name.

Directory Server Migration Wizard

If a `tnsnames.ora` file already exists, its net service names can be exported to a directory server with the Directory Server Migration Wizard.

To start the Directory Server Migration Wizard, choose **Command** > **Directory** > **Export Net Service Names** in the Oracle Net Manager menu bar.

Figure 6–2 Opening Page of the Directory Server Migration Wizard

See Also: "Exporting Local Naming Entries to a Directory Naming Server" on page 8-21

Selecting When to Use Oracle Enterprise Manager and Oracle Net Manager

In Oracle Database 11*g*, much of the functionality previously only available in Oracle Net Manager has been integrated within Oracle Enterprise Manager. Oracle Enterprise Manager provides the ability to manage configuration for multiple Oracle homes across multiple file systems; Oracle Net Manager only enables you to manage configuration for one Oracle home on a local host computer. Table 6–2 describes the key differences between the tools.

Table 6–2 Key Differences Between Oracle Enterprise Manager and Oracle Net Manager

User Interface Tool	Features
Oracle Enterprise Manager	▪ Configures the following features: - Local naming (`tnsnames.ora` files) - Directory naming - Listeners ▪ Provides multiple Oracle home support across multiple file system ▪ Provides the ability to search and sort local and directory naming entries ▪ Export directory naming entries to a `tnsnames.ora` file ▪ Performs the following administrative tasks for a selected listener: - Show current status - Change status - Change tracing settings - Change logging settings

Table 6–2 (Cont.) Key Differences Between Oracle Enterprise Manager and Oracle Net

User Interface Tool	Features
Oracle Net Manager	▪ Configures the following features: - Local naming (`tnsnames.ora` files) - Directory naming - Listeners - Profiles ▪ Provides Oracle home support for single host

Oracle Net Configuration Assistant

Oracle Net Configuration Assistant is provided to configure basic network components during installation, including:

- Listener names and protocol addresses
- Naming methods the client will use to resolve connect identifiers to connect descriptors
- Net service names in a `tnsnames.ora` file
- Directory server usage

Oracle Net Configuration Assistant runs automatically during software installation, as described in your Oracle installation guide.

It can also be run after installation in standalone mode to configure naming method usage, the listener, net service names in the `tnsnames.ora` file, and directory server usage in a similar way that is provided during installation.

To start Oracle Net Configuration Assistant:

- On UNIX, run `netca` from `$ORACLE_HOME/bin`.
- On Windows, choose **Start** > **Programs** > **Oracle - HOME_NAME** > **Configuration and Migration Tools** > **Net Configuration Assistant**.

See Also: Oracle Net Configuration Assistant online Help

Figure 6–3 Opening Page of Oracle Net Configuration Assistant

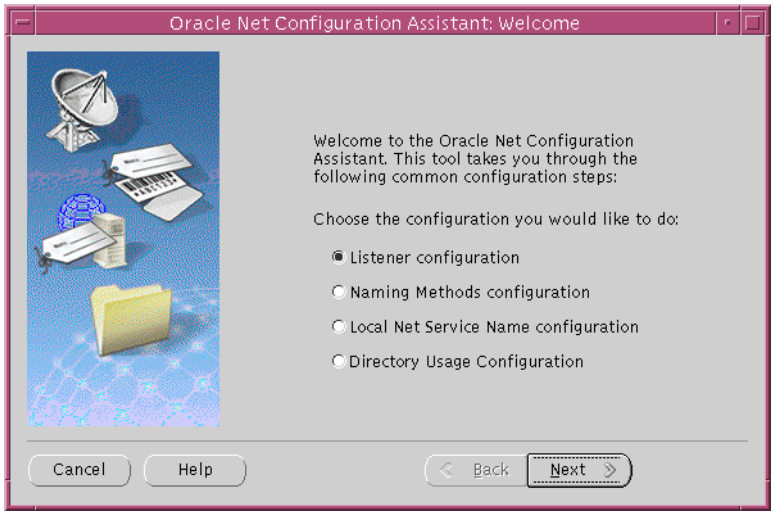


Table 6–3 describes the configuration options in the Welcome page:

Table 6–3 Oracle Net Configuration Assistant

Option	Description
Listener configuration	Click to create modify, delete, or rename a listener.
Naming Methods configuration	Click to configure this computer to resolve connect identifiers to connect descriptor through one or more of following naming methods: ■ Local naming ■ Directory naming ■ Host naming ■ Network Information Service ■ Cell Directory Service
Local Net Service Name configuration	Click to create, modify, delete, rename, or test connectivity of a connect descriptor stored in a local `tnsnames.ora` file.
Directory Usage Configuration	Click to configure a directory server for directory-enabled features.

Oracle Net Control Utilities

Oracle Net Services provides the following tools to help you start, stop, configure, and control each network component:

- Listener Control Utility
- Oracle Connection Manager Control Utility

Listener Control Utility

The Listener Control utility enables you to administer the listener. The basic syntax for this utility is as follows:

```
LSNRCTL command [listener_name]
```

For example, the following command starts a listener named `lsnr`:

```
LSNRCTL START lsnr
```

You can also issue Listener Control utility commands at the `LSNRCTL>` program prompt. To obtain the prompt, enter `lsnrctl` with no arguments at the operating system command line. When you run `lsnrctl`, the program is started. You can then enter the necessary commands from the program prompt.

For example:

```
lsnrctl
LSNRCTL> START lsnr
```

See Also:

- "Customizing Oracle Net Listener Configuration" on page 10-3 for further information about the listener
- *Oracle Database Net Services Reference* for further information about the Listener Control utility

Oracle Connection Manager Control Utility

The Oracle Connection Manager Control utility enables you to administer an Oracle Connection Manager. When you issue commands from the operating system, the basic syntax for this utility is as follows:

```
cmctl {command} [argument1. . . argumentN]instance_name} {-p}
```

For example, the following command starts both the CMGW and the CMADMIN processes:

```
cmctl STARTUP -c cman1 -p
```

You can also issue Oracle Connection Manager utility commands at the CMCTL> program prompt. To obtain the prompt, enter cmctl with no arguments at the operating system command line. When you run cmctl, the program is opened. You can then enter the necessary commands from the program prompt.

For example:

```
cmctl
CMCTL> STARTUP
```

Note: Before issuing STARTUP, you must issue the ADMINISTER command to choose an instance to start.

See Also:

- "Oracle Connection Manager Architecture" on page 5-11 for an overview of the Oracle Connection Manager processes
- *Oracle Database Net Services Reference* for a complete description of Oracle Connection Manager Control utility commands

Duties of a Network Administrator

Network configuration and administration tasks are described throughout this guide. Table 6–4 lists the common tasks, the tools associated with them, and points you to the topic in the guide that describes the task.

Table 6–4 Common Tasks for Configuring and Administering Oracle Net Services

Task	Tools to Perform Task	See Also
Configuring Directory Server for Oracle Net Usage		
Configure directory server usage.	Oracle Internet Directory Configuration Assistant	*Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*
Add users to the OracleNetAdmins group.	ldapmodify	"Administering the OracleNetAdmins Group" on page 8-19

Table 6–4 (Cont.) Common Tasks for Configuring and Administering Oracle Net Services

Task	Tools to Perform Task	See Also
Authenticate with the directory.	Oracle Enterprise Manager Oracle Net Manager	Online Help in Oracle Enterprise Manager Choose **Directory** > **Service Naming** > **How To** > **Change the Oracle Context** in the online Help for Oracle Net Manager
Change the Oracle Context.	Oracle Net Manager	Online Help in Oracle Enterprise Manager Choose **Directory** > **Service Naming** > **How To** > **Set Authentication Credentials** in the online Help for Oracle Net Manager
Configuring Naming Methods		
Configure the local naming method.	Oracle Enterprise Manager Oracle Net Manager Oracle Net Configuration Assistant	"Configuring the Local Naming Method" on page 8-8
Configure the directory naming method.	Oracle Enterprise Manager Oracle Net Manager	"Configuring the Directory Naming Method" on page 8-14
Configure the easy connect naming method.	Oracle Net Manager	"Using the Easy Connect Naming Method" on page 8-3
Configure external naming methods.	Oracle Net Manager	"Configuring External Naming Methods" on page 8-25
Migrating to Directory Naming		
Export from `tnsnames.ora` files.	Oracle Enterprise Manager Oracle Net Manager	"Exporting Net Service Names from a tnsnames.ora File" on page 8-21
Configuring Profiles		
Prioritize naming methods.	Oracle Net Manager Oracle Net Configuration Assistant	"Prioritizing Naming Methods" on page 9-3
Configure a default domain that is automatically appended to any unqualified net service name.	Oracle Net Manager Oracle Net Configuration Assistant	"Configuring a Default Domain for Clients" on page 9-2
Route connection requests.	Oracle Net Manager Oracle Net Configuration Assistant	"Routing Connection Requests" on page 9-4
Configure access control.	Oracle Net Manager	"Configuring Database Access Control" on page 9-4
Configure an **authentication method** available with **Oracle Advanced Security**.	Oracle Net Manager	"Configuring Oracle Advanced Security" on page 9-8 Choose **Oracle Advanced Security** > **How To** in the online help **See Also:** *Oracle Database Advanced Security Administrator's Guide*

Table 6–4 (Cont.) Common Tasks for Configuring and Administering Oracle Net Services

Task	Tools to Perform Task	See Also
Configure connect request timeouts.	Manual Configuration	"Configuring the Listener and the Oracle Database To Limit Resource Consumption By Unauthorized Users" on page 14-8
Configuring Listeners		
Configure listening protocol addresses.	Oracle Enterprise Manager Oracle Net Manager Oracle Net Configuration Assistant	"Configuring Listening Protocol Addresses" on page 10-3
Configure dynamic service registration.	Automatic	"Configuring Service Registration" on page 10-10
Configure static service registration.	Oracle Enterprise Manager Oracle Net Manager	"Configuring Static Service Information" on page 10-6
Configure password authentication.	Oracle Enterprise Manager Oracle Net Manager	"Configuring Passwords for Oracle Net Listener" on page 10-8
Configure connect request timeouts.	Manual Configuration	"Configuring the Listener and the Oracle Database To Limit Resource Consumption By Unauthorized Users" on page 14-8
Administering Listeners		
Start and stop listeners.	Listener Control Utility	"Starting and Stopping a Listener" on page 10-14
View registered information.	Listener Control Utility	"Monitoring Services of a Listener" on page 10-17
Configuring Oracle Connection Manager		
Configure session multiplexing.	Manual Configuration	"Enabling Session Multiplexing" on page 11-7
Configure access control.	Manual Configuration	"Enabling Access Control" on page 11-7

Part II

Configuration and Administration of Oracle Net Services

Part II describes how to set up and configure Oracle Net Services.

This part contains these chapters:

- Chapter 7, "Planning the Network"
- Chapter 8, "Configuring Naming Methods"
- Chapter 9, "Configuring Profiles"
- Chapter 10, "Configuring and Administering Oracle Net Listener"
- Chapter 11, "Configuring and Administering Oracle Connection Manager"
- Chapter 12, "Configuring Dispatchers"
- Chapter 13, "Enabling Advanced Features of Oracle Net Services"
- Chapter 14, "Optimizing Performance"

7

Planning the Network

Oracle Net Services provide a variety of options to help you design and manage networks that are both flexible and easy to use. With Oracle Net Services enhanced scalability and manageability features, you can develop a network to support a wide range of environments, whether they be simple workgroups or large mission critical enterprises.

This chapter describes considerations for planning a network using Oracle Net Services. It explains the relationships of the network products, and options for expanding and better managing your future network.

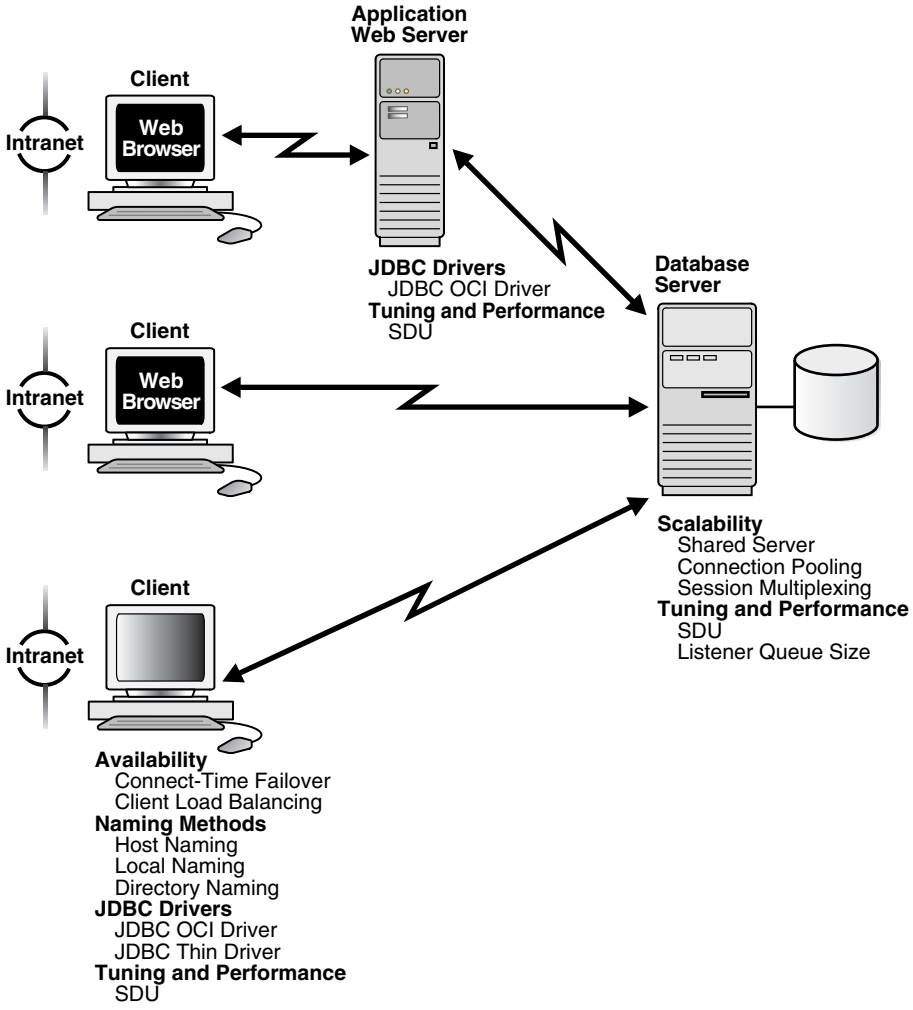
This chapter contains these topics:

- Deploying a Network Inside an Organization
- Deploying a Network for the Internet
- Naming Considerations
- Performance Considerations
- Planning Summary

Deploying a Network Inside an Organization

As Figure 7–1 shows, an internal network often has a diverse client makeup. Besides traditional clients that need access to the database servers, Web clients also need access. The type and number of clients, as well as other networking issues, affect the types of features to consider deploying.

Figure 7–1 Internal Network Layout



This section addresses deployment considerations for an internal network in terms of the following categories:

- Scalability
- Availability
- Naming Methods
- JDBC Drivers
- Security
- Tuning and Performance

Scalability

For an organization with hundreds or thousands of clients, scalability is of the utmost importance. Oracle Net Services offers a number of features that can improve database server scalability, including **shared server**, **connection pooling**, and **session multiplexing**.

Shared Server

With a shared server architecture, a small pool of server processes can serve a large number of clients. This reduces the server's memory requirements. Connection pooling and session multiplexing features are enabled through shared server.

Connection Pooling

Connection pooling enables the database server to time out protocol connections and to use those connections to service an active session.

Session Multiplexing

Session multiplexing, available with Oracle Connection Manager, enables multiple client sessions to funnel through a single protocol connection.

When session multiplexing is used with connection pooling, you can increase the total number of client sessions the server can handle.

Table 7–1 summarizes the relative advantages of using connection pooling and session multiplexing and provides recommendations for using them in the network.

Table 7–1 Connection Pooling and Session Multiplexing

Feature	Advantages	Disadvantages	Recommended for
Connection Pooling	■ Reduces the number of network resources used for each process ■ Supports larger client populations ■ Maximizes the number of client/server sessions over a limited number of process connections ■ Optimizes network traffic and network resource utilization, such as network connection bandwidth ■ Enables identification and monitoring of real users ■ Enables middle-tier application Web servers or applications that need to access backend database to support additional services, such as Oracle Application Server ■ Requires only a single transport for clients with multiple applications ■ Requires only a single network connection for database links	Database sessions should use the `IDLE_TIME` resource parameter.	Networks where many clients run interactive "high think/search time" applications such as messaging and OLAP
Session Multiplexing	■ Limits the number of network resources used for each process ■ Supports large client populations ■ Maximizes the number of client/server sessions over a limited number of process connections ■ Optimizes resource utilization ■ Enables identification and monitoring of real users ■ Enables mid-tier applications to support additional services ■ Requires only a single transport for clients with multiple applications ■ Requires only a single network connection for database links ■ Provides support for pre-Oracle8 clients	Clients must connect to Oracle Connection Manager.	Networks where continuous connectivity is required.

Availability

Availability to the database is crucial for any internal network. You can configure multiple listeners to handle client connection requests for the same database service. This is especially ideal in an Oracle Real Application Clusters configuration, where each instance has a listener associated with it. Multiple listener configurations enable you to utilize **connect-time failover** and **connection load balancing** features.

This section includes the following topics:

■ Connect-Time Failover

- Client Load Balancing

Connect-Time Failover

Connect-time failover enables clients to request a different listener (usually on a different node) if the first listener fails.

Client Load Balancing

Client load balancing enables clients to randomize requests to the multiple listeners (usually on different nodes).

These features can be used together or separately. Together, they ensure access to the database and distribute the load so as not to overburden a single listener.

Naming Methods

Selecting the appropriate **naming method** for mapping names to **connect descriptor**s depends upon the size of the organization.

For a small organization with only a few databases, use easy connect naming to make TCP/IP connections with the host name of the database server or **local naming** to store names in `tnsnames.ora` file on the clients.

For large organizations with several databases, use **directory naming** to store names in a centralized directory server.

> **See Also:** "Naming Considerations" on page 7-8 for further information about selecting a naming method

JDBC Drivers

Java client applications access an Oracle database through a **Java Database Connectivity (JDBC) Driver**—a standard Java interface for connecting from Java to a relational database. Oracle Corporation offers the following drivers:

- OCI driver for client side and application Web server use with an Oracle client installation
- Thin driver for client side use without an Oracle installation, particularly with applets

Security

Ensure that Internal networks are deployed inside a firewall.

> **See Also:** *Oracle Database Advanced Security Administrator's Guide* for further information about providing security for the internal network

Tuning and Performance

Oracle Net Services offers a number of features that can help reduce round-trip time across the network, increase listener performance, and reduce the number of protocols used.

This section includes the following topics:

- Listener Queue Size
- Session Data Unit (SDU) Size

- Protocol Conversion

Listener Queue Size

If you anticipate receiving a large number of connection requests for a listening process, you can increase the size of the listener queue.

Session Data Unit (SDU) Size

Before sending data across the network, Oracle Net buffers and encapsulates data into the **session data unit (SDU)**. Oracle Net sends the data stored in this buffer when the buffer is full, flushed, or when database server tries to read data. When large amounts of data are being transmitted or when the message size is consistent, adjusting the size of the SDU buffers can improve performance, network utilization, or memory consumption. You can deploy SDU at the client, the application Web server, and the database server.

Protocol Conversion

The database only needs to be configured to listen on one protocol address, even though clients may use other protocols. Oracle Connection Manager provides a **protocol conversion** feature that enables a client and database server configured with different networking protocols to communicate with one another.

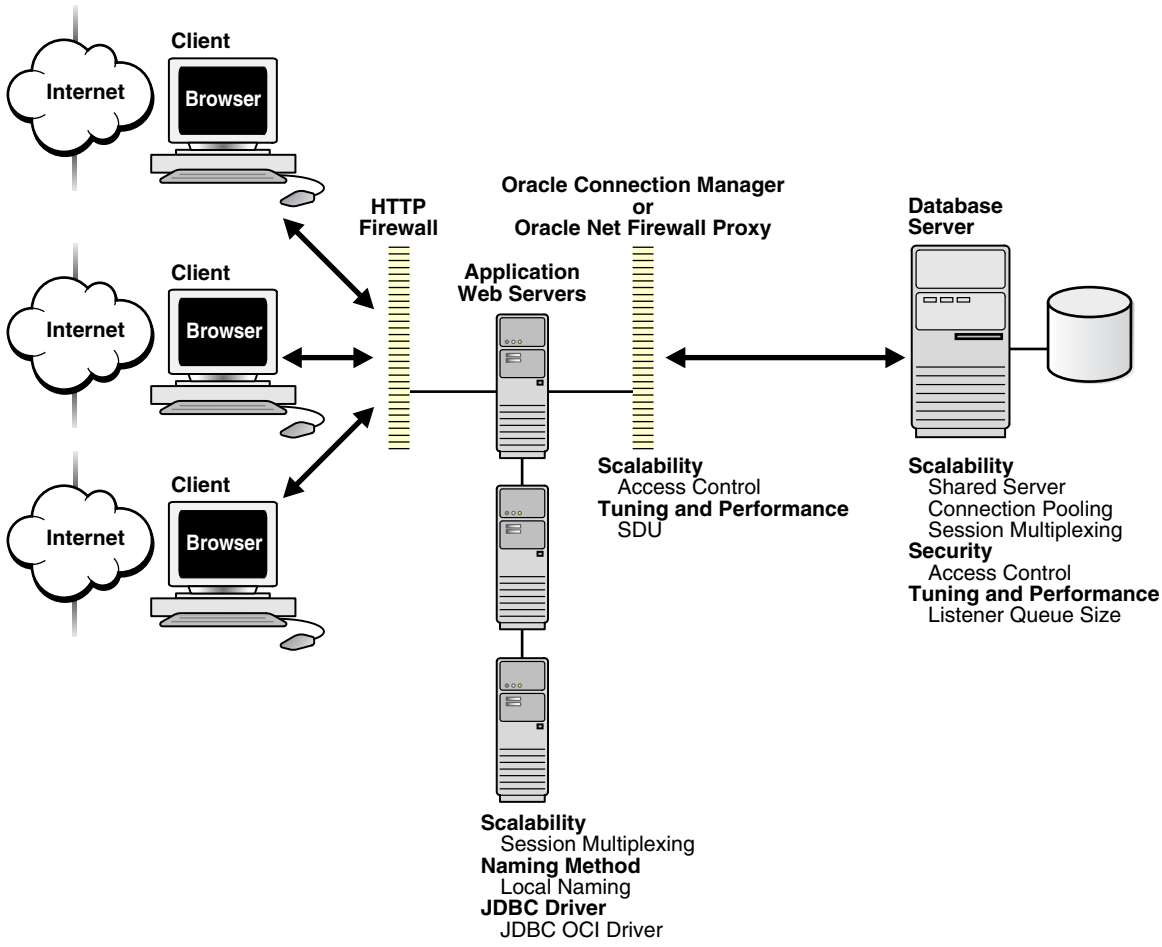
Deploying a Network for the Internet

As Figure 7–2 shows, an Internet network is usually consists of Web clients that access firewall-protected application Web servers, that then connect to a database. An Internet network has many of the same requirements as an internal network, but it also has its own unique set of requirements.

This section examines both requirements and includes the following topics:

- Scalability
- Availability
- Naming Methods
- JDBC Drivers
- Security

Figure 7–2 Internet Network Layout



Scalability

Like an internal network, scalability can be improved in an Internet network with shared server, connection pooling, and session multiplexing features configured on the database server. Session multiplexing can also be configured on the application Web server tier. This can help offload some of the network I/O of the application Web servers, increasing throughput.

Availability

Availability to the database is crucial for any Internet network with a firewall. It is important to deploy at least two Oracle Connection Manager firewalls or Oracle Net Firewall Proxies in the event that one firewall goes down.

Naming Methods

For an Internet network, configure the application Web servers needed to connect to the databases with the local naming method.

> **See Also:** "Naming Considerations" on page 7-8 for further information about selecting a naming method

JDBC Drivers

Java applications that reside on the application Web server require the JDBC OCI driver.

Security

Security in an Internet network is crucial to control access to the database.

This section includes the following topics:

- Access Control
- Listener and Database Server Connection Limits
- Tuning and Performance

Access Control

Granting and denying access to the a database is crucial for a secure network environment. You can configure access control through a firewall or on the database. For firewall support, you can configure Oracle Connection Manager to function as a firewall, whereby it grants or denies clients access to a particular database service or a computer based on a set of filtering rules. For the database, you can configure access control parameters in the `sqlnet.ora` file to specify that clients using a particular protocol are allowed or denied access.

Listener and Database Server Connection Limits

To prevent unauthorized clients from attempting denial-of-service attacks on the listener or database server, you can configure limits that constrain the time in which resources can be held prior to authentication. Client attempts to exceed the configured limits will result in connection terminations and an audit trail containing the IP address of the client being logged.

Tuning and Performance

The performance and tuning features described in "Tuning and Performance" on page 7-5 can also be deployed for an Internet network.

Naming Considerations

Table 7–2 summarizes the relative advantages and disadvantages of each naming method and provides recommendations for using them in the network.

Table 7–2 Naming Methods: Advantages and Disadvantages

Naming Method	Advantages/Disadvantages	Recommended for:
Local Naming	**Advantages**: ■ Provides a relatively straightforward method for resolving net service name addresses ■ Resolves net service names across networks running different protocols **Disadvantage**: Requires local configuration of all net service name and address changes	Simple distributed networks with a small number of services that change infrequently.
Directory Naming	**Advantages**: ■ Centralizes network names and addresses in a single place, facilitating administration of name changes and updates. This eliminates the need for an administrator to make changes to what potentially could be hundreds or even thousands of clients. ■ Directory stores names for other services. ■ Tools provide simple configuration. **Disadvantage**: Requires access to a directory server	Large, complex networks (over 20 databases) that change on a frequent basis.
Easy Connect Naming	**Advantages**: ■ Requires minimal user configuration. The user can provide only the name of the database host to establish a connection. ■ Eliminates the need to create and maintain a local names configuration file (`tnsnames.ora`) **Disadvantage**: Available only in a limited environment, as indicated in the **Recommended for** column	Simple TCP/IP networks that meet the criteria listed: ■ Your client and server are connecting using TCP/IP. ■ No features requiring a more advanced connect descriptor are required
External Naming	**Advantage**: Enables administrators to load Oracle net service name into their native name service using tools and utilities with which they are already familiar **Disadvantage**: Requires a third-party naming services that cannot be administered using Oracle Net products	Networks with existing name services.

Performance Considerations

This section covers performance considerations. It includes the following topics:

- Listener Queue Size
- Session Data Unit Size for Data Transfer Optimization
- Persistent Buffer Flushing for TCP/IP

Listener Queue Size

If you anticipate receiving a large number of connection requests for a listening process (such as a listener or Oracle Connection Manager) over TCP/IP, Oracle Net enables you to configure the listening queue to be higher than the system default.

Session Data Unit Size for Data Transfer Optimization

Tuning your application to reduce the number of round trips across the network is the best way to improve your network performance. If this is done, it is also possible to optimize data transfer by adjusting the size of the session data unit (SDU).

The SDU is a buffer that Oracle Net uses to place data into before transmitting it across the network. Oracle Net sends the data in the buffer either when requested or when it is full.

Table 7–3 outlines considerations for modifying the size of the SDU.

Table 7–3 SDU Considerations

Modify SDU size when:	Do not modify SDU size when:
■ The data coming back from the server is fragmented into separate packets ■ You are on a wide area network (WAN) that has long delays ■ The packet size is consistently the same ■ Large amounts of data are returned	■ The application can be tuned to avoid the delays listed in the **Modify SDU size when** column ■ You have a higher speed network where the effect of the data transmission is negligible ■ Your requests return small amounts of data from the server

See Also: "Configuring Session Data Unit" on page 14-1

Persistent Buffer Flushing for TCP/IP

Under certain conditions for some applications using TCP/IP, Oracle Net packets may not get flushed immediately to the network. Most often, this behavior occurs when large amounts of data are streamed. The implementation of TCP/IP itself is the reason for the lack of flushing, causing unacceptable delays. To remedy this problem, specify no delays in the buffer flushing process.

See Also: *Oracle Database Net Services Reference* for further information about the `TCP.NODELAY` parameter

Planning Summary

Table 7–4 summarizes the features you can deploy.

Table 7–4 Oracle Net Feature Summary

Feature	See Also
Scalability Features	
Connection Pooling	"Enabling Connection Pooling" on page 12-2
Session Multiplexing	"Enabling Session Multiplexing" on page 11-7
Shared Server	"Configuring Dispatchers" on page 12-1
Availability Features	
Client Load Balancing	"Configuring Address List Parameters" on page 13-3
Connect-Time Failover	"Configuring Address List Parameters" on page 13-3

Table 7–4 (Cont.) Oracle Net Feature Summary

Feature	See Also
Naming Method Features	
Directory Naming	"Configuring the Directory Naming Method" on page 8-14
Easy Connect Naming	"Using the Easy Connect Naming Method" on page 8-3
Local Naming	"Configuring the Local Naming Method" on page 8-8
JDBC Drivers	*Oracle Database JDBC Developer's Guide and Reference*
Security Features	
Access Control	"Enabling Access Control" on page 11-7 to configure Oracle Connection Manager "Configuring Database Access Control" on page 9-4 to configure access control parameters in sqlnet.ora
Listener and Database Server Connection Limits	"Configuring the Listener and the Oracle Database To Limit Resource Consumption By Unauthorized Users" on page 14-8
Tuning and Performance Features	
Listener Queue Size	"Starting and Stopping a Listener" on page 10-14
SDU Size	"Configuring Session Data Unit" on page 14-1
Protocol Conversion	Chapter 11, "Configuring and Administering Oracle Connection Manager"

8

Configuring Naming Methods

This chapter describes how to configure connectivity information for client connections to the database server.

This chapter contains these topics:

- Naming Method Configuration Overview
- About Connect Descriptors
- Using the Easy Connect Naming Method
- Configuring the Local Naming Method
- Configuring the Directory Naming Method
- Configuring External Naming Methods

> **See Also:** "Naming" on page 3-12 for an overview of naming methods

Naming Method Configuration Overview

To connect to a service, clients use a **connect identifier** in the **connect string** to connect to a service. The connect identifier can be a **connect descriptor** or a simple name that maps to a connect descriptor. The connect descriptor contains:

- Network route to the service, including the location of the listener through a protocol address
- Oracle8*i* or later release database **service name** or Oracle8 database **Oracle System Identifier (SID)**

A simple name is resolved to a connect descriptor by a **naming method**. A naming method configuration consists of the following steps:

1. Select a naming method.
2. Map connect descriptors to simple names.
3. Configure clients to use the naming method.

About Connect Descriptors

A connect descriptor is comprised of one or more protocol addresses of the listener and the connect information for the destination service.

The following example shows a connect descriptor mapped to simple name called `sales`:

```
sales=
 (DESCRIPTION=
  (ADDRESS= (PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)))
```

The ADDRESS section contains the listener protocol address, and the CONNECT_DATA section contains the destination service information. In this example, the destination service is a database service named sales.us.acme.com.

When creating a connect descriptor to an Oracle9*i* or Oracle8*i* database service, you must identify the service with the SERVICE_NAME parameter. Optionally, you can identify an instance with the INSTANCE_NAME parameter, as shown in the following:

```
sales=
 (DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (INSTANCE_NAME=sales)))
```

The values for these parameters come from the SERVICE_NAMES and INSTANCE_NAME parameters in the initialization parameter file. Note that SERVICE_NAMES uses a final S. The SERVICE_NAMES parameter in the initialization parameter file is typically the **global database name**, a name which includes the database name and domain name entered during installation or database creation. For example, sales.us.acme.com has a database name of sales and a domain of us.acme.com. The INSTANCE_NAME parameter in the initialization parameter file defaults to the SID entered during installation or database creation.

> **See Also:** "Database Service and Database Instance Identification" on page 3-1

When creating a connect a descriptor for an Oracle Database, you identify the service with the SID parameter. The following example shows a connect descriptor for an Oracle Database with a SID of sales:

```
sales=
 (DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
  (CONNECT_DATA=
     (SID=sales)))
```

Naming Methods

Table 8–1 describes the naming methods that Oracle Net supports.

Table 8–1 Naming Methods

Naming Method	Description

Easy Connect Naming	Enables clients to connect to a database server without any configuration. Clients use a connect string for a simple TCP/IP address, which includes a host name and optional port and service name:
	`CONNECT username@[//]host[:port][/service_name][:server][/instance_name]`
	`Enter password: password`
	This method is recommended for simple TCP/IP environments.
	See Also: "Using the Easy Connect Naming Method" on page 8-3

Local Naming	Resolves a **net service name** stored in a `tnsnames.ora` file stored on a client Local naming is most appropriate for simple distributed networks with a small number of services that change infrequently. **See Also:** "Configuring the Local Naming Method" on page 8-8
Directory Naming	Resolves a database **service name**, net service name, or **net service alias** stored in a centralized LDAP-compliant directory server **See Also:** "Configuring the Directory Naming Method" on page 8-14
External Naming	Resolves service information stored in an a third-party naming service **See Also:** "Configuring External Naming Methods" on page 8-25

See Also: "Naming" on page 3-1

Using the Easy Connect Naming Method

The easy connect naming method eliminates the need for service name lookup in the `tnsnames.ora` files for TCP/IP environments; in fact, no naming or directory system is required if you use this method.

This naming method provides out-of-the-box TCP/IP connectivity to databases. It extends the functionality of the **host naming** method by enabling clients to connect to a database server with an optional port and service name in addition to the host name of the database:

```
CONNECT username@[//]host[:port][/service_name][:server][/instance_name]
Enter password: password
```

If you installed Oracle Database in **Typical** mode, the default service name used by the `oracle` instance is `ORCL`, and the following easy connect syntax can be used to connect to that instance:

```
CONNECT username@host/ORCL
Enter password: password
```

Table 8–2 lists the easy connect syntax elements and descriptions for each.

Table 8–2 Connect Identifier for Easy Connection Naming Method

Syntax Element	Description
`//`	***Optional.*** Specify `//` for a URL.
host	***Required.*** Specify the host name or IP address of the database server computer. The host name is domain-qualified if the local operating system configuration specifies a domain.

Table 8–2 (Cont.) Connect Identifier for Easy Connection Naming Method

Syntax Element	Description
port	***Optional.*** Specify the listening port. The default is 1521.
service_name	***Optional.*** Specify the service name of the database. If a user specifies a service name, the listener connects the user to that specific database. Otherwise, the listener connects to the database specified by the DEFAULT_SERVICE_*listener_name* parameter in the listener.ora file. If DEFAULT_SERVICE_*listener_name* is not configured for the listener and a service name is not explicitly specified by the user as part of the easy connect syntax, the listener returns an error to the user. **See Also**: *Oracle Database Net Services Reference* for more information about configuring the DEFAULT_SERVICE_*listener_name* parameter
server	***Optional.*** Specify the database server type to use. This parameter instructs the listener to connect the client to a specific type of service handler. The values for the SERVER parameter are dedicated, shared, and pooled. If *server* is not specified in easy connect syntax, the default type of server is chosen by the listener (shared server if configured, otherwise a dedicated server is used). **Note**: In Oracle Call Interface documentation, *server* is referred to as *connect_type*. **See Also**: *Oracle Database Net Services Reference* for more information about configuring the DEFAULT_SERVICE_*listener_name* parameter
instance_name	Used to identify the database instance to access. The instance name can be obtained from the INSTANCE_NAME parameter in the initialization parameter file. **See Also**: *Oracle Database Net Services Reference* for more information about configuring the INSTANCE_NAME initialization parameter

The connect identifier converts into the following connect descriptor:

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=host)(PORT=port))
  (CONNECT_DATA=
    (SERVICE_NAME=service_name)
    (SERVER=server)
    (INSTANCE_NAME=instance_name)))
```

For example, the following connect strings connect the client to database service sales.us.acme.com with a listening endpoint of 1521 on database server sales-server.

```
CONNECT scott@sales-server:1521/sales.us.acme.com
CONNECT scott@//sales-server/sales.us.acme.com
CONNECT scott@//sales-server.us.acme.com/sales.us.acme.com
```

After each of the preceding connect strings, you must enter a password to connect to the database service.

These connect strings convert into the following connect descriptor:

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
  (CONNECT_DATA=
    (SERVICE_NAME=sales.us.acme.com)))
```

For URL or JDBC connections, prefix the connect identifier with a double-slash (`//`). For example:

```
scott@[//]nineva]
Enter password: password
```

For SQL connections, preceding the connect identifier with a double-slash (`//`) is optional. For example:

```
SQL> CONNECT scott@nineva
Enter password: password
```

or

```
SQL> CONNECT scott@//nineva
Enter password: password
```

Easy Connect Naming Method Examples

This section includes various examples of easy connect naming syntax and how each string converts into a connect descriptor.

Host only, where the host name is `nineva`:

```
nineva
```

This connect string converts into the following connect descriptor:

```
(DESCRIPTION=
  (CONNECT_DATA=
     (SERVICE_NAME=nineva.us.acme.com))
  (ADDRESS=
     (PROTOCOL=TCP)
     (HOST=130.35.45.131)
     (PORT=1521)))
```

Host and port, where the host name is `nineva` and the port number is 3456:

```
nineva:3456
```

This connect string converts into the following connect descriptor:

```
(DESCRIPTION=
   (CONNECT_DATA=
       (SERVICE_NAME=nineva.us.oracle.com))
   (ADDRESS=
       (PROTOCOL=TCP)
       (HOST=130.35.45.131)
       (PORT=3456)))
```

Host and service name, where the host name is `nineva` and the service name is `wanda`:

```
nineva/wanda
```

This connect string converts into the following connect descriptor:

```
(DESCRIPTION=
  (CONNECT_DATA=
```

```
    (SERVICE_NAME=wanda))
  (ADDRESS=
    (PROTOCOL=TCP)
    (HOST=130.35.45.131)
    (PORT=1521)))
```

Host, service name and server, and instance name where the host name is `nineva`, the service name is `wanda`, the server is `dedicated`, and the instance name is `inst1`:

```
nineva/wanda:dedicated/inst1
```

This connect string converts into the following connect descriptor:

```
(DESCRIPTION=
  (CONNECT_DATA=
      (SERVICE_NAME=wanda)
      (INSTANCE_NAME=inst1)
      (SERVER=dedicated))
  (ADDRESS=
      (PROTOCOL=TCP)
      (HOST=130.35.45.131)
      (PORT=1521)))
```

Host and instance name, where the host name is `nineva` and the instance name is `inst1`:

```
nineva//inst1
```

This connect string converts into the following connect descriptor:

```
(DESCRIPTION=
   (CONNECT_DATA=
      (SERVICE_NAME=nineva.us.oracle.com)
      (INSTANCE_NAME=inst1))
   (ADDRESS=
      (PROTOCOL=TCP)
      (HOST=130.35.45.131)
      (PORT=1521)))
```

Using Easy Connect Naming on the Client

Clients can connect to Oracle Database using easy connect naming if the following conditions are met:

- Oracle Net Services software installed on the client.
- Oracle TCP/IP protocol support on both the client and database server
- No features requiring a more advanced connect descriptor are required

For large or complex environments where advanced features, such as **connection pooling**, **external procedure** calls, or **Heterogeneous Services**, which require additional connect information, are desired, easy connect naming is not suitable. In these cases, another naming method is recommended.

Easy connect naming is automatically configured at installation. Prior to using it, you may want to ensure that `EZCONNECT` is specified by the `NAMES.DIRECTORY_PATH` parameter in the `sqlnet.ora` file. This parameter specifies the order of naming methods Oracle Net can use to resolve connect identifiers to connect descriptors.

To verify that the easy connect naming method is configured:

1. Start Oracle Net Manager.

See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.
3. From the list in the right pane, select **Naming**.
4. Click the **Methods** tab.

   Check that **EZCONNECT** listed in the **Selected Methods** list. If it is not, then proceed to Step 5.
5. From the **Available Methods** list, select **EZCONNECT**, and then click the right-arrow button.
6. From the **Selected Methods** list, select **EZCONNECT**, and then use the **Promote** button to move the selection to the top of the list.
7. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file updates with the `NAMES.DIRECTORY_PATH` parameter, listing `hostname` first:

   ```
   NAMES.DIRECTORY_PATH=(ezconnect, tnsnames)
   ```

Optionally Configuring Easy Connect Naming to Use a DNS Alias

You can optionally configure a DNS alias for the host name, as provided with the host naming method in Oracle Database 11*g*. With host naming, clients use a connect string of:

```
CONNECT username@DNSalias
Enter password: password
```

To configure an alias, perform these tasks:

Task 1: Ensure Database Service is Registered with the Listener

Task 2: Establish Host Name Resolution Environment

Task 3: Connect to the Database

Task 1: Ensure Database Service is Registered with the Listener

You must register database service information with the listener. If the database can find the listener, then information about the database service is dynamically registered with the listener during **service registration**, including the service name. The listener is found if:

- The default listener named `LISTENER` running on TCP/IP, port 1521 is running
- The `LOCAL_LISTENER` parameter is set in the initialization file

If the database cannot find the listener, then configure the `listener.ora` file with the `GLOBAL_DBNAME` parameter, as shown next:

```
SID_LIST_listener=
(SID_LIST=
 (SID_DESC=
  (GLOBAL_DBNAME=sales.acme.com)
  (SID_NAME=sales)
  (ORACLE_HOME=/u01/app/oracle))
```

See Also: "Configuring Static Service Information" on page 10-6

Task 2: Establish Host Name Resolution Environment

The service name must be resolved through an IP address translation mechanism, such as **Domain Name System (DNS)**, NIS, or a centrally-maintained TCP/IP host file, `/etc/hosts`.

For example, if a service name of `sales.us.acme.com` for a database exists on a computer named `sales-server`, the entry in the `/etc/hosts` file would look like the following:

```
#IP address of server     host name       alias
10.10.150.35              sales-server    sales.us.acme.com
```

Note that the domain section of the service name must match the network domain.

Task 3: Connect to the Database

Clients can connect to the database using the alias. Using the example in "Task 2: Establish Host Name Resolution Environment", the client can use `sales.acme.com` in the connect string:

```
CONNECT username@sales.us.acme.com
Enter password: password
```

If the client and server are in the same domain of `us.acme.com`, the client needs to enter only `sales` in the connect string.

Configuring the Local Naming Method

The local naming method adds net service names to the `tnsnames.ora` file. Each net service name maps to a connect descriptor. The following example shows a net service name mapped to a connect descriptor:

```
sales=
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)))
```

In this example, the net service name `sales` is mapped to the connect descriptor contained in `DESCRIPTION`. `DESCRIPTION` contains the protocol address and identifies the destination database service.

You can configure local naming during or after installation, as described in these topics:

- Configuring the tnsnames.ora File During Installation
- Configuring the tnsnames.ora File After Installation

Configuring the tnsnames.ora File During Installation

Oracle Net Configuration Assistant enables you to configure net service names for clients. Oracle Universal Installer launches Oracle Net Configuration Assistant after software installation. The configuration varies depending on the installation mode.

This section includes the following topics:

- Administrator or Runtime Installation
- Custom Installation

Administrator or Runtime Installation

Oracle Net Configuration Assistant prompts you to configure net service names in the `tnsnames.ora` file to connect to an Oracle Database service.

Custom Installation

Oracle Net Configuration Assistant prompts you to select naming methods to use. If Local is selected, then Oracle Net Configuration Assistant prompts you to configure net service names in a `tnsnames.ora` file to connect to an Oracle Database service.

Configuring the tnsnames.ora File After Installation

You can add net service names to the `tnsnames.ora` file at any time after installation. To configure the local naming method, perform the following tasks:

Task 1: Configure Net Service Names

Task 2: Configure TNSNAMES as the First Naming Method

Task 3: Distribute Configuration

Task 4: Configure the Listener

Task 5: Connect to the Database

> **Note:** The underlying network connection must be operational before attempting to configure connectivity with Oracle Net.

Task 1: Configure Net Service Names

To configure with the local naming method, use one of the following tools:

- Oracle Enterprise Manager
- Oracle Net Manager
- Oracle Net Configuration Assistant

Oracle Enterprise Manager

To configure net service names in the `tnsnames.ora` file with **Oracle Enterprise Manager**:

1. Access the Net Services Administration page in Oracle Enterprise Manager.

   > **See Also:** "Oracle Enterprise Manager" on page 6-1

2. Select **Local Naming** from the **Administer** list, and then select the Oracle home that contains the location of the configuration files.
3. Click **Go**.

   The Local Naming page appears.
4. Click **Create**.

   The Create Net Service Name page appears.
5. Enter any name in the **Net Service Name** field.

   You can qualify the net service name with the client's domain. The net service name is automatically domain qualified if the `sqlnet.ora` file parameter `NAMES.DEFAULT_DOMAIN` is set.

See Also: "Configuring a Default Domain for Clients" on page 9-2

6. In the **Database Information** section, configure service support:
   a. Enter a destination service.

      If the destination service is an Oracle8*i* or later release database, then select **Use Service Name**, and enter a service name in the **Service Name** field. If the destination service is an Oracle8 database, then select **Use SID**, and enter an Oracle System Identifier for an instance in the **SID** field.

      See Also: "About Connect Descriptors" on page 8-1 for further information about the service name string to use

   b. Select a database connection type

      The default setting of **Database Default** is recommended for the connection type. If **shared server** is configured in the initialization parameter file, you can select **Dedicated Server** to force the listener to spawn a dedicated server, bypassing shared server configuration. If shared server is configured in the initialization parameter file and you want to guarantee the connection always uses shared server, select **Shared Server**.

      See Also: Chapter 12, "Configuring Dispatchers" for further information about shared server configuration

7. In the **Addresses** section, configure protocol support:
   a. Click **Add**.

      The Add Address page appears.

   b. From the **Protocol** list, select the protocol on which the listener is configured to listen. This protocol must also be installed on the client.

   c. Enter the appropriate parameter information for the selected protocol in the fields provided.

      See Also: *Oracle Database Net Services Reference* for protocol parameter settings

   d. Optionally, in the **Advanced Parameters** section, specify the I/O buffer space limit for send and receive operations of sessions in the **Total Send Buffer Size** and **Total Receive Buffer Size** fields.

      See Also: "Configuring I/O Buffer Space" on page 14-3 for further information

   e. Click **OK**.

      The protocol address is added to the **Addresses** section.

8. Click **OK** to add the net service name.

   The net service name is added to the Local Naming page.

See Also:

- "Creating a List of Listener Protocol Addresses" on page 13-1 to configure multiple protocol addresses
- "Configuring Advanced Connect Data Parameters" on page 13-5 to configure additional `CONNECT_DATA` options

Oracle Net Manager

To configure net service names in the `tnsnames.ora` file with **Oracle Net Manager**:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Service Naming**.
3. Click plus (+) from the toolbar, or choose **Edit** > **Create** from the menu bar.

   The Welcome page of the Net Service Name Wizard appears.

4. Enter any name in the **Net Service Name** field.

   You can qualify the net service name with the client's domain. The net service name is automatically domain qualified if the `sqlnet.ora` file parameter `NAMES.DEFAULT_DOMAIN` is set.

   See Also: "Configuring a Default Domain for Clients" on page 9-2

5. Click **Next**.

   The Protocol page appears.

6. Select the protocol on which the listener is configured to listen. Note that this protocol must also be installed on the client.
7. Click **Next**.

   The Protocol Settings page appears.

8. Enter the appropriate parameter information for the selected protocol in the fields provided.

   See Also: *Oracle Database Net Services Reference* for protocol parameter settings

9. Click **Next**.

   The Service page appears.

10. Select a release, enter a destination service, and optionally, select a database connection type.

    If the destination service is an Oracle8*i* or later release database, then select **Oracle8*i* or later**, and enter a service name in the **Service Name** field. If destination service is an Oracle8 database, then select **Oracle8 or Previous**, and enter an Oracle System Identifier for an instance in the **Database SID** field.

    See Also: "About Connect Descriptors" on page 8-1 for further information about the service name string to use

Oracle Corporation recommends that you use the default setting of **Database Default** for the connection type. If **shared server** is configured in the initialization parameter file, you can select **Dedicated Server** to force the listener to spawn a dedicated server, bypassing shared server configuration. If shared server is configured in the initialization parameter file and you want to guarantee the connection always uses shared server, select **Shared Server**.

> **See Also:** Chapter 12, "Configuring Dispatchers" for further information about shared server configuration

11. Click **Next**.

    The Test page appears.

12. Click **Test** to verify that the net service name works, or click **Finish** to dismiss the Net Service Name Wizard.

    If you click **Test**, then Oracle Net connects to the database server by using the connect descriptor information you configured. Therefore, the database and the listener must be running for a successful test. If they are not, see "Starting Oracle Net Services Components" on page 15-1 to start components before testing. During testing, a Connection Test dialog box appears, providing status and test results. A successful test results in the following message:

    ```
    The connection test was successful.
    ```

    If the test was successful, click **Close** to dismiss the Connect Test dialog box, and proceed to Step 13.

    If the test was not successful:

    a. Ensure that the database and listener are running, and then click **Test**.

    b. Click **Change Login** to change the user name and password for the connection, and then click **Test**.

13. Click **Finish** to dismiss the Net Service Name Wizard.

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

> **See Also:**
>
> - "Creating a List of Listener Protocol Addresses" on page 13-1 to configure multiple protocol addresses
> - "Configuring Advanced Connect Data Parameters" on page 13-5 to configure additional `CONNECT_DATA` options

Oracle Net Configuration Assistant

To configure net service names in the `tnsnames.ora` file with Oracle Net Configuration Assistant:

1. Start Oracle Net Configuration Assistant.

> **See Also:** "Oracle Net Configuration Assistant" on page 6-6

   The Welcome page appears.

2. Select **Local Net Service Name Configuration**, and then click **Next**.

   The Net Service Name Configuration page appears.

3. Click **Add**, and then click **Next**.

The Service Name Configuration page appears.

4. Enter a service name in the **Service Name** field.
5. Click **Next**.
6. Follow the prompts in the wizard and online help to complete net service name creation.

Task 2: Configure TNSNAMES as the First Naming Method

Configure local naming as the first method specified in the `NAMES.DIRECTORY_PATH` parameter in the `sqlnet.ora` file. This parameter specifies the order of naming methods Oracle Net uses to resolve connect identifiers to connect descriptors.

To specify local naming as the first naming method:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.
3. From the list in the right pane, select **Naming**.
4. Click the **Methods** tab.
5. From the **Available Methods** list, select **TNSNAMES**, and then click the right-arrow button.
6. From the **Selected Methods** list, select **TNSNAMES**, and then use the **Promote** button to move the selection to the top of the list.
7. Choose **File** > **Save Network Configuration**.

   The `sqlnet.ora` file updates with the `NAMES.DIRECTORY_PATH` parameter, listing `tnsnames` first:

   ```
   NAMES.DIRECTORY_PATH=(tnsnames, hostname)
   ```

Task 3: Distribute Configuration

After one client is configured, it is best to simply copy the `tnsnames.ora` and `sqlnet.ora` configuration files to the same location on the other clients. This ensures that the files are consistent. Alternatively, you can use Net8 Assistant or Net8 Configuration Assistant on every client.

Task 4: Configure the Listener

Ensure that the listener (located on the server) is configured to "listen on" the same protocol address you configured for the net service name. By default, the listener should already be configured for the TCP/IP protocol on port 1521.

See Also: Chapter 10, "Configuring and Administering Oracle Net Listener" for listener configuration details

Task 5: Connect to the Database

Clients can connect to the database using the following syntax:

```
CONNECT username@net_service_name
Enter password: password
```

Configuring the Directory Naming Method

With the directory naming method, connect identifiers are mapped to connect descriptors contained in an LDAP-compliant directory server, including Oracle Internet Directory and Microsoft Active Directory. A directory provides central administration of database services and net service names, making it easier to add or relocate services.

A database service entry is created with **Database Configuration Assistant** during installation; net service name and **net service alias** entries can be created with Oracle Enterprise Manager or Oracle Net Manager. To modify Oracle Net attributes of a database service entry and the net service name entries, use Oracle Enterprise Manager or Oracle Net Manager.

Clients can use these entries to connect to the database.

This section contains these topics:

- Directory Naming Method Configuration Steps
- Administering the OracleNetAdmins Group
- Exporting Local Naming Entries to a Directory Naming Server
- Creating Multiple Default Contexts in a Directory Naming Server
- Exporting Directory Naming Entries to a tnsnames.ora File

Directory Naming Method Configuration Steps

To configure the directory naming method, perform the following tasks:

Task 1: Verify Directory Version Compatibility

Task 2: Create or Modify Net Entries

Task 3: Configure LDAP as the First Naming Method for Client Lookups

Task 4: Configure the Listener

Task 5: Connect to the Database

Task 1: Verify Directory Version Compatibility

On the computer from which you plan to create net service names, perform the following verification steps:

1. Ensure that computer has the latest version of Oracle Net Services software.
2. Run Oracle Internet Directory Configuration Assistant to verify directory server, Oracle Context, and Oracle schema versions.

> **See Also:** *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory* for further information about configuring directory server usage

Task 2: Create or Modify Net Entries

This section covers the following topics:

- Creating Net Service Names in the Directory
- Modifying Connectivity Information for Database Service Entries
- Creating Net Service Aliases

Creating Net Service Names in the Directory

Notes:

- Only users that are members of either the `OracleNetAdmins` or `OracleContextAdmins` group can create net service name entries in a directory. To add or remove users from the `OracleNetAdmins` group, see "Adding Users To the OracleNetAdmins Group" on page 8-20.
- You can export existing net service names from a `tnsnames.ora` file. See "Exporting Local Naming Entries to a Directory Naming Server" on page 8-21.

You can configure clients to use a net service name rather than the database service entry created by Database Configuration Assistant. To create net service names, use Oracle Enterprise Manager.

1. Access the Net Services Administration page in Oracle Enterprise Manager.

   See Also: "Oracle Enterprise Manager" on page 6-1

2. Select **Directory Naming** from the **Administer** list, and then select the Oracle home that contains the location of the directory server.
3. Click **Go**.

   The Directory Naming page appears.
4. Click the **Net Service Names** tab.
5. In the **Results** section, click **Create**.

   The Create Net Service Name page with the **General** tab appears.
6. Enter any name in the **Net Service Name** field.
7. In the **Database Information** section, configure service support:

   a. Enter a destination service.

      If the destination service is for an Oracle9*i* or Oracle8*i* database, then select **Use Service Name**, and enter a service name in the **Service Name** field. If destination service is an Oracle8 database, then select **Use SID**, and enter an Oracle System Identifier for an instance in the **SID** field.

      See Also: "About Connect Descriptors" on page 8-1 for further information about the service name string to use

   b. Select a database connection type

      Oracle Corporation recommends that you use the default setting of **Database Default** for the connection type. If **shared server** is configured in the initialization parameter file, you can select **Dedicated Server** to force the listener to spawn a dedicated server, bypassing shared server configuration. If shared server is configured in the initialization parameter file and you want to guarantee the connection always uses shared server, select **Shared Server**.

      See Also: Chapter 12, "Configuring Dispatchers" for further information about shared server configuration

8. In the **Addresses** section, configure protocol support:
   a. Click **Add**.

      The Add Address page appears.

   b. From the **Protocol** list, select the protocol on which the listener is configured to listen. This protocol must also be installed on the client.

   c. Enter the appropriate parameter information for the selected protocol in the fields provided.

      See Also: *Oracle Database Net Services Reference* for protocol parameter settings

   d. Optionally, in the **Advanced Parameters** section, specify the I/O buffer space limit for send and receive operations of sessions in the **Total Send Buffer Size** and **Total Receive Buffer Size** fields.

      See Also: "Configuring I/O Buffer Space" on page 14-3 for further information

   e. Click **OK**.

      The protocol address is added to the **Addresses** section.

9. Click **OK** to add the net service name.

   The net service name is added to the **Results** section of the **Net Service Names** tab.

   See Also:

   - "Creating a List of Listener Protocol Addresses" on page 13-1 to configure multiple protocol addresses
   - "Configuring Advanced Connect Data Parameters" on page 13-5 to configure additional `CONNECT_DATA` options

Modifying Connectivity Information for Database Service Entries

---

Note: Only users that are members of either the `OracleNetAdmins` or `OracleContextAdmins` group can modify network information for a database service in a directory. To add or remove users from these groups, see "Adding Users To the OracleNetAdmins Group" on page 8-20.

When database registration with the directory completes, Database Configuration Assistant creates a database service entry in the directory. By default, this entry contains network route information that includes the location of the listener through a protocol address. You can re-create this information, if it has been removed, or modify the existing network route information.

To create or modify network route information for a database service, use Oracle Enterprise Manager.

1. Access the Net Services Administration page in Oracle Enterprise Manager.

   See Also: "Oracle Enterprise Manager" on page 6-1

2. Select **Directory Naming** from the **Administer** list, and then select the Oracle home that contains the location of the directory server.
3. Click **Go**.

   The Directory Naming page appears.
4. Click the **Database Services** tab.
5. In the **Simple Search** section, select Oracle Context and search criteria to see the net service names for a particular Oracle Context.

   The database service names display in the **Results** section.
6. In the **Results** section, select a database service, and then click **Edit**.

Creating Net Service Aliases

Notes:

- Only users that are members of either the `OracleNetAdmins` or `OracleContextAdmins` group can create or modify net service alias entries in a directory. To add or remove users from the `OracleNetAdmins` group, see "Adding Users To the OracleNetAdmins Group" on page 8-20.
- To create or access net service aliases, ensure that the Oracle home is upgraded to at least 9.2.
- Net service aliases are not supported using Microsoft Active Directory.

Net service aliases in a directory server enable clients to refer to a database service or a net service name by an alternative name. For example, a net service alias of `salesalias` can be created for a net service name of `sales`. When `salesalias` is used to connect to a database, as in `CONNECT scott@salesalias`, it will actually resolve to and use the connect descriptor information for `sales`.

There are two main uses of net service aliases:

- Use a net service alias as a way for clients to refer to a database service or net service name by another name.
- Use a net service alias in one Oracle Context for a database service or net service name in a different Oracle Context. This enables a database service or net service name to be defined once in the directory server, but referred to by clients that use other Oracle Contexts.

See Also: "Net Service Alias Entries" on page 4-5 for an overview of net service aliases

To create or modify network route information for a database service, use Oracle Enterprise Manager.

1. Access the Net Services Administration page in Oracle Enterprise Manager.

   See Also: "Oracle Enterprise Manager" on page 6-1
2. Select **Directory Naming** from the **Administer** list, and then select the Oracle home that contains the location of the directory server.

3. Click **Go**.

   The Directory Naming page appears.

4. Click the **Net Service Aliases** tab.

5. In the **Results** section, click **Create**.

   The Create Net Service Alias page appears.

6. Enter a name for the alias in the **Net Service Alias Name** field.

7. In the **Referenced Service Detail** section, enter the information in the fields described in Table 8–3.

Table 8–3 Reference Service Detail Fields in Create Net Service Alias

Field	Description
Oracle Context	Select the Oracle Context of the database service or net service name by selecting one from the list or entering one in the field.
Referenced Service Name	Select the DN of the database service or net service name.

8. Click **OK** to add the net service alias.

   The net service alias is added to the Directory Naming page.

Task 3: Configure LDAP as the First Naming Method for Client Lookups

Configure directory naming as the first method specified in the `NAMES.DIRECTORY_PATH` parameter in the `sqlnet.ora` file. This parameter specifies the order of naming methods Oracle Net uses to resolve connect identifiers to connect descriptors.

To specify directory naming as the first naming method:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.

3. From the list in the right pane, select **Naming**.

4. Click the **Methods** tab.

5. From the **Available Methods** list, select **LDAP**, and then click the right-arrow button.

6. From the **Selected Methods** list, select **LDAP**, and then use the **Promote** button to move the selection to the top of the list.

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

   The `sqlnet.ora` file updates with the `NAMES.DIRECTORY_PATH` parameter, listing `ldap` first:

   ```
   NAMES.DIRECTORY_PATH=(ldap, tnsnames, hostname)
   ```

Task 4: Configure the Listener

Ensure that the listener (located on the server) is configured to listen on the same protocol address configured for the net service name. By default, the listener is configured to listen on the TCP/IP protocol, port 1521.

See Also: Chapter 10, "Configuring and Administering Oracle Net Listener" for listener configuration details

Task 5: Connect to the Database

Clients that are configured with a default directory entry that matches the directory location of the database service or net service name can connect to the database using the following syntax:

```
CONNECT username@connect_identifier
Enter password: password
```

Clients that are configured with a default directory entry that does not match the entry's directory location cannot use the connect identifier in the connect string. Instead, these connections require the entry's distinguished name or its absolute name.

See Also:

- "Connect Identifier and Connect Descriptor Syntax Characteristics" on page 15-4 for connect identifier syntax rules
- "Absolute Name Specification for Directory Naming" on page 15-5 for absolute name usage

Administering the OracleNetAdmins Group

Members of `OracleNetAdmins` (`cn=OracleOracleNetAdmins,cn=OracleContext,...`) have create, modify, and read access to Oracle Net objects and attributes. Oracle Net Configuration Assistant establishes these access rights for this group during Oracle Context creation.

This section contains the following topics:

- Establishing Access For the OracleNetAdmins Group
- Adding Users To the OracleNetAdmins Group
- Removing Users From the OracleNetAdmins Group

Note: Members of the `OracleContextAdmins` groups can also add and delete members of the `OracleNetAdmins` group.

Establishing Access For the OracleNetAdmins Group

The owner of the `OracleNetAdmins` group can perform the following functions:

- Add or delete members from the `OracleNetAdmins` group
- Add or delete groups which are owners of the `OracleNetAdmins` group

By default, the owner of the `OracleNetAdmins` group is the `OracleNetAdmins` group itself. This means that any member of the `OracleNetAdmins` group can add or delete other members from the `OracleNetAdmins` group. If you prefer that another group other than `OracleNetAdmins` add or delete other `OracleNetAdmins` members, you can change the owner attribute of the `OracleNetAdmins` group to another group.

The owner cannot be an individual user entry, such as `cn=scott`, but must be a group entry, where the group entry is one comprised of the LDAP schema object classes `GroupOfUniqueNames` and `orclPriviledgeGroup`.

To add a group as an owner of an `OracleNetAdmins` group:

1. Create an **LDAP Data Interchange Format (LDIF)** file:

   a. Specify the group you want to add as an owner.

   You can use the following sample LDIF file. Enter the appropriate **distinguished name (DN)** for `cn=OracleNetAdmins` and the DN of the group that you want to add.

```
dn: cn=OracleNetAdmins,cn=OracleContext,...
    changetype: modify
    add: owner
    owner: <DN of group to add>
```

   b. Optionally, specify the group to delete as an owner.

```
dn: cn=OracleNetAdmins,cn=OracleContext,...
    changetype: modify
    add: owner
    owner: <DN of group to add>
```

   For example, the following LDIF syntax changes the ownership from the `OracleNetAdmins` group to another group named `cn=AcmeSecurityAdmins`. The group can be either be inside or outside the Oracle Context; in this case, it is outside the Oracle Context.

```
dn: cn=OracleNetAdmins,cn=OracleContext,...
     changetype: modify
     add: owner
     owner: cn=AcmeSecurityAdmins

dn: cn=OracleNetAdmins,cn=OracleContext,...
     changetype: modify
     delete: owner
     owner: cn=OracleNetAdmins,cn=OracleContext,...
```

2. Enter the following `ldapmodify` syntax at the command line to delete the user:

```
$ ldapmodify -h directory_host -p port -D binddn -q -f ldif_file
```

Table 8–4 ldapmodify Arguments

Argument	Description
`-h` *directory_host*	Specify the directory server host.
`-p` *port*	Specify the listening TCP/IP port for the directory server. If you do not specify this option, the default port (389) is used.
`-D` *binddn*	Specify the directory administrator or user DN.
`-q`	Prompts for a single bind password to be entered on a subsequent line.
`-f` *ldif_file*	Specify the input file name.

Adding Users To the OracleNetAdmins Group

To add a user to the `OracleNetAdmins` group with `ldapmodify`:

1. Create an LDIF file that specifies that you want to add a user to the `OracleNetAdmins` group.

   You can use the following sample LDIF file. Use the appropriate DN for `cn=OracleNetAdmins` and the user that you want to add.

```
dn: cn=OracleNetAdmins,cn=OracleContext,...
```

```
changetype: modify
add: uniquemember
uniquemember: <DN of user being added to group>
```

2. Enter the following `ldapmodify` syntax at the command line to add a user:

```
$ ldapmodify -h directory_host -p port -D binddn -q -f ldif_file
```

Removing Users From the OracleNetAdmins Group

To remove a user from the `OracleNetAdmins` group with `ldapmodify`:

1. Create an LDIF file that specifies that you want to add a user to the `OracleNetAdmins` group.

   You can use the following sample LDIF file. Enter the appropriate DN for `cn=OracleNetAdmins` and the user that you want to delete.

```
dn: cn=OracleNetAdmins,cn=OracleContext,...
changetype: modify
delete: uniquemember
uniquemember: <DN of user being deleted from group>
```

2. Enter the following `ldapmodify` syntax at the command line to delete the user:

```
$ ldapmodify -h directory_host -p port -D binddn -q -f ldif_file
```

Exporting Local Naming Entries to a Directory Naming Server

This section explains how to export data stored in a `tnsnames.ora` file to a directory server.

- Exporting Net Service Names from a tnsnames.ora File

Exporting Net Service Names from a tnsnames.ora File

If a `tnsnames.ora` file already exists, then its net service names can be exported to a directory server. The export is performed for one domain at a time.

The tasks to export data from a `tnsnames.ora` file are as follows:

Task 1: Create Structure in Directory Server

Task 2: Create Oracle Contexts

Task 3: Configure Directory Server Usage

Task 4: Export Objects To a Directory Server

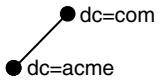
Note: These tasks assume that you have a directory server installed and running.

Task 1: Create Structure in Directory Server

In the directory server, create the **directory information tree (DIT)** with the structure in which you want to import net service names. Create the structure leading up to the top of the **Oracle Context**.

For example, if the `tnsnames.ora` file supports a domain structure `acme.com` and you want to replicate this domain in the directory, then create domain component entries of `dc=com` and `dc=acme` in the directory, as depicted in Figure 8–1.

Figure 8–1 acme.com in Directory Server



You can replicate the domain structure you currently use with `tnsnames.ora`, or you can develop an entirely different structure. Introducing an entirely different structure can change the way in which clients enter the net service name in the connect string. Therefore, Oracle Corporation recommends considering relative and absolute naming issues prior to changing the structure.

See Also:

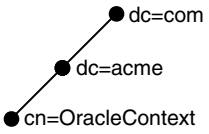
- Directory server vendor documentation for directory entry configuration instructions
- "Client Connections Using Directory Naming" on page 4-9

Task 2: Create Oracle Contexts

Create an Oracle Context under each DIT location that you created in Task 1. The Oracle Context has a **relative distinguished name (RDN)** of `cn=OracleContext`. The Oracle Context stores network object entries, as well as other entries for other Oracle components. In Figure 8–2, `cn=OracleContext` is created under `dc=acme,dc=com`.

To create the Oracle Context, use Oracle Internet Directory Configuration Assistant to create a DIT structure that looks similar to the one in Figure 8–2.

Figure 8–2 Oracle Context



See Also:

- Chapter 4, "Configuration Management Concepts" for further information about the Oracle Context
- *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory* for instructions on creating an Oracle Context

Task 3: Configure Directory Server Usage

If not already done as a part of creating the Oracle Contexts, configure the Oracle home for directory server usage. The Oracle home you configure should be the one that will perform the export.

See Also: *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory* for further information about configuring directory server usage

Task 4: Export Objects To a Directory Server

To export net service names contained in a `tnsnames.ora` file to a directory, use either Oracle Enterprise Manager or Oracle Net Manager.

Oracle Enterprise Manager

1. Access the Net Services Administration page in Oracle Enterprise Manager.

   See Also: "Oracle Enterprise Manager" on page 6-1

2. Select **Directory Naming** from the **Administer** list, and then select the Oracle home that contains the location of the directory server.
3. Click **Go**.

   The Directory Naming page appears.
4. Click the **Net Service Names** tab.
5. In the **Related Links** section, click **Import Net Service Names To Directory Server**.

   The Import Net Service Names To Directory Server page appears.
6. From the **Oracle Context** list in the **Oracle Internet Directory Server Destination** section, select the Oracle Context to which you want to export the selected net service names.
7. In the **Net Service Names to Import** section, select the net service names.
8. Click **Add** to add the net service names to the directory.

   The net service name is added to the Directory Naming page.

Oracle Net Manager

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. If the `tnsnames.ora` file you want to export is not the one loaded into Oracle Net Manager, then use **File** > **Open Network Configuration** to select the `tnsnames.ora` file to export to the directory.
3. Choose **Command** > **Directory** > **Export Net Service Names** from the menu bar.

   The Directory Server Migration Wizard starts.
4. Click **Next**.

   If net service names with multiple domain were detected in the `tnsnames.ora` file, then the Select Domain page appears. Continue to Step 5.

   If the net service names are not domain qualified, the Select Net Service Names page appears. Skip to Step 6.
5. Select the network domain whose net service names you want to export, and then click **Next**.

   The Select Net Service Names page appears.
6. Select the net service names from the list to export, and then click **Next**.

   The Select Destination Context page appears.
7. In the Select Destination Context page, perform the following:

   a. From the **Directory Naming Context** list, select the directory entry that contains the Oracle Context. The directory naming context is part of a directory subtree that contains one or more Oracle Contexts.

b. From the **Oracle Context** list, select the Oracle Context to which you want to export the selected net service names.

c. Click **Next**.

The Directory Server Update page appears with the status of the export operation.

8. Click **Finish** to dismiss the Directory Server Migration Wizard.

Creating Multiple Default Contexts in a Directory Naming Server

If you want clients to use discovery in directories which have more than one `oracleContext`, you can define the `orclCommonContextMap` attribute in the base `admin` context; this will override the `orclDefaultSubscriber` attribute. During name lookup the discovery operation will return both values, and the client will decide based on these which `oracleContext` to use.

If the `orclCommonContextMap` attribute is not defined the `orclDefaultSubscriber` will be used as the default. If `orclCommonContextMap` is defined, then the client will find the default `oracleContext` which is associated with its DNS domain in the `orclCommonContextMap`.

To enable multiple default contexts, define the `orclCommonContextMap` with a list of associations between a domain and a DN to be used as the default `oracleContext`. A sample LDIF file entry is shown here:

```
$ ldapmodify -v -h nineva -p 1389 -D cn=orcladmin -q
 dn: cn=Common,cn=Products,cn=OracleContext
 replace: orclCommonContextMap
 orclCommonContextMap:
 (contextMap=
   (domain_map=(domain=us.acme.com)(DN="dc=acme,dc=com"))
   (domain_map=(domain=uk.acme.com)(DN="dc=sales,dc=com"))
  )
```

In practice, the contextMap entry must be entered without line breaks, as in the following example:

```
orclCommonContextMap:(contextMap=(domain_map=(domain=us.acme.com)(DN="
dc=acme,dc=com"))(domain_map=(domain=uk.acme.com)(DN="dc=sales,dc=com")))
```

> **See Also:** *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory* for further information on how to configure the directory for context mapping

Exporting Directory Naming Entries to a tnsnames.ora File

Once you create the directory naming entries, consider exporting the entries to a local `tnsnames.ora` file, and distributing that files to clients. Clients can use the locally saved file when a directory server is temporarily unavailable.

To export directory naming entries to a local `tnsnames.ora` file, use Oracle Enterprise Manager:

1. Access the Oracle Net Administration page in Oracle Enterprise Manager.

   > **See Also:** "Oracle Enterprise Manager" on page 6-1

2. Select **Directory Naming** from the **Administer** list, and then select the Oracle home that contains the location of the directory server.

3. Click **Go**.

The Directory Naming page appears.

4. Click the **Net Service Names** tab.
5. In the **Simple Search** section, select Oracle Context and search criteria to see the net service names for a particular Oracle Context.

   The net service names display in the **Results** section.
6. In the **Results** section, click **Save to tnsnames.ora**.

   The Processing: Create tnsnames.ora File page appears, informing you of the creation process.

Configuring External Naming Methods

External naming refers to the method of resolving a net service name, stored in a non-Oracle naming service, to a network address. External naming services include:

- Network Information Service
- Distributed Computing Environment (DCE) Cell Directory Service (CDS)

Network Information Service

Organizations and corporations already using network information service (NIS) as part of their systems infrastructure have the option to store net service names and addresses in NIS, using NIS external naming.

When a user gives a command such as

```
sqlplus scott@payroll
Enter password: password
```

(where `payroll` is an Oracle service) NIS external naming on the node running the client program (or database server acting as a client program) contacts an NIS server located somewhere in the network, and passes the net service name to the NIS server. The NIS server resolves the net service name into a Oracle Net address and returns this address to the client program (or server acting as a client program). The client program then uses this address to connect to the Oracle Database.

A computer that acts as an NIS server runs a program called `ypserv`, which handles name requests. The `ypserv` program stores different types of data in special files called **map**s. For example, passwords are stored in a map called `passwd.byname`. Oracle Database service names are stored in a map called `tnsnames`.

When a user issues a connect string, NIS external naming uses an RPC call to contact the `ypserv` program and passes the Oracle net service name `payroll` and the name of the map—`tnsnames`. The `ypserv` program looks in the `tnsnames` map for the name `payroll` and its corresponding value, which is the address for the net service name. The address is returned to the client, and the client program (or server acting as a client program) uses this address to contact the database server.

Note: The NIS external naming method is not available on all platforms. Use the `adapters` command to check availability of NIS external naming on your system. If available, it will be listed under Oracle Net naming methods, as follows:

```
$ adapters

Installed Oracle Net naming methods are:

Local Naming (tnsnames.ora)
Oracle Directory Naming
Oracle Host Naming
NIS Naming
```

See Oracle platform-specific documentation for more information.

This section includes the following topics:

- Task 1: Configure NIS Servers to Support the NIS External Naming
- Task 2: Configure Clients

Task 1: Configure NIS Servers to Support the NIS External Naming

Before configuring servers to support the NIS external naming, make sure that NIS is configured and running on the NIS servers that need to resolve Oracle Database net service names. Consult your NIS documentation for specifics.

This task includes the following additional tasks:

- Adding the tnsnames Map to the Existing Set of NIS Maps
- Verifying that the tnsnames Map Has Been Properly Installed

Adding the tnsnames Map to the Existing Set of NIS Maps To add the `tnsnames` map to the existing set of NIS maps:

1. Create a `tnsnames.ora` file, as specified in "Configuring the Local Naming Method" on page 8-8.

   Note: Keep a copy of the `tnsnames.ora` file, preferably in `$ORACLE_HOME/network/admin` directory. You may need to use this file again later to load net service names into the NIS map.

   ---

2. Convert the contents of the `tnsnames.ora` file to a `tnsnames` map using the `tns2nis` program.

   Note: The `tns2nis` program is supplied with NIS External Naming.

   For example, run `tns2nis` on the command line with one argument:

   ```
   tns2nis tnsnames.ora
   ```

   The `tns2nis` program reads the `tnsnames.ora` file from the current directory. (If `tnsnames.ora` file is not located in the current directory, you can use a full

path name to specify its location—for example, `/etc/tnsnames.ora` or `$ORACLE_HOME/network/admin/tnsnames.ora`).

The `tnsnames` map is then written into the current working directory.

3. Copy `tnsnames` to the NIS server, if it is not already there.
4. Install the `tnsnames` map using `makedbm`, which is an NIS program.

> **Note:** This step should be performed by the person in charge of NIS administration.

The `makedbm` program converts the `tnsnames` map into two files that the NIS server can read. The location of these files is operating system specific.

> **See Also:** Oracle operating system-specific documentation for details

For example, to generate and install a `tnsnames` map on the Solaris Operating System, as the `root` user, enter the following at the command line:

```
# makedbm tnsnames /var/yp/'domainname'/tnsnames
```

Verifying that the tnsnames Map Has Been Properly Installed You can test the NIS server to see if the map has been installed properly by typing a command with the format:

```
ypmatch net_service_name tnsnames
```

For example, you might enter:

```
ypmatch payroll.com tnsnames
```

This returns the length of the address (in characters) followed by the address; for example:

```
99 (description=(address=(protocol=tcp)
 (host=garlic)(port=1999)))
 (connect_data=(service_name=dirprod)))
```

Task 2: Configure Clients

To configure clients, configure NIS as the first method specified in the `NAMES.DIRECTORY_PATH` parameter in the `sqlnet.ora` file. This parameter specifies the order of naming methods Oracle Net can use to resolve connect identifiers to connect descriptors.

1. Start Oracle Net Manager.

> **See Also:** "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.
3. From the list in the right pane, select **Naming**.
4. Click the **Methods** tab.
5. From the **Available Methods** list, select **NIS**, and then click the right-arrow button.
6. In the **Selected Methods** list, select **NIS**, and then use the **Promote** button to move the selection to the top of the list.

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

   The `sqlnet.ora` file updates with the `NAMES.DIRECTORY_PATH` parameter, listing `nis` first:

   ```
   NAMES.DIRECTORY_PATH=(nis, hostname, tnsnames)
   ```

Distributed Computing Environment (DCE) Cell Directory Service (CDS)

See Also: *Oracle Database Advanced Security Administrator's Guide* for instructions about how to configure **Cell Directory Services (CDS)**

9

Configuring Profiles

This chapter describes how to configure client and server configuration parameters in profiles. A **profile** is a collection of parameters that specifies preferences for enabling and configuring Oracle Net features on the client or database server. A profile is stored and implemented through the `sqlnet.ora` file.

This chapter contains these topics:

- Profile Configuration Overview
- Profile Configuration During Installation
- Configuring Client Attributes for Names Resolution
- Configuring Database Access Control
- Configuring Advanced Profile Information
- Configuring External Naming Methods
- Configuring Oracle Advanced Security

Profile Configuration Overview

You can use a profile to:

- Specify the client domain to append to unqualified names
- Prioritize **naming method**s
- Enable logging and tracing features
- Route connections through specific processes
- Configure parameters for **external procedure**
- Configure **Oracle Advanced Security**
- Use protocol-specific parameters to restrict access to the database

Profile Configuration During Installation

Oracle Universal Installer launches **Oracle Net Configuration Assistant** after software installation on the client and server. Oracle Net Configuration Assistant configures the order of the naming methods that the computer uses to resolve a **connect identifier** to a **connect descriptor**.

Configuration with the Oracle Net Configuration Assistant during installation results in the following entries in the `sqlnet.ora` file:

```
NAMES.DIRECTORY_PATH=(ezconnect,tnsnames)
```

NAMES.DIRECTORY_PATH specifies the priority order of the naming methods to use to resolve connect identifiers.

If the installed configuration is not adequate, you can use **Oracle Net Manager** to enhance the sqlnet.ora configuration.

Configuring Client Attributes for Names Resolution

The following sections describe available client configuration options:

- Configuring a Default Domain for Clients
- Prioritizing Naming Methods
- Routing Connection Requests

Configuring a Default Domain for Clients

In environments where the client often requests names from a specific domain, it is appropriate to set a default domain in the client sqlnet.ora file with the NAMES.DEFAULT_DOMAIN parameter.

When a default domain is set, it is automatically appended to any unqualified net service name given in the connect string, and then compared to net service names stored in a tnsnames.ora file.

For example, if the client tnsnames.ora file contains a net service name of sales.us.acme.com, the user can enter the following connect string:

```
CONNECT scott@sales
Enter password: password
```

In this example, sales gets searched as sales.us.acme.com.

If the connect string includes the domain extension, such as in CONNECT scott@sales.us.acme.com, the domain is not appended. If a net service name in a tnsnames.ora file is not domain qualified and this parameter is set, the net service name must be entered with a dot (.). For example, if domain is set to us.acme.com and the client tnsnames.ora file contains a net service name of sales, the user would enter the following connect string:

```
CONNECT scott@sales
Enter password: password
```

To specify a default domain:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.
3. From the list in the right pane, select **Naming**.
4. Click the **Methods** tab.
5. In the **Default Domain** field, enter the domain.
6. Choose **File** > **Save Network Configuration**.

   The sqlnet.ora file should contain an entry that looks like the following:

   ```
   NAMES.DEFAULT_DOMAIN=us.acme.com
   ```

Prioritizing Naming Methods

After naming methods are configured, as described in Chapter 8, "Configuring Naming Methods", they must be prioritized. The naming method at the top of the list is used first to resolve a connect identifier. If the first naming method in the list is unable to resolve the connect identifier, then the second method in the list is used.

To specify the order of naming methods:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.
3. From the list in the right pane, select **Naming**.
4. Click the **Methods** tab.

   Table 9–1 describes the naming method values listed in the **Methods** tab.

Table 9–1 Naming Method Values

Naming Method Value	Description
`TNSNAMES` (**local naming** method)	Select to resolve a **net service name** through the `tnsnames.ora` file on the client. **See Also:** "Configuring the Local Naming Method" on page 8-8
`LDAP` (**directory naming** method)	Select to resolve a database service name, net service name, or **net service alias** through a **directory server**. **See Also:** "Configuring the Directory Naming Method" on page 8-14
`EZCONNECT` or `HOSTNAME` (easy connect naming or **host naming** method)	Select to enable clients to use a TCP/IP connect identifier, consisting of a host name and optional port and service name, or resolve a host name alias through an existing names resolution service or centrally maintained set of `/etc/hosts` files. **See Also:** "Using the Easy Connect Naming Method" on page 8-3
`CDS` (**CDS** external naming method)	Set to resolve an Oracle Database name in a Distributed Computing Environment (DCE) environment. **See Also:** *Oracle Database Advanced Security Administrator's Guide*
`NIS` (**Network Information Service (NIS)** external naming method)	Set to resolve service information through an existing NIS. **See Also:** "Network Information Service" on page 8-25

5. Select naming methods from the **Available Methods** list, and then click the right-arrow button.

   The selected naming methods move to the **Selected Methods** list.

6. Order the naming methods according to the order in which you want Oracle Net to try to resolve the net service name or database service name. Select a naming method in the **Selected Methods** list, and then click **Promote** or **Demote** to move the selection up or down in the list.
7. Choose **File** > **Save Network Configuration**.

The sqlnet.ora file updates with the NAMES.DIRECTORY_PATH parameter:

```
NAMES.DIRECTORY_PATH=(ldap, tnsnames)
```

Routing Connection Requests

Clients and servers acting as clients can be configured so connection requests are directed to a specific process. To configure this feature so that all connections use a particular server, you choose the Always Use Dedicated Server option in Oracle Net Manager. This sets the sqlnet.ora parameter USE_DEDICATED_SERVER to force the listener to spawn a dedicated server for all network sessions from the client. The result is a dedicated server connection, even if shared server is configured.

To route connection requests:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.
3. From the list in the right pane, select **General**.
4. Click the **Routing** tab.
5. Select the preferred way that you want connection requests routed.

   See Also: Table 9–3, " Advanced Settings in sqlnet.ora" for a description of the fields and options

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

Configuring Database Access Control

You can configure the sqlnet.ora file to allow access to some clients and deny access to others. Table 9–2 describes the available settings.

Table 9–2 Access Control Settings in sqlnet.ora

Oracle Net Manager Field/Option	sqlnet.ora File Parameter	Description
Check TCP/IP client access rights	TCP.VALIDNODE_CHECKING	Use to specify whether to screen access to the database. If this field is selected, Oracle Net Manager checks the parameters TCP.EXCLUDED_NODES and TCP.VALIDNODE_CHECKING to determine which clients to allow access to the database. If this field is deselected, Oracle Net Manager does not screen clients.
Clients excluded from access	TCP.EXCLUDED_NODES	Use to specify which clients using the TCP/IP protocol are denied access to the database.
Clients allowed to access	TCP.INVITED_NODES	Use to specify which clients using the TCP/IP protocol are allowed access to the database.

To configure database access control:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 5-3

1. In the navigator pane, expand **Local** > **Profile**.
2. From the list in the right pane, select **General**.
3. Click the **Access Rights** tab.
4. Select the **Check TCP/IP client access rights** option.
5. In the **Clients excluded from access** and **Clients allowed to access** fields, enter either a host name or an IP address for a client that you wish to include or exclude, using commas to delimit entries placed on the same line.

Configuring Advanced Profile Information

Table 9–3 describes the advanced `sqlnet.ora` file settings that you can set.

Table 9–3 Advanced Settings in sqlnet.ora

Oracle Net Manager Field/Option	sqlnet.ora File Parameter	Description
Send operation Time Out	`SQLNET.SEND_TIMEOUT`	Use to specify the time, in seconds, for a database server to complete a send operation to clients to complete after connection establishment. For environments in which clients shut down on occasion or abnormally, setting this parameter is recommended. If the database server is unable to complete a send operation in the time specified, then it logs an `ORA-12535: TNS:operation timed out` and `ORA-12608: TNS: Send timeout occurred` to the `sqlnet.log` file. Without this parameter, the database server continues to send responses to clients that are unable to receive data due to a downed computer or a busy state. You can also set this parameter on the client side to specify the time, in seconds, for a client to complete send operations to the database server after connection establishment. Without this parameter, the client may continue to send requests to a database server already saturated with requests.
Receive operation Time Out	`SQLNET.RECV_TIMEOUT`	Use to specify the time, in seconds, for a database server to wait for client data after connection establishment. A client must send some data within time interval. For environments in which clients shut down on occasion or abnormally, setting this parameter is recommended. If a client does not send any data in time specified, then the database server logs an `ORA-12535: TNS:operation timed out` and `ORA-12609: TNS: Receive timeout occurred` to the `sqlnet.log` file. Without this parameter, the database server continues to wait for data from clients that may be down or are experiencing difficulties. You can also use this setting on the client side to specify the time, in seconds, for a client to wait for response data from the database server after connection establishment. Without this parameter, the client may wait for a long period of time for a response from a database server saturated with requests.

Table 9–3 (Cont.) Advanced Settings in sqlnet.ora

Oracle Net Manager Field/Option	sqlnet.ora File Parameter	Description
Connection Time Out	SQLNET.INBOUND_CONNECT_TIMEOUT	Specify the time, in seconds, for a client to connect with the database server and provide the necessary authentication information. **See Also:** "Configuring the Listener and the Oracle Database To Limit Resource Consumption By Unauthorized Users" on page 14-8 for complete information about configuring this setting
Total Send Buffer Size	SEND_BUF_SIZE	Specify the buffer space limit for send operations of sessions. **See Also:** "Configuring I/O Buffer Space" on page 14-3 for complete information about configuring this setting
Total Receive Buffer Size	RECV_BUF_SIZE	Specify the buffer space limit for receive operations of sessions. **See Also:** "Configuring I/O Buffer Space" on page 14-3 for complete information about configuring this setting
TNS Time Out Value	SQLNET.EXPIRE_TIME	Use to specify a specify the time interval, in minutes, to send a probe to verify that client/server connections are active. Setting a value greater than 0 ensures that connections are not left open indefinitely, due to an abnormal client termination. If the probe finds a terminated connection, or a connection that is no longer in use, it returns an error, causing the server process to exit. This setting is intended for the database server, which typically handles multiple connections at any one time. Limitations on using this terminated connection detection feature are: ■ It is not allowed on bequeathed connections. ■ Though very small, a probe packet generates additional traffic that may downgrade network performance. ■ Depending on which operating system is in use, the server may need to perform additional processing to distinguish the connection probing event from other events that occur. This can also result in downgrading network performance.
Client Registration ID	SQLNET.CLIENT_REGISTRATION	Use to specify a unique identifier for a client. This identifier is passed to the listener with any connection request. The identifier can be any string up to 128 characters long.

Table 9–3 (Cont.) Advanced Settings in sqlnet.ora

Oracle Net Manager Field/Option	sqlnet.ora File Parameter	Description
Logon Authentication Protocol Version	SQLNET.ALLOWED_LOGON_VERSION	Use to define the minimum Oracle Database client version that is allowed to attempt connections to database instances under the control of the given code tree. Each connection attempt is tested. If the client or server does not meet the minimum version specified by its partner, then authentication fails with an ORA-28040 error. Supported values include: ■ 11 for Oracle Database 11*g* authentication protocols (recommended for strongest protection) ■ 10 for Oracle Database 10*g* authentication protocols ■ 9 for Oracle9*i* authentication protocols ■ 8 for Oracle8*i* authentication protocols The default value is 8. Note the following implications of setting the value to 11: ■ To take advantage of the password protections introduced in Oracle Database 11*g*, users must change their passwords so that old password verifiers are purged from the system. ■ Pre-Oracle Database Release 11.1 client applications or JDBC thin clients cannot authenticate to the Oracle database using password-based authentication. **See Also:** *Oracle Database Advanced Security Administrator's Guide*
Turn Off UNIX Signal Handling	BEQUEATH_DETACH	Use to turn on or off UNIX signal handling. Since the client application spawns a server process internally through the Bequeath protocol as a child process, the client application becomes responsible for cleaning up the child process when it completes. When the server process completes its connection responsibilities, it becomes a defunct process. Signal handlers are responsible for cleaning up these defunct processes. Setting this parameter configures the client profile to pass this process to the UNIX initialization process by disabling signal handlers.
Disable Out-of-Band Break	DISABLE_OOB	Use to turn on or off out-of-band breaks. If deselected or set to off, enables Oracle Net to send and receive "break" messages using urgent data provided by the underlying protocol. If selected or set to on, disables the ability to send and receive "break" messages using urgent data provided by the underlying protocol. Once enabled, this feature applies to all protocols used by this client. **See Also:** Oracle operating system-specific documentation to determine if the protocol supports urgent data requests. TCP/IP is an example of a protocol that supports this feature.

To set advanced features:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.

3. From the list in the right pane, select **General**.
4. Click the **Advanced** tab.
5. Enter the values for the fields or options you want to set.

   See Also: Table 9–3, "Advanced Settings in sqlnet.ora" for a description of the fields and options

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

Configuring External Naming Methods

Configure required client parameters needed for the NIS external naming or the CDS external naming method in the profile. Table 9–4 describes the `sqlnet.ora` file external naming settings that you can set.

Table 9–4 External Naming Methods Settings in sqlnet.ora

Oracle Net Manager Field	sqlnet.ora File Parameter	Description
Cell Name	`NAMES.DCE.PREFIX`	Enter a valid DCE cell name (prefix).
Meta Map	`NAMES.NIS.META_MAP`	Specify the map, a special file that contains the database service name.

To configure external naming method parameters:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.
3. From the list in the right pane, select **Naming**.
4. Enter a value in the appropriate field for the external naming method you are using.
5. Choose **File** > **Save Network Configuration**.

Configuring Oracle Advanced Security

Oracle Advanced Security enables data encryption and integrity checking, enhanced authentication, single sign-on, and support for DCE. Oracle Advanced Security also provides centralized user management on LDAP-compliant directory servers and certificate-based single sign-on; this functionality relies on the **Secure Sockets Layer (SSL)**.

To configure a client or server to use Oracle Advanced Security features:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Profile**.
3. From the list in the right pane, select **Oracle Advanced Security**.

   Each Oracle Advanced Security tab page enables you to configure a separate set of parameters.

See Also:

- Choose the **Help** button on the particular tab page
- Oracle Advanced Security procedural topics in the Oracle Net Manager online help. To access these topics in the online help, choose **Oracle Advanced Security** > **How To**.
- *Oracle Database Advanced Security Administrator's Guide* for further information about configuration

4. Select or edit options as applicable.
5. Choose **File** > **Save Network Configuration**.

10

Configuring and Administering Oracle Net Listener

Oracle Net Listener is a separate process that runs on the database server computer. It receives incoming client connection requests and manages the traffic of these requests to the database server. This chapter describes how to configure the listener to accept client connections.

This chapter includes the following topics:

- Oracle Net Listener Configuration Overview
- Oracle Net Listener Configuration During Installation
- Customizing Oracle Net Listener Configuration
- Configuring Service Registration
- Listener Administration

See Also:

- Chapter 3, "Connectivity Concepts" for a description of how the listener is used during an initial connection request
- Chapter 5, "Architecture of Oracle Net Services" for an architectural overview of the listener

Oracle Net Listener Configuration Overview

Note: Oracle Database 10*g* and later databases require a version 10 or later listener. Earlier versions of the listener *are not* supported for use with Oracle Database 10*g* and later databases. However, you can use a version 10 listener with previous versions of Oracle Database.

A listener is configured with one or more listening protocol addresses, information about supported services, and parameters that control its runtime behavior. The listener configuration is stored in a configuration file named `listener.ora`.

Because all of the configuration parameters have default values, it is possible to start and use a listener with no configuration. This default listener has a name of `LISTENER`, supports no services on startup, and listens on the following TCP/IP protocol address:

```
(ADDRESS=(PROTOCOL=tcp)(HOST=host_name)(PORT=1521))
```

Supported services, that is, the services to which the listener forwards client requests, can be configured in the `listener.ora` file or this information can be dynamically registered with the listener. This dynamic registration feature is called **service registration**. The registration is performed by the **PMON process**—an instance background process—of each database instance that has the necessary configuration in the database initialization parameter file. Dynamic service registration does not require any configuration in the `listener.ora` file.

See Also: "Listener Architecture" on page 5-7

Service registration offers the following benefits:

- Simplified configuration

  Service registration reduces the need for the `SID_LIST_`*`listener_name`* parameter setting, which specifies information about the databases served by the listener, in the `listener.ora` file.

  Note: The `SID_LIST_`*`listener_name`* parameter is still required if you are using Oracle Enterprise Manager to manage the database.

- Connect-time failover

  Because the listener always monitors the state of the instances, service registration facilitates automatic failover of the client connect request to a different instance if one instance is down.

  In a static configuration model, a listener starts a dedicated server when it receives a client request. If the instance is not up, the server will return an "`Oracle not available`" error message.

- Connection load balancing

  Service registration enables the listener to forward client connect requests to the least-loaded instance and **dispatcher** or **dedicated server**. Service registration balances the load across the **service handler**s and nodes.

  See Also:

  - "Listener Architecture" on page 5-7
  - "Configuring Service Registration" on page 10-10
  - "Configuring Address List Parameters" on page 13-3
  - "Configuring Connection Load Balancing" on page 13-7

Oracle Net Listener Configuration During Installation

Oracle Universal Installer launches **Oracle Net Configuration Assistant** during software installation. Oracle Net Configuration Assistant configures the listening protocol address and service information for Oracle Database.

During an Enterprise Edition or Standard Edition installation on the database server, Oracle Net Configuration Assistant automatically configures a listener with a name of `LISTENER` that has a TCP/IP listening protocol address for Oracle Database. During a Custom installation, Oracle Net Configuration Assistant prompts you to configure a listener name and a protocol address of your choice.

Additionally, a listening IPC protocol address for **external procedure** calls is automatically configured, regardless of the installation type.

Oracle Net Configuration Assistant also automatically configures service information for the external procedures in the `listener.ora` file.

Example 10–1 shows a sample `listener.ora` file. The `LISTENER` entry defines the listening protocol address for a listener named `LISTENER`, and the `SID_LIST_LISTENER` entry provides information about the services statically supported by the listener `LISTENER`.

Example 10–1 Example listener.ora File

```
LISTENER=
  (DESCRIPTION=
    (ADDRESS_LIST=
      (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
      (ADDRESS=(PROTOCOL=ipc)(KEY=extproc))))
SID_LIST_LISTENER=
  (SID_LIST=
    (SID_DESC=
      (SID_NAME=plsextproc)
      (ORACLE_HOME=/oracle10g)
      (PROGRAM=extproc)))
```

If you are using the IPC protocol, you can improve performance by specifying the maximum number of concurrent IPC connection requests to match your expected connection requests. In `listener.ora` for example, you can specify the value as in the following example:

```
listener_name=(description=(address=(protocol=ipc)(key=listener0)(queuesize=50)))
```

> **See Also:** *Oracle Database Net Services Reference* for further information about identifying listeners by unique names and creating multiple listener entries in the `listener.ora` file

Customizing Oracle Net Listener Configuration

If the default or installed configuration is not adequate for a particular environment, you can use **Oracle Net Manager** to customize the `listener.ora` configuration.

This section contains these configuration topics:

- Configuring Listening Protocol Addresses
- Configuring Access to Oracle JServer
- Handling Large Volumes of Concurrent Connection Requests
- Configuring Static Service Information
- Default Oracle Net Listener Administration

Configuring Listening Protocol Addresses

To configure additional protocol addresses for the listener, use Oracle Enterprise Manager:

1. Access the Net Services Administration page in Oracle Enterprise Manager.

> **See Also:** "Oracle Enterprise Manager" on page 6-1

2. Select **Listeners** from the **Administer** list, and then select the Oracle home that contains the location of the configuration files.
3. Click **Go**.

   The Listeners page appears.
4. Select a listener, and then click **Edit**.

   The Edit Listener page appears.
5. In the **Addresses** section, configure protocol support:
   a. Click **Add**.

      The Add Address page appears.
   b. From the **Protocol** list, select the protocol on which the listener is configured to listen.
   c. Enter the appropriate parameter information for the selected protocol in the fields provided.

      When configuring the listener to listen on TCP/IP, you should enter the default port of 1521. If you do not, you must configure the `LOCAL_LISTENER` parameter in the initialization parameter file and resolve the listener name through a naming method.

      See Also:

      - *Oracle Database Net Services Reference* for further information about protocol addresses and TCP/IP privileged ports
      - "Registering Information with a Nondefault Listener" on page 10-11

      If the computer has more than one IP address and you want the listener to listen on all available IP addresses, configure TCP/IP or TCP/IP with SSL and enter the host name of the computer in the Host field.
   d. Optionally, in the **Advanced Parameters** section, specify the I/O buffer space limit for send and receive operations of sessions in the **Total Send Buffer Size** and **Total Receive Buffer Size** fields.

      See Also: "Configuring I/O Buffer Space" on page 14-3 for further information
   e. Click **OK**.

      The protocol address is added to the **Addresses** section.
6. Repeat Step 5 for additional protocols.

Note: You can also configure additional listening addresses with Oracle Net Manager. See the online Help for further instruction.

Configuring Access to Oracle JServer

Clients access Enterprise JavaBeans (EJBs) and Common Object Request Broker Architecture (CORBA) applications, provided with the Oracle JServer option, in an Oracle8*i* database over an Inter-Orb Protocol (IIOP) connection. IIOP is an implementation of General Inter-Orb Protocol (GIOP) over TCP/IP. To support access

to CORBA and EJB, you configure the listener with a protocol address with port 2481 for TCP/IP or port 2482 for TCP/IP with SSL.

To configure a protocol address for Oracle JServer in Oracle Database 11*g*:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Local** > **Listeners**.
3. Select a listener.
4. From the list in the right pane, select **Listening Locations**.
5. Click **Add Address**.

   A new **Address** tab appears.
6. Select the TCP/IP or TCP/IP with SSL protocol from the **Protocol** list.
7. Enter the host name of the database in the **Host** field.
8. Enter port 2481 for TCP/IP in the **Port** field, or enter port 2482 for TCP/IP with SSL in the **Port** field.
9. Click **Statically dedicate this address for JServer connections**.
10. Choose **File** > **Save Network Configuration**.

    The `listener.ora` file updates with the following:

```
listener=
 (DESCRIPTION_LIST=
  (DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=2481))
   (PROTOCOL_STACK=
     (PRESENTATION=giop)
     (SESSION=raw))))
```

Handling Large Volumes of Concurrent Connection Requests

If you expect the listener to handle large volumes of concurrent connection requests, then you can specify a listener queue size for its TCP/IP or IPC listening endpoints. To specify the listener queue size, specify the `QUEUESIZE` parameter at the end of the protocol address with its value set to the expected number of concurrent requests, similar to Example 10–2.

Example 10–2 listener.ora File with Queue Size

```
LISTENER=
 (DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521)(QUEUESIZE=20)))
```

Note: The default number of concurrent connection requests is operating-system specific. The defaults for TCP/IP on the Solaris Operating System and Windows follow:

- Solaris Operating System: 5
- Windows NT 4.0 Workstation: 5
- Windows NT 4.0 Server: 50

Configuring Static Service Information

The listener uses the dynamic service information about the database and instance it has received through service registration *before using statically configured information* in the `listener.ora` file.

Static configuration is also required for other services, such as external procedures, **Heterogeneous Services**.

Table 10–1 describes static service settings that you can set in the `listener.ora` file.

Table 10–1 Static Service Settings in listener.ora

Oracle Net Manager Field	listener.ora File Parameter	Description
SID	`SID_NAME`	Specifies the **Oracle System Identifier (SID)** of the instance. You can obtain the SID value from the `INSTANCE_NAME` parameter in the initialization parameter file.
Service Name	`GLOBAL_DBNAME`	Identifies the database service. While processing a client connection request, the listener tries to match the value of this parameter with the value of the `SERVICE_NAME` parameter in the client connect descriptor. If the client connect descriptor uses the `SID` parameter, then the listener does not attempt to map the values. This parameter is primarily intended for configurations with Oracle8 release 8.0 databases (where dynamic service registration is not supported for dedicated servers). This parameter may also be required for use with Oracle8*i* and higher database services by some configurations. The value for this parameter is typically obtained from the combination of the `DB_NAME` and `DB_DOMAIN` parameters (`DB_NAME.DB_DOMAIN`) in the initialization parameter file, but the value can also contain any valid name used by clients to identify the service.
Oracle Home Directory	`ORACLE_HOME`	On UNIX, this setting is optional. Use it to specify the Oracle home location of the instance. Without this setting, the listener assumes its Oracle home for the instance. On Windows, this setting is ignored. The Oracle home specified by the `ORACLE_HOME` parameter in `HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\HOME`*`ID`* of the Windows registry is used.

Important: If you are using **connect-time failover** or **Transparent Application Failover (TAF)**, such as in an Oracle Real Application Clusters environment, then do not set the `GLOBAL_DBNAME` parameter.

See Also:

- "Configuring Service Registration" on page 10-10 for more information about configuring dynamic service registration Oracle Databases
- Chapter 13, "Enabling Advanced Features of Oracle Net Services" for more information about statically configuring the listener for external procedures and Heterogeneous Services
- *Oracle Enterprise Manager Advanced Configuration* for further information about Oracle Enterprise Manager

To statically configure the listener:

1. Access the Net Services Administration page in Oracle Enterprise Manager.

   See Also: "Oracle Enterprise Manager" on page 6-1

2. Select **Listeners** from the **Administer** list, and then select the Oracle home that contains the configuration files.
3. Click **Go**.

   The Listeners page appears.
4. Select a listener, and then click **Edit**.

   The Edit Listener page appears.
5. Click the **Static Database Registration** tab, and then click **Add**.

   The Add Database Service page appears. Enter the required information in the fields.

   See Also: Table 10–1, " Static Service Settings in listener.ora" on page 10-6 for a description of the fields

6. Click **OK**.

Note: You can also configure static service information with Oracle Net Manager. See topic **Statically Configure Database Service Information** in the online Help for further instruction.

The following example shows an excerpt of a `listener.ora` file statically configured for a database service called `sales.us.acme.com`:

```
SID_LIST_listener=
(SID_LIST=
 (SID_DESC=
  (GLOBAL_DBNAME=sales.us.acme.com)
  (SID_NAME=sales)
  (ORACLE_HOME=/u01/app/oracle/11g)))
```

Default Oracle Net Listener Administration

By default, Oracle Net Listener permits only local administration for security reasons. Additionally, as a policy, the listener can be administered only by the user who started it and this is enforced through local operating system authentication. So, if `user1`

starts the listener, only `user1` can administer it, and any other user trying to administer it would get an error. The superuser is the only exception.

Oracle recommends that only local administration be allowed for the listener. However, in case a user wants to administer the listener remotely or a different user needs to administer the listener, either COST parameters or passwords can be used.

This section includes the following topics:

- Using COST Parameters and Other Secure Transports
- Configuring Passwords for Oracle Net Listener
- Changing the Oracle Net Listener Password

Using COST Parameters and Other Secure Transports

The security of a particular transport depends on the characteristics of the network on which it is used. If you are administering the listener remotely over an insecure network and require maximum security, you can configure the listener with a secure protocol address that uses the **TCP/IP with SSL protocol**. If the listener has multiple protocol addresses, ensure that the TCP/IP with SSL protocol address is listed first in the `listener.ora` file.

The class of secure transports (COST) parameters provide a way to specify the list of transports that are considered secure and can be used for listener administration. Using the COST parameters does not affect any connections other than registration and control.

The COST parameters identify two conditions: which transports are considered secure for that installation and whether the administration of a that listener requires secure transports.

> **See Also:** *Oracle Database Net Services Reference* for detailed descriptions of each of the four COST parameters.

Configuring Passwords for Oracle Net Listener

Oracle Net Listener Control (`lsnrctl`) is the command-line utility for managing Oracle Net Listener configuration, including passwords. A password can be configured for the listener to provide security for listener administrative operations, such as starting or stopping the listener, viewing a list of supported services, or saving changes to the Listener Control configuration. However, as mentioned earlier, local administration of the listener is secure by default through the local operating system. Therefore configuring a password is neither required nor recommended for secure local administration.

Changing the Oracle Net Listener Password

To change the listener password, use the Listener Control utility's `CHANGE_PASSWORD` command or Oracle Enterprise Manager to set or modify an encrypted password in the `PASSWORDS_`*`listener_name`* parameter in the `listener.ora` file. If the `PASSWORDS_`*`listener_name`* parameter is set to an unencrypted password, you must manually remove it from the `listener.ora` file prior to modifying it. If the unencrypted password is not removed, you will be unable to successfully set an encrypted password.

To set or modify an encrypted password with Oracle Enterprise Manager:

1. Access the Net Services Administration page in Oracle Enterprise Manager.

See Also: "Oracle Enterprise Manager" on page 6-1

2. Select **Listeners** from the **Administer** list, and then select the Oracle home that contains the location of the configuration files.
3. Click **Go**.

   The Listeners page appears.
4. Select a listener, and then click **Edit**.

   The Edit Listeners page appears.
5. Click the **Authentication** tab.
6. Click **Require a password for listener operations**.
7. Click **OK**.

Note: You can also configure static service information with Oracle Net Manager. See topic **Configure Password Authentication for the Listener** in the online Help for further instruction.

To set a new encrypted password with the CHANGE_PASSWORD command, issue the following commands from the Listener Control utility:

```
LSNRCTL> CHANGE_PASSWORD
Old password: old_password
New password: new_secure_password
Reenter new password: new_secure_password
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=tpc)(HOST=sales-server)(PORT=1521)))
Password changed for LISTENER
The command completed successfully
LSNRCTL> SAVE_CONFIG

Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521)))
Saved LISTENER configuration parameters.
Listener Parameter File   /oracle/network/admin/listener.ora
Old Parameter File   /oracle/network/admin/listener.bak
The command completed successfully
```

Bold denotes user input. The password is not displayed when entered.

To modify an encrypted password with the CHANGE_PASSWORD command:

```
LSNRCTL> SET PASSWORD
Password: password
The command completed successfully
LSNRCTL> CHANGE_PASSWORD
Old password: old_password
New password: new_secure_password
Reenter new password: new_secure_password
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=tpc)(HOST=sales-server)(PORT=1521)))
Password changed for LISTENER
The command completed successfully
LSNRCTL> SAVE_CONFIG
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521)))
Saved LISTENER configuration parameters.
Listener Parameter File   /oracle/network/admin/listener.ora
Old Parameter File   /oracle/network/admin/listener.bak
```

```
The command completed successfully
```

See Also:

- *Oracle Database Security Guide* for more information about minimum requirements for passwords
- *Oracle Database Net Services Reference* for further information about the CHANGE_PASSWORD command

Configuring Service Registration

Service registration allows processes, such as a database or OCS mail server, to identify their available services to the listener, which then acts as a portmapper for those services. The listener uses the dynamic service information about the database and instance it has received through service registration *before using statically configured information* in the `listener.ora` file.

Dynamic service registration is configured in the database initialization file. It does not require any configuration in the `listener.ora` file. However, listener configuration must be synchronized with the information in the database initialization file.

This section contains the following configuration topics related to service registration:

- Configuring Service Registration
- Registering Information with the Default, Local Listener
- Registering Information with a Nondefault Listener
- Registering Information with a Remote Listener
- Configuring a Naming Method

Configuring Service Registration

To ensure service registration works properly, the initialization parameter file should contain the following parameters:

- SERVICE_NAMES for the database service name
- INSTANCE_NAME for the instance name

For example:

```
SERVICE_NAMES=sales.us.acme.com
INSTANCE_NAME=sales
```

The value for the SERVICE_NAMES parameter defaults to the **global database name**, a name comprising the DB_NAME and DB_DOMAIN parameters in the initialization parameter file, entered during installation or database creation. The value for the INSTANCE_NAME parameter defaults to the SID entered during installation or database creation.

See Also: *Oracle Database Reference* for further information about the SERVICE_NAMES and INSTANCE_NAME parameters

Registering Information with the Default, Local Listener

By default, the PMON process registers service information with its local listener on the default local address of TCP/IP, port 1521. As long as the listener configuration is synchronized with the database configuration, PMON can register service information

with a nondefault local listener or a remote listener on another node. Synchronization is simply a matter of specifying the protocol address of the listener in the `listener.ora` file and the location of the listener in the initialization parameter file.

Registering Information with a Nondefault Listener

If you want PMON to register with a local listener that does not use TCP/IP, port 1521, configure the `LOCAL_LISTENER` parameter in the initialization parameter file to locate the local listener.

For a shared server environment, you can alternatively use the `LISTENER` attribute of the `DISPATCHERS` parameter in the initialization parameter file to register the dispatchers with a nondefault local listener. Because both the `LOCAL_LISTENER` parameter and the `LISTENER` attribute enable PMON to register dispatcher information with the listener, it is not necessary to specify both the parameter and the attribute if the listener values are the same.

Set the `LOCAL_LISTENER` parameter as follows:

```
LOCAL_LISTENER=listener_alias
```

Set the `LISTENER` attribute as follows:

```
DISPATCHERS="(PROTOCOL=tcp)(LISTENER=listener_alias)"
```

`listener_alias` is then resolved to the listener protocol addresses through a naming method, such as a `tnsnames.ora` file on the database server.

For example, if the listener is configured to listen on port 1421 rather than port 1521, you can set the `LOCAL_LISTENER` parameter in the initialization parameter file as follows:

```
LOCAL_LISTENER=listener1
```

Using the same listener example, you can set the `LISTENER` attribute as follows:

```
DISPATCHERS="(PROTOCOL=tcp)(LISTENER=listener1)"
```

You can then resolve `listener1` in the local `tnsnames.ora` as follows:

```
listener1=
 (DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1421)))
```

Notes:

- To dynamically update the `LOCAL_LISTENER` parameter, use the SQL statement `ALTER SYSTEM`:

  ```
  ALTER SYSTEM SET LOCAL_LISTENER='listener_alias'
  ```

  If you set the parameter to null with the statement that follows, then the default local address of TCP/IP, port 1521 is assumed.

  ```
  ALTER SYSTEM SET LOCAL_LISTENER=''
  ```

  See the *Oracle Database SQL Language Reference* for further information about the `ALTER SYSTEM` statement.

- The `LISTENER` attribute overrides the `LOCAL_LISTENER` parameter. As a result, the SQL statement `ALTER SYSTEM SET LOCAL_LISTENER` does not affect the setting of this attribute.

To register information with another local listener:

1. Configure the `listener.ora` file with the protocol address of the local listener.

   See Also: "Configuring Listening Protocol Addresses" on page 10-3

2. Configure the `LOCAL_LISTENER` parameter in the initialization parameter file to locate the local listener. If you are using shared server, you can also use the `LISTENER` attribute of the `DISPATCHERS` parameter in the initialization parameter file.
3. Resolve the listener name alias for the `LOCAL_LISTENER` or the `LISTENER` setting through a `tnsnames.ora` file.

   See Also: "Configuring a Naming Method" on page 10-14

Registering Information with a Remote Listener

Registration to remote listeners, such as in the case of Oracle Real Application Clusters, can be configured for shared server or dedicated server environments.

If you want PMON to register with a remote listener, configure the `REMOTE_LISTENER` parameter in the initialization parameter file to locate the remote listener.

For a shared server environment, you can alternatively use the `LISTENER` attribute of the `DISPATCHERS` parameter in the initialization parameter file to register the dispatchers with any listener. Because both the `REMOTE_LISTENER` parameter and the `LISTENER` attribute enable PMON to register dispatcher information with the listener, it is not necessary to specify both the parameter and the attribute if the listener values are the same.

Set the `REMOTE_LISTENER` parameter as follows:

```
REMOTE_LISTENER=listener_alias
```

Set the `LISTENER` attribute as follows:

```
DISPATCHERS="(PROTOCOL=tcp)(LISTENER=listener_alias)"
```

`listener_alias` is then resolved to the listener protocol addresses through a naming method, such as a `tnsnames.ora` file on the database server.

For example, if separate listeners are configured to listen on port 1521 on servers `sales1-server` and `sales2-server`, you can set the `REMOTE_LISTENER` parameter in the initialization file for the instance on host `sales1-server` as follows:

```
REMOTE_LISTENER=listener_sales2
```

You can set the `REMOTE_LISTENER` parameter in the initialization file for the instance on host `sales2-server` as follows:

```
REMOTE_LISTENER=listener_sales1
```

You can then resolve `listener_sales2` in the local `tnsnames.ora` on `sales1-server` as follows:

```
listener_sales2=
 (DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521)))
```

Likewise, you can resolve `listener_sales1` in the local `tnsnames.ora` on `sales2-server` as follows:

```
listener_sales1=
 (DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521)))
```

If shared server is configured, you can set the `DISPATCHERS` parameter in the initialization parameter file as follows:

```
DISPATCHERS="(PROTOCOL=tcp)(LISTENER=listeners_sales)"
```

You can then resolve `listener_sales` in the local `tnsnames.ora` as follows:

```
listeners_sales=
 (DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521))
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521)))
```

Notes:

- To dynamically update the `REMOTE_LISTENER` parameter, use the SQL statement `ALTER SYSTEM SET`:

  ```
  ALTER SYSTEM SET REMOTE_LISTENER='listener_alias'
  ```

  If you set the parameter to null with the statement that follows, then PMON de-registers information with the remote listener with which it had previously registered information.

  ```
  ALTER SYSTEM SET REMOTE_LISTENER=''
  ```

  See the *Oracle Database SQL Language Reference* for further information about the `ALTER SYSTEM SET` statement.

- The `LISTENER` attribute overrides the `REMOTE_LISTENER` parameter. As a result, the SQL statement `ALTER SYSTEM SET REMOTE_LISTENER` does not affect the setting of this attribute.

To register information with remote listener:

1. Configure the `listener.ora` file with the protocol addresses of the remote listeners.

   See Also: "Configuring Listening Protocol Addresses" on page 10-3

2. In a shared server environment, configure the `LISTENER` attribute of the `DISPATCHERS` parameter or the `REMOTE_LISTENER` parameter in the initialization parameter file. In a dedicated server environment, configure the `REMOTE_LISTENER` parameter in the database initialization parameter file.
3. Resolve the listener name alias for the `LISTENER` or the `REMOTE_LISTENER` setting through a `tnsnames.ora` file.

   See Also: "Configuring a Naming Method" on page 10-14

Configuring a Naming Method

The listener name alias specified for the `LOCAL_LISTENER` parameter, `REMOTE_LISTENER` parameter, or `LISTENER` attribute can be resolved through a `tnsnames.ora` file.

For example, if `LOCAL_LISTENER` is set to `listener1` and `listener1` uses TCP/IP on port 1421, the entry in the `tnsnames.ora` file would be:

```
listener1=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1421))
```

Note: Multiple addresses are supported, but connect-time failover and client load balancing features are not supported.

See Also: Chapter 13, "Enabling Advanced Features of Oracle Net Services" for further information about multiple address configuration

Listener Administration

Once the listener is configured, the listener can be administered with the Listener Control utility or Oracle Enterprise Manager. This section describes some of the common administrative tasks for the listener, including the following topics:

- Starting and Stopping a Listener
- Determining the Current Status of a Listener
- Monitoring Services of a Listener
- Monitoring Listener Log Files

See Also:

- *Oracle Database Net Services Reference* for a complete listing of all the Listener Control utility commands
- Oracle Enterprise Manager online Help

Starting and Stopping a Listener

To stop or start a listener, use either the Listener Control utility or Oracle Enterprise Manager.

Note: You can configure the listener to start automatically whenever the computer it is running on is restarted. See your operating-system specific documentation for details about establishing auto-restart.

Using the Listener Control Utility to Start or Stop a Listener

To stop a listener from the command line, enter:

```
lsnrctl STOP [listener_name]
```

where *listener_name* is the name of the listener defined in the `listener.ora` file. It is not necessary to identify the listener if you are using the default listener, named `LISTENER`.

To start the listener from the command line, enter:

```
lsnrctl START [listener_name]
```

where *listener_name* is the name of the listener defined in the `listener.ora` file. It is not necessary to identify the listener if you are using the default listener, named `LISTENER`.

In addition to starting the listener, the Listener Control utility verifies connectivity to the listener.

Using Oracle Enterprise Manager to Start or Stop a Listener

To start or stop a listener from Oracle Enterprise Manager:

1. Access the Net Services Administration page in Oracle Enterprise Manager.

   See Also: "Oracle Enterprise Manager" on page 6-1

2. Select **Listeners** from the **Administer** list, and then select the Oracle home that contains the location of the configuration files.
3. Click **Go**.

   The Listeners page appears.
4. Select a listener.
5. From the **Actions** list, select Start/Stop.

   The Start/Stop page appears.
6. Depending upon the current status of the selected listener, the operation will be either Stop or Start. Click OK to perform the operation.

Determining the Current Status of a Listener

To show the current status of a listener, use either the `STATUS` command of the Listener Control utility or Oracle Enterprise Manager. The status output provides basic status information about a listener, a summary of listener configuration settings, the listening protocol addresses, and a summary of services registered with the listener.

Using the Listener Control Utility to Determine the Listener Status

The `STATUS` command provides basic status information about a listener, including a summary of listener configuration settings, the listening protocol addresses, and a summary of services registered with the listener.

To show the status the listener from the command line, enter:

```
lsnrctl STATUS [listener_name]
```

where *listener_name* is the name of the listener defined in the `listener.ora` file. It is not necessary to identify the listener if you are using the default listener, named `LISTENER`.

Using Oracle Enterprise Manager to Determine the Listener Status

To show the status of a listener from Oracle Enterprise Manager:

1. Access the Net Services Administration page in Oracle Enterprise Manager.

   See Also: "Oracle Enterprise Manager" on page 6-1

2. Select **Listeners** from the **Administer** list, and then select the Oracle home that contains the location of the configuration files.
3. Click **Go**.

   The Listeners page appears.
4. Select a listener.
5. From the **Actions** list, select **Show Listener Control Status**.

   The Listener Control Status page appears.
6. After viewing the content, click **OK**.

The `STATUS` command generates output with the sections described in Table 10–2.

Table 10–2 Listener Control Utility STATUS Command

Output Section	Description
`STATUS of the LISTENER`	Specifies the following: ■ Alias of the listener ■ Version of listener ■ Start time and up time ■ Tracing level ■ Whether the listener can respond to queries from an SNMP-based network management system ■ `listener.ora` file being used ■ Logging and tracing configuration settings ■ Whether a password is set in `listener.ora` file
`Listening Endpoints Summary`	Lists the protocol addresses the listener is configured to listen on
`Services Summary`	Displays a summary of the services registered with the listener and the service handlers allocated to each service
`Service`	Identifies the registered service
`Instance`	Specifies the name of the instance associated with the service along with its status and number of service handlers associated with the service Status can be one of the following: ■ A `READY` status means that the instance can accept connections. ■ A `BLOCKED` status means that the instance cannot accept connections. ■ A `READY/SECONDARY` status means that this is a secondary instance in an Oracle Real Application Clusters primary/secondary configuration and is ready to accept connections. ■ An `UNKNOWN` status means that the instance is registered statically in the `listener.ora` file rather than dynamically with service registration. Therefore, the status is not known.

Example 10–3 shows example output of the `STATUS` command.

Example 10–3 Listener Control Utility's STATUS Command Output

```
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=net)))
STATUS of the LISTENER
------------------------
Alias                     LISTENER
Version                   TNSLSNR for Solaris: Version 10.1.0.2.0
Start Date                15-NOV-2006 20:22:00
Uptime                    0 days 0 hr. 5 min. 22 sec
Trace Level               support
Security                  OFF
SNMP                      OFF
Listener Parameter File   /oracle/admin/listener.ora
Listener Log File         /oracle/network/log/listener.log
Listener Trace File       /oracle/network/trace/listener.trc
Listening Endpoints Summary...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=net)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcps)(HOST=sales-server)(PORT=2484)))

Services Summary...
Service "sales.us.acme.com" has 1 instance(s).
  Instance "sales", status READY, has 3 handler(s) for this service...
Service "hr.us.acme.com" has 1 instance(s).
  Instance "hr", status READY, has 2 handler(s) for this service...
The command completed successfully
```

Monitoring Services of a Listener

The SERVICES command of the Listener Control utility provides detailed information about the services and instances registered with a listener and the service handlers allocated to each instance.

The SERVICES command generates output with the sections described in Table 10–3.

Table 10–3 Listener Control Utility SERVICES Command

Output Section	Description
Service	Identifies the registered service
Instance	Specifies the name of the instance associated with the service The status field indicates if the instance is able to accept connections. ■ A READY status means that the instance can accept connections. ■ A BLOCKED status means that the instance cannot accept connections. ■ A READY/SECONDARY status means that the is a secondary instance in an Oracle Real Application Clusters primary/secondary configuration and is ready to accept connections. ■ A RESTRICTED status means that the instance is in restricted mode. The listener blocks all connections to this instance. ■ An UNKNOWN status means that the instance is registered statically in the listener.ora file rather than dynamically with service registration. Therefore, the status is non known.

Table 10–3 (Cont.) Listener Control Utility SERVICES Command

Output Section	Description
Handlers	Identifies the name of the service handler. Dispatchers are named D000 through D999. Dedicated servers have a name of DEDICATED. This section also identifies the following about the service handler: ■ established: The number of client connections this service handler has established ■ refused: The number of client connections it has refused ■ current: The number of client connections it is handling, that is, its current load ■ max: The maximum number of connections for the service handler, that is, its maximum load ■ state: The state of the handler: - A READY state means that the service handler can accept new connections. - A BLOCKED state means that the service handler cannot accept new connections. Following this, additional information about the service handler displays, such as whether the service handler is a dispatcher, a local dedicated server, or a remote dedicated server on another node.

Example 10–4 shows example output of the SERVICES command.

Example 10–4 Listener Control Utility's SERVICES Command Output

```
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=net)))
Services Summary...
Service "sales.us.acme.com" has 1 instance(s).
  Instance "sales", status READY, has 3 handler(s) for this service...
    Handler(s):
      "DEDICATED" established:0 refused:0 state:ready
         LOCAL SERVER
      "D000" established:0 refused:0 current:0 max:10000 state:ready
         DISPATCHER <machine: sales-server, pid: 1689>
         (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=52414))
      "D001" established:0 refused:0 current:0 max:10000 state:ready
         DISPATCHER <machine: sales-server, pid: 1691>
         (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=52415))
Service "hr.us.acme.com" has 1 instance(s).
  Instance "hr", status READY, has 2 handler(s) for this service...
    Handler(s):
      "DEDICATED" established:0 refused:0 state:ready
         LOCAL SERVER
      "D000" established:0 refused:0 current:0 max:10000 state:ready
         DISPATCHER <machine: sales-server, pid: 11326>
         (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=58361))
The command completed successfully
```

This output shows that two database services, sales.us.acme.com and hr.us.acme.com, are registered with the listener.

Client connection requests to sales.us.acme.com are handled by two dispatchers named D000 and D001 and one dedicated server. All handlers have a status of ready, indicating that they are ready to receive connections.

Client connection requests to `hr.us.acme.com` are handled by one dispatcher named `D001` and one dedicated server.

Monitoring Listener Log Files

When you notice any of the following conditions, review and monitor the listener log file for the following:

- Long connection establishment times
- Connectivity problems and refusals
- Unexpected shutdown of the listener that could indicate a denial-of-service attack

See Also: "Analyzing Listener Log Files" on page 16-26

11

Configuring and Administering Oracle Connection Manager

This chapter describes how to configure **Oracle Connection Manager** features.

This chapter contains these topics:

- Oracle Connection Manager Configuration Overview
- Configuring Oracle Connection Manager
- Enabling Oracle Connection Manager Features
- Migrating cman.ora from Oracle9i to Oracle Database 11g

Note: Oracle Connection Manager is available for installation with Oracle Database 11*g*Enterprise Edition.

See Also:

- Chapter 1, "Introduction to Oracle Net Services" for an introductory level overview of Oracle Connection Manager concepts
- Chapter 5, "Architecture of Oracle Net Services" for an architectural overview of Oracle Connection Manager

Oracle Connection Manager Configuration Overview

Oracle Connection Manager is a **proxy server**, an intermediate server that forwards connection requests to database servers or to other proxy servers. It has two primary functions:

- Session multiplexing
- Access control

With session multiplexing, you can quickly enable Oracle Connection Manager to funnel multiple client sessions through a network connection to a **shared server** destination.

With access control, you can use rule-based configuration to filter out certain client requests and accept others.

Configuring Oracle Connection Manager

To configure Oracle Connection Manager:

1. Configure the `cman.ora` file on the Oracle Connection Manager computer. This file specifies the listening endpoint for the server, access control rules, and Oracle Connection Manager performance parameters.
2. Configure clients with the protocol addresses of the Oracle Connection Manager listener.
3. Optionally Configure the database server for session multiplexing.

This section contains these topics:

- Configuring the Oracle Connection Manager Computer
- Configuring Clients for Oracle Connection Manager
- Configuring the Oracle Database Server for Oracle Connection Manager

Configuring the Oracle Connection Manager Computer

Note: Oracle Net Manager does not support configuration of the `cman.ora` file, so changes must be made manually.

To configure the computer where Oracle Connection Manager is installed, you can define three types of parameters in the `cman.ora` file:

- Listening endpoint (`ADDRESS`)
- Access control rule list (`RULE_LIST`)
- Parameter list (`PARAMETER_LIST`)

The `cman.ora` file is located in the `$ORACLE_HOME/network/admin` directory on UNIX and in the *`ORACLE_HOME`*`\network\admin` directory on Windows. Example 11–1 shows an example `cman.ora` file that contains a configuration entry for an Oracle Connection Manager called `CMAN1`.

Example 11–1 Example cman.ora File

```
CMAN1=
  (CONFIGURATION=
    (ADDRESS=(PROTOCOL=tcp)(HOST=proxysvr)(PORT=1521))
    (RULE_LIST=
      (RULE=(SRC=206.62.226.32/27)(DST=sales-server)(SRV=*)(ACT=accept)
        (ACTION_LIST=(AUT=on)(MCT=120)(MIT=30)))
      (RULE=(SRC=206.26.226.32)(DST=proxysvr)(SRV=cmon)(ACT=accept)))
    (PARAMETER_LIST=
      (MAX_GATEWAY_PROCESSES=8)
      (MIN_GATEWAY_PROCESSSES=3)
      (REMOTE_ADMIN=YES)))
```

One computer can host any number of Oracle Connection Managers, each with its own configuration entry in `cman.ora`. When defining more than one Oracle Connection Manager in the file, you can assign a default by giving only one a fully qualified host name.

See Also: *Oracle Database Net Services Reference* to learn more about this feature and the `ADMINISTER` and `STARTUP` commands

This section includes the following topics:

- Listening Endpoint (ADDRESS)
- Access Control Rule List (RULE_LIST)
- Parameter List (PARAMETER_LIST)

Listening Endpoint (ADDRESS)

The listening endpoint specifies the protocol address for the Oracle Connection Manager listener. CMON, the Oracle Connection Manager monitoring process, uses this address to register information about gateway processes with the listener. The database server, in turn, uses the address to register service information at the Oracle Connection Manager node.

Note that the Oracle Connection Manager listener always listens on the TCP/IP protocol. The address shown in Example 11–1 on page 11-2 is the default address of TCP/IP, port 1521.

```
(ADDRESS=(PROTOCOL=tcp)(HOST=proxysvr)(PORT=1521))
```

Note: Oracle Connection Manager can connect to the database server using protocols such as TCP/IP and IPC. The protocol TCPS is not supported.

Access Control Rule List (RULE_LIST)

The access control rule list specifies which connections are accepted, rejected, or dropped by the listener.

```
(RULE=(SRC=206.62.226.32/27)(DST=sales-server)(SRV=*)(ACT=accept)
  (ACTION_LIST=(AUT=on)(MCT=120)(MIT=30)))
(RULE=(SRC=206.26.226.32)(DST=proxysvr)(SRV=cmon)(ACT=accept))
```

The example shows two rules. The first one is for client connections. The second is for the Oracle Connection Manager Control utility (CMCTL). In the first rule, `src=206.62.226.32/27` designates the IP address of the client, or source. `DST=sales-server` designates the destination host name. The abbreviation `ACT` stands for "action"—that is, accept, reject, or drop. In the second rule, `SRC=206.26.226.32` and `DST=proxysvr` represent the same server, indicating that Oracle Connection Manager and CMCTL must reside on the same computer.

The parameter `ACTION_LIST` in the first rule sets attributes for a connection if it is accepted. This parameter enables you to override default parameter settings on a connection by connection basis. See "Oracle Connection Manager Parameters" in Oracle Database Net Services Reference for a complete definition of `ACTION_LIST` subparameters.

You can specify multiple rules for both client and CMCTL connections.

Notes:

- You must enter at least one rule for client connections and one rule for CMCTL connections. Omitting one or the other results in the rejection of all connections for the rule type omitted.
- If the CMCTL connection is remote, the `REMOTE_ADMIN` parameter in `cman.ora` must be set to `on`, regardless of the rules specified.
- If `cman.ora` does not exist, Oracle Connection Manager cannot start.
- Oracle Connection Manager does not support wildcards for partial IP addresses. If you use a wildcard, use it in place of a full IP address. The IP address of the client may, for example, be (SRC=*).
- Oracle Connection Manager supports only the /nn notation for subnet addresses. In the first rule in the example, /27 represents a subnet mask that comprises 27 left-most bits. This means that only the first 27 bits in the client's IP address are compared with the IP address in the rule.

Parameter List (PARAMETER_LIST)

The parameter list sets attributes for an Oracle Connection Manager. Parameters take two forms: global and rule level.

A global parameter applies to all Oracle Connection Manager connections, unless a rule-level parameter overrides it. To change a global parameter's default setting, enter it into the `PARAMETER_LIST`, together with an allowable value.

A rule-level parameter is enabled in the `ACTION_LIST` section of the `RULE_LIST` and applies only to connections specified by the rule. It overrides its global counterpart.

See Also: *Oracle Database Net Services Reference* for a complete list of parameters and their default and allowed values

Configuring Clients for Oracle Connection Manager

To route clients to the database server through Oracle Connection Manager, configure the `tnsnames.ora` file with a **connect descriptor** that specifies the protocol address of Oracle Connection Manager. This address enables clients to connect to the Oracle Connection Manager computer. The connect descriptor looks like this:

```
sales=
  (DESCRIPTION=
    (ADDRESS=
      (PROTOCOL=tcp)
      (HOST=cman-pc)
      (PORT=1521))
    (CONNECT_DATA=
      (SERVICE_NAME=sales.com)))
```

To configure a protocol address for Oracle Connection Manager:

1. Start Oracle Net Manager.

See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Directory** or **Local** > **Service Naming**.
3. Click plus (+) from the toolbar, or choose **Edit** > **Create**.

   The Welcome page of the Net Service Name Wizard appears.
4. Enter any name in the **Net Service Name** field.
5. Click **Next**.

   The Protocol page appears.
6. Select the protocol on which Oracle Connection Manager is configured to listen on. By default this protocol is TCP/IP.
7. Click **Next**.

   The Protocol Settings page appears.
8. Enter the appropriate parameter information for the selected protocol in the fields provided. If you are using TCP/IP, the default port to use is 1521.

   See Also: *Oracle Database Net Services Reference* for protocol parameter settings
9. Click **Next**.

   The Service page appears.
10. Select a release, and then enter the name of destination database service.

    If the destination service is an Oracle Database 11*g*, Oracle9*i* or Oracle8*i* database, select **Oracle8*i* or later**, and enter a service name in the **Service Name** field. If destination service is an Oracle8 database, select **Oracle8 or Previous**, and enter an **Oracle System Identifier (SID)** for an instance in the **Database SID** field.

    See Also: "About Connect Descriptors" on page 8-1 for further information about setting the service name string
11. Click **Next**.

    Note: Do not click **Test**, because a connection cannot be tested at this point.
12. Click **Finish** to save your configuration and dismiss Net Service Name Wizard.

    The new net service name and the Oracle Connection Manager protocol address is added to the Service Naming folder.

Configuring the Oracle Database Server for Oracle Connection Manager

Configuring the database server is a two-part process that involves registering database information remotely with Oracle Connection Manager and, optionally, configuring the server for multiplexing.

This section includes the following topics:

- Service Registration Configuration
- Session Multiplexing Configuration

Service Registration Configuration

To enable the database server to communicate with Oracle Connection Manager, the initialization parameter file `init.ora` must contain a descriptor that specifies the listening address of Oracle Connection Manager. Because this address is TCP, port 1521 but not the default local listening address of TCP, port 1521, you must specify an alias, using the `REMOTE_LISTENER` parameter:

```
REMOTE_LISTENER=cman_listener_alias
```

After the alias is specified, it must be resolved with a service name entry in the `tnsnames.ora` file.

For example, an alias for an Oracle Connection Manager listener located at proxyserver1 might look like this in the `init.ora` file:

```
REMOTE_LISTENER=listener_cman
```

The alias `listeners_cman` would then be resolved to the following entry in the `tnsnames.ora` file:

```
listener_cman=
 (DESCRIPTION=
  (ADDRESS_LIST=
   (ADDRESS=(PROTOCOL=tcp)(HOST=proxyserver1)(PORT=1521))))
```

Once the initialization parameter file is configured with the listening address of Oracle Connection Manager, the **PMON process**—the database instance background process—can register database information with the Oracle Connection Manager listener. This registration is similar to what occurs on the proxy node, where the **CMADMIN (Connection Manager Administration)** process—the Oracle Connection Manager background process—registers the location and load of proxy processes with the listener of Oracle Connection Manager.

> **See Also:** "Registering Information with a Remote Listener" on page 10-12

Session Multiplexing Configuration

To enable Connection Manager to take advantage of session multiplexing, set the `DISPATCHERS` parameter in the initialization parameter file with the attributes `PROTOCOL` and `MULTIPLEX`.

```
DISPATCHERS="(PROTOCOL=tcp)(MULTIPLEX=on)"
```

> **See Also:**
>
> - "Enabling Session Multiplexing" on page 11-7 for configuration details.
> - Chapter 12, "Configuring Dispatchers" for more information about configuring shared server.

Enabling Oracle Connection Manager Features

This section contains these topics:

- Enabling Session Multiplexing
- Enabling Access Control

The first feature is enabled by using the parameter DISPATCHERS in the initialization parameter file, the second by using the parameter RULE_LIST in the cman.ora file.

Enabling Session Multiplexing

Once the attributes PROTOCOL and MULTIPLEX have been added to the parameter DISPATCHERS in the initialization parameter file, enabling session multiplexing is simply a matter of ensuring that MULTIPLEX is set to on or to an equivalent value.

You can set different levels of multiplexing, as Table 11–1 on page 11-7 shows.

Table 11–1 Session Multiplexing Parameters

Attribute	Description
PROTOCOL (PRO or PROT)	The network protocol for which the dispatcher generates a listening endpoint.
MULTIPLEX (MUL or MULT)	Used to enable session multiplexing If 1, on, yes, true, or both is specified, then multiplexing is enabled for both incoming and outgoing network sessions. If in is specified, then multiplexing is enabled for incoming network sessions from the client. If out is specified, then multiplexing is enabled for outgoing network sessions. If 0, no, off, or false is specified, then multiplexing is disabled for both incoming and outgoing network sessions.

Note: You can configure the DISPATCHERS parameter using the **Database Configuration Assistant**.

Enabling Access Control

As stated in "Configuring the Oracle Connection Manager Computer" on page 11-2, you can use the parameter RULE_LIST to control client access to designated database servers in a TCP/IP environment. By entering filtering rules under this parameter, you can allow or restrict specific clients access to a database server.

To configure access control:

1. Manually create a cman.ora file, if one does not already exist.
2. Add the parameter RULE_LIST and its subparameters, using the following format:

```
(RULE_LIST=
 (RULE=(SRC=source_host)
       (DST=destination_host)
       (SRV=service)
       (ACT=accept | reject | drop)))
```

3. Add the following parameters for each rule described in Table 11–2 on page 11-7 as needed.

Table 11–2 Rule-Level Parameters

Parameter	Description
SRC	Specify the source host name or IP address of the client. The IP address can be a subnet such as 152.10.10.62/24.

Table 11–2 (Cont.) Rule-Level Parameters

Parameter	Description
DST	Specify the destination host name or IP address of the database server. The IP address can be a subnet such as 152.10.10.62/24.
SRV	Specify the service name of the Oracle Database 11*g*, Oracle9*i* or Oracle8*i* database (obtained from the SERVICE_NAME parameter in the initialization parameter file).
ACT	Specify to accept, reject, or drop incoming requests based on the preceding three parameters.

See Also: *Oracle Database Net Services Reference* for default values and allowed values of Oracle Connection Manager parameters

You can define multiple rules in the RULE_LIST. The action (ACT) in the first matched RULE is applied to the connection request. If no rules are defined, all connections are rejected.

In the following example, client computer client1-pc is denied access to the service sales.us.acme.com, but client 144.25.23.45 is granted access to the service db1.

```
(RULE_LIST=
  (RULE=(SRC=client1-pc)(DST=sales-server)(SRV=sales.us.acme.com)(ACT=reject))
  (RULE=(SRC=144.25.23.45)(DST=144.25.187.200)(SRV=db1)(ACT=accept)))
```

See Also: *Oracle Database Net Services Reference* for further information about Oracle Connection Manager parameters

Migrating cman.ora from Oracle9*i* to Oracle Database 11*g*

If you want to migrate an Oracle9*i* cman.ora file to Oracle Database 11*g*, use the cmmigr tool. Here is the syntax for the tool:

```
cmmigr [cman.ora_location]
```

Specifying the file location is optional. If you omit it, cmmigr tries to find the file in the TNS_ADMIN directory; then it looks in $ORACLE_HOME/network/admin.

When it runs, cmmigr renames the Oracle9*i* cman.ora file cman.bak. It names the Oracle Database 11*g* file cman.ora.

The tool migrates three of the four sections that are in the Oracle9*i* file:

- Address section: cmmigr converts the listener protocol address from the Oracle9*i* format to the Oracle Database 11*g* format
- Admin section: cmmigr ignores this section.
- Profile section: cmmigr translates the parameter names in cman_profile into Oracle Database 11*g* names. With the exception of log level and trace level, the tool leaves parameter values untouched. Obsolete parameters appear in a commented list in the new file.
- Rules section: cmmigr copies existing rules to the new file. It adds a rule that enables CMCTL to contact CMADMIN. If the old file contains no rules, cmmigr adds two rules to the new file: one for the connection between CMCTL and CMADMIN and one for the client connection. See "Access Control Rule List (RULE_LIST)" on page 11-3 for examples of these two rules.

Table 11–3 lists messages used by `cmmigr`.

Table 11–3 cmmigr Messages

Message	Description
1.4140-"Migration completed successfully."	This message appears when `cman.ora` has been migrated successfully.
2.4141-"Unable to find CMAN.ORA."	This message appears when the file location that you specify is incorrect.
3.4142-"CMAN.ORA has an invalid format."	This message appears when the file is in a format that `cmmigr` cannot understand. Need formatting guidelines
4.4143-"Unable to write the new CMAN.ORA file."	
5.4144-"Nothing to migrate."	The tool found nothing in the file that it could migrate.

12

Configuring Dispatchers

This chapter describes how to configure dispatcher.

The **shared server** architecture enables a database server to allow many user processes to share very few server processes, so the number of users that can be supported is increased. With shared server, many user processes connect to a **dispatcher**. The dispatcher directs multiple incoming network session requests to a common queue. An idle shared server process from a shared pool of server processes picks up a request from the queue. This means a small pool of server processes can serve a large number of clients.

This chapter includes the following topics:

- Configuring Dispatchers
- Enabling Connection Pooling
- Enabling Session Multiplexing
- Grouping Services by Dispatcher
- Configuring Clients for Environments Using Both Dedicated Server and Shared Server

> **See Also:** *Oracle Database Administrator's Guide* for further information about shared server configuration

Configuring Dispatchers

Shared memory resources for dispatchers, virtual circuits, and shared servers are preconfigured to allow the enabling of shared servers at runtime. Database administrators can start dispatchers and shared servers with the SQL `ALTER SYSTEM` statement without having to restart the instance to start shared servers. A dispatcher is started automatically on the TCP/IP protocol when shared server mode is turned on and the dispatchers parameter

has not been set. The default configuration for the dispatcher is equivalent to the following `DISPATCHERS` parameter setting in the database initialization parameter file.

```
dispatchers="(PROTOCOL=tcp)"
```

In configurations in which client load is causing a strain on memory and other system resources, database administrators can remedy load issues by starting shared server resources.

Configure the `DISPATCHERS` parameter if either of the following conditions apply:

- **You need to configure another protocol other than TCP/IP**

  Configure a protocol address with one of the following attributes:

  - ADDRESS (ADD or ADDR)
  - DESCRIPTION (DES or DESC)
  - PROTOCOL (PRO or PROT)

- **You want to configure one or more of the optional dispatcher attributes**
  - CONNECTIONS (CON or CONN)
  - DISPATCHERS (DIS or DISP)
  - LISTENER (LIS or LIST)
  - MULTIPLEX (MUL or MULT)
  - POOL (POO)
  - SERVICE (SER or SERV)
  - SESSIONS (SES or SESS)
  - TICKS (TIC or TICK)

After setting this parameter, you do not need to restart the instance. You can alter dispatchers configurations with the SQL statement ALTER SYSTEM.

Note: **Database Configuration Assistant** enables you to configure the DISPATCHERS parameter.

See Also:

- *Oracle Database Administrator's Guide* for further information about shared server configuration
- *Oracle Database Reference* for further information about configuring the DISPATCHERS parameter and supported attributes
- *Oracle Database SQL Language Reference* for further information about the ALTER SYSTEM statement

Enabling Connection Pooling

Connection pooling is a resource utilization feature that enables you to reduce the number of physical network connections to a dispatcher. This is achieved by sharing or pooling a set of connections among the client processes.

To configure connection pooling, set the DISPATCHERS parameter in the initialization parameter file with the POOL attribute and the following optional attributes:

- CONNECTIONS (CON or CONN)
- SESSIONS (SES or SESS)
- TICKS (TIC or TICK)

Consider a system that can support 950 connections for each process and has:

- 4000 users concurrently connected through TCP/IP
- 2500 sessions concurrently connected through TCP/IP with SSL

In this case, the DISPATCHERS attribute for TCP/IP should be set to a minimum of five dispatchers and TCP/IP with SSL should be set to three dispatchers:

```
DISPATCHERS="(PROTOCOL=tcp)(DISPATCHERS=5)"
DISPATCHERS="(PROTOCOL=tcps)(DISPATCHERS=3)"
```

Connection pooling can allow each dispatcher to handle more than 950 sessions. For example, suppose that clients are idle most of the time and one dispatcher can route requests and responses for 4000 TCP/IP sessions, or 2500 TCP/IP with SSL sessions. The following configuration reduces the number of dispatchers required to one for each protocol and allows the dispatchers to be more fully utilized:

```
DISPATCHERS="(PROTOCOL=tcp)(DISPATCHERS=1)(POOL=on)(TICK=1)
(CONNECTIONS=950)(SESSIONS=4000)"

DISPATCHERS="(PROTOCOL=tcps)(DISPATCHERS=1)(POOL=on)(TICK=1)
(CONNECTIONS=950)(SESSIONS=2500)"
```

The DISPATCHERS and CONNECTIONS attributes do not have to be specified, since the default for DISPATCHERS is 1 and the default for CONNECTIONS is the maximum number of connections for each process, which is operating system dependent.

> **See Also:** "Connection Pooling" on page 1-9

Enabling Session Multiplexing

Session multiplexing, available with Oracle Connection Manager, enables multiple client sessions to funnel through a single protocol connection. For example, several user processes can connect to one dispatcher by way of a single connection from Oracle Connection Manager.

Oracle Connection Manager communicates from users to the dispatcher by way of a shared connection. At any one time, zero, one, or a few users might need the connection, while other user processes linked to the dispatcher by way of the connection manager process are idle. This way, session multiplexing is beneficial because it maximizes use of the dispatcher process connections.

Session multiplexing is also useful for multiplexing database link sessions between dispatchers. The limit on the number of sessions for each dispatcher is operating system specific.

To enable session multiplexing, simply set attribute MULTIPLEX in the DISPATCHERS parameter to on or to an equivalent value.

```
DISPATCHERS="(PROTOCOL=tcp)(MULTIPLEX=on)"
```

> **See Also:** "Enabling Session Multiplexing" on page 11-7 for configuration details and "Enabling Connection Pooling" on page 12-2

Grouping Services by Dispatcher

An Oracle database can be represented by multiple service names. Because of this, a pool of dispatchers can be allocated exclusively for clients requesting a particular service. This way, the mission critical requests may be given more resources and, thus, in effect increase their priority.

For example, the following initialization parameter file sample shows two dispatchers. The first dispatcher services requests for clients requesting `sales.us.acme.com`.

The other dispatcher services requests only for clients requesting `adminsales.us.acme.com`.

```
SERVICE_NAMES=sales.us.acme.com
INSTANCE_NAME=sales
DISPATCHERS="(PROTOCOL=tcp)"
DISPATCHERS="(PROTOCOL=tcp)(SERVICE=adminsales.us.acme.com)"
```

Configuring Clients for Environments Using Both Dedicated Server and Shared Server

If shared server is configured and a client connection request arrives when no dispatchers are registered, the request is processed by a dedicated server process. If you want a particular client always to use a dispatcher, configure `(server=shared)` in the `CONNECT_DATA` section of the connect descriptor. For example:

```
sales=
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (SERVER=shared))
```

If a dispatcher is not available, the client connection request is rejected.

> **See Also:** "Configuring Advanced Connect Data Parameters" on page 13-5 to set the `SERVER` parameter

If the database is configured for shared server and a particular client requires a dedicated server, you can configure the client to use a dedicated server in one of the following ways:

- You can configure a net service name with a connect descriptor that contains `(server=dedicated)` in the `CONNECT_DATA` section. For example:

  ```
  sales=
  (DESCRIPTION=
    (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
    (CONNECT_DATA=
       (SERVICE_NAME=sales.us.acme.com)
       (SERVER=dedicated))
  ```

- You can configure the client profile (`sqlnet.ora` file) with `USE_DEDICATED_SERVER=on`. This adds `(server=dedicated)` to the `CONNECT_DATA` section of the connect descriptor the client uses.

  > **Note:** If `USE_DEDICATED_SERVER` is set to `ON`, then existing `(server=value)` entries in connect descriptors are overwritten with `(server=dedicated)`.

  See Also:

  - "Configuring Advanced Connect Data Parameters" on page 13-5 to set the `SERVER` parameter
  - "Routing Connection Requests" on page 9-4 to set the `USE_DEDICATED_SERVER` parameter

If database resident connection pooling is enabled on the server, use the value `pooled` to get a connection from the pool.

> **See Also:** *Oracle Call Interface Programmer's Guide* and *Oracle Database Administrator's Guide* for more information about enabling and configuring database resident connection pooling

13

Enabling Advanced Features of Oracle Net Services

This chapter describes how to configure advanced features of Oracle Net Services, including advanced connect data parameters, **load balancing**, **failover**, and connections to non-database services.

This chapter contains these topics:

- Configuring Advanced Network Address and Connect Data Information
- Configuring Connection Load Balancing
- Configuring Transparent Application Failover
- Specifying the Instance Role for Primary and Secondary Instance Configurations
- Configuring Connections to Non-Oracle Database Services

Configuring Advanced Network Address and Connect Data Information

This section contains the following advanced connect descriptor topics:

- Creating a List of Listener Protocol Addresses
- Configuring Address List Parameters
- Configuring Advanced Connect Data Parameters

Creating a List of Listener Protocol Addresses

A database service may be accessed by more than one network route, or protocol address. In the following example, `sales.us.acme.com` can connect to `sales.us.acme.com` using listeners on either `sales1-server` or `sales2-server`.

```
sales.us.acme.com=
 (DESCRIPTION=
  (ADDRESS_LIST=
    (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521))
    (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521)))
  (CONNECT_DATA=
    (SERVICE_NAME=sales.us.acme.com)))
```

To add a network protocol address to an existing net service name or database service, use either Oracle Enterprise Manager or Oracle Net Manager.

Oracle Enterprise Manager

1. Access the Directory Naming or Local Naming page in Oracle Enterprise Manager:
   a. Access the Net Services Administration page in Oracle Enterprise Manager.

      See Also: "Oracle Enterprise Manager" on page 6-1

   b. Select **Local Naming** or **Directory Naming** from the **Administer** list, and then select the Oracle home for the directory server or the location of the local configuration files.
   c. Click **Go**.

      The Directory Naming or Local Naming pages appear.
2. Select the directory service or net service name.

   For **Directory Naming**, perform a search of the net service name in the **Simple Search** section, select the net service or database service from the **Results** list, and then click **Edit**. For **Local Naming**, select a net service from the list, and then click **Edit**.
3. In the **Addresses** section, click **Add**.

   The Add Address page appears.
4. From the **Protocol** list, select the protocol on which the listener is configured to listen. This protocol must also be installed on the client.
5. Enter the appropriate parameter information for the selected protocol in the fields provided.

   See Also: *Oracle Database Net Services Reference* for protocol parameter settings
6. Optionally, in the **Advanced Parameters** section, specify the I/O buffer space limit for send and receive operations of sessions in the **Total Send Buffer Size** and **Total Receive Buffer Size** fields.

   See Also: "Configuring I/O Buffer Space" on page 14-3 for further information
7. Click **OK**.

   The protocol address is added to the **Addresses** section.
8. Click **OK** to update the address information.

Oracle Net Manager

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2
2. In the navigator pane, expand **Directory** or **Local** > **Service Naming**.
3. Select either the net service name or a database service.

   The right pane displays the current destination service and address list.
4. In the **Address Configuration** box, click plus (+) to add a new address.

A new **Address** tab appears:

a. Select the protocol and enter appropriate address information.

See Also: *Oracle Database Net Services Reference* for details about protocol address parameters

b. Optionally, on the **Address** tab, click **Advanced** to specify the I/O buffer space limit for send and receive operations of sessions in the **Total Send Buffer Size** and **Total Receive Buffer Size** fields.

See Also: "Configuring I/O Buffer Space" on page 14-3 for further information

c. Order the protocol addresses according to where they should be in the protocol address list with the left-arrow and right-arrow buttons. Unless multiple address options are configured, the first address in the list is contacted.

See Also: "Configuring Address List Parameters" on page 13-3 for address list options

5. If you are making these changes to the **Local** folder, then choose **File** > **Save Network Configuration**.

Configuring Address List Parameters

When a database service is accessible by multiple listener protocol addresses, specify the order in which the addresses are to be used. The addresses can be chosen randomly or tried sequentially.

When multiple protocol addresses have been configured for a net service name or database service, you can configure the parameters described in Table 13–1.

Table 13–1 Address List Parameters

Parameter	Description
SOURCE_ROUTE (Source Routing)	When set to on, instructs Oracle Net to use each address in the order presented until the destination is reached. This parameter is required for reaching the destination using a specific route, that is, by specific computers. This parameter is used to enable connections to **Oracle Connection Manager**. **See Also:** "Configuring Clients for Oracle Connection Manager" on page 11-4
FAILOVER (Connect-Time Failover)	At connect time, instructs Oracle Net to fail over to a different listener if the first listener fails when set to on. The number of addresses in the list determines how many addresses are tried. When set to off, instructs Oracle Net to try one address. **Connect-time failover** is turned on by default for multiple address lists (ADDRESS_LIST), connect descriptors (DESCRIPTION), and multiple connect descriptors (DESCRIPTION_LIST). **Important:** When using a connect descriptor with a SERVICE_NAME, ensure that the value is neither a GLOBAL_DBNAME in any SID_DESC entry, nor a SID_NAME in any SID_DESC entry without a GLOBAL_DBNAME set.

Table 13–1 (Cont.) Address List Parameters

Parameter	Description
LOAD_BALANCE (Client Load Balancing)	When set to on, instructs Oracle Net to progress through the list of protocol addresses in a random sequence, balancing the load on the various listeners. When set to off, instructs Oracle Net to try the addresses sequentially until one succeeds. **Client load balancing** is turned on by default for multiple connect descriptors (DESCRIPTION_LIST).

Note: It is not possible to set client load balancing or connect-time failover with source routing. While connect-time failover and client load balancing select an address from a list, source routing connects to each address in the list sequentially.

Source routing involves other configuration that goes beyond the scope of this section.

See Also: "Configuring Clients for Oracle Connection Manager" on page 11-4 for more information about configuring clients for source routing

To configure connect-time failover or client load balancing:

1. Perform the procedure in "Creating a List of Listener Protocol Addresses" on page 13-1.
2. Use either Oracle Enterprise Manager or Oracle Net Manager to configure address list options.

   For Oracle Enterprise Manager, select the appropriate option in the **Connect-time Failover and Client Load Balancing** section.

   For Oracle Net Manager, click **Advanced** in the **Address Configuration** box. The Address List Options dialog box appears. Select the appropriate option.

   Table 13–2 describes the address list options.

Table 13–2 Address List Options Dialog Box

Option	Parameter Setting
Try each address, in order, until one succeeds	FAILOVER=on
Try each address, randomly, until one succeeds **Note:** This option is not enabled if **Use Options Compatible with Net8 8.0 Clients** is selected in Oracle Net Manager.	LOAD_BALANCE=on FAILOVER=on
Try one address, selected at random **Note:** This option is not enabled if **Use Options Compatible with Net8 8.0 Clients** is selected in Oracle Net Manager.	LOAD_BALANCE=on
Use each address in order until destination reached	SOURCE_ROUTE=on

Table 13–2 (Cont.) Address List Options Dialog Box

Option	Parameter Setting
Use only the first address **Note:** This option is not enabled if **Use Options Compatible with Net8 8.0 Clients** is selected in Oracle Net Manager.	LOAD_BALANCE=off FAILOVER=off SOURCE_ROUTE=off

The following example shows a `tnsnames.ora` file configured for client load balancing:

```
sales.us.acme.com=
 (DESCRIPTION=
  (ADDRESS_LIST=
  (LOAD_BALANCE=on)
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521))
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521)))
  (CONNECT_DATA=
    (SERVICE_NAME=sales.us.acme.com)))
```

The following example shows a `tnsnames.ora` file configured for connect-time failover:

```
sales.us.acme.com=
 (DESCRIPTION=
  (ADDRESS_LIST=
   (LOAD_BALANCE=off)
   (FAILOVER=ON)
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521))
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521)))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)))
```

Configuring Advanced Connect Data Parameters

The `CONNECT_DATA` section of a connect descriptor defines the destination database service. In the following example, `SERVICE_NAME` defines a service called `sales.us.acme.com`:

```
sales.us.acme.com=
 (DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
   (CONNECT_DATA=
    (SERVICE_NAME=sales.us.acme.com)))
```

Besides the service name, you can optionally configure the connect data information with the parameters described in Table 13–3.

Table 13–3 Advanced Connect Data Settings

Oracle Enterprise Manager/Oracle Net Manager Option	tnsnames.ora File Parameter	Description
Instance Name	INSTANCE_NAME	Used to identify the database instance to access. The instance name can be obtained from the INSTANCE_NAME parameter in the initialization parameter file. **Note:** This parameter is not enabled if **Use Oracle8i Release 8.0 Compatible Identification** is checked. **See Also:** "About Connect Descriptors" on page 8-1

Table 13–3 (Cont.) Advanced Connect Data Settings

Oracle Enterprise Manager/Oracle Net Manager Option	tnsnames.ora File Parameter	Description
Session Data Unit Size	SDU	To optimize the transfer rate of data packets being sent across the network, you can specify the **session data unit (SDU)** size to change the performance characteristics having to do with the packets sent across the network. **See Also:** "Configuring Session Data Unit" on page 14-1
Use for Heterogeneous Services	HS	If you want an Oracle database server to access a non-Oracle system through **Heterogeneous Services**, turn this option on. **See Also:** "Configuring Oracle Net Services for Oracle Heterogeneous Services" on page 13-26
Oracle Rdb Settings		
Oracle RDB Database	RDB_DATABASE	Specify the file name of the Oracle Rdb database. **See Also:** "Configuring Oracle Net Services for an Oracle Rdb Database" on page 13-27
Type of Service	TYPE_OF_SERVICE	Specify the type of service to use for the Oracle Rdb database. **See Also:** "Configuring Oracle Net Services for an Oracle Rdb Database" on page 13-27
Global Database Name	GLOBAL_NAME	Use to identify an Oracle Rdb database. **See Also:** "Configuring Oracle Net Services for an Oracle Rdb Database" on page 13-27

To configure advanced CONNECT_DATA parameters for either a net service name or a database service:

To add a network protocol address to an existing net service name or database service, use either Oracle Enterprise Manager or Oracle Net Manager.

Oracle Enterprise Manager

1. Access the Directory Naming or Local Naming page in Oracle Enterprise Manager:
   a. Access the Net Services Administration page in Oracle Enterprise Manager.

      See Also: "Oracle Enterprise Manager" on page 6-1

   b. Select **Local Naming** or **Directory Naming** from the **Administer** list, and then select the Oracle home for the directory server or the location of the local configuration files.
   c. Click **Go**.

      The Directory Naming or Local Naming pages appear.
2. Select the directory service or net service name.

   For **Directory Naming**, perform a search of the net service name in the **Simple Search** section, select the net service or database service from the **Results** list, and then click **Edit**. For **Local Naming**, select a net service from the list, and then click **Edit**.
3. Click the **Advanced** tab.
4. Enter fields or select options as appropriate, and then click **OK**.

See Also: Table 13–3, " Advanced Connect Data Settings" on page 13-5 for a description of the fields and options

5. Click **OK** to update the connect data information.

Oracle Net Manager

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator pane, expand **Directory** or **Local** > **Service Naming**.
3. Select either the net service name or a database service.

   The right pane displays the current destination service and address list.
4. In the Service Identification box, click **Advanced**.

   The Advanced Service Options dialog box appears.

   See Also: Table 13–3, " Advanced Connect Data Settings" on page 13-5 for a description of the fields and options

5. Enter fields or select options as appropriate, and then click **OK**.
6. If you are making these changes to the **Local** folder, choose **File** > **Save Network Configuration**.

Configuring Connection Load Balancing

The **connection load balancing** feature improves connection performance by balancing the number of active connections among multiple **dispatcher**s. In an **Oracle Real Application Clusters** environment, connection pool load balancing also has the capability to balance the number of active connections among multiple instances.

Because the **PMON process** can register with remote listeners, a listener can always be aware of all instances and dispatchers, regardless of their location. Depending on the load information, a listener decides which instance and, if shared server is configured, which dispatcher to send the incoming client request to.

In a **shared server** configuration, a listener selects a dispatcher in the following order: 1) least loaded node, 2) least loaded instance, and 3) least loaded dispatcher for that instance. In a **dedicated server** configuration, a listener selects an instance in the following order: 1) least loaded node, and 2) least loaded instance.

If a database service has multiple instances on multiple nodes, the listener selects the least loaded instance on the least loaded node. If shared server is configured, then the least loaded dispatcher of the selected instance is chosen.

An Oracle9*i* Real Application Clusters environment requires that the dispatchers on each instance be cross registered with the other listeners on the other nodes. This is achieved by the use of the `LISTENER` attribute of the `DISPATCHERS` parameter.

See Also:

- "Registering Information with a Remote Listener" on page 10-12 for complete information about cross registration
- *Oracle Database Reference* for complete information about the `SERVICE_NAMES` and `INSTANCE_NAME` parameters
- Chapter 12, "Configuring Dispatchers" for complete information about the `LISTENER` attribute

---

Note: For optimum connection pool load balancing results, the instances that belong to the same database service should be on equivalent hardware and software configurations.

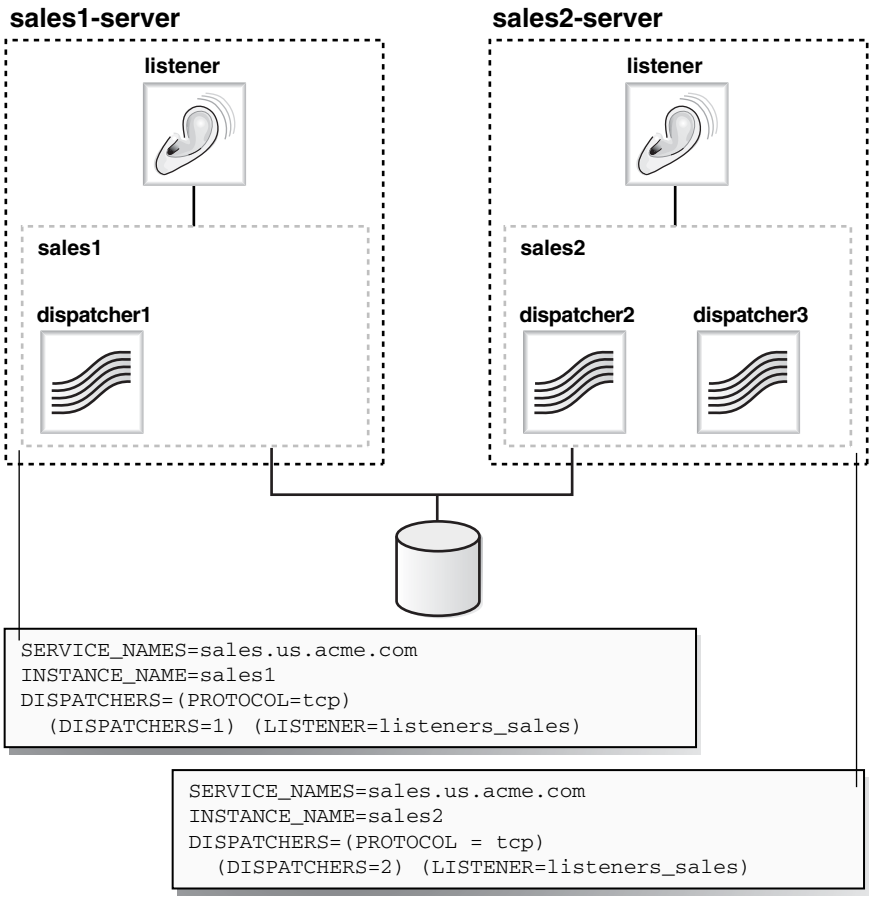
This section includes the following two examples:

- Example: Connection Pool Load Balancing for Shared Server Configuration
- Example: Connection Pool Load Balancing for Dedicated Server Configuration

Example: Connection Pool Load Balancing for Shared Server Configuration

Figure 13–1 shows an Oracle Real Application Clusters shared server database with two instances, `sales1` and `sales2`, of the same service, `sales.us.acme.com`. The instances `sales1` and `sales2` reside on computers `sales1-server` and `sales2-server`, respectively. `sales1` has one dispatcher and `sales2` has two dispatchers. Listeners named `listener` run on nodes 1 and 2, respectively. The `listener` attribute in the `DISPATCHERS` parameter has been configured to allow for service registration of information to both listeners.

Figure 13–1 Load Balancing Environment for a Shared Server Configuration



The `listeners_sales` value in `(LISTENER=listeners_sales)` can be then resolved through a local `tnsnames.ora` file on the both servers as follows:

```
listeners_sales=
 (DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521))
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521)))
```

Based on the environment, the following actions occur. The numbered actions correspond to the arrows shown in Figure 13–2 on page 13-10:

1. PMON processes for instances `sales1` and `sales2` register with both listeners. The listeners are updated on the load of the instances and dispatchers dynamically. The following load information is registered:

   The one minute load average for each instance is `600` for `sales1` and `400` for `sales2`.

   The number of connections to each instance is `200` for `sales1` and `300` for `sales2`.

   The number of dispatcher connections to each instance is `200` for `dispatcher1`, `100` for `dispatcher2`, and `200` for `dispatcher3`.

   The load average on `sales2-server` (400) is less than the load average on `sales1-server` (600). This can happen if more processing is required on `sales1-server`. The number of connections to `sales1` (200) is the same as that

of its only dispatcher, dispatcher1. The number of connections on sales2 (300) is the sum of the connections on its two dispatchers, dispatcher2 (100) and dispatcher3 (200). Therefore, sales2 has more connections than sales1. In this example, sales2-server is the least loaded node, sales2 is the least loaded instance, and dispatcher2 is the least loaded dispatcher.

2. The client sends a connect request.

   A connect descriptor is configured to try each protocol address randomly until one succeeds:

```
sales.us.acme.com=
 (DESCRIPTION=
  (LOAD_BALANCE=on)
  (FAILOVER=on)
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521))
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)))
```

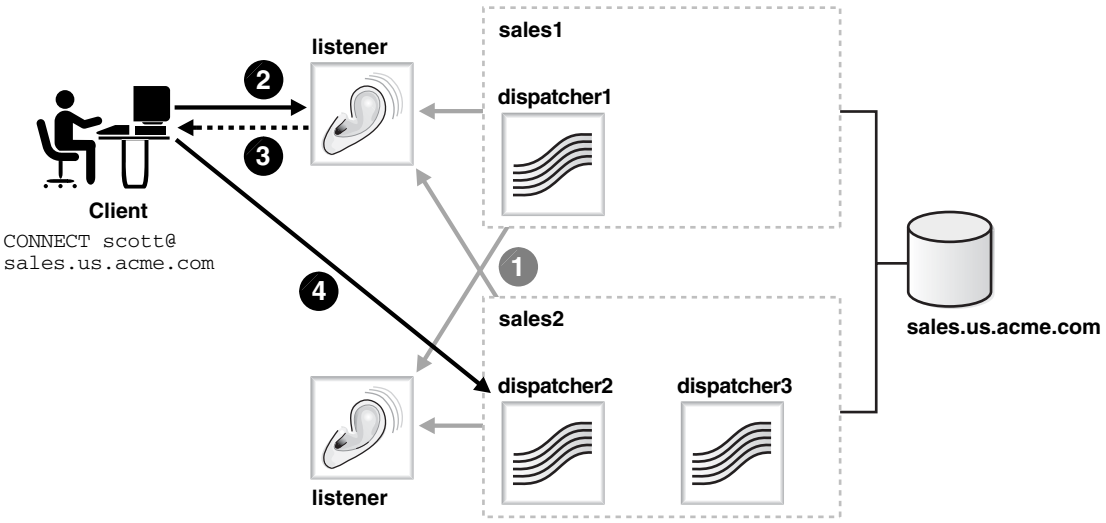
   The listener on sales1-server was randomly chosen to receive the client connect request.

   The listener on sales1-server compares the load of the instances sales1 and sales2. The comparison takes into account the load on nodes sales1-server and sales2-server, respectively. Since sales2-server is less loaded than sales1-server, the listener selects sales2-server over sales1-server.

3. The listener compares the load on dispatchers dispatcher2 and dispatcher3. Because dispatcher2 is less loaded than dispatcher3, the listener redirects the client connect request to dispatcher2.

4. The client connects directly to dispatcher2.

Figure 13–2 Load Balancing Example for a Shared Server Configuration

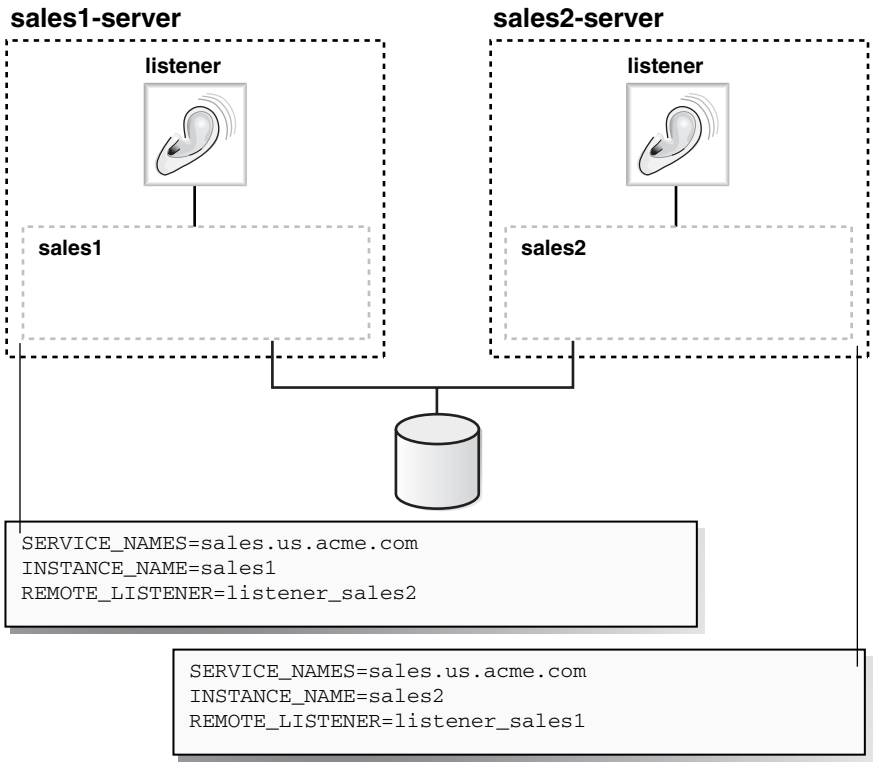


Example: Connection Pool Load Balancing for Dedicated Server Configuration

Figure 13–3 shows an Oracle Real Application Clusters dedicated server database with two instances, sales1 and sales2, of the same service, sales.us.acme.com. The instances sales1 and sales2 reside on computers sales1-server and sales2-server, respectively. Listeners named listener run on nodes 1 and 2,

respectively. The REMOTE_LISTENER parameter has been configured to allow for service registration of information to both listeners.

Figure 13–3 Load Balancing Environment for a Dedicated Server Configuration



The listener_sales2 value in (REMOTE_LISTENER=listener_sales2) can be then resolved through a local tnsnames.ora file on the sales1-server as follows:

```
listener_sales2=
 (DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521)))
```

The listener_sales1 value in (REMOTE_LISTENER=listener_sales1) can be then resolved through a local tnsnames.ora file on the sales2-server as follows:

```
listener_sales1=
 (DESCRIPTION=
   (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521)))
```

Based on the environment, the following actions occur. The numbered actions correspond to the arrows shown in Figure 13–4 on page 13-13:

1. PMON processes for instances sales1 and sales2 register with both listeners. The listeners are updated on the load of the instances dynamically. The load information in Table 13–4 is registered:

Table 13–4 Instance Load Information Upon Which Listeners are Updated

Server or Instance	1 Minute Node Load Average	Number of Connections to Instance
sales1-server	450	

Table 13–4 (Cont.) Instance Load Information Upon Which Listeners are Updated

Server or Instance	1 Minute Node Load Average	Number of Connections to Instance
sales2-server	200	
sales1		200
sales2		150

In Table 13–4, `sales2-server` is the least loaded node and `sales2` is the least loaded instance.

2. The client sends a connect request.

   A connect descriptor is configured to try each protocol address randomly until one succeeds:

```
sales.us.acme.com=
 (DESCRIPTION=
  (LOAD_BALANCE=on)
  (FAILOVER=on)
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521))
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)))
```

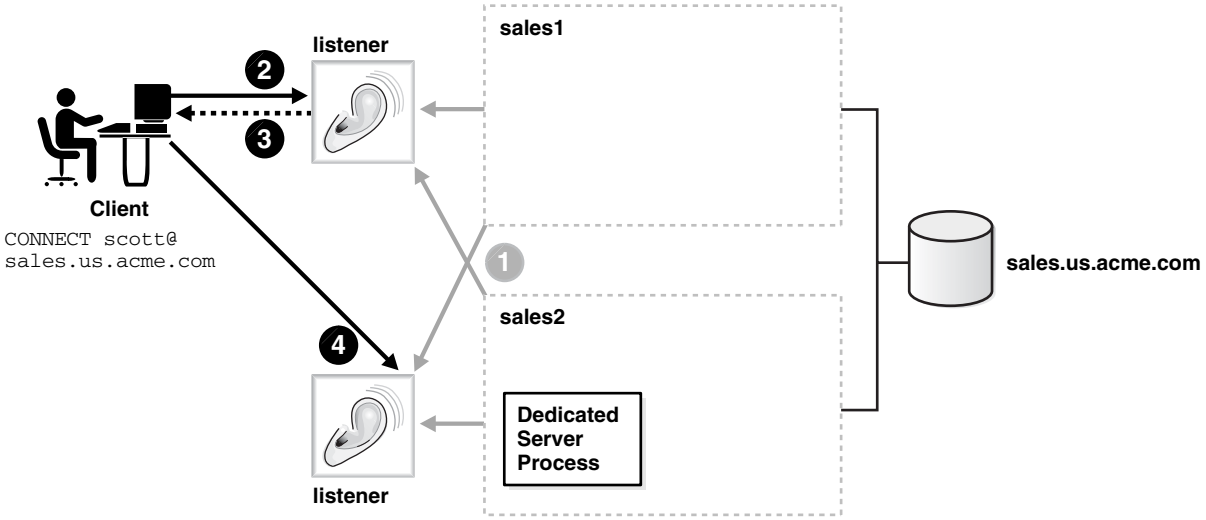
   The listener on `sales1-server` was randomly chosen to receive the client connect request.

   The listener on `sales1-server` compares the load of the instances `sales1` and `sales2`. The comparison takes into account the load on nodes `sales1-server` and `sales2-server`, respectively. Since `sales2-server` is less loaded than `sales1-server`, the listener selects `sales2-server` over `sales1-server`.

3. The listener on `sales1-server` redirects the client connect request to the listener on `sales2-server`.

4. The client connects to the listener on `sale2-server`. The listener starts a dedicated server process, and the dedicated server process inherits the connection request from the listener.

Figure 13–4 Load Balancing Example for a Dedicated Server Configuration



Configuring Transparent Application Failover

Note: **Transparent Application Failover (TAF)** is available with Oracle Database 11*g* Enterprise Edition.

TAF instructs Oracle Net to fail over a failed connection to a different listener. This enables the user to continue to work using the new connection as if the original connection had never failed.

TAF involves manual configuration of a net service name that includes the `FAILOVER_MODE` parameter included in the `CONNECT_DATA` section of the connect descriptor.

This sections covers the following topics:

- About TAF
- What TAF Restores
- TAF Database Configurations
- FAILOVER_MODE Parameters
- TAF Implementation
- TAF Verification

About TAF

Transparent Application Failover (TAF) is a client-side feature that allows for clients to reconnect to surviving databases in the event of a failure of a database instance. Notifications are used by the server to trigger TAF callbacks on the client-side.

TAF is configured using either client-side specified TNS connect string or using server-side service attributes. However, if both methods are used to configure TAF, the server-side service attributes will supersede the client-side settings. The server-side service attributes are the preferred way to set up TAF.

TAF can operate in one of two modes, Session Failover and Select Failover. Session Failover will recreate lost connections and sessions. Select Failover will replay queries that were in progress.

When there is a failure, callback functions will be initiated on the client-side via OCI callbacks. This will work with standard OCI connections as well as Connection Pool and Session Pool connections. Please see the OCI manual for more details on callbacks, Connection Pools, and Session Pools.

TAF will work with RAC. For more details and recommended configurations, please see the RAC Administration Guide.

TAF will operate with Physical Data Guard to provide automatic failover.

What TAF Restores

TAF automatically restores some or all of the following elements associated with active database connections. Other elements, however, may need to be embedded in the application code to enable TAF to recover the connection.

- Client-Server Database Connections
- Users' Database Sessions
- Executed Commands
- Open Cursors Used for Fetching
- Active Transactions
- Serverside Program Variables

Client-Server Database Connections

TAF automatically reestablishes the connection using the same connect string or an alternate connect string that you specify when configuring failover.

Users' Database Sessions

TAF automatically logs a user in with the same user ID as was used prior to failure. If multiple users were using the connection, then TAF automatically logs them in as they attempt to process database commands. Unfortunately, TAF cannot automatically restore other session properties. These properties can, however, be restored by invoking a callback function.

Executed Commands

If a command was completely executed upon connection failure, and it changed the state of the database, TAF does not resend the command. If TAF reconnects in response to a command that may have changed the database, TAF issues an error message to the application.

Open Cursors Used for Fetching

TAF allows applications that began fetching rows from a cursor before failover to continue fetching rows after failover. This is called "select" failover. It is accomplished by re-executing a `SELECT` statement using the same snapshot, discarding those rows already fetched and retrieving those rows that were not fetched initially. TAF verifies that the discarded rows are those that were returned initially, or it returns an error message.

Active Transactions

Any active transactions are rolled back at the time of failure because TAF cannot preserve active transactions after failover. The application instead receives an error message until a `ROLLBACK` is submitted.

Serverside Program Variables

Serverside program variables, such as PL/SQL package states, are lost during failures; TAF cannot recover them. They can be initialized by making a call from the failover callback.

See Also: *Oracle Call Interface Programmer's Guide*

TAF Database Configurations

TAF works with the following database configurations to effectively mask a database failure:

- Oracle Real Application Clusters
- Replicated systems
- Standby databases
- Single instance Oracle database

See Also: *Oracle Real Application Clusters Installation and Configuration Guide*

FAILOVER_MODE Parameters

The `FAILOVER_MODE` parameter must be included in the `CONNECT_DATA` section of a connect descriptor. `FAILOVER_MODE` can contain the subparameters described in Table 13–5.

Table 13–5 Subparameters of the FAILOVER_MODE Parameter

FAILOVER_MODE Subparameter	Description
`BACKUP`	Specify a different net service name for backup connections. A backup should be specified when using `preconnect` to pre-establish connections.
`TYPE`	Specify the type of failover. Three types of Oracle Net failover functionality are available by default to Oracle Call Interface (OCI) applications: ■ `session`: Set to failover the session. If a user's connection is lost, a new session is automatically created for the user on the backup. This type of failover does not attempt to recover selects. ■ `select`: Set to enable users with open cursors to continue fetching on them after failure. However, this mode involves overhead on the client side in normal select operations. ■ `none`: This is the default. No failover functionality is used. This can also be explicitly specified to prevent failover from happening.

Table 13–5 (Cont.) Subparameters of the FAILOVER_MODE Parameter

FAILOVER_MODE Subparameter	Description
METHOD	Determines how fast failover occurs from the primary node to the backup node: ■ basic: Set to establish connections at failover time. This option requires almost no work on the backup server until failover time. ■ preconnect: Set to pre-established connections. This provides faster failover but requires that the backup instance be able to support all connections from every supported instance.
RETRIES	Specify the number of times to attempt to connect after a failover. If DELAY is specified, RETRIES defaults to five retry attempts. **Note:** If a callback function is registered, then this subparameter is ignored.
DELAY	Specify the amount of time in seconds to wait between connect attempts. If RETRIES is specified, DELAY defaults to one second. **Note:** If a callback function is registered, then this subparameter is ignored.

Note: Oracle Net Manager does not provide support for TAF parameters. These parameters must be manually added.

TAF Implementation

Important: Do not set the GLOBAL_DBNAME parameter in the SID_LIST_*listener_name* section of the listener.ora. A statically configured global database name disables TAF.

Depending on the FAILOVER_MODE parameters, you can implement TAF in a number of ways. Oracle recommends the following methods:

- Example: TAF with Connect-Time Failover and Client Load Balancing
- Example: TAF Retrying a Connection
- Example: TAF Pre-Establishing a Connection

Example: TAF with Connect-Time Failover and Client Load Balancing

Implement TAF with connect-time failover and client load balancing for multiple addresses. In the following example, Oracle Net connects randomly to one of the protocol addresses on sales1-server or sales2-server. If the instance fails after the connection, the TAF application fails over to the other node's listener, reserving any SELECT statements in progress.

```
sales.us.acme.com=
 (DESCRIPTION=
  (LOAD_BALANCE=on)
  (FAILOVER=on)
  (ADDRESS=
       (PROTOCOL=tcp)
```

```
      (HOST=sales1-server)
      (PORT=1521))
  (ADDRESS=
      (PROTOCOL=tcp)
      (HOST=sales2-server)
      (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (FAILOVER_MODE=
       (TYPE=select)
       (METHOD=basic))))
```

Example: TAF Retrying a Connection

TAF also provides the ability to automatically retry connecting if the first connection attempt fails with the `RETRIES` and `DELAY` parameters. In the following example, Oracle Net tries to reconnect to the listener on `sales1-server`. If the failover connection fails, Oracle Net waits 15 seconds before trying to reconnect again. Oracle Net attempts to reconnect up to 20 times.

```
sales.us.acme.com=
 (DESCRIPTION=
  (ADDRESS=
      (PROTOCOL=tcp)
      (HOST=sales1-server)
      (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (FAILOVER_MODE=
       (TYPE=select)
       (METHOD=basic)
       (RETRIES=20)
       (DELAY=15))))
```

Example: TAF Pre-Establishing a Connection

A backup connection can be pre-established. The initial and backup connections must be explicitly specified. In the following example, clients that use net service name `sales1.us.acme.com` to connect to the listener on `sales1-server` are also preconnected to `sales2-server`. If `sales1-server` fails after the connection, Oracle Net fails over to `sales2-server`, preserving any `SELECT` statements in progress. Likewise, Oracle Net preconnects to `sales1-server` for those clients that use `sales2.us.acme.com` to connect to the listener on `sales2-server`.

```
sales1.us.acme.com=
 (DESCRIPTION=
  (ADDRESS=
      (PROTOCOL=tcp)
      (HOST=sales1-server)
      (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (INSTANCE_NAME=sales1)
     (FAILOVER_MODE=
       (BACKUP=sales2.us.acme.com)
       (TYPE=select)
       (METHOD=preconnect))))
sales2.us.acme.com=
 (DESCRIPTION=
  (ADDRESS=
      (PROTOCOL=tcp)
```

```
      (HOST=sales2-server)
      (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (INSTANCE_NAME=sales2)
     (FAILOVER_MODE=
       (BACKUP=sales1.us.acme.com)
       (TYPE=select)
       (METHOD=preconnect)))
```

TAF Verification

You can query FAILOVER_TYPE, FAILOVER_METHOD, and FAILED_OVER columns in the V$SESSION view to verify that TAF is correctly configured.

Use the V$SESSION view to obtain information about the connected clients and their TAF status. For example, query the FAILOVER_TYPE, FAILOVER_METHOD, and FAILED_OVER columns to verify that you have correctly configured TAF as in the following SQL statement:

```
SELECT MACHINE, FAILOVER_TYPE, FAILOVER_METHOD, FAILED_OVER, COUNT(*)
FROM V$SESSION
GROUP BY MACHINE, FAILOVER_TYPE, FAILOVER_METHOD, FAILED_OVER;
```

The output before failover resembles the following:

```
MACHINE              FAILOVER_TYPE FAILOVER_M FAI   COUNT(*)
-------------------- ------------- ---------- --- ----------
sales1               NONE          NONE       NO          11
sales2               SELECT        PRECONNECT NO           1
```

The output after failover is:

```
MACHINE              FAILOVER_TYPE FAILOVER_M FAI   COUNT(*)
-------------------- ------------- ---------- --- ----------
sales2               NONE          NONE       NO          10
sales2               SELECT        PRECONNECT YES          1
```

Note: You can monitor each step of TAF using an appropriately configured OCI TAF CALLBACK function.

See Also:

- *Oracle Call Interface Programmer's Guide*
- *Oracle Database Reference* for more information about the V$SESSION view

Specifying the Instance Role for Primary and Secondary Instance Configurations

The INSTANCE_ROLE parameter is an optional parameter for the CONNECT_DATA section of a connect descriptor. It enables you to specify a connection to the primary or secondary instance of Oracle Real Application Clusters configurations.

This parameter is useful when:

- You want to explicitly connect to a primary or secondary instance. The default is the primary instance.
- You want to use TAF to preconnect to a secondary instance.

INSTANCE_ROLE supports the following values:

primary — Specifies a connection to the primary instance

secondary — Specifies a connection to the secondary instance

any — Specifies a connection to whichever instance has the lowest load, regardless of primary or secondary instance role

Example: Connection to Instance Role Type

In the following example, net service name sales_primary enables connections to the primary instance, and net service name sales_secondary enables connections to the secondary instance.

```
sales_primary=
 (DESCRIPTION=
  (ADDRESS=
       (PROTOCOL=tcp)
       (HOST=sales1-server)
       (PORT=1521))
  (ADDRESS=
       (PROTOCOL=tcp)
       (HOST=sales2-server)
       (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (INSTANCE_ROLE=primary)))
sales_secondary=
 (DESCRIPTION=
  (ADDRESS=
       (PROTOCOL=tcp)
       (HOST=sales1-server)
       (PORT=1521))
  (ADDRESS=
       (PROTOCOL=tcp)
       (HOST=sales2-server)
       (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (INSTANCE_ROLE=secondary)))
```

Example: Connection To a Specific Instance

There are times when Oracle Enterprise Manager and other system management products need to connect to a specific instance regardless of its role to perform administrative tasks. For these types of connections, configure (INSTANCE_NAME=*instance_name*) and (INSTANCE_ROLE=any) to connect to the instance regardless of its role.

In the following example, net service name sales1 enables connections to the instance on sales1-server and sales2 enables connections to the instance on sales2-server. (SERVER=dedicated) is specified to force a dedicated server connection.

```
sales1=
 (DESCRIPTION=
  (ADDRESS=
```

```
      (PROTOCOL=tcp)
      (HOST=sales1-server)
      (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (INSTANCE_ROLE=any)
     (INSTANCE_NAME=sales2)
     (SERVER=dedicated)))
sales2=
 (DESCRIPTION=
  (ADDRESS=
      (PROTOCOL=tcp)
      (HOST=sales2-server)
      (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (INSTANCE_ROLE=any)
     (INSTANCE_NAME=sales2)
     (SERVER=dedicated)))
```

Example: TAF Pre-Establishing a Connection

If Transparent Application Failover (TAF) is configured, a backup connection can be pre-established to the secondary instance. The initial and backup connections must be explicitly specified. In the following example, Oracle Net connects to the listener on `sales1-server` and preconnects to `sales2-server`, the secondary instance. If `sales1-server` fails after the connection, the TAF application fails over to `sales2-server`, the secondary instance, preserving any `SELECT` statements in progress.

```
sales1.acme.com=
 (DESCRIPTION=
  (ADDRESS=
      (PROTOCOL=tcp)
      (HOST=sales1-server)
      (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (INSTANCE_ROLE=primary)
     (FAILOVER_MODE=
       (BACKUP=sales2.acme.com)
       (TYPE=select)
       (METHOD=preconnect))))
sales2.acme.com=
 (DESCRIPTION=
  (ADDRESS=
      (PROTOCOL=tcp)
      (HOST=sales2-server)
      (PORT=1521))
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)
     (INSTANCE_ROLE=secondary)))
```

Configuring Connections to Non-Oracle Database Services

The following topics describe how to configure connections to non-Oracle database services:

- Default Configuration for External Procedures

- Configuring Oracle Net Services for Oracle Heterogeneous Services
- Configuring Oracle Net Services for an Oracle Rdb Database

Default Configuration for External Procedures

An **external procedure** is a procedure called from another program, but written in a different language. An example would be a PL/SQL program calling one or more C routines that are required to perform special-purpose processing.

When an application calls an external procedure, Oracle Database starts an external procedure agent named `extproc`. Using the network connection established by Oracle Database, the application passes the following information to the agent:

- DLL or shared library name
- External procedure name
- Any parameters

The agent then loads the DLL or the shared library, and runs the external procedure and passes back to the application any values returned by the external procedure.

The agent must reside on the same computer as the application making the external procedure call.

Note: The default configuration for external procedures no longer requires a network listener to work with Oracle Database and `extproc` agent. The `extproc` agent is spawned directly by Oracle Database and eliminates the risks that extproc might be spawned by Oracle Listener, unexpectedly. This default configuration is recommended for maximum security.

You can change the default configuration for external procedures and have your `extproc` agent spawned by Oracle Listener. To do this, however, you must perform additional network configuration steps.

Having your `extproc` agent spawned by Oracle Listener is necessary if you use:

- Multi-threaded Agent
- Oracle Database in MTS mode on Windows
- `AGENT` clause of the `LIBRARY` specification or `AGENT IN` clause of the `PROCEDURE` specification such that you can redirect external procedures to a different extproc agent

---

When you use the default configuration for external procedures, the `extproc` agent is spawned directly by Oracle Database. There are no configuration changes required for either `listener.ora` or `tnsnames.ora`.

When the default configuration for external procedures is used, define the environment variables to be used by external procedures in the `extproc.ora` file located in the `$ORACLE_HOME/hs/admin` directory on UNIX operating systems or the `%ORACLE_HOME%\hs\admin` directory on Windows.

Configuring Oracle Net Services for External Procedures

You can change the default configuration for external procedures and have your `extproc` agent spawned by the listener similar to prior releases of Oracle Database.

To do this, modify the default configuration, as follows:

1. Configure and run a separate or existing listener to serve external procedures.

   Oracle Net Configuration Assistant configures a listener to accept connections for both the database and external procedures during a database server installation. In addition, Oracle Net Configuration Assistant configures a net service name for the external procedures in `tnsnames.ora` file on the database server. The external procedure agent will only be able to load DLLS from the `bin` or `lib` directories in the `ORACLE_HOME`.

   Example 13–1 shows a sample configuration in the `listener.ora` file.

Example 13–1 listener.ora File with a Sample External Procedure Setup

```
LISTENER=
  (DESCRIPTION=
    (ADDRESS_LIST=
      (ADDRESS=(PROTOCOL=tcp)(HOST=sale-server)(PORT=1521))
      (ADDRESS=(PROTOCOL=ipc)(KEY=extproc)))
SID_LIST_LISTENER=
  (SID_LIST=
    (SID_DESC=
      (GLOBAL_DBNAME=sales.us.acme.com)
      (ORACLE_HOME=/oracle)
      (SID_NAME=sales))
    (SID_DESC=
      (SID_NAME=plsextproc)
      (ORACLE_HOME=/oracle)
      (PROGRAM=extproc))
```

   Example 13–2 shows a sample configuration in the `tnsnames.ora` file.

Example 13–2 tnsnames.ora File a Sample External Procedure Setup

```
EXTPROC_CONNECTION_DATA=
 (DESCRIPTION=
   (ADDRESS=(PROTOCOL=ipc)(KEY=extproc))
   (CONNECT_DATA=
    (SID=plsextproc)))
```

   The `extproc` agent spawned by the listener inherits the operating system privileges of the listener. Therefore, if you configure a separate listener, run with operating system privileges lower than those of the listener for the database.

2. Restrict the DLLs that the `extproc` agent can load by listing them explicitly in the `listener.ora` file.

The details of these tasks follow.

To modify the default configuration for external procedures, configure and run a separate or existing listener to serve external procedures:

1. To configure an existing listener to serve external procedures, choose the existing listener and configure it using Oracle Net Configuration Assistant.

   For most installation types, this listener is named `LISTENER`.

   a. Access the Oracle Net Administration page in Oracle Enterprise Manager.

b. Select **Listeners** from the **Administer** list, and then select the Oracle home that contains the location of the configuration files.

c. Click **Go**.

   The Listeners page appears.

d. Select the existing listener created by Oracle Net Configuration Assistant, and then click **Edit**.

   The Edit Listeners page appears.

e. In the **Addresses** section, select the protocol address for external procedures, and then click **Add**.

f. Click the **Other Services** tab.

g. Select the row representing the service information for external procedures, and then click **Add**.

2. Add service information about extproc in the listener.ora file, including the parameters described in Table 13–6.

   See Also: "Oracle Enterprise Manager" on page 6-1

To configure and run a separate listener to serve external procedures, remove the external procedure entries for a different listener using Oracle Net Configuration Assistant.

1. Create another listener to exclusively handle external procedures:

   a. Navigate back to the Listeners page.

   b. Click Create.

      The Create Listener page appears.

   c. In the Listener Name field, enter a unique listener name, such as `LISTENEREXTPROC`, in the **Listener Name** field.

2. In the **Addresses** section, configure an IPC protocol address.

   a. Click **Add**.

      The Add Address page appears.

   b. From the **Protocol** list, select IPC.

   c. In the **Key** field, enter a key value of `extproc`.

      See Also: "Configuring Listening Protocol Addresses" on page 10-3 for more information about configuring listener protocol addresses

   d. Click **OK**.

3. Add service information about `extproc` in the `listener.ora` file, including the parameters described in Table 13–6.

Table 13–6 External Procedures Settings in listener.ora

Oracle Enterprise Manager Field	listener.ora Parameter	Description
Program Name	PROGRAM	Specify the name of the external procedure agent executable. **Note:** On Windows, the executable must reside in the *ORACLE_HOME*\bin directory.
Environment Variables	ENVS	When the default configuration for external procedures is used, define the environment variables to be used by external procedures in the extproc.ora file located in the $ORACLE_HOME/hs/admin directory on UNIX operating sytems or the %ORACLE_HOME%\hs\admin directory on Windows. **Note:** When extproc.ora is in use, it precedes the same environment variables of ENVS in listener.ora. **Syntax:** SET *name=value* **Example:** SET EXTPROC_DLLS=ANY Specify the EXTPROC_DLLS environment variable to restrict the DLLs that extproc is allowed to load. Without the EXTPROC_DLLS environment variable, extproc loads DLLs from $ORACLE_HOME/lib on UNIX operating systems and *ORACLE_HOME*\bin on Windows. Set EXTPROC_DLLS to one of the following values: ■ Colon-separated list of the DLLs **Syntax:** "*DLL*:*DLL*" **Description:** This value allows extproc to load the specified DLLs and the DLLs from $ORACLE_HOME/lib on UNIX operating systems and *ORACLE_HOME*\bin on Windows. You must enter the complete directory path and file name of the DLLs. ■ ONLY (Recommended for maximum security) **Syntax:** "ONLY:*DLL*:*DLL*" **Description:** This value allows extproc to load only the specified DLLs. You must enter the complete directory path and file name of the DLLs. ■ ANY **Syntax:** "ANY" **Description:** This value allows extproc to load any DLL. ANY disables DLL checking. **Examples:** "EXTPROC_DLLS=/home/xyz/mylib.so:/home/abc/urlib.so,LD_LIBRARY_PATH=/private/xpm/lib:/private/mylibs, MYPATH=/usr/ucb:/usr/local/packages,APL_ENV_FILE=/apl/conf/env.txt" "EXTPROC_DLLS=ONLY:/home/xyz/mylib.so:/home/abc/urlib.so,LD_LIBRARY_PATH=/private/xpm/lib:/private/mylibs, MYPATH=/usr/ucb:/usr/local/packages,APL_ENV_FILE=/apl/conf/env.txt" "EXTPROC_DLLS=ANY,LD_LIBRARY_PATH=/private/xpm/lib:/private/mylibs, MYPATH=/usr/ucb:/usr/local/packages,APL_ENV_FILE=/apl/conf/env.txt"
Oracle Home Directory	ORACLE_HOME	Specify the Oracle home location of the agent.
SID	SID_NAME	Specify a system identifier for the external procedure agent by any name.

To configure service information about extproc:

1. Click the **Other Services** tab.
2. Click **Add**.

   The Create Other Service page appears.
3. Enter `extproc` in the **Program Name** field, and the Oracle home where the `extproc` executable resides in the **Oracle Home Directory** field, and a system identifier, such as `extproc`, in the **SID** field.
4. In the **Environment Variables** section, click **Add Another Row**.
5. Enter the `EXTPROC_DLLS` environment variable in the **Name** field and the directory path and file name of the DLLs in the **Value** field.
6. Click **OK**.

   The Create Listener page appears.
7. Click **OK** to add the listener.

   The listener is added to the Listeners page.

   The `listener.ora` file updates with information for external procedures, as shown in the following output:

```
LISTENEREXTPROC=
 (DESCRIPTION=
  (ADDRESS=
     (PROTOCOL=ipc)(KEY=extproc)))
SID_LIST_LISTENEREXTPROC=
  (SID_LIST=
    (SID_DESC=
     (PROGRAM=extproc)
     (ENVS="EXTPROC_DLLS=ONLY:/home/xyz/mylib.so:/home/abc/urlib.so,
      LD_LIBRARY_PATH=/private/xpm/lib:/private/mylibs,
      MYPATH=/usr/ucb:/usr/local/packages,APL_ENV_FILE=/apl/conf/env.txt")
     (SID_NAME=extproc)
     (ORACLE_HOME=/oracle)))
```

8. Start the listener for external procedures from a user account with lower privileges than the `oracle` user.

   Ensure that this user account does not have general access to `oracle`-owned files. Specifically, this user should not have permission to read or write to database files or to the Oracle server address space. In addition, this user should have read access to the `listener.ora` file, but must not have write access to it.

   Running the listener with lower privileges also prevents you from using Listener Control utility `SET` commands to alter the configuration of this listener in the `listener.ora` file. For this reason, Oracle Corporation recommends that you complete `listener.ora` file configuration prior to running the listener.

   See Also:

   - "Task 1: Start the Listener" on page 15-2 for instructions on using the Listener Control utility `START` command to start the listener
   - *Oracle Database Advanced Application Developer's Guide* for instruction on enabling external procedure calls

Configuring Oracle Net Services for Oracle Heterogeneous Services

Heterogeneous Services are an integrated component within the Oracle database server, and provides the generic technology for accessing non-Oracle systems from the Oracle database server. Heterogeneous Services enable you to:

- Use Oracle SQL to transparently access data stored in non-Oracle systems as if the data resides within an Oracle database server
- Use Oracle procedure calls to transparently access non-Oracle systems, services, or application programming interfaces (APIs), from your Oracle distributed environment

While Heterogeneous Services provides the generic technology in the Oracle database server, a Heterogeneous Service agent is required to access a particular non-Oracle system.

To initiate a connection to the non-Oracle system, the Oracle database server starts an agent process through the listener on the gateway. For the Oracle database server to be able to connect to the agent, perform the following steps:

1. Configure the listener on the gateway to listen for incoming requests from the Oracle database server and spawn Heterogeneous Services agents by configuring the following parameters in the `listener.ora` file:
   - `PROGRAM`: Specify the name of the agent executable
   - `ORACLE_HOME`: Specify the Oracle home location of the agent executable
   - `SID_NAME`: Specify the Oracle System Identifier (SID)
2. Configure the `PROGRAM`, `ORACLE_HOME`, and `SID` parameters in Oracle Enterprise Manager.

   Access the Oracle Net Administration page in Oracle Enterprise Manager.

   See Also: "Oracle Enterprise Manager" on page 6-1
3. Select **Listeners** from the **Administer** list, and then select the Oracle home that contains the location of the configuration files.
4. Click **Go**.

   The Listeners page appears.
5. Select the listener created by Oracle Net Configuration Assistant, and then click **Edit**.

   The Edit Listeners page appears.
6. In the Addresses section, select the protocol address for external procedures, and then click **Remove**.
7. Click the **Other Services** tab.
8. Click **Add**.

   The Create Other Service page appears.
9. Enter the program name in the **Program Name** field that will be executed to create a gateway, the Oracle home where the agent executable resides in the **Oracle Home Directory** field, and the SID or service name of the non-Oracle system in the **SID** field.
10. Click **OK**.

The Edit Listener page appears.

11. Click **OK** to modify the listener.

    The Listeners page appears.

    The `listener.ora` file updates information about the Heterogeneous Services, as shown in the following:

```
SID_LIST_LISTENER=
 (SID_LIST=
  (SID_DESC=
   (SID_NAME=sybasegw)
   (ORACLE_HOME=/oracle10g)
   (PROGRAM=tg4sybs))
```

    See Also: *Oracle Database Heterogeneous Connectivity User's Guide*

12. On the computer where the Oracle database resides, set up a net service name to connect to the listener on the gateway. The connect descriptor must also include the `HS=ok` clause to make sure the connection uses Heterogeneous Services:

    a. Create a net service name that can be used for connections from the Oracle database server to a non-Oracle system.

       See Also: "Task 1: Configure Net Service Names" on page 8-9 for local naming instructions and "Task 2: Create or Modify Net Entries" on page 8-14 for directory naming instructions

    b. Use either Oracle Enterprise Manager or Oracle Net Manager to configure `HS=ok`.

       For Oracle Enterprise Manager, click the **Advanced** tab in the Create Net Service Name page, and then click the **Use for Heterogeneous Services**.

       For Oracle Net Manager, click **Advanced** in the **Service Identification** box. The Advanced Service Options dialog box appears. Click **Use for Heterogeneous Services**.

    c. Click **OK** to confirm the change.

       The `tnsnames.ora` file updates with the new net service name configured for Heterogeneous Services, as shown in the following:

```
sybase_gtw=
 (DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=gate-server)(PORT=1521))
  (CONNECT_DATA=
    (SERVICE_NAME=sybasegw)
  )
    (HS=ok))
 )
```

Configuring Oracle Net Services for an Oracle Rdb Database

Oracle Rdb is a database for Digital's 64-bit operating systems. Because Oracle Rdb has its own listener, the client interacts with Rdb in the same manner as it does with an Oracle database.

To initiate a connection to an Oracle Rdb, set up a net service name to connect to the Oracle Rdb database using the parameters described in Table 13–7.

Table 13–7 Oracle RDB Database Settings in a Connect Descriptor

Oracle Enterprise Manager Field	tnsnames.ora Parameter	Description
Rdb Database	RDB_DATABASE	Specify the file name of an Oracle Rdb database.
Type of Service	TYPE_OF_SERVICE	Specify the type of service to use for an Oracle Rdb database. It is used by Rdb interface tools. This feature should only be used if the application supports both Oracle Rdb and Oracle database services, and you want the application to load balance between the two.
Global Database Name	GLOBAL_NAME	(Optional) Specify the Oracle Rdb database.

See Also: Oracle Rdb documentation

To configure a client for an Oracle Rdb database, use Oracle Net Manager:

1. Create a net service name that can be used for connections from the Oracle server to a non-Oracle system.

   See Also: "Task 1: Configure Net Service Names" on page 8-9 for local naming instructions and "Task 2: Create or Modify Net Entries" on page 8-14 for directory naming instructions

2. Use either Oracle Enterprise Manager or Oracle Net Manager to the Oracle Rdb parameters.

   For Oracle Enterprise Manager, click the **Advanced** tab in the Create Net Service Name page.

   For Oracle Net Manager, click **Advanced** in the **Service Identification** box. The Advanced Service Options dialog box appears.

3. Enter the file name of an Oracle Rdb database in the **Rdb Database** field.

4. Optionally, enter the global database name in the **Global Database Name** field, and, if needed, specify the type of service in the **Type of Service** field, and then click **OK.**

   The `tnsnames.ora` file updates with the new net service name configured for the Oracle Rdb database, as shown in the following:

```
alpha5=
  (DESCRIPTION=
    (ADDRESS=...)
    (CONNECT_DATA=
      (SERVICE_NAME=generic)
      (RDB_DATABASE=[.mf]mf_personnel.rdb)
     (GLOBAL_NAME=alpha5)))
```

   In the following example, TYPE_OF_SERVICE is used to load balance between an Oracle Rdb database service and an Oracle database service:

```
alpha5=
 (DESCRIPTION_LIST=
  (DESCRIPTION=
    (ADDRESS=...)
    (CONNECT_DATA=
     (SERVICE_NAME=generic)
     (RDB_DATABASE=[.mf]mf_personnel.rdb)
     (GLOBAL_NAME=alpha5))
```

```
(DESCRIPTION=
 (ADDRESS=...)
  (CONNECT_DATA=
   (SERVICE_NAME=sales.us.acme.com))
 (TYPE_OF_SERVICE=oracle9_database))
```

See Also: Oracle Rdb documentation

14

Optimizing Performance

This chapter describes how to optimize connection performance.

This chapter contains these topics:

- Configuring Session Data Unit
- Configuring I/O Buffer Space
- Configuring SDP Protocol Support for Infiniband Network Communication to the Database Server
- Configuring the Listener and the Oracle Database To Limit Resource Consumption By Unauthorized Users

Configuring Session Data Unit

Under typical database configuration, Oracle Net encapsulates data into buffers the size of the **session data unit (SDU)** before sending the data across the network. Oracle Net sends each buffer when it is filled, flushed, or when an application tries to read data. Adjusting the size of the SDU buffers relative to the amount of data provided to Oracle Net to send at any one time can improve performance, network utilization and memory consumption.

The amount of data provided to Oracle Net to send at any one time can be referred to as the message size. Oracle Net assumes by default that the message size will normally vary between 0 and 8192 bytes, and infrequently, will be larger than 8192. If this assumption is true, then most of the time, the data will be sent using one SDU buffer. This assumption is why the default value for the SDU size is 8192.

Consider changing the SDU size when the predominant message size is smaller or larger than 8192. The SDU size you choose should be 70 bytes larger than the predominant message size, as long as the maximum SDU size is not exceeded. If the predominant message size plus 70 bytes exceeds the maximum SDU, then the SDU should be set such that the message size is divided into the smallest number of equal parts where each part is 70 bytes less than the SDU size.

The SDU size can range from 512 bytes to 32767 bytes. If the `DEFAULT_SDU_SIZE` parameter is not configured in the `sqlnet.ora` file, then the default SDU for the client and a dedicated server is 8192 bytes, while for a shared server the default SDU is 32767 bytes.

The actual SDU size used is negotiated between the client and the server at connect time and will be the smaller of the client and server values. As such, configuring an SDU size different from the default requires configuring the SDU on both the client and server computers, unless you are using shared servers, in which case only the client needs to be changed because the shared server defaults to the maximum value.

For example, if the majority of the messages sent and received by the application are smaller than 8K in size, taking into account about 70 bytes for overhead, setting the SDU to 8K will likely produce good results. If sufficient memory is available, using the maximum value for the SDU will minimize the number of system calls and overhead for Oracle Net Services.

Client-Side Configuration

To configure the client, set the SDU size in the following places:

- `sqlnet.ora` File

  For global configuration on the client side, configure the `DEFAULT_SDU_SIZE` parameter in the `sqlnet.ora` file:

  ```
  DEFAULT_SDU_SIZE=32767
  ```

- Connect Descriptors

  For a particular connect descriptor, you can override the current settings in the client side `tnsnames.ora` file. In a connect descriptor, you specify the `SDU` parameter for a description.

  ```
  sales.us.acme.com=
  (DESCRIPTION=
      (SDU=11280)
        (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
      (CONNECT_DATA=
       (SERVICE_NAME=sales.us.acme.com))
  )
  ```

SDU size applies to all Oracle Net protocols.

Server-Side Configuration

To configure the database server, set the SDU size in the following places:

- `sqlnet.ora` File

  Configure the `DEFAULT_SDU_SIZE` parameter in the `sqlnet.ora` file:

  ```
  DEFAULT_SDU_SIZE=32767
  ```

- If using shared server processes, set the SDU size in the `DISPATCHERS` parameter in the initialization parameter file as follows:

  ```
  DISPATCHERS="(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp))(SDU=8192))"
  ```

- If you have configured the listener with a list of targets in the `listener.ora` file, the value for SDU in the `SID_LIST` element overrides the current setting in the `sqlnet.ora` file when using dedicated server processes.

  ```
  SID_LIST_listener_name=
    (SID_LIST=
      (SID_DESC=
       (SDU=8192)
       (SID_NAME=sales)))
  ```

  Note that the smaller value of the SDU size and the value configured for the client will win.

Configuring I/O Buffer Space

Reliable network protocols like TCP/IP buffer data into send and receive buffers while sending and receiving to or from lower and upper layer protocols. The sizes of these buffers affect network performance, as these buffer sizes influence flow control decisions.

The RECV_BUF_SIZE and SEND_BUF_SIZE parameters specify sizes of socket receive and send buffers, respectively, associated with an Oracle Net connection.

To ensure the continuous flow of data and better utilization of network bandwidth, specify the I/O buffer space limit for receive and send operations of sessions with the RECV_BUF_SIZE and SEND_BUF_SIZE parameters.

For best performance, the size of the send and receive buffers should be set large enough to hold all of the data that may be sent concurrently on the network connection. For a simple database connection, this typically maps to the OCI_PREFETCH_MEMORY size.

Setting the SEND_BUF_SIZE and RECV_BUF_SIZE to at least the bandwidth-delay product, will insure that when large amounts of data are being sent that the network bandwidth will be optimally utilized.

For example, suppose that the network link between a primary database and a standby database has a round trip time of 34 ms and a bandwidth of 15 Mbps. The bandwidth-delay product of this network link is approximately 64KB. The largest message used to transfer redo data between a primary database and a standby database is 1MB so the optimum value for the SEND_BUF_SIZE and RECV_BUF_SIZE parameters in this scenario is therefore 1MB however a setting of at least 64KB should be sufficient to optimize use of the available bandwidth.

Note: The actual value of the SEND_BUF_SIZE and RECV_BUF_SIZE parameters may be less than the value specified either because of limitations in the host operating system or due to memory constraints.

Note Also: It is important to consider the total number of concurrent connections that your system must support and the memory resources that are available because the total amount of memory that will be consumed by these connections will depend on both the number of concurrent connections and the size of their respective buffers.

For most network protocols, ensure that the RECV_BUF_SIZE parameter at one end of the network connection, typically at the client, is equal to the value of the SEND_BUF_SIZE parameter at the other end, typically at the server.

Specify these parameters in the sqlnet.ora file or in the connect descriptor on the client side, and the listener.ora and sqlnet.ora files on the server side.

Note: Use these parameters with caution as they affect network and system performance. The default values for these parameters are operating-system specific. Following are the defaults for the Solaris 2.8 Operating System:

- `SEND_BUF_SIZE`: 16,384 bytes
- `RECV_BUF_SIZE`: 24,576 bytes

The default size for both `SEND_BUF_SIZE` and `RECV_BUF_SIZE` for Solaris 2.9 is 49,152 bytes.

These parameters are supported for TCP, TCP/IP with SSL, and SDP protocols. Additional protocols may support these parameters on certain operating systems. Refer to operating-system specific documentation of Oracle Net for additional information.

Determining Bandwidth Delay Product

Bandwidth-delay product is the product of network bandwidth and the round trip time of data going over the network. The easiest way to determine the round trip time is to use a command such as ping from one host to another and use the response times returned by ping.

For example, if a network has a bandwidth of 100 Mbps and a round trip time of 5ms, the send and receive buffers should be at least (100X106) * (5X10-3) bits or approximately 62.5 Kilobytes.

For a better understanding of the relationships among the units and factors involved, refer to the following equation:

```
100,000,000 bits   1 byte   5 seconds
---------------- x ------ x --------- = 62,500 bytes
 1 second          8 bits     1000
```

See Also: "Using the Trace Assistant to Examine Trace Files" on page 16-44

Clientside Configuration

To configure the client, set the buffer space size in the following places:

- `sqlnet.ora` File

  For global configuration on the clientside, configure the `sqlnet.ora` file. Setting just the RECV_BUF_SIZE parameter is typically adequate. If the client is sending large requests, then also set the SEND_BUF_SIZE.

  ```
  RECV_BUF_SIZE=11784
  ```

  See Also: "Configuring Advanced Profile Information" on page 9-5

- Connect Descriptors

  For a particular connect descriptor, you can override the current settings in the clientside `sqlnet.ora` file. In a connect descriptor, you either specify the buffer space parameters for a particular protocol address or description.

```
sales.us.acme.com=
 (DESCRIPTION=
   (ADDRESS_LIST=
     (ADDRESS=(PROTOCOL=tcp)(HOST=sales1-server)(PORT=1521)
        (SEND_BUF_SIZE=11784)
        (RECV_BUF_SIZE=11784))
     (ADDRESS=(PROTOCOL=tcp)(HOST=sales2-server)(PORT=1521)
        (SEND_BUF_SIZE=11784)
        (RECV_BUF_SIZE=11784))
   (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)))
hr.us.acme.com=
 (DESCRIPTION=
   (SEND_BUF_SIZE=11784)
   (RECV_BUF_SIZE=11784)
     (ADDRESS=(PROTOCOL=tcp)(HOST=hr1-server)(PORT=1521))
   (CONNECT_DATA=
     (SERVICE_NAME=hr.us.acme.com)))
```

Serverside Configuration

Because the database server writes data to clients, setting just the SEND_BUF_SIZE parameter at the serverside is typically adequate. If the database server is receiving large requests, then also set the RECV_BUF_SIZE parameter.

To configure the database server, set the buffer space size in the `listener.ora` and in `sqlnet.ora` files.

listener.ora

In the `listener.ora` file, you can either specify the buffer space parameters for a particular protocol address or for a description.

```
LISTENER=
(DESCRIPTION=
(ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521)
(SEND_BUF_SIZE=11784)
(RECV_BUF_SIZE=11784))
(ADDRESS=(PROTOCOL=ipc)(KEY=extproc)
(SEND_BUF_SIZE=11784)
(RECV_BUF_SIZE=11784)))
LISTENER2=
(DESCRIPTION=
(SEND_BUF_SIZE=11784)
(RECV_BUF_SIZE=11784)
(ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521)))
```

sqlnet.ora

```
RECV_BUF_SIZE=65536
SEND_BUF_SIZE=65536
```

DISPATCHERS Initialization Parameter

If using shared server processes, you can override the current settings obtained from the server's `sqlnet.ora` file by setting the buffer space parameters in the `DISPATCHERS` initialization parameter as follows:

```
DISPATCHERS="(ADDRESS=(PROTOCOL=tcp)(SEND_BUF_SIZE=65536))"
```

Configuring SDP Protocol Support for Infiniband Network Communication to the Database Server

Oracle Net Services provides support for the **SDP protocol** for Infiniband high-speed networks.

The SDP protocol is a standard communication protocol for clustered server environments. SDP is an interface between a network interface card and the application. By using SDP, applications place most of the messaging burden upon the network interface card, freeing the CPU for other tasks. As a result, SDP decreases network **latency** and CPU utilization.

SDP is designed specifically for System Area Networks (SANs). A SAN is characterized by short-distance, high-performance communications between multiple server systems, such as Oracle Application Server (OracleAS) or any other third-party middle-tier client and database servers clustered into one switch.

This section describes how to set up Oracle Net support of SDP for middle tier and database server communication. It includes the following topics:

- Prerequisites to Using SDP Protocol Support
- Serverside Configuration
- Clientside Configuration

> **See Also:** "Performance Between the Middle Tier and Oracle Database" on page 1-11 for an overview of supported deployments

---

Note: Please check with your individual vendor for their version compatibility with Oracle Database 11*g*.

Visit the Oracle Technology Network for further information about SDP protocol support at

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

Prerequisites to Using SDP Protocol Support

Prior to configuring support for the SDP protocol, install the required hardware, and set up Infiniband hardware and software compatible with OpenFabrics Enterprise Distribution (OFED) 1.2 from a designated vendor on both the application Web server and database server.

In the process of installing the Infiniband software, identify the constant that defines the SDP protocol or address family for your system. This can be obtained from your operating system or OFED documentation.

Perform the following:

1. Rerun the Oracle Universal Installer.
2. In the Available Products page, select **Oracle Database 11*g* Server** or **Oracle Database 11*g* Client**.
3. In the Installation Types page, select **Custom**.
4. In the Available Product Components page, select only **Oracle Net Services**.

Serverside Configuration

To configure the database server, configure an SDP protocol address in the `listener.ora` file on the database server.

> **Note:** If the SDP protocol or address protocol family constant is not 27, the default value for Oracle Net Services, define the constant in the `SDP.PF_INET_SDP` parameter in the `sqlnet.ora` file.

The following example shows an SDP endpoint that uses port number 1521 on the computer `sales-server`.

```
LISTENER=
  (DESCRIPTION=
    (ADDRESS_LIST=
      (ADDRESS=(PROTOCOL=sdp)(HOST=sales-server)(PORT=1521))
      (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
      (ADDRESS=(PROTOCOL=ipc)(KEY=extproc))))
```

> **See Also:** "Creating a List of Listener Protocol Addresses" on page 13-1

Clientside Configuration

> **Note:** If the SDP protocol or address protocol family constant is not 27, the default value for Oracle Net Services, define the constant in the `SDP.PF_INET_SDP` parameter in the `sqlnet.ora` file.

To configure the OracleAS servers or third-party middle-tier client:

1. If configuring third-party middle-tier client, upgrade the clients to use Oracle Database 11*g* Client software. From the Oracle Universal Installer, In the Available Products page, select Oracle Database 11*g* Client.
2. For both OracleAS servers and third-party middle-tier client, create a net service name to connect to the database server:
   - For OracleAS servers, specify a net service name that uses the same TCP/IP protocol address configured in the `tnsnames.ora` file. For example:

     ```
     sales=
      (DESCRIPTION=
        (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)))
        (CONNECT_DATA=
          (SERVICE_NAME=sales.us.acme.com)))
     ```

   - For third-party middle-tier clients, specify a net service name that uses the same SDP protocol address configured in the `tnsnames.ora` file.

     For example:

     ```
     sales=
      (DESCRIPTION=
        (ADDRESS=(PROTOCOL=sdp)(HOST=sales-server)))
        (CONNECT_DATA=
          (SERVICE_NAME=sales.us.acme.com)))
     ```

See Also: Chapter 8, "Configuring Naming Methods" for more information about creating connect descriptors

Configuring the Listener and the Oracle Database To Limit Resource Consumption By Unauthorized Users

Unauthorized access to the listener or database server can result in denial-of-service attacks, whereby an unauthorized client attempts to block authorized users' ability to access and use the system when needed. Malicious clients may attempt to flood the listener or database server with connect requests that have the sole purpose of consuming resources, such as connections, processes, or threads. To mitigate these types of attacks, configure limits that constrain the time in which resources can be held prior to authentication. Client attempts to exceed the configured limits will result in connection terminations and an audit trail containing the IP address of the client being logged.

To limit the resource consumption by unauthorized users and enable the audit trail, set time-limit values for the parameters described in Table 14–1. These parameters do not have default values.

Table 14–1 Connect-Timeout Parameters

Parameter	Description
`INBOUND_CONNECT_TIMEOUT_`*`listener_name`* in `listener.ora`	Specify the time, in seconds, for the client to complete its connect request to the listener after the network connection had been established. If the listener does not receive the client request in the time specified, then it terminates the connection. In addition, the listener logs the IP address of the client and an ORA-12525: TNS:listener has not received client's request in time allowed error message to the `listener.log` file. **See Also:** ■ "Resolving the Most Common Error Messages for Oracle Net Services" on page 16-10 for a description of error message workarounds ■ "Analyzing Listener Log Files" on page 16-26 for further information about logged entry in the `listener.log` file
`SQLNET.INBOUND_CONNECT_TIMEOUT` parameter in `sqlnet.ora` on the database server	Specify the time, in seconds, for a client to connect with the database server and provide the necessary authentication information. If the client fails to establish a connection and complete authentication in the time specified, then the database server terminates the connection. In addition, the database server logs the IP address of the client and an ORA-12170: TNS:Connect timeout occurred error message to the `sqlnet.log` file. The client receives either an `ORA-12547: TNS:lost contact` or an `ORA-12637: Packet receive failed` error message. **See Also:** ■ "Configuring Advanced Profile Information" on page 9-5 for information on configuring this parameter ■ "Resolving the Most Common Error Messages for Oracle Net Services" on page 16-10 for a description of error message workarounds

When specifying values for these parameters, consider the following recommendations:

- Set both parameters to an initial low value.
- Set the value of the `INBOUND_CONNECT_TIMEOUT_`*`listener_name`* parameter to a lower value than the `SQLNET.INBOUND_CONNECT_TIMEOUT` parameter.

For example, you can set INBOUND_CONNECT_TIMEOUT_*listener_name* to 2 seconds and INBOUND_CONNECT_TIMEOUT parameter to 3 seconds. If clients are unable to complete connections within the specified time due to system or network delays that are normal for the particular environment, then increment the time as needed.

Part III

Testing and Troubleshooting Oracle Net Services

Part III describes how to establish connections, and identify and diagnose problems with Oracle Net Services.

This part includes the following chapters:

- Chapter 15, "Establishing a Network and Testing the Connection"
- Chapter 16, "Troubleshooting Oracle Net Services"

15

Establishing a Network and Testing the Connection

Once you have completed configuring the network, you should make a connection and test each component to ensure that the network is functioning properly. Oracle Net Services provide a variety of tools to help you start, test, and control the listener and Oracle Connection Manager.

This chapter outlines procedures to make a connection and test network components. This chapter contains these topics:

- Connecting to a Database
- Testing the Network

Connecting to a Database

Connecting to a database involves starting network components and entering a connect string with a net service name, such as the following:

```
CONNECT username@connect_identifier
Enter password: password
```

This section contains these topics:

- Starting Oracle Net Services Components
- Entering a Connect String
- Initiating Connections

Starting Oracle Net Services Components

Client workstations and other servers connect to a listener with a net service name when logging onto an Oracle database server.

After installing and configuring all the network components, you need to start them to make the network functional. Following is an outline of the tasks you should perform to start the network components.

Task 1: Start the Listener

Task 2: Start the Database

Task 3: Start Oracle Connection Manager

Task 1: Start the Listener

For Oracle Net to accept connections on the database server, start the listener with the Listener Control utility on the server:

1. Determine the status of the listener. From the command line, enter:

   ```
   lsnrctl
   LSNRCTL> STATUS [listener_name]
   ```

   where *listener_name* is the name of the listener defined in the `listener.ora` file. It is not necessary to identify the listener if you are using the default listener, named `LISTENER`.

   If the `STATUS` command indicates that the listener is running stop the listener, as follows:

   ```
   LSNRCTL> SET PASSWORD password
   LSNRCTL> STOP [listener_name]
   ```

   `SET PASSWORD` is only required if the password is set in the `listener.ora` file. The password defaults to `ORACLE`.

2. Start the listener. Enter:

   ```
   LSNRCTL> START [listener_name]
   ```

   The Listener Control utility will display a status message indicating that the listener has started successfully. Check that all expected services for that listener are listed in the services summary in the status message.

3. Exit from the Listener Control utility. Enter:

   ```
   LSNRCTL> EXIT
   ```

On Windows, the listener can also be started through the Control Panel:

1. Select the **Services** icon in the Control Panel window.
2. Select the `Oracle`*HOME_NAME*`TNSListener` service—the service name if you are using the default listener name `LISTENER`—or `Oracle`*HOME_NAME*`TNSListener`*lsnr*, where *lsnr* is the nondefault listener name.
3. Click **Start** to start the service.
4. In the Services window, click **Close**.

Task 2: Start the Database

Use the tool of choice, such as SQL*Plus, to start the database:

1. Start SQL*Plus without connecting to the database:

   ```
   sqlplus /nolog
   ```

2. Connect to Oracle as `SYSDBA`:

   ```
   SQL> CONNECT username as sysdba
   Enter password: password
   ```

3. When you enter a `STARTUP` command, specify the database name and full path of the parameter file:

   ```
   SQL> STARTUP database_name pfile=file
   ```

If you do not specify the PFILE option, the Oracle database uses the standard initialization parameter file located in the $ORACLE_BASE/admin/*db_name*/pfile/*sid* directory on UNIX platforms, and *ORACLE_BASE*\admin*db_name*\pfile*sid* directory on Windows. If you do not specify a database name, then the database uses the value of the DB_NAME parameter specified in the initialization parameter file.

> **See Also:** *Oracle Database Administrator's Guide* for further information about starting the database

Task 3: Start Oracle Connection Manager

If Oracle Connection Manager is installed and configured, start it with the Oracle Connection Manager Control utility (CMCTL), entering commands in the following order:

1. From the command line, enter:

```
CMCTL
CMCTL> ADMINISTER [instance_name]
```

instance_name is the name of the Oracle Connection Manager that you would like to administer. You can determine the name by viewing cman.ora, the Oracle Connection Manager configuration file. The file can be found at the following location on the Oracle Connection Manager computer:

- UNIX:

```
$ORACLE_HOME/network/admin
```

- Windows:

```
ORACLE_HOME\network\admin
```

Oracle Connection Manager displays a status message indicating the name of the instance and informing you that the instance has not yet been started.

Note: If you do not provide an instance name as an argument, the Oracle Connection Manager with a fully qualified host name is administered. This is the default. After you issue the ADMINISTER command, CMCTL displays the instance name this way:

CMAN_*fully_qualified_host_name*

---

2. Start the Oracle Connection Manager that you have chosen to administer:

```
CMCTL> STARTUP
```

Oracle Connection Manager indicates that the instance has been started. In addition, it provides a status report for the instance.

3. Exit from the Oracle Connection Manager Control utility. Enter:

```
CMCTL> EXIT
```

On Windows, Oracle Connection Manager can also be started through the Control Panel:

4. Select the **Services** icon in the Control Panel window.

5. Select the `Oracle`*`HOME_NAME`*`CMan` service to start Oracle Connection Manager, and then click **Start**.
6. In the Services window, click **Close**.

Entering a Connect String

After the network components are started, as described in "Connecting to a Database" on page 15-1, you should be able to make a connection across the network. How you make a connection depends upon the **naming method** you configured in Chapter 8, "Configuring Naming Methods", and the tool used for the connection.

The **connect string** takes the following basic form:

```
CONNECT username@connect_identifier
```

On most operating systems, you can define a default **connect identifier**. This way, a connect identifier does not need to be specified in the connect string. To define a default connect identifier, use the `TWO_TASK` environment variable on UNIX platforms or the `LOCAL` environment variable or registry entry on Windows.

For example, if the `TWO_TASK` environment variable is set to `sales`, you can connect to a database from SQL*Plus with `CONNECT` *`username`* rather than `CONNECT` *`username`*`@sales`. Oracle Net checks if `TWO_TASK` is set and uses the value `sales` as the connect identifier. If it exists, Oracle Net connects.

> **See Also:** Oracle operating system-specific documentation for instructions on setting `TWO_TASK` and `LOCAL`

Further information about connect string format is provided in the following topics:

- Connect Identifier and Connect Descriptor Syntax Characteristics
- Absolute Name Specification for Directory Naming

Connect Identifier and Connect Descriptor Syntax Characteristics

Connect identifiers used in a connect string cannot contain spaces, unless enclosed within single quotes (') or double quotes ("). In the following examples, a connect identifier and a connect descriptor that contain spaces are enclosed within single quotes:

```
CONNECT scott@'(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)
(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)))'
Enter password: password

CONNECT scott'cn=sales, cn=OracleContext, dc=us, dc=acme, dc=com'
Enter password: password
```

Single quotes (') are required if a double quote (") is used in a connect identifier. For example:

```
CONNECT scott@'sales@Good"Fast"Food.com'
Enter password: password
```

Likewise, double quotes (") are required if a single quote (') is used in a connect identifier. For example:

```
CONNECT scott@"cn=sales, cn=OracleContext, ou=Mary's Dept, o=acme"
Enter password: password
```

Absolute Name Specification for Directory Naming

> **Note:** **JDBC OCI Driver**s support absolute naming. **JDBC Thin Driver**s support absolute naming only when the complete DN is used. See the *Oracle Database JDBC Developer's Guide and Reference* for further information.

This section describes how to configure absolute names for the following namespaces:

- Absolute Names for X.500 Namespaces
- Absolute Names for Domain Component Namespaces

Absolute Names for X.500 Namespaces

For X.500 namespaces, the default directory entry defined for the client must be in one of the following formats:

```
[ou],o
```

```
[ou],o,c
```

where `[ou]` represents an optional organizationalUnitName.

The absolute name the client uses as the connect identifier must be in one of the following formats:

```
cn[.ou].o
```

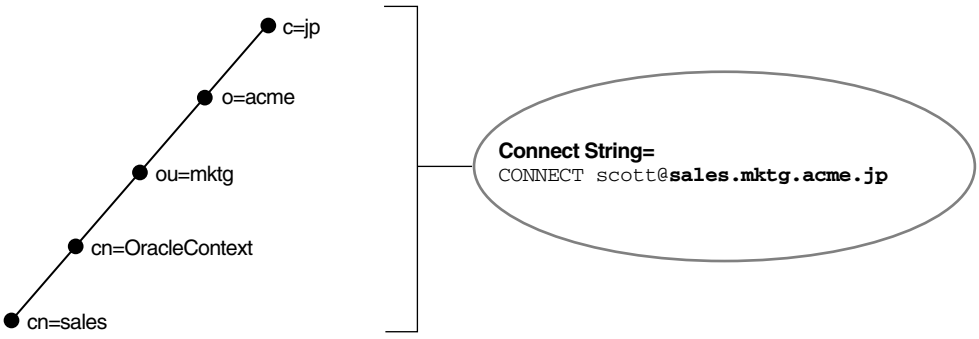
```
cn[.ou].o.c
```

where `[cn]` represents the Oracle Net entry.

For example, consider a client that has been configured with a default Oracle Context of `cn=OracleContext,ou=acctg,o=acme,c=us`.

The directory contains database object `sales` with a DN of `cn=sales,cn=OracleContext,ou=mktg,o=acme,c=jp`. In this scenario, the client requires a connect identifier of `sales.mktg.acme.jp` (`cn.ou.o.c`). Figure 15–1 depicts this example.

Figure 15–1 Absolute Name for X.500 Namespaces



Absolute Names for Domain Component Namespaces

For domain component namespaces, the default directory entry defined for the client must be in one of the following formats:

```
dc[,dc][...]
```

```
ou,dc[,dc][...]
```

where `[dc]` represents an optional domain component and `[...]` represents additional domain component entries.

The absolute name the client must use in the connect identifier must be in one of the following formats:

```
cn.dc[.dc][...]
```

```
cn[.ou]@dc[.dc][...]
```

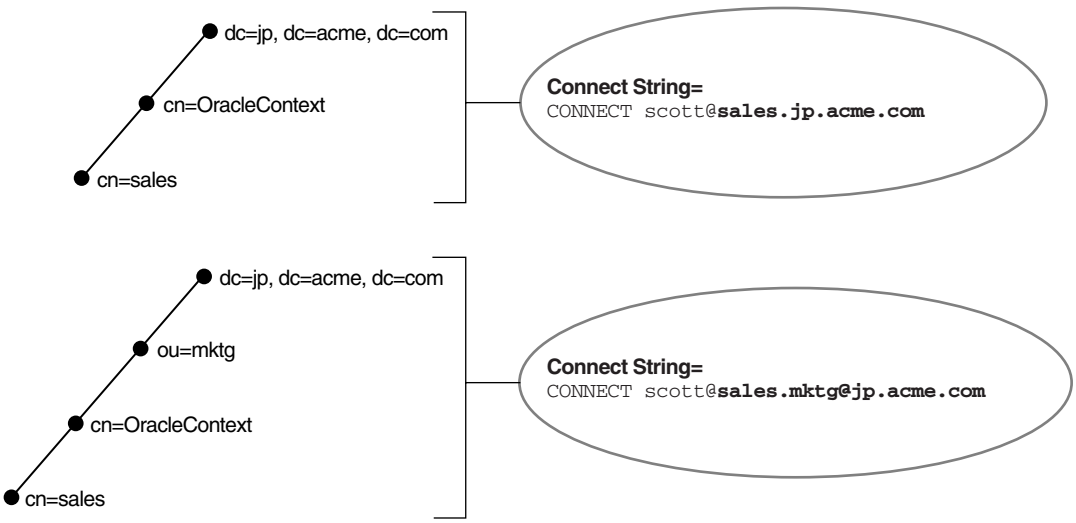
where `[cn]` represents the Oracle Net entry.

Example 1 Consider a client that has been configured with a default Oracle Context of `cn=OracleContext,dc=us,dc=acme,dc=com`.

The directory server contains an entry for database object `sales` with a DN of `cn=sales,cn=OracleContext,dc=jp,dc=acme,dc=com`. In this scenario, the client requires a connect identifier of `sales.jp.acme.com` (`cn.dc.dc.dc`). Figure 15–2 depicts this example.

Example 2 Consider the same default directory entry as Example 1. The directory server contains database object `sales` with a DN of `cn=sales,cn=OracleContext,ou=mktg,dc=jp,dc=acme,dc=com`. Notice `ou=mktg`. Because domain components must be separated from organization units, the client requires a connect identifier of `sales.mktg@jp.acme.com` (`cn.ou@dc.dc.dc`). Figure 15–2 depicts this example.

Figure 15–2 Absolute Name for Domain Component Namespaces



Initiating Connections

There are a number of ways to initiate a connection to an Oracle server. Commonly used methods are described in these topics:

- Connecting from the Operating System to Test a Client
- Connecting from the Tool Logon Screen to Test a Client
- Connecting from 3GL to Test a Client
- Connecting Using Special Commands Within Tools

The specifics of use are slightly different in each case. Each of the general methods listed is briefly covered here. To identify the method used in a specific tool, refer to the user guide for the tool.

Connecting from the Operating System to Test a Client

The general form of connecting an application to a database server from the command line is:

```
tool username@connect_identifier
Enter password: password
```

You will be prompted to enter your password which will be encrypted.

For example:

```
SQLPLUS system@sales
Enter password: password
```

Most Oracle tools can use the operating system command line to connect; some provide alternatives.

Connecting from the Tool Logon Screen to Test a Client

Some tools provide a logon screen as an alternative form of logon. A user can log on to a database server by identifying both the username and connect identifier (*username*@*connect_identifier*) in the username field of the tool logon screen, and entering the password as usual in the password field.

Connecting from 3GL to Test a Client

In applications written using 3GL, the program must establish a connection to a server using the following syntax:

```
exec sql connect :username identified by :password
```

In this connection request, :*username* and :*password* are 3GL variables that can be set within the program either statically or by prompting the user. When connecting to a database server, the value of the :*username* variable is in the form:

```
username@net_service_name
```

The :*password* variable contains the password for the database account to which you are connecting.

Connecting Using Special Commands Within Tools

Some Oracle tools have commands for database connections, once the tool has been started, to allow an alternative username to be specified without leaving the tool. SQL*Plus allows the CONNECT command using the following syntax:

```
SQL> CONNECT username@net_service_name
Enter password: password
```

For example:

```
SQL> CONNECT scott@serverx
Enter password: password
```

This is very similar to the operating system command line method, except that it is entered in response to the tool prompt instead of the operating system prompt.

Other Oracle tools use slightly different methods specific to their function or interface. For example, Oracle CDE tools use logon buttons and a pop-up window with the username, password, and remote database ID field.

Testing the Network

The preferred sequence for testing the network is as follows:

1. Start and test each listener.
2. Start and test each Oracle Connection Manager (if included in your network layout).
3. Test the server with a loopback test.
4. Test client with a connection.

This section contains these topics:

- Testing a Listener
- Testing Oracle Connection Manager
- Testing Configuration on the Database Server
- Testing Network Connectivity from the Client

Testing a Listener

To test a listener, initiate a connection from a client to any active database controlled by that listener, as described in "Testing Configuration on the Database Server" on page 15-8.

Testing Oracle Connection Manager

To test Oracle Connection Manager, initiate a connection from a client to any active database that has been registered with Oracle Connection Manager.

Testing Configuration on the Database Server

Once you have configured the network, test the configuration by performing a **loopback test** on the database server.

A loopback test uses Oracle Net to go from the database server back to itself, bypassing the Interprocess Communication (IPC). Performing a successful loopback verifies that Oracle Net is functioning on the database server.

To perform the loopback test, use Oracle Enterprise Manager or Net8 Assistant.

Oracle Net Manager

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator, expand **Directory** or **Local** > **Service Naming**.
3. Select the net service name or database service.
4. Choose **Command** > **Test Net Service**.

   Testing assumes the database and listener are running. If they are not, see "Starting Oracle Net Services Components" on page 15-1 to start components.

During testing, a Connection Test dialog box appears, providing status and test results. A successful test results in the following message:

The connection test was successful.

If the test was successful, proceed to Step 5.

If the test was not successful:

a. Ensure that the database and listener are running, and then click **Test**.

b. Click **Change Login** to change the username and password for the connection, and then click **Test**.

5. Click **Close** to dismiss the Connect Test dialog box.

Testing Network Connectivity from the Client

To test several different clients in your network, initiate a connection to a database server from each of them by following the instructions in "Entering a Connect String" on page 15-4.

Oracle Net also provides the following tools to help evaluate network connectivity:

- TNSPING Utility
- TRCROUTE Utility
- Oracle Net Manager

TNSPING Utility

The TNSPING utility determines whether or not a service (for example, an Oracle database or any other Oracle service) on an Oracle Net network can be successfully reached.

If you can connect successfully from a client to a server (or a server to another server) using the TNSPING utility, it displays an estimate of the round trip time (in milliseconds) it takes to reach the Oracle Net service.

If it fails, it displays a message describing the error that occurred. This enables you to see the network error that is occurring without the overhead of a database connection.

Using TNSPING To invoke the TNSPING utility, enter the following:

```
tnsping net_service_name count
```

> **Note:** Different platforms may have different interfaces, but the program accepts the same arguments. Invoke TNSPING for the display of the proper interface requirements.

- *net_service_name*: must exist in `tnsnames.ora` file or the name service in use, such as NIS or DCE's CDS.
- *count* (optional): determines how many times the program attempts to reach the server.

If the net service name specified is a database name, TNSPING attempts to contact the corresponding listener. It does not actually determine whether or not the database itself is running. Use SQL*Plus to attempt a connection to the database.

Following are some examples of TNSPING.

Example: Reaching a Database with TNSPING To connect to a database using a net service name of `sales`, the following is entered:

```
TNSPING sales
```

This produces the following message:

```
TNS Ping Utility for Solaris: Version 10.1.0.2.0 on 15-NOV-2003 14:46:28

Copyright (c) 1997 Oracle Corporation.  All rights reserved.

Used parameter files:
Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL =
TCP)(HOST = sales-server)(PORT = 1521))) (CONNECT_DATA = (SERVICE_NAME =
sales.us.acme.com)))
OK (10 msec)
```

To determine whether a connection can be made to the `sales` database, and to specify that TNSPING try to connect eight times and then give up, use the following syntax:

```
tnsping sales 8
```

This command produces the following message:

```
TNS Ping Utility for Solaris: Version 10.1.0.2.0 on 15-NOV-2003 14:49:26

Copyright (c) 1997 Oracle Corporation.  All rights reserved.

Used parameter files:

Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL =
TCP)(HOST = sales-server)(PORT = 1521))) (CONNECT_DATA = (SERVICE_NAME =
sales.us.acme.com)))
OK (10 msec)
OK (0 msec)
OK (10 msec)
OK (0 msec)
OK (10 msec)
OK (10 msec)
OK (10 msec)
OK (0 msec)
```

Example: Invalid Net Service Name with TNSPING Below is an example of TNSPING attempting to connect to an invalid net service name:

```
tnsping badname
```

This attempt produces the following message:

```
TNS Ping Utility for Solaris: Version 10.1.0.2.0 on 15-NOV-2003 14:51:12

Copyright (c) 1997 Oracle Corporation.  All rights reserved.

Used parameter files:

TNS-03505: Failed to resolve name
```

Example: Valid Net Service Name with TNSPING Following is an example of using TNSPING to connect to a name that is valid, but that resolves to an address where no listener is located (for example, the listener may not be started):

```
TNS Ping Utility for Solaris: Version 10.1.0.2.0 on 15-NOV-2003 14:46:28

Copyright (c) 1997 Oracle Corporation.  All rights reserved.

Used parameter files:
Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL =
TCP)(HOST = sales-server)(PORT = 1521))) (CONNECT_DATA = (SERVICE_NAME =
sales.us.acme.com)))
TNS-12541: TNS:no listener
```

TRCROUTE Utility

The Trace Route Utility (TRCROUTE) enables administrators to discover the path or route a connection is taking from a client to a server. If TRCROUTE encounters a problem, it returns an error stack to the client instead of a single error. These additional error messages make troubleshooting easier.

TRCROUTE is different from TNSPING in that it travels as a special type of connect packet, and is routed as such. As it travels toward its destination, the TRCROUTE connect packet collects the TNS addresses of every node it travels through. If an error occurs, TRCROUTE collects error information that shows where the error occurred. The TRCROUTE displays the information collected on the client screen. You can redirect the TRCROUTE output to a file, and print it if you wish.

The TRCROUTE uses minimal resources. It gathers information in the connect data of a special connect packet; standard connect packets are not affected.

The server is not affected by TRCROUTE. The listener receives and processes the TRCROUTE connect packet. It returns the information to the client by putting it into a refuse packet. The server does not need to start up any new processes or deal with dummy connections.

Using TRCROUTE To invoke TRCROUTE, enter the following from the command line:

```
trcroute net_service_name
```

The following are two examples of trace route output.

Example: Successful Trace Route

The following example shows a successful Trace Route packet that traveled from a client to a listener.

```
trcroute sales
Trace Route Utility for Solaris: Version 10.1.0.2.0 on 15-NOV-2003 14:35:05

Copyright (c) 1999 Oracle Corporation.  All rights reserved.

Route of TrcRoute:
------------------

Node: Client            Time and address of entry into node:
-------------------------------------------------------------
25-JAN-2002 14:35:05 ADDRESS= PROTOCOL=TCP  HOST=sales-server  PORT=1521

Node: Server            Time and address of entry into node:
-------------------------------------------------------------
25-JAN-2002 14:35:06
```

Example: Trace Route with Error

The following examples shows an unsuccessful Trace Route packet that could not reach the listener because the listener was not up.

```
trcroute sales
Trace Route Utility for Solaris: Version 10.1.0.2.0 on 15-NOV-2003 14:43:05

Copyright (c) 1999 Oracle Corporation.  All rights reserved.

Route of TrcRoute:
------------------

Node: Client            Time and address of entry into node:
------------------------------------------------------------
25-FEB-2002 14:43:05 ADDRESS= PROTOCOL=TCP  HOST=sales-server  PORT=1521

TNS-12543: TNS:unable to connect to destination
TNS-12541: TNS:no listener
TNS-12560: TNS:protocol adapter error
TNS-03601: Failed in route information collection
```

Oracle Net Manager

To verify connectivity for a client computer, use Net8 Assistant:

1. Start Oracle Net Manager.

   See Also: "Oracle Net Manager" on page 6-2

2. In the navigator, expand **Directory** or **Local** > **Service Naming.**
3. Select the net service name or database service.
4. Choose **Command** > **Test Net Service**.

   Testing assumes that the database and listener are running. If they are not, see "Starting Oracle Net Services Components" on page 15-1 to start components.

   During testing, a Connection Test dialog box appears, providing status and test results. A successful test results in the following message:

   The connection test was successful.

   If the test was successful, proceed to Step 5.

   If the test was not successful:

   a. Ensure that the database and listener are running, and then click **Test**.
   b. Click **Change Login** to change the username and password for the connection, and then click **Test**.

5. Click **Close** to dismiss the Connect Test dialog box.

16

Troubleshooting Oracle Net Services

Oracle Net Services provides methods for understanding and resolving network problems through the use of log and trace files. These files keep track of the interaction between network components as errors occur. Evaluating this information will help you to diagnose and troubleshoot even the most complex network problems.

This chapter describes common network errors and outlines procedures for resolving them. It also describes methods for logging and tracing error information to diagnose and troubleshoot more complex network problems. This chapter contains these topics:

- Diagnosing Oracle Net Services
- Resolving the Most Common Error Messages for Oracle Net Services
- Troubleshooting Tips from the Field for Oracle Net Services
- Troubleshooting the TNS-12154 Error
- Troubleshooting Network Problems Using Log and Trace Files
- Logging Error Information for Oracle Net Services
- Tracing Error Information for Oracle Net Services
- Contacting Oracle Support Services

Diagnosing Oracle Net Services

If an attempt to make a basic peer-to-peer (single protocol network) connection returns an `ORA` *`Error`*, this section may help you diagnose the cause of the problem.

Any underlying fault, noticeable or not, is reported by Oracle Net Services with an error number or message that is not always indicative of the actual problem. This section helps you determine which parts of Net8 Services do function properly rather than the parts that do not work. It also helps you to decide in which of the following categories the fault belongs:

- Oracle software
- Operating system layer
- Other network layers

Testing the various network layers progressively should, in most cases, uncover any problem.

This section includes the following topics:

- Automatic Diagnostic Repository
- ADRCI: ADR Command Interpreter

- Server Diagnostics
- Client Diagnostics

Automatic Diagnostic Repository

In an effort to reduce both downtime and interruptions in business when database problems occur, Oracle has implemented a standardized diagnostic method employed across all Oracle products.

Part of this method consolidates Oracle Net diagnostics and tracing information into a standardized, readable format. This information is stored in a single hierarchical repository. Oracle Net diagnostics data consists of tracing and logging information produced by Oracle clients, application servers, and the database server.

The **automatic diagnostic repository** (ADR) is a system-wide tracing and logging central repository. The repository is a file-based hierarchical datastore for depositing diagnostic information, including network tracing and logging information.

The ADR Home is the unit of the ADR directory that is assigned to an instance of an Oracle product. Each database instance will have its own ADR Home. Similarly, each listener, connection manager (CMAN), and client instance will have its own ADR Home.

The location of an ADR Home is given by the following path, which starts at the ADR base directory:

`diag/`*`product_type`*`/`*`product_id`*`/`*`instance_id`*

Table 16–1 lists the values of the various path components for an Oracle Net Listener instance.

Table 16–1 ADR Home Path Components for an Oracle Net Listener Instance

Path Component	Value for Oracle Net Listener
product_type	tnslsnr
product_id	host name
instance_id	listener alias name

Figure 16–1 illustrates the directory hierarchy of the ADR for an Oracle Net Listener instance. Other ADR homes for other Oracle products or components (such as ASM or Oracle Database) can exist within this hierarchy, under the same ADR base.

Figure 16–1 Directory Structure for an Oracle Net Listener Instance

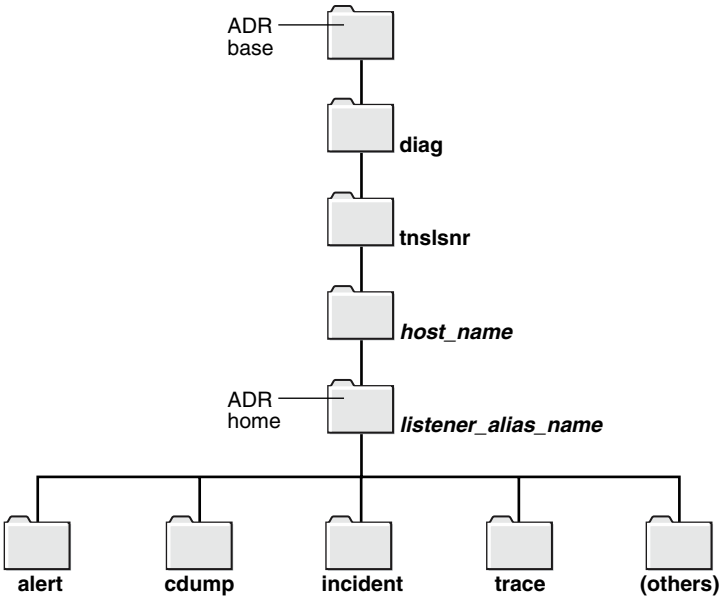


Table 16–2 lists the values of the various path components for a CMAN instance.

Table 16–2 ADR Home Path Components for a CMAN Instance

Path Component	Value for Oracle Net Listener
product_type	netcman
product_id	host name
instance_id	CMAN instance name

Within the ADR Home directory are subdirectories where each instance (be it database, listener, CMAN, or client) stores diagnostic data. Table 16–3 lists some of these subdirectories and their contents.

Figure 16–2 illustrates the directory hierarchy of the ADR for a CMAN instance. Other ADR homes for other Oracle products or components (such as ASM or Oracle Database) can exist within this hierarchy, under the same ADR base.

Figure 16–2 Directory Structure for a CMAN Instance

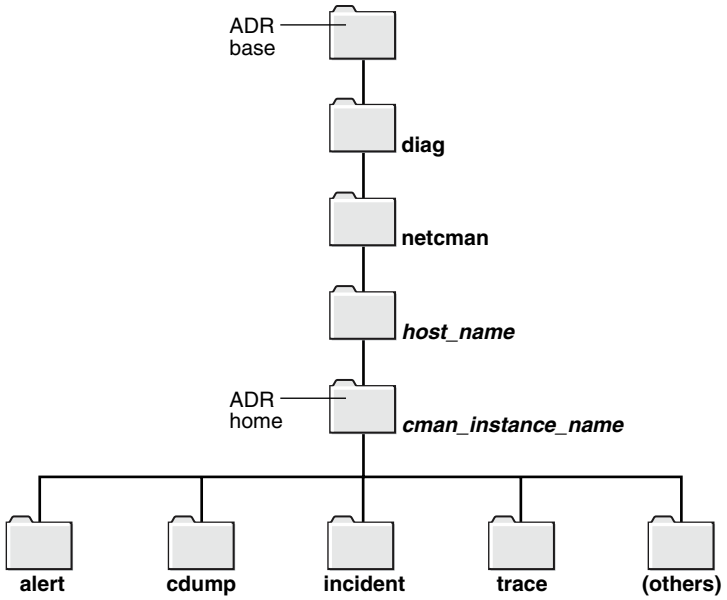


Table 16–3 ADR Home Subdirectories

Subdirectory Name	Contents
`alert`	The XML-formatted alert log
`cdump`	Core files
`incident`	Multiple subdirectories, where each subdirectory is named for a particular incident, and where each contains dumps pertaining only to that incident
`trace`	Background and server process trace files and SQL trace files
(others)	Other subdirectories of ADR Home, which store incident packages, health monitor reports, and other information

> **See Also:** *Oracle Call Interface Programmer's Guide* for information about the location of the client ADR Home

The `ADR_BASE` is the physical location in which one or more ADR Homes are placed. Conceptually, it is the root directory of ADR.

> **See Also:** *Oracle Database Administrator's Guide* for more information about ADR

Non-ADR (meaning that the `DIAG_ADR_ENABLED` parameter is set to `OFF`) diagnostic and tracing methods are still current and applicable but the parameters are ignored if ADR is enabled.

Diagnostic parameters are found in the three configuration files: `sqlnet.ora` for clients, `listener.ora` for listeners, and `cman.ora` for connection managers.

> **See Also:** *Oracle Database Net Services Reference* for descriptions of the following diagnostic parameters

Table 16–4 compares usage of various diagnostic parameters found in the `sqlnet.ora` file used in both non-ADR and ADR-based diagnostics.

Table 16–4 sqlnet.ora File Diagnostic Parameter Comparison

Parameter	Non-ADR DIAG_ADR_ ENABLED=OFF	ADR DIAG_ADR_ ENABLED=ON
ADR_BASE[1]	No	Yes
TRACE_LEVEL_CLIENT[2]	Yes	Yes
TRACE_LEVEL_SERVER[2]	Yes	Yes
TRACE_DIRECTORY_CLIENT[3]	Yes	No
TRACE_FILE_CLIENT[3]	Yes	No
TRACE_UNIQUE_CLIENT[3]	Yes	No
LOG_DIRECTORY_CLIENT[3]	Yes	No
LOG_FILE_CLIENT[3]	Yes	No
LOG_DIRECTORY_SERVER[3]	Yes	No
TRACE_DIRECTORY_SERVER[3]	Yes	No
TRACE_FILE_SERVER[3]	Yes	No

1 ADR only parameter. This parameter is not used for non-ADR-based tracing/logging.

2 These parameters are functionally equivalent for both non-ADR and ADR-based tracing and logging.

3 If DIAG_ADR_ENABLED is set to ON, this parameter is ignored. The trace and log files are created in the location defined by ADR_BASE (See *Oracle Database Administrator's Guide* for description of ADR directory tree and log/trace file naming conventions).

Table 16–5 compares usage of various diagnostic parameters found in the `listener.ora` file used in both non-ADR and ADR-based diagnostics.

Table 16–5 listener.ora File Diagnostic Parameter Comparison

Parameter	Non-ADR DIAG_ADR_ ENABLED=OFF	ADR DIAG_ADR_ ENABLED=ON
ADR_BASE_*listener_name*[1]	No	Yes
LOGGING_*listener_name*[2]	Yes	Yes
TRACE_LEVEL_*listener_name*[2]	Yes	Yes
TRACE_TIMESTAMP_*listener_name*[2]	Yes	Yes
LOG_DIRECTORY_CLIENT_*listener_name*[3]	Yes	No
LOG_FILE_CLIENT_*listener_name*[3]	Yes	No
TRACE_DIRECTORY_CLIENT_*listener_name*[3]	Yes	No
TRACE_FILELEN_*listener_name*[3]	Yes	No
TRACE_FILENO_*listener_name*[3]	Yes	No

1 ADR only parameter. This parameter is not used for non-ADR-based tracing/logging.

2 These parameters are functionally equivalent for both non-ADR and ADR-based tracing and logging.

3 If DIAG_ADR_ENABLED_*listener_name* is set to ON, this parameter is ignored. The trace and log files are created in the location defined by ADR_BASE_*listener_name* (See *Oracle Database Administrator's Guide* for description of ADR directory tree and log/trace file naming conventions).

Table 16–6 compares usage of various diagnostic parameters found in the `cman.ora` file used in both non-ADR and ADR-based diagnostics.

Table 16–6 cman.ora File Diagnostic Parameter Comparison

Parameter	Non-ADR DIAG_ADR_ ENABLED=OFF	ADR DIAG_ADR_ ENABLED=ON
ADR_BASE[1]	No	Yes
LOG_LEVEL[2]	Yes	Yes
TRACE_LEVEL[2]	Yes	Yes
TRACE_TIMESTAMP[2]	Yes	Yes
LOG_DIRECTORY[3]	Yes	No
TRACE_DIRECTORY[3]	Yes	No
TRACE_FILELEN[3]	Yes	No
TRACE_FILENO[3]	Yes	No

1 ADR only parameter. This parameter is not used for non-ADR-based tracing/logging.

2 These parameters are functionally equivalent for both non-ADR and ADR-based tracing and logging.

3 If DIAG_ADR_ENABLED is set to ON, this parameter is ignored. The trace and log files are created in the location defined by ADR_BASE (See *Oracle Database Administrator's Guide* for description of ADR directory tree and log/trace file naming conventions).

Trace Assistant

Trace Assistant is a diagnostic tool that completely decodes each packet of Oracle Net tracing data and presents it in a readable and understandable format. Trace Assistant also provides useful statistical information.

ADRCI: ADR Command Interpreter

ADRCI is a command-line tool that is part of the fault diagnosability infrastructure introduced in Oracle Database 11*g*. ADRCI enables you to:

- View diagnostic data within ADR
- Package incident and problem information into a zip file for transmission to Oracle Support

Diagnostic data includes incident and problem descriptions, trace files, dumps, health monitor reports, alert log entries, and more.

ADRCI has a rich command set, and can be used in interactive mode or within scripts. In addition, ADRCI can execute scripts of ADRCI commands in the same way that SQL*Plus executes scripts of SQL and PL/SQL commands.

To view trace files using ADRCI, enter ADRCI at a command prompt. Following are common commands used for Oracle Net trace diagnosis:

Client Side

```
adrci>> SHOW BASE -product client
adrci>> SET BASE -product client
adrci>> SHOW TRACEFILE
adrci>> SHOW TRACE trace_file.trc
```

In the preceding command, SHOW BASE -product client displays the value of ADR_BASE for the client. Use that value for *client* in the SET BASE command.

Server Side

```
adrci>> SHOW BASE
adrci>> SHOW TRACEFILE
adrci>> SHOW TRACE trace_file.trc
```

In the preceding command, `BASE` is defined as `$ORACLE_HOME/log`.

Other ADRCI command options are available for a more targeted Oracle Net trace file analysis. Type `HELP` at the `adrci>>` prompt for in-line help documentation.

> **See Also:** *Oracle Database Utilities* for more information about ADRCI

Server Diagnostics

Answer the following questions:

- Is any other system (workstation/server) able to connect to the server using Net8?
- Has the server, database, or listener configuration remained the same for some time?

If you answered YES to any of the preceding questions/statements, then skip this section and continue to "Client Diagnostics" on page 16-8.

If you are unsure, or answered NO to any of the preceding questions, then continue.

Diagnosing Net8 Services on the server involves the following tasks:

- Task 1: Verify the Database Is Running
- Task 2: Perform a Loopback Test

Task 1: Verify the Database Is Running

To check that the database is up, login to the database and connect with a valid username and password. For example:

```
SQLPLUS system
Enter password: password
```

A message appears, confirming that you are connected with the database. If you receive the following errors, ask your Database Administrator to assist you:

- `ORA-1017: invalid U/P`
- `ORA-1034: Oracle not available`

Task 2: Perform a Loopback Test

To perform a **loopback test** from the server to the database:

1. Ensure that the `listener.ora`, `tnsnames.ora`, and `sqlnet.ora` files exist in the correct locations, as described in "Localized Configuration File Support" on page 4-1.
2. Follow the instructions in "Testing Configuration on the Database Server" on page 15-8 to perform a loopback test.
   - If the loopback test continues to fail, continue to the next step.
   - If the loopback test passes, skip to "Client Diagnostics".
3. Contact Oracle Worldwide Support.

Client Diagnostics

> **See Also:** *Oracle Call Interface Programmer's Guide* for the location of the ADR diagnostic files

At this point, you know the server-side listener works properly, because you could verify at least one of the following statements:

- The database server passed a loopback test, showing that the connection worked.
- Other computers connect also using Net8 Services to this same database.
- Connections from this workstation worked previous to making changes on this computer, such as the installation of a new product or a modification to the network configuration.

To perform diagnostics on the client:

1. Check that you have installed the same protocol support as was installed on the database server.

   On UNIX you can use the ADAPTERS utility to verify protocol support. On the database server, run the `adapters 'which oracle'` command from `$ORACLE_HOME/bin` to display the protocol support, naming methods, and security options linked with the `oracle` executable. The `adapters` utility displays output similar to the following:

```
Oracle Net transport protocols linked with ./oracle are:

    IPC
    BEQ
    TCP/IP
    SSL
    RAW

Oracle Net naming methods linked with ./oracle are:

    Local Naming (tnsnames.ora)
    Oracle Directory Naming
    Oracle Host Naming
    NIS Naming

Oracle Advanced Security options linked with ./oracle are:

    RC4 40-bit encryption
    RC4 56-bit encryption
    RC4 128-bit encryption
    RC4 256-bit encryption
    DES40 40-bit encryption
    DES 56-bit encryption
    3DES 112-bit encryption
    3DES 168-bit encryption
    AES 128-bit encryption
    AES 192-bit encryption
    AES 256-bit encryption
    MD5 crypto-checksumming
    SHA crypto-checksumming (for FIPS)
    SHA-1 crypto-checksumming
    Kerberos v5 authentication
    RADIUS authentication
    ENTRUST authentication
```

On the client, run the `adapters` command from `$ORACLE_HOME/bin` to display the configured Oracle protocol support, naming methods, and security options. The ADAPTERS utility displays output similar to the following:

```
Installed Oracle Net transport protocols are:

    IPC
    BEQ
    TCP/IP
    SSL
    RAW

Installed Oracle Net naming methods are:

    Local Naming (tnsnames.ora)
    Oracle Directory Naming
    Oracle Host Naming
    NIS Naming

Installed Oracle Advanced Security options are:

    RC4 40-bit encryption
    RC4 56-bit encryption
    RC4 128-bit encryption
    RC4 256-bit encryption
    DES40 40-bit encryption
    DES 56-bit encryption
    3DES 112-bit encryption
    3DES 168-bit encryption
    AES 128-bit encryption
    AES 192-bit encryption
    AES 256-bit encryption
    MD5 crypto-checksumming
    SHA-1 crypto-checksumming
    Kerberos v5 authentication
    RADIUS authentication
    ENTRUST authentication
```

Note: RAW is an internal protocol used by Oracle Net.

See Also: Oracle UNIX operating system-specific *Administrator's Reference* for further information about the `adapters` utility

2. Check base connectivity for underlying network transport. Net8 technology depends on the underlying network for a successful connection.

Protocol	Verify that you can...
TCP/IP	Use terminal emulation or file transfer utilities, (PING, FTP, TELNET) from the client to the database server.
Named Pipes	■ See other computers or servers on the Microsoft network. ■ Ensure that you are able to share drives within the network.

3. To ensure that both the Net8 foundation layer and the appropriate Oracle protocol support are present, verify that all Net8 Services software for the client has been installed.
4. Ensure that the client computer has the `tnsnames.ora` and the `sqlnet.ora` files exist in the correct locations.

   See Also: "Localized Configuration File Support" on page 4-1

   If you have any other working client computers connecting to the selected Oracle Database, back up your existing files and copy both the working `tnsnames.ora` and `sqlnet.ora` files from the working computer onto the non-working client workstations. This eliminates the possibility of errors in the files.
5. Test the Net8 foundation layer.

   See Also: "Testing Network Connectivity from the Client" on page 15-9

   Note: Do *not* use the TNSPING utility. The TNSPING utility works like the TCP/IP PING utility and does *not* create and open a socket, nor does it connect with the listener. It ensures that the listener is present on the database server.

   ---
6. If the connection still fails:
   - Use tracing, as described in section "Troubleshooting Network Problems Using Log and Trace Files" on page 16-20
   - Check the **Problem/Solution Database** Web site on the Oracle Support web site for a specific diagnostics bulletin on the error received
   - Contact Oracle Support Services

Resolving the Most Common Error Messages for Oracle Net Services

Due to the complexity of network communications, network errors may originate from a variety of sources, for a variety of reasons. If an error occurs, applications such as SQL*Plus, that depend on network services from Oracle Net Services, will normally generate an error message.

A list of the most common network error messages follows:

- **ORA-03113: TNS:end-of-file on communication channel**
- **ORA-03121: no interface driver connection - function not performed**
- **ORA-12154: TNS:could not resolve service name**
- **ORA-12170: TNS:Connect timeout occurred**
- **TNS-12500/ORA-12500: TNS: listener failed to start a dedicated server process**
- **ORA-12514: TNS:listener does not currently know of service requested in connectdescriptor**
- **ORA-12520: TNS:listener could not find available handler for requested type of server**

- **ORA-12521: TNS:listener could not resolve INSTANCE_NAME given in connect descriptor**
- **ORA-12525: TNS:listener has not received client's request in time allowed**
- **ORA-12533: TNS:illegal ADDRESS parameters**
- **TNS-12540/ORA-12540: TNS:internal limit restriction exceeded and TNS-00510: Internal limit restriction exceeded**
- **TNS-12541/ORA-12541: TNS:no listener**
- **TNS-12549/ORA-12549: TNS:operating system resource quota exceeded and TNS-00519: Operating system resource quota exceeded**
- **TNS-12560/ORA-12560: TNS:protocol adapter error occurred**

> **See Also:** *Oracle Database Error Messages* for a complete listing of error messages

ORA-03113: TNS:end-of-file on communication channel

Cause: An error has occurred on the database server.

Action: Check the alert_sid.log on the server. The location of alert_sid.log is specified by the BACKGROUND_DUMP_DEST initialization parameter. An unexpected end of file was processed on the communication channel. This may be an indication that the communications link may have gone down at least temporarily; it may indicate that the server has gone down.You may need to modify your retransmission count.

ORA-03121: no interface driver connection - function not performed

Cause: A SQL*Net version 1 prefix was erroneously used in the connect string.

Action: Do not use the following prefixes in the connect string.

- T:
- X:
- P:

The username and password were specified from a client computer that had no local Oracle Database installed. Specify a connect string.

ORA-12154: TNS:could not resolve service name

Cause: Oracle Net could not locate the net service name specified in the `tnsnames.ora` configuration file.

Action: Perform these steps:

1. Verify that a `tnsnames.ora` file exists.

   > **See Also:** "Localized Configuration File Support" on page 4-1 for configuration file location information

2. Verify that there are not multiple copies of the `tnsnames.ora` file.
3. In the `tnsnames.ora` file, verify that the net service name specified in your connect string is mapped to a connect descriptor.
4. Verify that there are no duplicate copies of the `sqlnet.ora` file.

5. If you are using domain names, verify that your `sqlnet.ora` file contains a `NAMES.DEFAULT_DOMAIN` parameter. If this parameter does not exist, you must specify the domain name in your connect string.
6. If you are not using domain names, and this parameter exists, delete it or disable it by commenting it out.
7. If you are connecting from a login dialog box, verify that you are not placing an "@" symbol before your connect net service name.
8. Activate client tracing and repeat the operation.

Cause: Oracle Net could not locate the database service name or net service name specified in the directory server.

Action: Perform these steps:

1. Verify that the database service or net service name entry exists in the directory that this computer was configured to use.

   See Also: *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory* for directory setup instructions

2. Verify that the `sqlnet.ora` file includes the following entry:

   ```
   NAMES.DIRECTORY_PATH=(ldap, other_naming_methods)
   ```

ORA-12170: TNS:Connect timeout occurred

Cause: The client failed to establish a connection and complete authentication in the time specified by the `SQLNET.INBOUND_CONNECT_TIMEOUT` parameter in the `sqlnet.ora` file. This error may be a result of network or system delays, or it may indicate that a malicious client is trying to cause a denial-of-service attack on the database server.

See Also: "Configuring the Listener and the Oracle Database To Limit Resource Consumption By Unauthorized Users" on page 14-8 further information about setting the `SQLNET.INBOUND_CONNECT_TIMEOUT` parameter

Action: If the error occurred due to system or network delays that are normal for the particular environment, then perform these steps:

1. Turn on tracing to determine where clients are timing out.

   See Also: "Tracing Error Information for Oracle Net Services" on page 16-32

2. Reconfigure the `SQLNET.INBOUND_CONNECT_TIMEOUT` parameter in `sqlnet.ora` to a larger value.

If you suspect a malicious client, then perform these steps:

1. Locate the IP address of the client in the `sqlnet.log` file on the database server to identify the source.

   For example, the following `sqlnet.log` excerpt shows a client IP address of `10.10.150.35`.

   ```
   Fatal NI connect error 12170.

     VERSION INFORMATION:
   TNS for Solaris: Version 10.1.0.2.0
   ```

```
Oracle Bequeath NT Protocol Adapter for Solaris: Version 10.1.0.2.0
TCP/IP NT Protocol Adapter for Solaris: Version 10.1.0.2.0
  Time: 03-JUL-2002 13:51:12
  Tracing to file: /ora/trace/svr_13279.trc
  Tns error struct:
    nr err code: 0
    ns main err code: 12637
    TNS-12637: Packet receive failed
    ns secondary err code: 12604
    nt main err code: 0
    nt secondary err code: 0
    nt OS err code: 0
  Client address: (ADDRESS=(PROTOCOL=tcp)(HOST=10.10.150.35)(PORT=52996))
```

Beware that an IP address can be forged.

If the time out occurs before the IP address can be retrieved by the database server, then enable listener tracing to determine the client that made the request.

See Also: Tracing Error Information for Oracle Net Services on page 16-32

2. Restrict access to the client. For example, you can configure parameters for access rights in the `sqlnet.ora` file.

   See Also: "Configuring Database Access Control" on page 9-4

TNS-12500/ORA-12500: TNS: listener failed to start a dedicated server process

Cause: The listener failed to start the Oracle program. Possible reasons include:

- The maximum number of processes allowed for a single user was exceeded
- The listener does not have execute permission on the Oracle program
- The associated Windows service is not started

In some cases, these errors can be caused by the same conditions which cause TNS-12549/ORA-12549, TNS-00519, TNS-12540/ORA-12540, TNS-00510, and TNS-12560/ORA-12560 errors.

Action: Perform the appropriate action:

- Increase the number of processes by setting the PROCESSES parameter in the database initialization file to a larger value.
- Check the listener.log file for detailed error stack information.

ORA-12514: TNS:listener does not currently know of service requested in connectdescriptor

Cause: The listener received a request to establish a connection to a database or other service. The connect descriptor received by the listener specified a service name for a service (usually a database service) that has either not yet dynamically registered with the listener or has not been statically configured for the listener. This may be a temporary condition such as after the listener has started, but before the database instance has registered with the listener.

Action: Perform these steps:

1. Wait a moment and try to connect a second time.

2. Check which services are currently known by the listener by executing the Listener Control utility `STATUS` or `SERVICES` command.

   See Also: "Determining the Current Status of a Listener" on page 10-15 and "Monitoring Services of a Listener" on page 10-17

3. Check that the `SERVICE_NAME` parameter in the connect descriptor specifies a service name known by the listener.
4. Check for an event in the `listener.log` file.

   See Also: "Analyzing Listener Log Files" on page 16-26

ORA-12520: TNS:listener could not find available handler for requested type of server

Cause: The type of service handler requested by the client is incorrect or not registered for the requested `SERVICE_NAME/INSTANCE_NAME`, or the database instance is not registered with the listener.

Action: If you suspect the problem is the wrong type of service handler, perform these steps:

1. If `(server=`*`value`*`)` is set is in the connect descriptor, ensure that the value is set to the appropriate service handler type for the database, that is, `dedicated` for dedicated server or `shared` for dispatchers. You can use the Listener Control utility `SERVICES` command to see what service handlers are currently registered with the listener.

   See Also: "Monitoring Services of a Listener" on page 10-17

2. If `USE_DEDICATED_SERVER` is set to `ON` in the `sqlnet.ora` file, then ensure the database is configured to use dedicated servers. If it is not, set this parameter to `off`.
3. Ensure that the database instance is running. If the instance not running, start it so that it can register with the listener.

ORA-12521: TNS:listener could not resolve INSTANCE_NAME given in connect descriptor

Cause: The `INSTANCE_NAME` in the connect descriptor is incorrect, or the database instance is not registered with the listener.

Action: Perform these steps:

1. Check to make sure the service name specified in the connect descriptor is correct.
2. Ensure the database instance is running. If the instance not running, start it so that it can register with the listener. You can use the Listener Control utility `SERVICES` command to see what instances are currently registered with the listener.

   See Also: "Monitoring Services of a Listener" on page 10-17

ORA-12525: TNS:listener has not received client's request in time allowed

Cause: The client failed to complete its connect request in the time specified by the `INBOUND_CONNECT_TIMEOUT_`*`listener_name`* parameter in the `listener.ora` file. This error may be a result of network or system delays, or it

may indicate that a malicious client is trying to cause a denial-of-service attack on the listener.

> **See Also:** "Configuring the Listener and the Oracle Database To Limit Resource Consumption By Unauthorized Users" on page 14-8 for further information about setting the `INBOUND_CONNECT_TIMEOUT_`*`listener_name`* parameter

Action: If the error occurred due to system or network delays that are normal for the particular environment, then reconfigure the `INBOUND_CONNECT_TIMEOUT_`*`listener_name`* parameter in `listener.ora` to a larger value.

If you suspect a malicious client, then perform these steps:

1. Locate the IP address of the client in `listener.log` to identify the source.

   For example, the following `listener.log` excerpt shows a client IP address of `10.10.150.35`.

   ```
   03-JUL-2002 16:42:35 * <unknown connect data> *
   (ADDRESS=(PROTOCOL=tcp)(HOST=10.10.150.35)(PORT=53208)) * establish *
   <unknown sid> * 12525
   TNS-12525: TNS:listener has not received client's request in time
   allowed
   TNS-12604: TNS: Application timeout occurred
   ```

   Beware that an IP address can be forged.

2. Restrict access to the client. For example, you can configure parameters for access rights in the `sqlnet.ora` file.

   > **See Also:** "Configuring Database Access Control" on page 9-4

ORA-12533: TNS:illegal ADDRESS parameters

Cause: The protocol specific parameters in the `ADDRESS` section of the designated connect descriptor are incorrect.

Action: Correct the protocol address.

> **See Also:** *Oracle Database Net Services Reference* for correct protocol syntax

TNS-12540/ORA-12540: TNS:internal limit restriction exceeded and TNS-00510: Internal limit restriction exceeded

Cause: An internal limit has been exceeded. Possible limits include:

- Number of open connection that Oracle Net can process simultaneously
- Number of memory buffers which can be used simultaneously
- Number of processes a particular database instance is allowed

The first two are examples of hard limits. The third is an example of a limit which can be increased by setting PROCESSES parameter in the database initialization file to a larger value. In this case, a TNS-12500/ORA-12500 error is also returned.

In some cases, these errors can be caused by the same conditions which cause TNS-12549/ORA-12549 and TNS-00519 errors.

Action: Perform these steps:

Wait for the open connections to close and retry. If the error persists, then check the sqlnet.log or listener.log file for detailed error stack information.

TNS-12541/ORA-12541: TNS:no listener

Cause: The connection request could not be completed because the listener is not running.

Action: Perform these steps:

Ensure that the supplied destination address matches one of the addresses used by the listener.Verify that this is not a version compatibility problem.

TNS-12549/ORA-12549: TNS:operating system resource quota exceeded and TNS-00519: Operating system resource quota exceeded

Cause: A quota or hard limit imposed by the operating system has been exceeded.

Possible limits include:

- The maximum number of processes allowed for a single user
- The operating system is running low on paging space

Action: Perform the appropriate action:

- Increase the number of processes by setting the PROCESSES parameter in the database initialization file to a larger value.
- Check the sqlnet.log or listener.log file for detailed error stack information, such as an operating system error code to help identify which quota has been exceeded.

TNS-12560/ORA-12560: TNS:protocol adapter error occurred

Cause: There was an error when using a particular protocol. This error may be due to incorrect configuration of an ADDRESS parameter or may occur due to errors returned from the underlying protocol or operating system interface.

In some cases, these errors will be caused by the same conditions which cause TNS-00510, TNS-00519, TNS-12540/ORA-12540, TNS-12549/ORA-12549 errors.

Action: Check the sqlnet.log or listener.log file for detailed error stack information.

Troubleshooting Directory Naming Errors

Directory naming issues associated with connectivity errors such as `ORA-12154`, `ORA-12543`, or `ORA-12541` for database service or net service name entries in a directory server require analysis of the data. You can analyze the data contained within a directory server with the `ldifwrite` command line tool.

`ldifwrite` enables you to convert all or part of the information residing in a directory server to **LDAP Data Interchange Format (LDIF)**. The `ldifwrite` tool performs a subtree search, including all entries following the specified **distinguished name (DN)**, including the DN itself.

The `ldifwrite` tool syntax is as follows:

```
ldifwrite -c net_service_name/database_service -b base_DN -f ldif_file
```

Table 16–7 lists `ldifwrite` tool arguments and descriptions for each.

Table 16–7 Idifwrite Arguments

Argument	Description
`-c net_service_name/database_service`	Specify the net service name or database service name that will connect you to the directory server.
`-b base_DN`	Specify the base of the subtree to be written out in LDIF format.
`-f ldif_file`	Specify the input file name.

The following example writes all the directory naming entries under `dc=us,dc=acme,dc=com` to the `output1.ldi` file:

```
ldifwrite -c ldap -b "dc=us,dc=acme,dc=com" -f output.ldif
```

Troubleshooting Tips from the Field for Oracle Net Services

Here are some tips you may find helpful when you are having difficulty diagnosing network problems:

- **Use the node or network address during configuration instead of the name of the server computer**

  This eliminates any internal lookup problems and make the connection slightly faster.

- **If you are using TCP/IP addresses, use the IP address rather than host name**

  For example, change the (`HOST=server_name`) line in the `tnsnames.ora` file with the internet address, for example (`HOST=198.32.3.5`).

- **Perform a loopback test**

  Perform a loopback test on the server as described in "Testing Configuration on the Database Server" on page 15-8. If the test passes, `ftp` the `tnsnames.ora` and `sqlnet.ora` files to the client.

- **Check what is between you and the server**

  If it is a wide area network (WAN), identify any intermediate systems that may not work correctly. If all computers are fine, the problem may be a timing issue.

- **Verify whether or not there is a timing issue**

  Timing issues are associated with an `ORA-12535` error in the client log files.

  To resolve this, try speeding up the connection by using exact addresses instead of names and increase the `INBOUND_CONNECT_TIMEOUT_listener_name` parameter in the `listener.ora` file. The default value for this parameter is 10 seconds.

- **Determine which Oracle applications are failing**

  SQL*Plus may work, but CASE tools may not. If you determine the problem is a data volume issue, try to transfer a large (5 MB) file with the base connectivity.

Questions to Ask When Troubleshooting Oracle Net Services

Here are some questions to ask yourself when diagnosing a problem:

- **Do all computers have a problem, or is it just one?**

If one computer works and another does not, and you are confident that the same software (Oracle and third-party products) is installed, on each computer, swap out the network cables, if they are close enough, to see if the problem moves. If it does move, it indicates that the problem has something to do with the client/server connection and is not local to the PC.

- **What kind of links exist between the client and the server, for example, X.25, ISDN, Token Ring, or leased line?**

  Sniffers and LAN analyzers are useful for intermittent failing connections or detecting time outs and resent packets. You can also see what side of the conversation is waiting for a response.

Troubleshooting the TNS-12154 Error

This section offers some solutions to the TNS-12154 error. The TNS-12154 error is encountered when SQL*Net cannot find the alias specified for a connection in the `tnsnames.ora` file or other naming adapter.

Before attempting to resolve the problem, it may be helpful to have a printout or view of both the `tnsnames.ora` file and the `sqlnet.ora` file. Looking at these files at the same time is helpful since references will be made to both.

This section includes the following topics:

- Problem Description for TNS-12154
- Troubleshooting TNS-12154 on UNIX

Problem Description for TNS-12154

The TNS-12154 error appears when SQL*Net cannot find the alias specified for a connection in the `tnsnames.ora` file or other naming adapter.

Before attempting to resolve this problem, it is helpful to print out or view both the `tnsnames.ora` file and the `sqlnet.ora` file. Looking at these files at the same time is helpful because references will be made to both.

The `tnsnames.ora` and `sqlnet.ora` files are located in the default network administration directory on the client system.

Troubleshooting TNS-12154 on UNIX

Be sure that the `tnsnames.ora` file and the `sqlnet.ora` file resemble the following examples.

Example 16–1 tnsnames.ora Sample

```
DEV1.WORLD =
     (DESCRIPTION =
       (ADDRESS_LIST =
           (ADDRESS =
             (PROTOCOL = TCP)
             (Host = 145.45.78.56)
             (Port = 1521)
           )
       )
       (CONNECT_DATA = (SID = ORCL)
       )
     )
```

Example 16–2 sqlnet.ora Sample

```
TRACE_LEVEL_CLIENT = OFF
SQLNET.AUTHENTICATION_SERVICES = (NONE)
NAMES.DIRECTORY_PATH = (TNSNAMES)
AUTOMATIC_IPC = OFF
```

To begin the diagnostic process, determine which section of this document applies to the problem. In the sample files shown in Example 16–1 and Example 16–2, the alias in Example 16–1 is `DEV1.WORLD` while the `NAMES.DEFAULT_DOMAIN=world` parameter does not exist in Example 16–2.

In this case, add the `NAMES.DEFAULT_DOMAIN=world` parameter anywhere in the `sqlnet.ora` file. Save the file, and try the connection again.

If the TNS-12154 error still persists, determine whether the files were transferred from the client to the server and check the configuration files to ensure that CTRL-M (`^M`) or CTRL-R (`^R`) characters were not inserted at the ends of any lines. Remove any such characters you may find.

If the characters do not exist, check to see whether the `NAMES.DIRECTORY_PATH` parameter exists in the `sqlnet.ora` file and make sure the value in parenthesis is `TNSNAMES`, as follows:

```
NAMES.DIRECTORY_PATH=(TNSNAMES)
NAMES.DIRECTORY_PATH=(TNSNAMES, HOSTNAME)
```

This parameter is not necessary but if it exists in the `sqlnet.ora` file and appears as in the preceding example, the configuration files are most likely technically accurate.

At the UNIX prompt, echo the `TNS_ADMIN` environment variable, as follows:

```
% echo $TNS_ADMIN
```

If nothing is returned, set the `TNS_ADMIN` environment variable to explicitly point to the location of the `tnsnames.ora` file.

- In C shell:

  ```
  % setenv TNS_ADMIN full_path_to_tnsnames.ora_file
  ```

- In K shell:

  ```
  % TNS_ADMIN=full_path_to_tnsnames.ora_file; export TNS_ADMIN
  ```

Try the connection again.

If the error persists, add the `AUTOMATIC_IPC=OFF` parameter to the `sqlnet.ora` file. If `AUTOMATIC_IPC` is already set to `ON`, change the value to `OFF`. Try the connection again.

If the error persists, check the permissions of the `tnsnames.ora` and `sqlnet.ora` files and parent directories—usually `.ora` files are either `-rwxrwxrwx` or `-rwxrwx---`. Change the permissions of the configuration files to `777` to set the permissions to fully open and try the connection again.

> **Note:** Setting permissions to `777` enables anyone on the system to access the configuration files. Do this *only* as a temporary test and reset the permissions when finished.

If the error persists, redo the configuration as follows:

1. Set the `TNS_ADMIN` environment variable to `/tmp`.
2. Go to the `/tmp` directory and create a new `tnsnames.ora` file using a text editor.
3. Copy the sample `tnsnames.ora` file from Example 16–1 into the text editor and save the new `tnsnames.ora` file.
4. Exit the text editor and at the command prompt, type:

```
% sqlplus scott@dev1.world
Enter password: password
```

This should connect or progress to the next logical error.

Troubleshooting Network Problems Using Log and Trace Files

Oracle Net Services provide detailed information about the source and context of problems as they arise. This information is generated and stored in log and trace files. The process of logging and tracing error information will help you to diagnose and resolve network problems.

Logging Error Information for Oracle Net Services

All errors encountered in Oracle Net Services are appended to a log file for evaluation by a network or database administrator. The log file provides additional information for an administrator when the error message on the screen is inadequate to understand the failure. The log file, by way of the error stack, shows the state of the software at various layers.

To ensure that all errors are recorded, logging cannot be disabled on clients or Names Servers. Furthermore, only an administrator may replace or erase log files. The log file for the listener also includes Audit Trail information about every client connection request, as well as most listener control commands.

This section contains these topics:

- Oracle Net Error Stacks
- Oracle Net Services Log File Names
- Setting Logging Parameters
- Setting Logging Settings During Runtime of Control Utilities
- Using Log Files
- Analyzing Listener Log Files
- Analyzing Oracle Connection Manager Logs

Oracle Net Error Stacks

Log files provide information contained in an error stack. An error stack refers to the information that is produced by each layer in an Oracle communications stack as the result of a network error.

The error stack components are described in Table 16–8.

Table 16–8 Error Stack Components

Error Stack Component	Description
NI	Network Interface. This layer provides a generic interface for Oracle clients, servers, or external processes to access Oracle Net functions. The NI layer handles the "break" and "reset" requests for a connection. NI uses the Network Routing (NR) layer to obtain network route information for pre-Oracle9*i* clients, and the Network Naming (NN) layer to resolve names to connect descriptors. For Oracle9*i* clients, NI goes directly to the Network Session (NS) layer.
NS	Network Session (main and secondary layers). These layers receive requests from NI, and settle all generic computer-level connectivity issues, such as: the location of the server or destination (open, close functions); whether one or more protocols will be involved in the connection (open, close functions); and how to handle interrupts between client and server based on the capabilities of each (send, receive functions).
NA	Network Authentication. This layer negotiates authentication and encryption requirements.
NT	Network Transport (main, secondary, and operating system layers). This layer maps Oracle Net foundation layer functionality to industry-standard protocols.

Example: Error Stack

As an example, suppose that a user of a client application tries to establish a connection with a database server using Oracle Net and TCP/IP, and the user enters:

```
sqlplus scott@hrserver.com
Enter password: password
```

The following error displays:

```
ORA-12543: TNS:Unable to connect to destination
```

This message indicates that the connection to the server failed because the database could not be contacted. Although the application displays only a one-line error message, an error stack that is much more informative is recorded in the log file by the network layer.

On the client side, the `sqlnet.log` file (Example 16–3) contains an error stack corresponding to the `ORA-12543` error.

Example 16–3 sqlnet.log File

```
************************************************************

Fatal OSN connect error 12543, connecting to:
 (DESCRIPTION=(CONNECT_DATA=(SID=trace)(CID=(PROGRAM=)
   (HOST=lala)(USER=sviavant)))(ADDRESS_LIST=(ADDRESS=
   (PROTOCOL=ipc)(KEY=trace))(ADDRESS=(PROTOCOL=tcp)
   (HOST=lala)(PORT=1521))))

VERSION INFORMATION:
TNS for SunOS:
Oracle Bequeath NT Protocol Adapter for SunOS:
Unix Domain Socket IPC NT Protocol Adaptor for SunOS:
TCP/IP NT Protocol Adapter for SunOS:
```

```
Tracing to file: /home/sviavant/trace_admin.trc
Tns error struct:
  TNS-12543: TNS:unable to connect to destination
  ns main err code: 12541
  TNS-12541: TNS:no listener
  ns secondary err code: 12560
  nt main err code: 511
  TNS-00511: No listener
  nt secondary err code: 61
  nt OS err code: 0
```

Oracle Net Services Log File Names

Each Oracle Net Services component produces its own log file. Table 16–9 provides the default log file names and lists the components that generate the log files.

Table 16–9 Log Files

Log File	Component
`listener.log`	Listener
`sqlnet.log`	Client or Database Server
`instance-name_pid``.log`	Oracle Connection Manager listener
`instance-name``_cmgw_`*`pid`*`.log`	Oracle Connection Manager **CMGW (Connection Manager gateway)** process
`instance-name``_cmadmin_`*`pid`*`.log`	Oracle Connection Manager **CMADMIN (Connection Manager Administration)** process
instance-name_alert.log	Oracle Connection Manager alert log

Setting Logging Parameters

Parameters that control logging, including the type and amount of information logged, as well as the location where the files are stored, are set in the configuration file of each network component as described in Table 16–10.

Table 16–10 Location of Log Parameters

Network Component	Configuration File
Oracle Connection Manager Processes	`cman.ora`
Listener	`listener.ora`
Client	`sqlnet.ora`
Database Server	`sqlnet.ora`

This section contains these topics:

- sqlnet.ora Log Parameters
- listener.ora Log Parameters
- cman.ora Log Parameters
- Setting Logging Parameters in Configuration Files

> **See Also:** *Oracle Database Net Services Reference* for more information about these parameters

sqlnet.ora Log Parameters

Table 16–11 describes the log parameters settings that can be set in the `sqlnet.ora` file.

Table 16–11 sqlnet.ora Log Parameters

sqlnet.ora Parameter	Oracle Net Manager Field	Description
`ADR_BASE`		The `ADR_BASE` parameter specifies the base directory into which tracing and logging incidents are stored. Use this parameter when `DIAG_ADR_ENABLED` is set to `ON`.
`DIAG_ADR_ENABLED`		The `DIAG_ADR_ENABLED` parameter indicates whether ADR tracing is enabled. When the `DIAG_ADR_ENABLED` parameter is set to `OFF`, non-ADR file tracing is used.
`LOG_DIRECTORY_CLIENT`	**Client Information: Log Directory**	Establishes the destination directory for the client log file. By default, the client directory is the current working directory.
`LOG_DIRECTORY_SERVER`	**Server Information: Log Directory**	Establishes the destination directory for the database server log files. By default the server directory is `$ORACLE_HOME/network/log` on UNIX and `%ORACLE_HOME%\network\log` on Windows.
`LOG_FILE_CLIENT`	**Client Information: Log File**	Sets the name of the log file for the client. By default the log name is `sqlnet.log`.
`LOG_FILE_SERVER`	Not applicable	Sets the name of the log file for the database server. By default the log name is `sqlnet.log`.

listener.ora Log Parameters

Table 16–12 describes the log parameters settings that can be set in the `listener.ora` file.

Table 16–12 listener.ora Log Parameters

listener.ora Parameter	Oracle Net Manager Field	Description
`ADR_BASE_`*`listener_name`*		The `ADR_BASE_`*`listener_name`* parameter specifies the base directory into which tracing and logging incidents are stored. Use this parameter when `DIAG_ADR_ENABLED_`*`listener_name`* is set to `ON`.
`DIAG_ADR_ENABLED_`*`listener_name`*		The `DIAG_ADR_ENABLED_`*`listener_name`* parameter indicates whether ADR tracing is enabled. When the `DIAG_ADR_ENABLED_`*`listener_name`* parameter is set to `OFF`, non-ADR file tracing is used.
`LOG_DIRECTORY_`*`listener_name`* `LOG_FILE_`*`listener_name`*	**Log File**	Establishes the destination directory and file for the log file that is automatically generated for listener events. By default the directory is `$ORACLE_HOME/network/log` on UNIX and *`ORACLE_HOME`*`\network\log` on Windows, and the file name is defaulted to `listener.log`.

cman.ora Log Parameters

Table 16–13 describes the log parameters settings that can be set in the `cman.ora` file.

Table 16–13 cman.ora Log Parameters

cman.ora Parameter	Description
ADR_BASE	The ADR_BASE parameter specifies the base directory into which tracing and logging incidents are stored. Use this parameter when DIAG_ADR_ENABLED is set to ON.
DIAG_ADR_ENABLED	The DIAG_ADR_ENABLED parameter indicates whether ADR tracing is enabled. When the DIAG_ADR_ENABLED parameter is set to OFF, non-ADR file tracing is used.
EVENT_GROUP	Specifies which event groups are logged. Multiple events may be designated using a comma-separated list. This parameter accepts the following values: ■ INIT_AND_TERM—initialization and termination ■ MEMORY_OPS—memory operations ■ CONN_HDLG—connection handling ■ PROC_MGMT—process management ■ REG_AND_LOAD—registration and load update ■ WAKE_UP—events related to CMADMIN wakeup queue ■ TIMER—gateway time outs ■ CMD_PROC—command processing ■ RELAY—events associated with connection control blocks
LOG_DIRECTORY	Establishes the destination directory for log files. By default, the directory is $ORACLE_HOME/network/log on UNIX and %ORACLE_HOME%\network\log on Windows.
LOG_LEVEL	Establishes the level of logging. Four levels are supported: ■ off (default)—no logging ■ user—user log information ■ admin—administrative log information ■ support—Oracle Support Services information The Oracle Connection Manager listener, gateway, and CMADMIN processes create log files on both UNIX and Windows.

Setting Logging Parameters in Configuration Files

You configure logging parameters for the sqlnet.ora file with Oracle Net Manager and listener.ora file with either Oracle Enterprise Manager or Oracle Net Manager.

You must manually configure cman.ora file logging parameters.

See Also: *Oracle Database Net Services Reference*

To set logging parameters with Oracle Enterprise Manager and Oracle Net Manager, refer to Table 16–14.

Table 16–14 Setting Logging Parameters in Configuration Files

Log File	Tool	Set Logging Parameters Here...
`sqlnet.log`	Oracle Net Manager	1. Start Oracle Net Manager. **See Also:** "Oracle Net Manager" on page 6-2 2. In the navigator pane, expand **Local** > **Profile**. 3. From the list in the right pane, select **General**. 4. Click the **Logging** tab. 5. Specify the settings. 6. Choose **File** > **Save Network Configuration**.
`listener.log`	Oracle Enterprise Manager	1. Access the Oracle Net Administration page in Oracle Enterprise Manager. **See Also:** "Oracle Enterprise Manager" on page 6-1 2. Select **Listeners** from the **Administer** list, and then select the Oracle home that contains the location of the configuration files. 3. Click **Go** to display the Listeners page. 4. Select a listener, and then click **Edit** to display the Edit Listeners page. 5. Click the **Logging & Tracing** tab. 6. Specify the settings. 7. Click **OK**.
`listener.log`	Oracle Net	1. Start Oracle Net Manager. **See Also:** "Oracle Net Manager" on page 6-2 2. In the navigator pane, expand **Local** > **Listeners**. 3. Select a listener. 4. From the list in the right pane, select **General**. 5. Click the **Logging and Tracing** tab. 6. Specify the settings. 7. Choose **File** > **Save Network Configuration**.

Setting Logging Settings During Runtime of Control Utilities

You can set logging during control utility runtime. Setting logging with a control utility does not set parameters in the `*.ora` files; the setting is only valid for the session of the control utility:

- For a listener, use the `SET LOG_FILE` and `SET LOG_DIRECTORY` commands from the Listener Control utility.
- For an Oracle Connection Manager, use the `SET LOG_DIRECTORY`, `SET LOG_LEVEL`, and `SET EVENT` commands from the Oracle Connection Manager control utility.

See Also: *Oracle Database Net Services Reference*

Using Log Files

To use a log file to diagnose a network error:

1. Review the log file for the most recent error number you received from the application. Note that this is almost always the last entry in the log file.
2. Starting from the bottom of the file, locate the first nonzero entry in the error report. This is usually the actual cause.
3. If that error does not provide the desired information, review the next error in the stack until you locate the correct error information.
4. If the cause of the error is still not clear, turn on tracing and repeat the statement that produced the error message.

Analyzing Listener Log Files

This section describes what is recorded in the listener log file, including:

- Listener Log Audit Trail Information
- Listener Service Registration Event Information
- Listener Direct Hand-Off Information
- Listener Subscription for ONS Node Down Event Information
- Listener Oracle Clusterware Notification Information

Listener Log Audit Trail Information

The listener log file contains audit trail information that enables you to gather and analyze network usage statistics, as well as information indicating the following:

- A client connection request
- A `RELOAD`, `START`, `STOP`, `STATUS`, or `SERVICES` command issued by the Listener Control utility

You can use Audit Trail information to view trends and user activity by first storing it in a table and then collating it into a report format. To import the data into a table, use an import utility such as SQL*Loader.

Format of the Listener Log Audit Trail

The audit trail formats text into the following fields:

```
Timestamp * Connect Data [* Protocol Info] * Event [* SID | Service] * Return Code
```

Properties of the audit trail are as follows:

- Each field is delimited by an asterisk (`*`).
- Protocol address information and service name or SID information appear only when a connection is attempted.
- A successful connection or command returns a code of zero.
- A failure produces a code that maps to an error message.

See Also:

- "Resolving the Most Common Error Messages for Oracle Net Services" on page 16-10 for information about resolving the most common Oracle Net errors
- *Oracle Database Error Messages* for a complete listing of error messages

Example: Listener Log Event for Successful Reload Request

The following output shows a log file excerpt with `RELOAD` command request.

```
14-JUL-2002 00:29:54 *
(connect_data=(cid=(program=)(host=sales-server)(user=jdoe))(command=stop)
(arguments=64)(service=listener)(version=135290880))
* stop * 0
```

Example: Listener Log Events for a Successful Connection Request

The following output shows a log file excerpt with a successful connection request.

```
14-JUL-2002 15:28:58 *
(connect_data=(service_name=sales.us.acme.com)(cid=(program=)(host=sales-server)
(user=jdoe)))
* (address=(protocol=tcp)(host=10.10.150.35)(port=41349)) * establish
* sales.us.acme.com * 0
```

Example: Listener Log Events for an Unsuccessful Connection Request

The following output shows a log file excerpt with a successful execution of the `STATUS` command by host `sales-server`, followed by an unsuccessful connection attempt by a client with an IP address of `10.10.150.35`. This connection attempt resulted in an ORA-12525: TNS:listener has not received client's request in time allowed error message, which occurs when a client fails to complete its connect request in the time specified by the `INBOUND_CONNECT_TIMEOUT_`*`listener_name`* parameter in the `listener.ora` file. This client may be attempting a denial-of-service attack on the listener.

```
03-JUL-2002 16:41:57 *
(CONNECT_DATA=(CID=(PROGRAM=)(HOST=sales-server)(USER=jdoe))(COMMAND=status)
(ARGUMENTS=64)(SERVICE=LISTENER)(VERSION=153092352)) * status * 0
03-JUL-2002 16:42:35 * <unknown connect data> *
(ADDRESS=(PROTOCOL=tcp)(HOST=10.10.150.35)(PORT=53208)) * establish *
<unknown sid> * 12525
TNS-12525: TNS:listener has not received client's request in time allowed
TNS-12604: TNS: Application timeout occurred
```

Listener Service Registration Event Information

The listener records service registration events. During service registration, the **PMON process** provides the listener with information about the following:

- Service names for each running instance of the database
- Instance names of the database
- Service handlers (dispatchers or dedicated servers) available
- Dispatcher, instance, and node load information
- Dynamic listening endpoints

The service registration-related events listed in Table 16–15 are recorded in the `listener.log` file:

Table 16–15 Service Registration Event Log Information

Event	Description
`service_register`	The listener received registration information for an instance.

Table 16–15 (Cont.) Service Registration Event Log Information

Event	Description
`service_update`	The listener received updated registration information for a particular instance, such as dispatcher or instance load information.
`service_died`	The listener lost its connection to PMON. All registration information for the instance is discarded. Clients will be unable to connect to the instance until PMON registers it again.

Format of the Listener Service Registration Information

The service registration events are formatted into the following fields:

```
Timestamp * Event *  Instance Name * Return Code
```

Properties of service registration fields are as follows:

- Each field is delimited by an asterisk (`*`).
- It is normal for the events to appear multiple times in a row for one instance.
- A successful registration returns a code of zero, meaning the client can connect to the instance.
- A failure produces a code that maps to an error message.

See Also:

- "Resolving the Most Common Error Messages for Oracle Net Services" on page 16-10 for the most common Oracle Net errors
- *Oracle Database Error Messages* for a complete listing of error messages

Example: Listener Log with Service Registration Events

The following example shows a log file with service registration events. Notice how the listener is able to receive a client request after a successful `service_register` event, but is unable to receive client requests after a `service_died` event.

```
------------------------------
14-JUL-2002 15:28:43 * service_register * sales * 0
14-JUL-2002 15:28:43 * service_register * sales * 0
14-JUL-2002 15:28:58 *
(connect_data=(service_name=sales.us.acme.com)(cid=(program=)(host=sales-server)
(user=jdoe)))
* (address=(protocol=tcp)(host=10.10.150.35)(port=41349)) * establish
* sales.us.acme.com * 0
14-JUL-2002 15:38:44 * service_update * sales * 0
14-JUL-2002 15:38:44 * service_update * sales * 0
14-JUL-2002 15:48:45 * service_update * sales * 0
14-JUL-2002 15:48:45 * service_update * sales * 0
14-JUL-2002 15:50:57 *
(connect_data=(service_name=sales.us.acme.com)(cid=(program=)(host=sales-server)(u
ser=jdoe)))
* (address=(protocol=tcp)(host=10.10.150.35)(port=41365)) * establish
* sales.us.acme.com * 0
14-JUL-2002 15:51:26 * service_died * sales * 12537
14-JUL-2002 15:51:26 * service_died * sales * 12537
14-JUL-2002 15:52:06 *
```

```
(connect_data=(service_name=sales.us.acme.com)(cid=(program=)(host=sales-server)(u
ser=jdoe)))
* (address=(protocol=tcp)(host=10.10.150.35)(port=41406)) * establish
* sales.us.acme.com * 12514
TNS-12514: TNS:listener could not resolve SERVICE_NAME given in connect
descriptor
-------------------------------
```

Listener Direct Hand-Off Information

The listener records direct hand-off events to **dispatcher**s. These events are formatted into the following fields:

```
Timestamp * Presentation * Handoff  * Error Code
```

Properties of direct hand-off fields are as follows:

- Each field is delimited by an asterisk (*).
- A successful connection or command returns a code of zero.
- A failure produces a code that maps to an error message.

> **See Also:** "Resolving the Most Common Error Messages for Oracle Net Services" on page 16-10 for the most common Oracle Net errors or *Oracle Database Error Messages* for a complete listing of error messages

Example: Listener Log Event for Direct Hand-Off

A direct hand-off event in the log file is shown in the following example.

```
21-JUL-2002 10:54:55 * oracle.aurora.net.SALESHttp2 * handoff * 0
```

Listener Subscription for ONS Node Down Event Information

Listener will subscribe to the Oracle Notification Service (ONS) `node down` event on startup if ONS configuration file is available. This subscription enables the listener to remove the affected service when it receives `node down` event notification from ONS. The listener uses asynchronous subscription for the event notification. The following warning message will be recorded to listener log file on each STATUS command if the subscription has not completed; for example if the ONS daemon is not running on the host.

```
WARNING: Subscription for node down event still pending
```

Listener will not be able to receive the ONS event while subscription is pending. Other than that, no other listener functionality is affected.

Listener Oracle Clusterware Notification Information

If the required Oracle Clusterware (CRS in the following log messages) libraries are installed and Oracle Clusterware is started on the host, Oracle Listener will notify Oracle Clusterware about its status upon start and stop. After successful notification, listeners records the event in the log. No message will be recorded if the notification fails.

```
Listener completed notification to CRS on start
Listener completed notification to CRS on stop
```

Analyzing Oracle Connection Manager Logs

Oracle Connection Manager generates four types of log files: one each for its listener, gateway, and CMADMIN processes and one for alerts. The last is a chronological record of all critical errors. In addition to logging critical errors, the alert log captures information about instance startup and shutdown. It also records the value of all configuration parameters at the beginning and end of a session. See Table 16–9, " Log Files" on page 16-22 for file name syntax.

The CMADMIN and gateway log files are reproduced here. Table 16–16, " CMADMIN and Gateway Log Entries: What They Mean" explains some of the log entries. Each entry consists of a timestamp and an event. You can configure cman.ora to log events in the following categories:

- Initialization and termination
- Memory operations
- Connection handling
- Process management
- Registration and load update
- Events related to CMADMIN wakeup queue
- Gateway timeouts
- Command processing
- Events associated with connection control blocks

Use the SET EVENT command to specify which events to log.

This section includes the following examples:

- CMADMIN Log File Example
- Gateway Log File Example

CMADMIN Log File Example

```
------------------------------
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:40)(EVENT=Parameter list)
    (listener_address=(address=(protocol=tcp)(host=usunnae16)(port=1574)))
    (aso_authentication_filter=OFF)
    (connection_statistics=ON)
    (log_directory=/home/user/network/admin/log)
    (log_level=support)
    (max_connections=256)
    (idle_timeout=5)
    (inbound_connect_timeout=0)
    (session_timeout=20)
    (outbound_connect_timeout=0)
    (max_gateway_processes=1)
    (min_gateway_processes=1)
    (password=OFF)
    (remote_admin=ON)
    (trace_directory=/home/user/network/admin/log)
    (trace_level=off)
    (trace_timestamp=OFF)
    (trace_filelen=0)
    (trace_fileno=0)
)
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:40)(EVENT=Shared Memory Size)
```

```
(BYTES=82524))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:40)(EVENT=GMON Attributes validated)
(Type=Information))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:40)(EVENT=NS Listen Successful)
((ADDRESS=(PROTOCOL=tcp)(HOST=usunnae16)(PORT=55878))))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:44)(EVENT=Received command)(CMD=verify
password))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:44)(EVENT=Received command)
(CMD=version))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:44)(EVENT=Received command)
(CMD=show status))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:44)(EVENT=Failed to get procedure id))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:49:12)(EVENT=Received command)(CMD=verify
password))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:49:15)(EVENT=Failed to get procedure id))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:49:29)(EVENT=Received command)(CMD=verify
password))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:49:46)(EVENT=Failed to get procedure id))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:49:50)(EVENT=Received command)(CMD=verify
password))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:49:50)(EVENT=Received command)
(CMD=probe monitor))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:49:50)(EVENT=Received command)
(CMD=shutdown normal))
-------------------------------
```

Gateway Log File Example

```
-------------------------------
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=NS Initialised))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=Memory Allocated)
(BYTES=1024))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=NCR Initialised))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=Connected to Monitor))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=State Change from Empty to
Init))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=Memory Allocated)
(BYTES=251904))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=Memory Allocated)
(BYTES=2048))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=CCB Initialised))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=Started Listening))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:41)(EVENT=State Change from Init to
Ready))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:46:47)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:06)(EVENT=Ready)(CONN NO=0))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:06)(EVENT=Ready)(CONN NO=0))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:07)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:12)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:13)(EVENT=Idle Timeout)(CONN NO=0))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:17)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:22)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:25)(EVENT=Ready)(CONN NO=0))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:25)(EVENT=Ready)(CONN NO=0))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:27)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:30)(EVENT=Idle Timeout)(CONN NO=0))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:32)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:37)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:42)(EVENT=Ready)(CONN NO=0))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:42)(EVENT=Ready)(CONN NO=0))
```

```
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:42)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:47)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:52)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:48:57)(EVENT=Housekeeping))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:49:02)(EVENT=Session Timeout)(CONN NO=0))
(LOG_RECORD=(TIMESTAMP=08-MAY-2003 08:49:02)(EVENT=Housekeeping))
-------------------------------
```

Table 16–16 CMADMIN and Gateway Log Entries: What They Mean

Event	Description	Log File
GMON Attributes validated	Informational message. The parameters needed for CMADMIN to come up are specified correctly.	CMADMIN
Failed to get procedure ID	The CMCTL session connected to CMADMIN has disconnected.	CMADMIN
Out of CCB	CMADMIN is unable to process a connection request. There could be two reasons: ■ Faulty load update between CMADMIN and listener ■ Someone is trying to connect to CMADMIN directly (possibly a denial of service attack)	Gateway
No connect data	An unknown client is trying to connect to CMADMIN. This is most likely a denial of service attack.	CMADMIN
Invalid connect data	An unknown client is trying to connect to CMADMIN. This is most likely a denial of service attack.	CMADMIN
Housekeeping	Informational message. Internal housekeeping for the gateway process is in order. The gateway process is properly connected to the CMADMIN process.	Gateway
Connected to Monitor	The gateway has connected to CMADMIN.	Gateway
State change from Empty to Init	State change message from the gateway. Once it reaches a ready state, the gateway begins accepting connections from the client.	Gateway
State change from Init to Ready	State change message from the gateway. Once it reaches a ready state, the gateway begins accepting connections from the client.	Gateway
Idle Timeout	The connection was disconnected because it was idle longer than the time specified in cman.ora.	Gateway
Session Timeout	The connection was disconnected because it exceeded the session timeout specified in cman.ora.	Gateway

Tracing Error Information for Oracle Net Services

Tracing produces a detailed sequence of statements that describe network events as they are executed. Tracing an operation enables you to obtain more information on the internal operations of the components of Oracle Net Services than is provided in a log file. This information is output to files that can be evaluated to identify the events that led to an error.

Note: Tracing uses a large amount of disk space and may have a significant impact upon system performance. Therefore, you should enable tracing only when necessary.

This section contains topics:

- Oracle Net Services Trace File Names
- Setting Tracing Parameters
- Setting Tracing Settings During Runtime of Control Utilities
- Evaluating Oracle Net Services Traces
- Using the Trace Assistant to Examine Trace Files

Oracle Net Services Trace File Names

Each Oracle Net Services component produces its own trace file. Table 16–17 provides the default trace file names and lists the components that generate the trace files.

Table 16–17 Trace Files

Trace File	Component
instance-name_pid.trc	Oracle Connection Manager listener
*instance-name*_cmgw_*pid*.trc	Oracle Connection Manager **CMGW (Connection Manager gateway)** process
*instance-name*_cmadmin_*pid*.trc	Oracle Connection Manager **CMADMIN (Connection Manager Administration)** process
listener.trc	Listener
sqlnet.trc	Client
svr_*pid*.trc	Database Server
tnsping.trc	TNSPING Utility

Setting Tracing Parameters

Parameters that control tracing, including the type and amount of information trace, as well as the location where the files are stored, are set in the configuration file of each network component as described in Table 16–18.

Table 16–18 Location of Trace Parameters

Component	Configuration File
Oracle Connection Manager Processes	cman.ora
Listener	listener.ora
Client	sqlnet.ora
Database Server	sqlnet.ora
TNSPING Utility	sqlnet.ora

This section contains these topics:

- sqlnet.ora Trace Parameters

- listener.ora Trace Parameters
- cman.ora Trace Parameters
- Setting Tracing Parameters in Configuration Files

See Also: *Oracle Database Net Services Reference* for more information about these parameters

sqlnet.ora Trace Parameters

Table 16–19 describes the trace parameters settings that can be set in the `sqlnet.ora` file.

Table 16–19 sqlnet.ora Trace Parameters

sqlnet.ora Parameter	Oracle Net Manager Field	Description
`TRACE_DIRECTORY_CLIENT`	**Client Information: Trace Directory**	Establishes the destination directory for the client trace output. By default, the client directory is `$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`\network\trace` on Windows.
`TRACE_DIRECTORY_SERVER`	**Server Information: Trace Directory**	Establishes the destination directory for the database server trace output. By default, the server directory is `$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`\network\trace` on Windows.
`TRACE_FILE_CLIENT`	**Client Information: Trace File**	Sets the name of the trace file for the client. By default the trace file name is `sqlnet.trc`.
`TRACE_FILE_SERVER`	**Server Information: Trace File**	Sets the name of the trace file for the database server. By default the trace file name is `svr_`*`pid`*`.trc`.
`TRACE_FILELEN_CLIENT`	Not Applicable	Specifies the size of the client trace files in kilobytes (KB). When the size is met, the trace information is written to the next file. The number of files is specified with the `TRACE_FILENO_CLIENT` parameter.
`TRACE_FILELEN_SERVER`	Not Applicable	Specifies the size of the database server trace files in kilobytes (KB). When the size is met, the trace information is written to the next file. The number of files is specified with the `TRACE_FILENO_CLIENT` parameter.
`TRACE_FILENO_CLIENT`	Not Applicable	Specifies the number of trace files for client tracing. When this parameter is set along with the `TRACE_FILELEN_CLIENT` parameter, trace files are used in a cyclical fashion. The first file is filled first, then the second file, and so on. When the last file has been filled, the first file is re-used, and so on. The trace file names are distinguished from one another by their sequence number. For example, if the default trace file of `sqlnet.trc` is used, and this parameter is set to 3, the trace files would be named `sqlnet1_`*`pid`*`.trc`, `sqlnet2_`*`pid`*`.trc` and `sqlnet3_`*`pid`*`.trc`. In addition, trace events in the trace files are preceded by the sequence number of the file.

Table 16–19 (Cont.) sqlnet.ora Trace Parameters

sqlnet.ora Parameter	Oracle Net Manager Field	Description
TRACE_FILENO_SERVER	Not Applicable	Specifies the number of trace files for database server tracing. When this parameter is set along with the TRACE_FILELEN_SERVER parameter, trace files are used in a cyclical fashion. The first file is filled first, then the second file, and so on. When the last file has been filled, the first file is re-used, and so on. The trace file names are distinguished from one another by their sequence number. For example, if the default trace file of svr_*pid*.trc is used, and this parameter is set to 3, the trace files would be named svr1_*pid*.trc, svr2_*pid*.trc and svr3_*pid*.trc. In addition, trace events in the trace files are preceded by the sequence number of the file.
TRACE_LEVEL_CLIENT	**Client Information: Trace Level**	Specifies the level of detail the trace facility records for the client. The trace level value can either be a value within the range of 0 (zero) to 16 (where 0 is no tracing and 16 represents the maximum amount of tracing) or a value of off, admin, user, or support. ■ off (equivalent to 0) provides no tracing ■ user (equivalent to 4) traces to identify user-induced error conditions ■ admin (equivalent to 6) traces to identify installation-specific problems ■ support (equivalent to 16) provides trace information for troubleshooting information for Oracle Support Services
TRACE_LEVEL_SERVER	**Server Information: Trace Level**	Specifies the level of detail the trace facility records for the database server. The trace level value can either be a value within the range of 0 (zero) to 16 (where 0 is no tracing and 16 represents the maximum amount of tracing) or a value of off, admin, user, or support. ■ off (equivalent to 0) provides no tracing ■ user (equivalent to 4) traces to identify user-induced error conditions ■ admin (equivalent to 6) traces to identify installation-specific problems ■ support (equivalent to 16) provides trace information for troubleshooting information for Oracle Support Services

Table 16–19 (Cont.) sqlnet.ora Trace Parameters

sqlnet.ora Parameter	Oracle Net Manager Field	Description
TRACE_TIMESTAMP_CLIENT	Not Applicable	Adds a time stamp in the form of *dd-mon-yyyy hh:mi:ss:mil* to every trace event in the client trace file, sqlnet.trc.
TRACE_TIMESTAMP_SERVER	Not Applicable	Adds a time stamp in the form of *dd-mon-yyyy hh:mi:ss:mil* to every trace event in the client trace file, sqlnet.trc.
TRACE_UNIQUE_CLIENT	**Client Information: Unique Trace File Name**	When the value is set to on, Oracle Net creates a unique file name for each trace session by appending a process identifier to the name of each trace file generated, enabling several files to coexist. For example, trace files named sqlnet*pid*.trc are created if default trace file name sqlnet.trc is used. When the value is set to off, data from a new client trace session overwrites the existing file.

You can manually add the following TNSPING utility tracing parameters described in Table 16–20 to sqlnet.ora. The TNSPING utility determines whether or not a service (such as a database or other TNS services) on a Oracle Net network can be successfully reached.

Table 16–20 TNSPING Trace Parameters

sqlnet.ora Parameter	Description
TNSPING.TRACE_DIRECTORY	Establishes the destination directory for TNSPING trace file, tnsping.trc. By default, the directory is $ORACLE_HOME/network/trace on UNIX and %ORACLE_HOME%\network\trace on Windows.
TNSPING.TRACE_LEVEL	Specifies the level of detail the trace facility records for the TNSPING utility. The trace level value can either be a value within the range of 0 (zero) to 16 (where 0 is no tracing and 16 represents the maximum amount of tracing) or a value of off, admin, user, or support. ▪ off (equivalent to 0) provides no tracing ▪ user (equivalent to 4) traces to identify user-induced error conditions ▪ admin (equivalent to 6) traces to identify installation-specific problems ▪ support (equivalent to 16) provides trace information for troubleshooting information for Oracle Support Services

listener.ora Trace Parameters

Table 16–21 describes the trace parameters settings for the listener that can be set in the listener.ora file.

Table 16–21 listener.ora Trace Parameters

listener.ora Parameter	Oracle Enterprise Manager/Oracle Net Manager Field	Description
TRACE_LEVEL_*listener_name*	**Select a trace level/Trace Level**	Specifies the level of detail the trace facility records for the listener. The trace level value can either be a value within the range of 0 (zero) to 16 (where 0 is no tracing and 16 represents the maximum amount of tracing) or a value of off, admin, user, or support. ■ off (equivalent to 0) provides no tracing ■ user (equivalent to 4) traces to identify user-induced error conditions ■ admin (equivalent to 6) traces to identify installation-specific problems ■ support (equivalent to 16) provides trace information for troubleshooting information for Oracle Support Services
TRACE_DIRECTORY_*listener_name* TRACE_FILE_*listener_name*	**Trace File**	Establishes the destination directory and file for the trace file. By default the directory is $ORACLE_HOME/network/trace on UNIX and %ORACLE_HOME%\network\trace on Windows, and the file name is listener.trc.
TRACE_FILELEN_*listener_name*	Not Applicable	Specifies the size of the listener trace files in kilobytes (KB). When the size is met, the trace information is written to the next file. The number of files is specified with the TRACE_FILENO_*listener_name* parameter
TRACE_FILENO_*listener_name*	Not Applicable	Specifies the number of trace files for listener tracing. When this parameter is set along with the TRACE_FILELEN_*listener_name* parameter, trace files are used in a cyclical fashion. The first file is filled first, then the second file, and so on. When the last file has been filled, the first file is re-used, and so on. The trace file names are distinguished from one another by their sequence number. For example, if the default trace file of listener.trc is used, and this parameter is set to 3, the trace files would be named listener1.trc, listener2.trc and listener3.trc. In addition, trace events in the trace files are preceded by the sequence number of the file.
TRACE_TIMESTAMP_*listener_name*	Not Applicable	Adds a time stamp in the form of *dd-mon-yyyy hh:mi:ss:mil* to every trace event in the listener trace file.

cman.ora Trace Parameters

Table 16–22 describes the trace parameters settings for Oracle Connection Manager that can be set in the cman.ora file.

Table 16–22 cman.ora Trace Parameters

cman.ora Parameter	Description
TRACE_DIRECTORY	Establishes the destination directory for trace files. By default, the directory is $ORACLE_HOME/network/trace on UNIX and %ORACLE_HOME%\network\trace on Windows.
TRACE_FILELEN	Specifies the size of the trace file in kilobytes (KB). When the size is met, the trace information is written to the next file. The number of files is specified with the TRACE_FILENO parameter.
TRACE_FILENO	Specifies the number of trace files for tracing. When this parameter is set along with the TRACE_FILELEN parameter, trace files are used in a cyclical fashion. The first file is filled first, then the second file, and so on. When the last file has been filled, the first file is reused, and so on. The trace file names are distinguished from one another by their sequence number. For example, if this parameter is set to 3, the Oracle Connection Manager trace files for the gateway processes would be named *instance-name*_cmgw1_*pid*.trc, *instance-name*_cmgw2_*pid*.trc and *instance-name*_cmgw3_*pid*.trc. In addition, trace events in the trace files are preceded by the sequence number of the file.
TRACE_LEVEL	Specifies the trace level for the Oracle Connection Manager instance. This parameter accepts four trace levels: ■ off (default)—no logging ■ user—user log information ■ admin—administrative log information ■ support—Oracle Support Services information The Oracle Connection Manager listener, gateway, and CMADMIN processes create trace files on both UNIX and Windows. See Table 16–17, " Trace Files" on page 16-33 for file name syntax.
TRACE_TIMESTAMP TRACE_TIMESTAMP	If the TRACING parameter is enabled, adds a time stamp in the form of *dd-mon-yyyy hh:mi:ss:mil* to every trace event in the trace files.

Setting Tracing Parameters in Configuration Files

You configure tracing parameters for the sqlnet.ora file with Oracle Net Manager and listener.ora file with either Oracle Enterprise Manager or Oracle Net Manager.

You must manually configure cman.ora file tracing parameters.

See Also: *Oracle Database Net Services Reference*

To set tracing parameters with Oracle Enterprise Manager and Oracle Net Manager, refer to Table 16–23

Table 16–23 Set Tracing Parameters in Configuration Files

Trace File	Tool	Set Logging Parameters Here...
`sqlnet.trc` (for the client) `svr_pid.trc` (for the server)	Oracle Net Manager	1. Start Oracle Net Manager. **See Also:** "Oracle Net Manager" on page 6-2 2. In the navigator pane, expand **Local** > **Profile**. 3. From the list in the right pane, select **General**. 4. Click the **Tracing** tab. 5. Specify the settings. 6. Choose **File** > **Save Network Configuration**.
`listener.trc`	Oracle Enterprise Manager	1. Access the Oracle Net Administration page in Oracle Enterprise Manager. **See Also:** "Oracle Enterprise Manager" on page 6-1 2. Select **Listeners** from the **Administer** list, and then select the Oracle home that contains the location of the configuration files. 3. Click **Go** to display the Listeners page. 4. Select a listener, and click **Edit** to display the Edit Listeners page. 5. Click the **Logging & Tracing** tab. 6. Specify the settings. 7. Click **OK**.
		1. Start Oracle Net Manager. **See Also:** "Oracle Net Manager" on page 6-2 2. In the navigator pane, expand **Local** > **Listeners**. 3. Select a listener. 4. From the list in the right pane, select **General**. 5. Click the **Logging and Tracing** tab. 6. Specify the settings. 7. Choose **File** > **Save Network Configuration**.

Setting Tracing Settings During Runtime of Control Utilities

You can set tracing during control utility runtime. Setting tracing with a control utility does not set parameters in the `*.ora` files; the setting is only valid for the session of the control utility:

- For the listener, use the `SET TRC_DIRECTORY`, `SET TRC_FILE`, and `SET TRC_LEVEL` commands from the Listener Control utility.
- For an Oracle Connection Manager, use the `SET TRACE_DIRECTORY` and `SET TRACE_LEVEL`, and `SET TRACE_TIMESTAMP` commands from the Oracle Connection Manager control utility.

See Also: *Oracle Database Net Services Reference*

This section includes the following topics:

- Evaluating Oracle Net Services Traces
- Using the Trace Assistant to Examine Trace Files

Evaluating Oracle Net Services Traces

Trace files can help Oracle Support Services diagnose and troubleshoot network problems.

This section explains how to perform basic analysis of trace files. The topics discussed include:

- Flow of Data Packets Between Network Nodes
- Oracle Net Data Packet Formats
- Pertinent Oracle Net Trace Error Output

Flow of Data Packets Between Network Nodes

Oracle Net performs its functions by sending and receiving data packets.By specifying a trace level of `support`, you can view the actual contents of the Oracle Net packet in your trace file. The order of the packet types sent and received will help you to determine how your connection was established.

Oracle Net Data Packet Formats

Each line in the trace file begins with a procedure followed by a message. Following each procedure is a line of hexadecimal data representing actual data. The actual data that flows inside the packet is sometimes viewable to the right of the hexadecimal data.

Next is a list of the Oracle Net packet keywords and descriptions of the types of packets they represent:

Keyword	Packet Type
`NSPTCN`	Connect
`NSPTAC`	Accept
`NSPTRF`	Refuse
`NSPTRS`	Resend
`NSPTDA`	Data
`NSPCNL`	Control
`NSPTMK`	Marker

For example, the following line describes a procedure called "`nscon`" sending a `NSPTCN` packet over the network:

```
nscon: sending NSPTCN packet
```

Each packet has a keyword that denotes the packet type. All packet types begin with the prefix "`nsp`". It is helpful to remember this when reviewing trace files for specific packet information

Example 16–4 provides typical packet information.

Example 16–4 Packet Information

```
nscon: entry
nscon: doing connect handshake...
nscon: sending NSPTCN packet
nspsend: entry
```

```
nspsend: plen=187, type=1
nspsend: 187 bytes to transport
nspsend:packet dump
nspsend:00 BB 00 00 01 00 00 00  |........|
nspsend:01 33 01 2C 0C 01 08 00  |.3.,....|
nspsend:7F FF 7F 08 00 00 00 01  |........|
nspsend:00 99 00 22 00 00 08 00  |..."....|
nspsend:01 01 28 44 45 53 43 52  |..(DESCR|
nspsend:49 50 54 49 4F 4E 3D 28  |IPTION=(|
nspsend:43 4F 4E 4E 45 43 54 5F  |CONNECT_|
nspsend:44 41 54 41 3D 28 53 49  |DATA=(SI|
nspsend:44 3D 61 70 33 34 37 64  |D=ap347d|
nspsend:62 31 29 28 43 49 44 3D  |b1)(CID=|
nspsend:28 50 52 4F 47 52 41 4D  |(PROGRAM|
nspsend:3D 29 28 48 4F 53 54 3D  |=)(HOST=|
nspsend:61 70 32 30 37 73 75 6E  |ap207sun|
nspsend:29 28 55 53 45 52 3D 6D  |)(USER=m|
nspsend:77 61 72 72 65 6E 29 29  |warren))|
nspsend:29 28 41 44 44 52 45 53  |)(ADDRES|
nspsend:53 5F 4C 49 53 54 3D 28  |S_LIST=(|
nspsend:41 44 44 52 45 53 53 3D  |ADDRESS=|
nspsend:28 50 52 4F 54 4F 43 4F  |(PROTOCO|
nspsend:4C 3D 74 63 70 29 28 48  |L=tcp)(H|
nspsend:4F 53 54 3D 61 70 33 34  |OST=ap34|
nspsend:37 73 75 6E 29 28 50 4F  |7sun)(PO|
nspsend:52 54 3D 31 35 32 31 29  |RT=1521)|
nspsend:29 29 29 00 00 00 00 00  |)))......|
nspsend: normal exit
nscon: exit (0)
```

Pertinent Oracle Net Trace Error Output

When there is a problem a connection, the error code is logged in the trace file. Example 16–5 depicts typical trace file output for a failed SQL*Plus connection to a database server.

Example 16–5 Trace Example

```
[22-JUL-2002 13:34:07:687] nsprecv: entry
[22-JUL-2002 13:34:07:687] nsbal: entry
[22-JUL-2002 13:34:07:687] nsbgetfl: entry
[22-JUL-2002 13:34:07:687] nsbgetfl: normal exit
[22-JUL-2002 13:34:07:687] nsmal: entry
[22-JUL-2002 13:34:07:687] nsmal: 44 bytes at 0x132d90
[22-JUL-2002 13:34:07:687] nsmal: normal exit
[22-JUL-2002 13:34:07:687] nsbal: normal exit
[22-JUL-2002 13:34:07:687] nsprecv: reading from transport...
[22-JUL-2002 13:34:07:687] nttrd: entry
[22-JUL-2002 13:35:09:625] nttrd: exit
[22-JUL-2002 13:35:09:625] ntt2err: entry
[22-JUL-2002 13:35:09:625] ntt2err: Read unexpected EOF ERROR on 10
[22-JUL-2002 13:35:09:625] ntt2err: exit
[22-JUL-2002 13:35:09:625] nsprecv: transport read error
[22-JUL-2002 13:35:09:625] nsprecv: error exit
[22-JUL-2002 13:35:09:625] nserror: entry
[22-JUL-2002 13:35:09:625] nserror: nsres: id=0, op=68, ns=12537, ns2=12560;
nt[0]=507, nt[1]=0, nt[2]=0; ora[0]=0, ora[1]=0, ora[2]=0
[22-JUL-2002 13:35:09:625] nscon: error exit
[22-JUL-2002 13:35:09:625] nsdo: nsctxrnk=0
[22-JUL-2002 13:35:09:625] nsdo: error exit
```

```
[22-JUL-2002 13:35:09:625] nscall: unexpected response
[22-JUL-2002 13:35:09:625] nsclose: entry
[22-JUL-2002 13:35:09:625] nstimarmed: entry
[22-JUL-2002 13:35:09:625] nstimarmed: no timer allocated
[22-JUL-2002 13:35:09:625] nstimarmed: normal exit
[22-JUL-2002 13:35:09:625] nsdo: entry
[22-JUL-2002 13:35:09:625] nsdo: cid=0, opcode=98, *bl=0, *what=0,
uflgs=0x440, cflgs=0x2
[22-JUL-2002 13:35:09:625] nsdo: rank=64, nsctxrnk=0
[22-JUL-2002 13:35:09:625] nsdo: nsctx: state=1, flg=0x4201, mvd=0
[22-JUL-2002 13:35:09:625] nsbfr: entry
[22-JUL-2002 13:35:09:625] nsbaddfl: entry
[22-JUL-2002 13:35:09:625] nsbaddfl: normal exit
[22-JUL-2002 13:35:09:625] nsbfr: normal exit
[22-JUL-2002 13:35:09:625] nsbfr: entry
[22-JUL-2002 13:35:09:625] nsbaddfl: entry
[22-JUL-2002 13:35:09:625] nsbaddfl: normal exit
[22-JUL-2002 13:35:09:625] nsbfr: normal exit
[22-JUL-2002 13:35:09:625] nsdo: nsctxrnk=0
[22-JUL-2002 13:35:09:625] nsdo: normal exit
[22-JUL-2002 13:35:09:625] nsclose: closing transport
[22-JUL-2002 13:35:09:625] nttdisc: entry
[22-JUL-2002 13:35:09:625] nttdisc: Closed socket 10
[22-JUL-2002 13:35:09:625] nttdisc: exit
[22-JUL-2002 13:35:09:625] nsclose: global context check-out (from slot 0)
complete
[22-JUL-2002 13:35:09:703] nsnadisc: entry
[22-JUL-2002 13:35:09:703] nadisc: entry
[22-JUL-2002 13:35:09:703] nacomtm: entry
[22-JUL-2002 13:35:09:703] nacompd: entry
[22-JUL-2002 13:35:09:703] nacompd: exit
[22-JUL-2002 13:35:09:703] nacompd: entry
[22-JUL-2002 13:35:09:703] nacompd: exit
[22-JUL-2002 13:35:09:703] nacomtm: exit
[22-JUL-2002 13:35:09:703] nas_dis: entry
[22-JUL-2002 13:35:09:703] nas_dis: exit
[22-JUL-2002 13:35:09:703] nau_dis: entry
[22-JUL-2002 13:35:09:703] nau_dis: exit
[22-JUL-2002 13:35:09:703] naeetrm: entry
[22-JUL-2002 13:35:09:703] naeetrm: exit
[22-JUL-2002 13:35:09:703] naectrm: entry
[22-JUL-2002 13:35:09:703] naectrm: exit
[22-JUL-2002 13:35:09:703] nagbltrm: entry
[22-JUL-2002 13:35:09:703] nau_gtm: entry
[22-JUL-2002 13:35:09:703] nau_gtm: exit
[22-JUL-2002 13:35:09:703] nagbltrm: exit
[22-JUL-2002 13:35:09:703] nadisc: exit
[22-JUL-2002 13:35:09:703] nsnadisc: normal exit
[22-JUL-2002 13:35:09:703] nsbfr: entry
[22-JUL-2002 13:35:09:703] nsbaddfl: entry
[22-JUL-2002 13:35:09:703] nsbaddfl: normal exit
[22-JUL-2002 13:35:09:703] nsbfr: normal exit
[22-JUL-2002 13:35:09:703] nsmfr: entry
[22-JUL-2002 13:35:09:703] nsmfr: 2256 bytes at 0x130508
[22-JUL-2002 13:35:09:703] nsmfr: normal exit
[22-JUL-2002 13:35:09:703] nsmfr: entry
[22-JUL-2002 13:35:09:703] nsmfr: 484 bytes at 0x1398a8
[22-JUL-2002 13:35:09:703] nsmfr: normal exit
[22-JUL-2002 13:35:09:703] nsclose: normal exit
[22-JUL-2002 13:35:09:703] nscall: connecting...
```

```
[22-JUL-2002 13:35:09:703] nsclose: entry
[22-JUL-2002 13:35:09:703] nsclose: normal exit
[22-JUL-2002 13:35:09:703] nladget: entry
[22-JUL-2002 13:35:09:734] nladget: exit
[22-JUL-2002 13:35:09:734] nsmfr: entry
[22-JUL-2002 13:35:09:734] nsmfr: 144 bytes at 0x132cf8
[22-JUL-2002 13:35:09:734] nsmfr: normal exit
[22-JUL-2002 13:35:09:734] nsmfr: entry
[22-JUL-2002 13:35:09:734] nsmfr: 156 bytes at 0x138e70
[22-JUL-2002 13:35:09:734] nsmfr: normal exit
[22-JUL-2002 13:35:09:734] nladtrm: entry
[22-JUL-2002 13:35:09:734] nladtrm: exit
[22-JUL-2002 13:35:09:734] nscall: error exit
[22-JUL-2002 13:35:09:734] nioqper:  error from nscall
[22-JUL-2002 13:35:09:734] nioqper:    ns main err code: 12537
[22-JUL-2002 13:35:09:734] nioqper:    ns (2)  err code: 12560
[22-JUL-2002 13:35:09:734] nioqper:    nt main err code: 507
[22-JUL-2002 13:35:09:734] nioqper:    nt (2)  err code: 0
[22-JUL-2002 13:35:09:734] nioqper:    nt OS   err code: 0
[22-JUL-2002 13:35:09:734] niomapnserror: entry
[22-JUL-2002 13:35:09:734] niqme: entry
[22-JUL-2002 13:35:09:734] niqme: reporting NS-12537 error as ORA-12537
[22-JUL-2002 13:35:09:734] niqme: exit
[22-JUL-2002 13:35:09:734] niomapnserror: returning error 12537
[22-JUL-2002 13:35:09:734] niomapnserror: exit
[22-JUL-2002 13:35:09:734] niotns: Couldn't connect, returning 12537
[22-JUL-2002 13:35:10:734] niotns: exit
[22-JUL-2002 13:35:10:734] nsbfrfl: entry
[22-JUL-2002 13:35:10:734] nsbrfr: entry
[22-JUL-2002 13:35:10:734] nsbrfr: nsbfs at 0x132d90, data at 0x132dc8.
[22-JUL-2002 13:35:10:734] nsbrfr: normal exit
[22-JUL-2002 13:35:10:734] nsbrfr: entry
[22-JUL-2002 13:35:10:734] nsbrfr: nsbfs at 0x1248d8, data at 0x132210.
[22-JUL-2002 13:35:10:734] nsbrfr: normal exit
[22-JUL-2002 13:35:10:734] nsbrfr: entry
[22-JUL-2002 13:35:10:734] nsbrfr: nsbfs at 0x12d820, data at 0x1319f0.
[22-JUL-2002 13:35:10:734] nsbrfr: normal exit
[22-JUL-2002 13:35:10:734] nsbfrfl: normal exit
[22-JUL-2002 13:35:10:734] nigtrm: Count in the NI global area is now 1
[22-JUL-2002 13:35:10:734] nigtrm: Count in the NL global area is now 1
```

The most efficient way to evaluate error codes is to find the most recent `nserror` entry logged, as the session layer controls the connection. The most important error messages are the ones at the bottom of the file. They are the most recent errors and the source of the problem with the connection.

For information about the specific return codes, use the Oracle UNIX error tool `oerr`, by entering the following at any command line:

```
oerr tns error_number
```

As an example, consider the following `nserror` entry logged in the trace file shown in Example 16–5 on page 16-41:

```
[22-JUL-2002 13:35:09:625] nserror: nsres: id=0, op=68, ns=12537, ns2=12560;
nt[0]=507, nt[1]=0, nt[2]=0; ora[0]=0, ora[1]=0, ora[2]=0
```

Using `oerr`, you can find out more information about return codes 12537 and 507. (Bold denotes user input.)

```
oerr tns 12537
```

```
12537, 00000, "TNS:connection closed"
// *Cause: "End of file" condition has been reached; partner has
disconnected.
// *Action: None needed; this is an information message.

oerr tns 507
00507, 00000, "Connection closed"
// *Cause: Normal "end of file" condition has been reached; partner has
// disconnected.
// *Action: None needed; this is an information message.
```

Using the Trace Assistant to Examine Trace Files

Oracle Net Services provides a tool called the Trace Assistant to help you understand the information provided in trace files by converting existing lines of trace file text into a more readable paragraph. Note that the Trace Assistant runs against only a level `16` (`support`) Oracle Net Services trace file.

> **Note:** The Trace Assistant can only be used when `DIAG_ADR_ENABLED` is set to `off`.

This section contains the following topics:

- Trace Assistant Syntax
- Packet Examples
- Two-Task Common Packet Examples
- Connection Example
- Statistics Example

Trace Assistant Syntax

To run the Trace Assistant, enter the following at any command line prompt:

```
trcasst [options] <filename>
```

The options are described in Table 16–24.

Table 16–24 Trace Assistant Syntax

Option	Description
`-e`*level*	Displays error information. After the `-e`, zero or one error decoding level may follow: ■ `0` or nothing translates the NS error numbers dumped from the `nserror` function plus lists all other errors ■ `1` displays only the NS error translation from the `nserror` function ■ `2` displays error numbers without translation

Table 16–24 (Cont.) Trace Assistant Syntax

Option	Description
`-la`	If a connection ID exists in the NS connect packet, then the output displays the connection IDs. Connection IDs are displayed as hexadecimal, eight-byte IDs. A generated ID is created by Trace Assistant if the packet is not associated with any connection, that is, the connect packet is overwritten in the trace file. This can occur with cyclic trace files. For each ID, the output lists the following: ■ Socket ID, if the connection has one ■ Connect packet send or receive operation ■ Current setting of the `MULTIPLEX` attribute of the `DISPATCHERS` parameter in the initialization parameter file. When `MULTIPLEX` is set to `ON`, **session multiplexing** is enabled. ■ Session ID, if `MULTIPLEX` is set to `ON` ■ Connect data information **Notes:** ■ Do not use this option with other options. ■ The IDs generated by the Trace Assistant do not correlate with client/server trace files.
`-li` *ID*	Displays the trace for a particular ID from the `-la` output **Note:** Only use this option with output from the `-la` option.
`-o`*type*	Displays the amount and type of information to be output. After the `-o` the following options can be used: ■ `c` to display summary connectivity information ■ `d` to display detailed connectivity information ■ `u` to display summary **Two-Task Common (TTC)** information ■ `t` to display detailed TTC information ■ `q` to display SQL commands enhancing summary TTC information. Use this option with `u`, such as `-ouq`. **Note:** As output for `d` contains the same information as displayed for `c`, do not submit both `c` and `d`. If you submit both, then only output `d` will be processed.
`-p`	Oracle internal use only
`-s`	Displays the following statistical information: ■ Total number of bytes sent and received ■ Maximum open cursors ■ Currently open cursors ■ Count and ratio of operations ■ Parsing and execution count for PL/SQL ■ Total calls sent and received ■ Total, average, and maximum number of bytes sent and received ■ Total number of transports and sessions present ■ Timestamp information, if any ■ Sequence numbers, if any

If no options are provided, then the default is `-odt -e0 -s`, providing detailed connectivity and TTC events, error level zero (`0`), and statistics in the trace file.

Example 16–6 shows how the Trace Assistant converts trace file information into a more readable format.

Example 16–6 Trace File with Error

```
ntus2err: exit
ntuscni: exit
ntusconn: exit
nserror: entry
-<ERROR>- nserror: nsres: id=0, op=65, ns=12541, ns2=12560; nt[0]=511, nt[1]=2,
nt[2]=0
```

Example 16–7 shows how the Trace Assistant converts the trace file information into a more readable format with the `-e1` option.

Example 16–7 trcasst -e1 Output

```
    ********************************************************************************
    *                         Trace Assistant                                      *
    ********************************************************************************

ntus2err: exit
ntuscni: exit
ntusconn: exit
nserror: entry
-<ERROR>- nserror: nsres: id=0, op=65, ns=12541, ns2=12560; nt[0]=511, nt[1]=2,
nt[2]=0
////////////////////////////////////////////////////////////////
Error found. Error Stack follows:
             id:0
  Operation code:65
      NS Error 1:12541
      NS Error 2:12560
NT Generic Error:511
  Protocol Error:2
       OS Error:0
 NS & NT Errors Translation
12541, 00000 "TNS:no listener"
 // *Cause: The connection request could not be completed because the listener
 // is not running.
 // *Action: Ensure that the supplied destination address matches one of
 // the addresses used by the listener - compare the TNSNAMES.ORA entry with
 // the appropriate LISTENER.ORA file (or TNSNAV.ORA if the connection is to
 // go by way of an Interchange). Start the listener on the remote machine.
/
12560, 00000 "TNS:protocol adapter error"
 // *Cause: A generic protocol adapter error occurred.
 // *Action: Check addresses used for proper protocol specification. Before
 // reporting this error, look at the error stack and check for lower level
 // transport errors.For further details, turn on tracing and reexecute the
 // operation. Turn off tracing when the operation is complete.
/
00511, 00000 "No listener"
 // *Cause: The connect request could not be completed because no application
 // is listening on the address specified, or the application is unable to
 // service the connect request in a sufficiently timely manner.
 // *Action: Ensure that the supplied destination address matches one of
 // the addresses used by the listener - compare the TNSNAMES.ORA entry with
 // appropriate LISTENER.ORA file (or TNSNAV.ORA if the connection is to go
```

```
 // by way of an Interchange. Start the listener on the remote machine.
/
////////////////////////////////////////////////////////////////
    **********************************************************************
    *                    Trace Assistant has completed                   *
    **********************************************************************
```

However, other errors may also exist within the trace file that were not logged from the `nserror` function.

Packet Examples

Trace Assistant also enables you to view data packets from both the Oracle Net and TTC communication layers. Trace Assistant offers you two options to view these packets:

- Summary connectivity (using option `-oc`)
- Detailed connectivity (using option `-od`)

Example: Summary Data Packets Sent in a Connection

Example 16–8 shows summary information from the `-oc` option. The output shows....

Example 16–8 trcasst -oc Output

```
    **********************************************************************
    *                         Trace Assistant                            *
    **********************************************************************

---> Send 198 bytes - Connect packet
Connect data length: 140
Connect Data:
    (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521))
    (CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)(CID=(PROGRAM=)
    (HOST=sales-server)(USER=joe))))

<--- Received 76 bytes - Redirect packet
Redirect data length: 66
Redirect Data:
    (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))

---> Send 198 bytes - Connect packet
Connect data length: 140
Connect Data:
    (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521))
    (CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)(CID=(PROGRAM=)
    (HOST=sales-server)(USER=joe))))

<--- Received 32 bytes - Accept packet
Connect data length: 0
---> Send 153 bytes - Data packet
        Native Services negotiation packet

<--- Received 127 bytes - Data packet
        Native Services negotiation packet

---> Send 32 bytes - Data packet

<--- Received 140 bytes - Data packet
```

```
    ************************************************************************
    *                    Trace Assistant has completed                     *
    ************************************************************************
```

Note that the packets being sent or received have a prefix of "`--->  Send` *nnn* `bytes`" or "`<---  Received` *nnn* `bytes`" showing that this node is sending or receiving a packet of a certain type and with *nnn* number of bytes. This prefix enables you to determine if the node is the client or the database server. The connection request is always sent by the client, but received by the database server (or listener).

Example: Detailed Data Packets Sent in a Connection

Example 16–9 shows detailed information from the `-od` option. The output shows all of the details sent along with the connect data in negotiating a connection.

Example 16–9 trcasst -od Output

```
    ************************************************************************
    *                         Trace Assistant                              *
    ************************************************************************
--->  Send   241 bytes - Connect packet
Current NS version number is: 311.
Lowest NS version number can accommodate is: 300.
Global options for the connection:
      can receive attention
      no attention processing
      Don't care
      Maximum SDU size:8192
      Maximum TDU size:32767
      NT protocol characteristics:
            Test for more data
            Test operation
            Full duplex I/O
            Urgent data support
            Generate SIGURG signal
            Generate SIGPIPE signal
            Generate SIGIO signal
            Handoff connection to another
      Line turnaround value :0
      Connect data length :183
      Connect data offset :58
      Connect data maximum size :512
            Native Services wanted
            NAU doing O3LOGON - DH key foldedin
            Native Services wanted
            NAU doing O3LOGON - DH key foldedin
      Cross facility item 1: 0
      Cross facility item 2: 0
      Connection id : Ox000059F70000004C
(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)(SRVR=SHARED)(CID=(PROGRAM=)
(HOST=sales-server)(USER=joe))))

<--- Received 76 bytes - Redirect packet
     Redirect data length: 66
(ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))

--->  Send   241 bytes - Connect packet
Current NS version number is: 311.
Lowest NS version number can accommodate is: 300.
```

```
Global options for the connection:
      can receive attention
      no attention processing
      Don't care
      Maximum SDU size:8192
      Maximum TDU size:32767
      NT protocol characteristics:
      Test for more data
      Test operation
      Full duplex I/O
      Urgent data support
      Generate SIGURG signal
      Generate SIGPIPE signal
      Generate SIGIO signal
      Handoff connection to another
Line turnaround value :0
Connect data length :183
Connect data offset :58
Connect data maximum size :512
      Native Services wanted
      NAU doing O3LOGON - DH key foldedin
      Native Services wanted
      NAU doing O3LOGON - DH key foldedin
Cross facility item 1: 0
Cross facility item 2: 0
Connection id : 0x000059F70000007A
(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=sales.us.acme.com)(SRVR=SHARED)(CID=(PROGRAM=)
(HOST=sales-server)(USER=joe))))
<--- Received 32 bytes - Accept packet
      Accepted NS version number is: 310.
Global options for the connection:
      no attention processing
      Don't care
      Accepted maximum SDU size: 8192
      Accepted maximum TDU size: 32767
      Connect data length: 0
            Native Services wanted
            NAU doing O3LOGON - DH key foldedin
            Native Services wanted
            NAU doing O3LOGON - DH key foldedin

--->  Send   153 bytes - Data packet
      Native Services negotiation packet version#: 150999040
            Service data packet #0 for Supervisor has 3 subpackets
               Subpacket #0:  Version #150999040
               Subpacket #1: 0000000000000000
               Subpacket #2: DEADBEEF0003000000040004000100010002
            Service data packet #1 for Authentication has 3 subpackets
               Subpacket #0:  Version #150999040
               Subpacket #1: UB2: 57569
               Subpacket #2: FCFF
            Service data packet #2 for Encryption has 2 subpackets
               Subpacket #0:  Version #150999040
               Subpacket #1: 000000000000000000
            Service data packet #3 for Data Integrity has 2 subpackets
               Subpacket #0:  Version #150999040
               Subpacket #1: 000000

<--- Received 127 bytes - Data packet
```

```
     Native Services negotiation packet version#: 135290880
            Service data packet #0 for Supervisor has 3 subpackets
               Subpacket #0:  Version #135290880
               Subpacket #1: 0000
               Subpacket #2: DEADBEEF00030000000200040001
            Service data packet #1 for Authentication has 2 subpackets
               Subpacket #0:  Version #135290880
               Subpacket #1: FBFF
            Service data packet #2 for Encryption has 2 subpackets
               Subpacket #0:  Version #135290880
               Subpacket #1: UB1: 0
            Service data packet #3 for Data Integrity has 2 subpackets
               Subpacket #0:  Version #135290880
               Subpacket #1: UB1: 0
....

--->  Send   11 bytes - Marker packet
      One data byte.
      Hex character sent over to the server: 2

<--- Received 11 bytes - Marker packet
     One data byte.
     Hex character sent over to the server: 2

<--- Received 155 bytes - Data packet

--->  Send   25 bytes - Data packet

<--- Received 11 bytes - Data packet

--->  Send   13 bytes - Data packet

<--- Received 11 bytes - Data packet

--->  Send   10 bytes - Data packet
      Data Packet flags:
      End of file
    ********************************************************************
    *                   Trace Assistant has completed                  *
    ********************************************************************
```

Two-Task Common Packet Examples

TTC handles requests such as open cursor, select rows, and update rows that are directed to the database server. All requests are answered by the database server. If you request to logon, a response is returned from the database server that the request was completed.

Example: Two-Task Common Summary Information Summary information for TTC from the `-ou` option is different from other displays in that it shows two packets on each line, rather than one. This is done to mirror the request/response pairings process by which TTC operates.

Example 16–10 shows all of the details sent along with the connect data in negotiating a connection.

Example 16–10 trcasst -ou Output

```
    ********************************************************************
    *                       Trace Assistant                            *
```

```
    ***************************************************************************

                                                                      Bytes  Bytes
                                                                       Sent   Rcvd

Send operation(TTIPRO)                                                  32    140
Send operation(TTIDTY)                                                  33     22
Get the session key (OSESSKEY)                                         229    145
Generic authentication call (OAUTH)                                    368   1001
Send operation(TTIPFN)                                                  44    144
Send operation(TTIPFN)                                                  36     16
Parse a statement (OSQL)                # 1  SELECT USER FROM ...       47    100
Fast upi calls to opial7 (OALL7)        # 1                           130    111
Fetch row (OFETCH)                      # 1                            21    137
Close cursor (OCLOSE)                   # 1                            17     11
New v8 bundled call (OALL8)             # 0  !Keep Parse  BEGI...      156    145
Send operation(TTIPFN)                                                  51     16
Parse a statement (OSQL)                # 1  SELECT ATTRIBUTE,...      186    100
Fast upi calls to opial7 (OALL7)        # 1                           246    111
Fetch row (OFETCH)                      # 1                            21    126
Close cursor (OCLOSE)                   # 1                            17     11
Send operation(TTIPFN)                                                  36     16
Parse a statement (OSQL)                # 1  SELECT CHAR_VALUE...      208    100
Fast upi calls to opial7 (OALL7)        # 1                           130    111
Fetch row (OFETCH)                      # 1                            21    126
Close cursor (OCLOSE)                   # 1                            17     11
Send operation(TTIPFN)                                                  36     16
Fast upi calls to opial7 (OALL7)        # 1  !Keep Parse  BEGI...      183     41
Send operation(TTIRXD)                                                  20    111
Close cursor (OCLOSE)                   # 1                            17     11
New v8 bundled call (OALL8)             # 0  Parse Fetch  SELE...      165    278
Send operation(TTIPFN)                                                  51     16
Parse a statement (OSQL)                # 1  commit                    31    100
Execute statement (OEXEC)               # 1  number of rows: 1         25    100
Close cursor (OCLOSE)                   # 1                            17     11
Send operation(TTIPFN)                                                  36     16
Fast upi calls to opial7 (OALL7)        # 1  !Keep Parse  BEGI...      183     41
Send operation(TTIRXD)                                                  60    111
Close cursor (OCLOSE)                   # 1                            17     11
Send operation(TTIPFN)                                                  36     16
Fast upi calls to opial7 (OALL7)        # 1  !Keep Parse  BEGI...      183     41
Send operation(TTIRXD)                                                  20    111
Close cursor (OCLOSE)                   # 1                            17     11
New v8 bundled call (OALL8)             # 0  Parse Fetch  sele...      144    383
New v8 bundled call (OALL8)             # 1  !Keep Fetch               121    315
Logoff off of Oracle (OLOGOFF)                                          13     11

    ***************************************************************************
    *                    Trace Assistant has completed                        *
    ***************************************************************************
```

Output is displayed in the following format:

description TTC_message cursor_number SQL_statement bytes_sent bytes_received

On each line of the output, the first item displayed is the actual request made. The second item shows on what cursor that operation has performed. The third item is either a listing of the SQL command or flag that is being answered. The number of bytes sent and received are displayed at the far right. A flag can be one of the following:

```
!PL/SQL = Not a PL/SQL request
COM = Commit
IOV = Get I/O Vector
DEFN = Define
EXEC = Execute
FETCH = Fetch
CAN = Cancel
DESCSEL = Describe select
DESCBND = Describe Bind
BND = Bind
PARSE = Parse
EXACT = Exact
```

Example: Detailed SQL Information on Top of Summary Two-Task

Example 16–11 shows detailed SQL information from the `-ouq` option.

Example 16–11 trcasst -ouq Output

```
    ********************************************************************************
    *                          Trace Assistant                                     *
    ********************************************************************************
                                                                          Bytes  Bytes
                                                                           Sent   Rcvd

Send operation(TTIPRO)                                                      32    140
Send operation(TTIDTY)                                                      33     22
Get the session key (OSESSKEY)                                             229    145
Generic authentication call (OAUTH)                                        368   1001
Send operation(TTIPFN)                                                      44    144
Send operation(TTIPFN)                                                      36     16
Parse a statement (OSQL)                  # 1                             47    100
          SELECT USER FROM DUAL

Fast upi calls to opial7 (OALL7)          # 1                            130    111
Fetch row (OFETCH)                        # 1                             21    137
Close cursor (OCLOSE)                     # 1                             17     11
New v8 bundled call (OALL8)               # 0  !Keep Parse                156    145
          BEGIN DBMS_OUTPUT.DISABLE; END;

Send operation(TTIPFN)                                                      51     16
Parse a statement (OSQL)                  # 1                            186    100
          SELECT ATTRIBUTE,SCOPE,NUMERIC_VALUE,CHAR_VALUE,DA
          TE_VALUE FROM SYSTEM.PRODUCT_PRIVS WHERE (UPPER('S
          QL*Plus') LIKE UPPER(PRODUCT)) AND (UPPER(USER) LI
          KE USERID)

Fast upi calls to opial7 (OALL7)          # 1                            246    111
Fetch row (OFETCH)                        # 1                             21    126
Close cursor (OCLOSE)                     # 1                             17     11
Send operation(TTIPFN)                                                      36     16
Parse a statement (OSQL)                  # 1                            208    100
          SELECT CHAR_VALUE FROM SYSTEM.PRODUCT_PRIVS WHERE
          (UPPER('SQL*Plus') LIKE UPPER(PRODUCT)) AND ((UPPE
          R(USER) LIKE USERID) OR (USERID = 'PUBLIC')) AND (
          UPPER(ATTRIBUTE) = 'ROLES')

Fast upi calls to opial7 (OALL7)          # 1                            130    111
Fetch row (OFETCH)                        # 1                             21    126
Close cursor (OCLOSE)                     # 1                             17     11
```

```
Send operation(TTIPFN)                                              36     16
Fast upi calls to opial7 (OALL7)         # 1  !Keep Parse          183     41
          BEGIN DBMS_APPLICATION_INFO.SET_MODULE(:1,NULL); E
          ND;

Send operation(TTIRXD)                                              20    111
Close cursor (OCLOSE)                    # 1                        17     11
New v8 bundled call (OALL8)              # 0  Parse Fetch          165    278
          SELECT DECODE('A','A','1','2') FROM DUAL

Send operation(TTIPFN)                                              51     16
Parse a statement (OSQL)                 # 1                        31    100
          commit

Execute statement (OEXEC)                # 1  number of rows: 1     25    100
Close cursor (OCLOSE)                    # 1                        17     11
Send operation(TTIPFN)                                              36     16
Fast upi calls to opial7 (OALL7)         # 1  !Keep Parse          183     41
          BEGIN DBMS_APPLICATION_INFO.SET_MODULE(:1,NULL); E
          ND;

Send operation(TTIRXD)                                              60    111
Close cursor (OCLOSE)                    # 1                        17     11
Send operation(TTIPFN)                                              36     16
Fast upi calls to opial7 (OALL7)         # 1  !Keep Parse          183     41
          BEGIN DBMS_APPLICATION_INFO.SET_MODULE(:1,NULL); E
          ND;

Send operation(TTIRXD)                                              20    111
Close cursor (OCLOSE)                    # 1                        17     11
New v8 bundled call (OALL8)              # 0  Parse Fetch          144    383
          select * from dept

New v8 bundled call (OALL8)              # 1  !Keep Fetch          121    315
Logoff off of Oracle (OLOGOFF)                                      13     11

    ************************************************************************
    *                   Trace Assistant has completed                      *
    ************************************************************************
```

Example: Two-Task Common Detailed Summary Information Example 16–12 shows detailed TTC information from the `-ot` option.

Example 16–12 trcasst -ot Output

```
    ************************************************************************
    *                    Trace  Assistant                                  *
    ************************************************************************

Set protocol (TTIPRO)
      Operation 01 (con) Send protocol version=6
      Originating platform: SVR4-be-8.1.0

Set protocol (TTIPRO)
      Operation 01 (con) Receive protocol version=6
      Destination platform: SVR4-be-8.1.0

Set datatypes (TTIDTY)
```

```
Set datatypes (TTIDTY)

Start of user function (TTIFUN)
      (OSESSKEY)

Return opi parameter (TTIRPA)

Start of user function (TTIFUN)
      (OAUTH)

Return opi parameter (TTIRPA)

Start of user function (TTIFUN)
      session operations 71 (O71SESOPN) (switch session)

Return opi parameter (TTIRPA)

Start of user function (TTIFUN)
      Get Oracle version/date string in new format (OVERSION)

Return opi parameter (TTIRPA)
Oracle Enterprise Edition Release 10.1.0.2.0
With the Partitioning option
JServer Release 10.1.0.2.0

Start of user function (TTIFUN)
      session operations 71 (O71SESOPN) (switch session)

Return opi parameter (TTIRPA)

Start of user function (TTIFUN)
      Open a cursor (OOPEN)

Return opi parameter (TTIRPA)
      Cursor #: 1

Start of user function (TTIFUN)
      Parse a statement (OSQL) Cursor # 1
SELECT USER FROM DUAL
    ************************************************************************
    *                    Trace Assistant has completed                     *
    ************************************************************************
```

Connection Example

Example 16–13 shows output from the `-la` option. The output shows the following information:

- Connect IDs received
- Socket ID on which this connection has come
- Operation

  `Receive` identifies the trace as a database server trace; `Send` identifies the trace as a client trace. In this output, `Receive` is the operation.
- `MULTIPLEX` attribute of the `DISPATCHERS` parameter is set to `ON`
- 32-bit session ID
- Connect data information received

Example 16–13 trcasst -la Output

```
    ************************************************************************
    *                          Trace Assistant                             *
    ************************************************************************

Connection ID: 00000B270000000B
        Socket Id: 15
        Operation: Receive
        Multiplex: ON
        Session Id: 8362785DE4FC0B19E034080020F793E1
        Connect Data:
        (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521))
        (CONNECT_DATA=(SERVER=shared)
        (SERVICE_NAME=sales.us.acme.com)(CID=(PROGRAM=)(HOST=sales-server)
        (USER=oracle))))
Connection ID: 00000B240000000B
        Socket Id: 15
        Operation: Receive
        Multiplex: ON
        Session Id: 8362785DE4FB0B19E034080020F793E1
        Connect Data:
        (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521))
        (CONNECT_DATA=(SERVER=shared)
        (SERVICE_NAME=sales.us.acme.com)(CID=(PROGRAM=)(HOST=sales-server)
        (USER=oracle))))
Connection ID: 00000B1F00000008
        Socket Id: 15
        Operation: Receive
        Multiplex: ON
        Session Id: 8362785DE4F90B19E034080020F793E1
        Connect Data:
        (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521))
        (CONNECT_DATA=(SERVER=shared)
        (SERVICE_NAME=sales.us.acme.com)(CID=(PROGRAM=)(HOST=sales-server)
        (USER=oracle))))
    ************************************************************************
    *                    Trace Assistant has completed                     *
    ************************************************************************
```

Example 16–14 shows output for connection ID `00000B1F00000008` from the `-li 00000B1F00000008` option.

Example 16–14 trcasst -li Output

```
    ************************************************************************
    *                              Trace Assistant                         *
    ************************************************************************
<--- Received 246 bytes - Connect packet
Current NS version number is: 310.
Lowest NS version number can accommodate is: 300.
Global options for the connection:
        Can receive attention
        No attention processing
        Don't care
        Maximum SDU size: 8192
        Maximum TDU size: 32767
        NT protocol characteristics:
                Test for more data
                Test operation
                Full duplex I/O
```

```
                Urgent data support
                Generate SIGURG signal
                Generate SIGPIPE signal
                Generate SIGIO signal
                Handoff connection to another
        Line turnaround value: 0
        Connect data length: 188
        Connect data offset: 58
        Connect data maximum size: 512
                Native Services wanted
                NAU doing O3LOGON - DH key foldedin
                Native Services wanted
                NAU doing O3LOGON - DH key foldedin
        Cross facility item 1: 0
        Cross facility item 2: 0
        Connection id: 0x00000B1F00000008
    (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=sales-server)(PORT=1521))
    (CONNECT_DATA=(SERVER=shared)(SERVICE_NAME=sales.us.acme.com)
    (CID=(PROGRAM=)(HOST=sales-server)(USER=oracle))))

---> Send 114 bytes - Accept packet
Accepted NS version number is: 310.
Global options for the connection:
        No attention processing
        Don't care
        Accepted maximum SDU size: 8192
        Accepted maximum TDU size: 32767
        Connect data length: 0
                Native Services wanted
                NAU doing O3LOGON - DH key foldedin
                Native Services wanted
                NAU doing O3LOGON - DH key foldedin
        Connection Time out: 1000
        Tick Size: 100
        Reconnect Data: (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=34454))
        Session Id: 8362785DE4F90B19E034080020F793E1
<--- Received 164 bytes - Data packet
        Native Services negotiation packet version#: 135290880
                 Service data packet #0 for Supervisor has 3 subpackets
                        Subpacket #0:  Version #135290880
                        Subpacket #1: 0000000000000000
                        Subpacket #2: DEADBEEF00030000000040004000100010002
                 Service data packet #1 for Authentication has 3 subpackets
                        Subpacket #0:  Version #135290880
                        Subpacket #1: UB2: 57569
                        Subpacket #2: FCFF
                 Service data packet #2 for Encryption has 2 subpackets
                        Subpacket #0:  Version #135290880
                        Subpacket #1: 0000000000
                 Service data packet #3 for Data Integrity has 2 subpackets
                        Subpacket #0:  Version #135290880
                        Subpacket #1: 0000
---> Send 143 bytes - Data packet
        Native Services negotiation packet version#: 135290880
                 Service data packet #0 for Supervisor has 3 subpackets
                        Subpacket #0:  Version #135290880
                        Subpacket #1: 0000
                        Subpacket #2: DEADBEEF00030000000200040001
                 Service data packet #1 for Authentication has 2 subpackets
                        Subpacket #0:  Version #135290880
```

```
                        Subpacket #1: FBFF
                Service data packet #2 for Encryption has 2 subpackets
                        Subpacket #0:  Version #135290880
                        Subpacket #1: UB1: 0
                Service data packet #3 for Data Integrity has 2 subpackets
                        Subpacket #0:  Version #135290880
                        Subpacket #1: UB1: 0
<--- Received 48 bytes - Data packet
Set protocol (TTIPRO)
        Operation 01 (con) Receive protocol version=6
        Destination platform: SVR4-be-8.1.0
---> Send 156 bytes - Data packet
Set protocol (TTIPRO)
        Operation 01 (con) Send protocol version=6
        Originating platform: SVR4-be-8.1.0
<--- Received 49 bytes - Data packet
Set datatypes (TTIDTY)
---> Send 38 bytes - Data packet
Set datatypes (TTIDTY)
<--- Received 245 bytes - Data packet
Start of user function (TTIFUN)
        Get the session key (OSESSKEY)
---> Send 161 bytes - Data packet
Return opi parameter (TTIRPA)
...
    ************************************************************************
    *                    Trace Assistant has completed                     *
    ************************************************************************
```

Statistics Example

The type of statistics gathered is approximately on how many TTC calls, packets and bytes were sent and received between the network partners. Example 16–15 shows typical trace file statistics from the `-s` option.

Example 16–15 trcasst -s Output

```
    ************************************************************************
    *                              Trace Assistant                         *
    ************************************************************************
----------------------
Trace File Statistics:
----------------------
Total number of Sessions: 3

DATABASE:
  Operation Count:    0 OPENS,    21 PARSES,    21 EXECUTES,     9 FETCHES
    Parse Counts:
       9 PL/SQL,     9 SELECT,      0 INSERT,     0 UPDATE,     0 DELETE,
       0 LOCK,       3 TRANSACT,    0 DEFINE,     0 SECURE,     0 OTHER
    Execute counts with SQL data:
       9 PL/SQL,     0 SELECT,      0 INSERT,     0 UPDATE,     0 DELETE,
       0 LOCK,       0 TRANSACT,    0 DEFINE,     0 SECURE,     0 OTHER

  Packet Ratio: 6.142857142857143 packets sent per operation
  Currently opened Cursors: 0
  Maximum opened Cursors  : 0

ORACLE NET SERVICES:
  Total Calls  :       129 sent,        132 received,          83 oci
```

```
Total Bytes  :     15796 sent,      13551 received
  Average Bytes:       122 sent per packet,        102 received per packet
  Maximum Bytes:      1018 sent,        384 received

Grand Total Packets:    129  sent,     132 received

  ************************************************************************
  *                    Trace Assistant has completed                     *
  ************************************************************************
```

Contacting Oracle Support Services

If you are still unable to resolve your problems, or if you are requested to contact Oracle Support Services to report the error, please have the following information at hand:

- The hardware and operating system release number on which the application is running
- The up-to-five-digit release number of all the Oracle networking products involved in the current problem
- The third-party vendor and version you are using
- If you encountered one or more error codes or messages, the exact code numbers and message texts in the order they appeared
- The kind of links that exist between the client and server
- A description of what does work
- The exact error message, if there is one
- An Net8 Services trace, if possible; if not, the log file is sufficient

Glossary

access control list (ACL)

The group of access directives that you define. The directives grant levels of access to specific data for specific clients or groups of clients.

ACL

See **access control list (ACL)**.

access control

A feature of Oracle Connection Manager that sets rules for denying or allowing certain clients to access designated servers.

address

See **protocol address**.

ADR

See **automatic diagnostic repository**.

alias

An alternative name for a network object in an Oracle Names server. An alias stores the name of the object is referencing. When a client requests a lookup of an alias, Oracle completes the lookup as if it is the referenced object.

application gateway

A host computer that runs the **Oracle Net Firewall Proxy**. An application gateway looks and acts like a real server from the client's point of view, and a real client from the server's point of view. An application gateway sits between the Internet and company's internal network and provides middleman services (or proxy services) to users on either side.

ASCII character set

American Standard Code for Information Interchange character set, a convention for representing alphanumeric information using digital data. The collation sequence used by most computers with the exception of IBM and IBM-compatible computers.

attribute

A piece of information that describes some aspect of a directory entry. An entry comprises a set of attributes, each of which belongs to an **object class**. Moreover, each attribute has both a type—which describes the kind of information in the attribute—and a value—which contains the actual data.

authentication method

A security method that enables you to have high confidence in the identity of users, clients, and servers in distributed environments. Network authentication methods can also provide the benefit of single sign-on for users. The following authentication methods are supported in Oracle9*i*, depending on whether or not **Oracle Advanced Security** is installed:

- RADIUS
- Kerberos
- **SSL**
- **Windows NT native authentication**

automatic diagnostic repository

The automatic diagnostic repository (ADR) is a system-wide tracing and logging central repository. The repository is a file-based hierarchical datastore for depositing diagnostic information, including network tracing and logging information.

cache

Memory that stores recently-accessed data to so that subsequent requests to access the same data can be processed quickly.

CDS

See **Cell Directory Services (CDS)**.

Cell Directory Services (CDS)

An **external naming** method that enables users to use Oracle tools transparently and applications to access Oracle databases in a Distributed Computing Environment (DCE) environment.

client

A user, software application, or computer that requests the services, data, or processing of another application or computer. The client is the user process. In a network environment, the client is the local user process and the server may be local or remote.

client load balancing

Load balancing, whereby if more than one listener services a single database, a client can randomly choose between the listeners for its connect requests. This randomization enables all listeners to share the burden of servicing incoming connect requests.

client profile

The properties of a client, which may include the preferred order of **naming method**s, client and server **logging** and **tracing**, the domain from which to request names, and other client options for **Oracle Advanced Security**.

client/server architecture

Software architecture based on a separation of processing between two CPUs. One CPU acts as the client in the transaction, requesting and receiving services. The other acts as the server that provides the requests.

cman.ora file

A configuration file that specifies protocol addresses for incoming requests and administrative commands, as well as Oracle Connection Manager parameters and **access control** rules.

CMADMIN (Connection Manager Administration)

An **Oracle Connection Manager** process that monitors the health of the listener and Oracle Connection Manager gateway processes, shutting down and starting processes as needed. `CMADMIN` registers information about gateway processes with the listener and processes commands executed with the Oracle Connection Manager Control utility.

CMGW (Connection Manager gateway)

An **Oracle Connection Manager** process that receives client connections screened and forwarded by the listener located at the Oracle Connection Manager instance. The gateway process forwards the requests to the database server. In addition, it can multiplex or funnel multiple client connections through a single protocol connection.

connect data

A portion of the **connect descriptor** that defines the destination database **service name** or **Oracle System Identifier (SID)**. In the following example, `SERVICE_NAME` defines a database service called `sales.us.acme.com`:

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521)
  (CONNECT_DATA=
     (SERVICE_NAME=sales.us.acme.com)))
```

connect descriptor

A specially formatted description of the destination for a network connection. A connect descriptor contains destination service and network route information.

The destination service is indicated by using its **service name** for Oracle9*i* or Oracle8*i* databases or its **Oracle System Identifier (SID)**for Oracle release 8.0 databases. The network route provides, at a minimum, the location of the listener through use of a network address.

connect identifier

A **connect descriptor** or a name that maps to a connect descriptor. A connect identifier can be a **net service name**, database **service name**, or **net service alias**. Users initiate a connect request by passing a username and password along with a connect identifier in a connect string for the service to which they wish to connect:

`CONNECT` *`username/password@connect_identifier`*

connect string

Information the user passes to a service to connect, such as username, password, and **connect identifier**:

`CONNECT` *`username/password@net_service_name`*

connect-time failover

A client connect request is forwarded to a another listener if a listener is not responding. Connect-time failover is enabled by **service registration**, because the listener knows if an instance is running to attempting a connection.

connection

An interaction between two processes on a network. Connections are originated by an initiator (client) that requests a connection with a destination (server).

connection load balancing

Load balancing, whereby the number of active connections among various instances and dispatchers for the same service are balanced. This enables listeners to make their routing decisions based on how many connections each dispatcher has and on how loaded the nodes that the instances run.

connection pooling

A resource utilization and user scalability feature that enables you to maximize the number of sessions over a limited number of protocol connections to a **shared server**.

connection request

A notification sent by an initiator and received by a listener that indicates that the initiator wants to start a connection.

data packet

See **packet**.

database administrator (DBA)

(1) A person responsible for operating and maintaining an Oracle Server or a database application. (2) An Oracle username that has been given DBA privileges and can perform database administration functions. Usually the two meanings coincide. Many sites have multiple DBAs.

Database Configuration Assistant

A tool that enables you to create, delete, and modify a database.

database link

A pointer that defines a one-way communication path from an Oracle database server to another database server. The link pointer is actually defined as an entry in a data dictionary table. To access the link, you must be connected to the local database that contains the data dictionary entry.

A database link connection is one-way in the sense that a client connected to local database A can use a link stored in database A to access information in remote database B, but users connected to database B cannot use the same link to access data in database A. If local users on database B want to access data on database A, then they must define a link that is stored in the data dictionary of database B.

The following database links types are supported:

- A **private database link** in a specific schema of a database. Only the owner of a private database link can use it.
- A **public database link** for a database. All users in the database can use it.

dedicated connection

A dedicated server together with a database session.

dedicated server

A server process that is dedicated to one client connection. Contrast with **shared server process**.

default domain

The **domain** within which most client requests take place. It could be the domain where the client resides, or it could be a domain from which the client requests network services often. Default domain is also the client configuration parameter that determines what domain should be appended to unqualified network name requests. A name request is unqualified if it does not have a dot (`.`) character within it.

directory information tree (DIT)

A hierarchical tree-like structure in a **directory server** of the **distinguished names (DNs)** of the entries.

directory naming

A **naming method** that resolves a database service, **net service name**, or **net service alias** to a **connect descriptor** stored in a central directory server. A **directory server** provides central administration of directory naming objects, reducing the work effort associated with adding or relocating services.

directory server

A directory server that is accessed with the **Lightweight Directory Access Protocol (LDAP)**. Support of LDAP-compliant directory servers provides a centralized vehicle for managing and configuring a distributed Oracle network. The directory server can replace client-side and server-side localized `tnsnames.ora` files.

dispatcher

A process that enables many clients to connect to the same server without the need for a dedicated server process for each client. A dispatcher handles and directs multiple incoming network session requests to shared server processes. See also **shared server**.

distinguished name (DN)

Name of entry in a **directory server**. The DN specifies where the entry resides in the LDAP directory hierarchy, much the way a directory path specifies the exact location of a file.

distributed processing

Division of front-end and back-end processing to different computers. Oracle Network Services support distributed processing by transparently connecting applications to remote databases.

domain

Any tree or subtree within the **Domain Name System (DNS)** namespace. Domain most commonly refers to a group of computers whose host names share a common suffix, the domain name.

domain hint

A `NAMES.DOMAIN_HINTS` parameter in the `names.ora` file that contains the name of the domain and at least one address of an Oracle server in that domain. This enables an Oracle server to forward the client requests to a specific address, reducing network traffic.

Domain Name System (DNS)

A system for naming computers and network services that is organized into a hierarchy of **domain**s. DNS is used in TCP/IP networks to locate computers through

user-friendly names. DNS resolves a friendly name into an **IP address**, which is understood by computers.

For Oracle Network Services, DNS translates the host name in a TCP/IP address into an IP address.

DNS

Domain Name System. See **Domain Name System (DNS)**.

enterprise role

An enterprise role is analogous to a regular database role, except that it spans authorization on multiple databases. An enterprise role is a category of roles that define privileges on a particular database. An enterprise role is created the database administrator of a particular database. An enterprise role can be granted to or revoked to one or more enterprise users. The information for granting and revoking these roles is stored in the directory server.

enterprise user

A user that has a unique identity across an enterprise. Enterprise users connect to individual databases through a schema. Enterprise users are assigned enterprise roles that determine their access privileges on databases.

entry

The building block of a directory server, it contains information about an object of interest to directory users.

external naming

A **naming method** that uses a third-party naming service, such as **NIS** or **CDS**.

external procedure

Function or procedure written in a third-generation language (3GL) that can be called from PL/SQL code. Only C is supported for external procedures.

failover

See **connect-time failover**.

firewall support

See **access control**.

foreign domains

The set of domains not managed within a given administrative region. Domains are foreign only in relation to a region; they are not foreign in any absolute sense. A network administrator typically defines foreign domains relative to a particular region to optimize caching performance.

FTP protocol

File Transfer Protocol. A client/server protocol which allows a user on one computer to transfer files to and from another computer over a TCP/IP network.

global database name

The full name of the database which uniquely identifies it from any other database. The global database name is of the form "*database_name*.*database_domain*," for example, `sales.us.acme.com`.

The database name portion, `sales`, is a simple name you wish to call your database. The database domain portion, `us.acme.com`, specifies the database domain in which the database is located, making the global database name unique. When possible, Oracle Corporation recommends that your database domain mirror the network domain.

The global database name is the default service name of the database, as specified by the `SERVICE_NAMES` parameter in the initialization parameter file.

Heterogeneous Services

An integrated component that provides the generic technology for accessing non-Oracle systems from the Oracle database server. Heterogeneous Services enables you to:

- Use Oracle SQL to transparently access data stored in non-Oracle systems as if the data resides within an Oracle server.
- Use Oracle procedure calls to transparently access non-Oracle systems, services, or application programming interfaces (APIs), from your Oracle distributed environment.

hierarchical naming model

An infrastructure in which names are divided into multiple hierarchically-related domains. For Oracle Names, hierarchical naming model can be used with either central or delegated administration.

host naming

A **naming method** resolution that enables users in a TCP/IP environment to resolve names through their existing name resolution service. This name resolution service might be **Domain Name System (DNS)**, **Network Information Service (NIS)**, or simply a centrally-maintained set of `/etc/hosts` files. Host Naming enables users to connect to an Oracle database server by simply providing the server computer's host name or host name alias. No client configuration is required to take advantage of this feature. This method is recommended for simple TCP/IP environments.

HTTP protocol

Hypertext Transfer Protocol. A protocol that provides the language that enables Web browsers and application Web servers to communicate.

identity management realm

A collection of identities, all of which are governed by the same administrative policies. In an enterprise, all employees having access to the intranet may belong to one realm, while all external users who access the public applications of the enterprise may belong to another realm. An identity management realm is represented in the directory by a specific entry with a special object class associated with it.

instance

The combination of the **System Global Area (SGA)** and the Oracle background processes. When a database is started on a database server (regardless of the type of computer), Oracle allocates a memory area called the SGA and starts one or more Oracle processes. The memory and processes of an instance efficiently manage the associated database's data and serve the database users. You can connect to any instance to access information within a cluster database.

instance name

A name of an Oracle database instance. The instance name is identified by the `INSTANCE_NAME` parameter in the database initialization parameter file. `INSTANCE_NAME` corresponds to the **Oracle System Identifier (SID)** of the instance. Clients can connect to a specific instance by specifying the `INSTANCE_NAME` parameter in the connect descriptor.

The instance name is included in the **connect data** part of the **connect descriptor**.

Interprocess Communication

A protocol used by client applications that resides on the same node as the listener to communicate with the database. IPC can provide a faster local connection than TCP/IP.

IP address

Used to identify a node on a network. Each computer on the network is assigned a unique IP address, which is made up of the network ID, and a unique host ID. This address is typically represented in dotted-decimal notation, with the decimal value of each octet separated by a period, for example 144.45.9.22.

IPC

See **Interprocess Communication**.

Java Database Connectivity (JDBC) Driver

A driver that provides Java applications and applets access to an Oracle database.

JDBC OCI Driver

A Type II driver for use with client/server Java applications. This driver requires an Oracle client installation.

JDBC Thin Driver

A Type IV driver for Oracle JDBC applets and applications. Because it is written entirely in Java, this driver is platform-independent. It does not require any additional Oracle software on the client side. The Thin driver communicates with the server using **Two-Task Common (TTC)**, a protocol developed by Oracle to access the database server.

keyword-value pair

The combination of a keyword and a value, used as the standard unit of information in connect descriptors and many configuration files. Keyword-value pairs may be nested; that is, a keyword may have another keyword-value pair as its value.

latency

Networking round-trip time.

Lightweight Directory Access Protocol (LDAP)

A standard, extensible directory access protocol. It is a common language that LDAP clients and servers use to communicate. The framework of design conventions supporting industry-standard **directory server**s.

LDAP Data Interchange Format (LDIF)

The set of standards for formatting an input file for any of the LDAP command line utilities.

ldap.ora file

A file created by Oracle Internet Directory Configuration Assistant or Oracle Net Configuration Assistant that contains the following directory server access information:

- Type of directory server
- Location of the directory server
- Default Oracle Context that the client or server will use to look up or configure connect identifiers for connections to database services

When created with Oracle Internet Directory Configuration Assistant, `ldap.ora` is located in the `$ORACLE_HOME/ldap/admin` directory on UNIX operating systems and the *`ORACLE_HOME`*`\ldap\admin` directory on Windows operating systems. When created with Oracle Net Configuration Assistant, `ldap.ora` is located in the `$ORACLE_HOME/network/admin` directory on UNIX operating systems and the *`ORACLE_HOME`*`\network\admin` directory on Windows operating systems.

link qualifier

A qualifier appended to a global database link to provide alternate settings for the database username and password credentials. For example, a link qualifier of `fieldrep` can be appended to a global database link of `sales.us.acme.com`.

```
SQL> SELECT * FROM emp@sales.us.acme.com@fieldrep
```

listener

See **Oracle Net Listener**.

listener.ora file

A configuration file for Oracle Net Listener that identifies the following:

- Unique name
- Protocol addresses that it is accepting connection requests on
- Services it is listening for

The `listener.ora` file typically resides in `$ORACLE_HOME/network/admin` on UNIX platforms and *`ORACLE_HOME`*`\network\admin` on Windows.

Oracle9*i* does not require identification of the database service because of **service registration**. However, static service configuration is required for if you plan to use Oracle Enterprise Manager.

Listener Control utility

A utility included with Oracle Network Services to control various listener functions, such as to starting, stopping, and getting the status of the listener.

load balancing

A feature by which client connections are distributed evenly among multiple listeners, dispatchers, instances, and nodes so that no single component is overloaded.

Oracle Network Services support **client load balancing** and **connection load balancing**.

local naming

A **naming method** that locates network addresses by using information configured and stored on each individual client's **tnsnames.ora file**. Local naming is most

appropriate for simple distributed networks with a small number of services that change infrequently.

location transparency

A distributed database characteristic that enables applications to access data tables without knowing where they reside. All data tables appear to be in a single database, and the system determines the actual data location based on the table name. The user can reference data on multiple nodes in a single statement, and the system automatically and transparently routes (parts of) SQL statements to remote nodes for execution if needed. The data can move among nodes with no impact on the user or application.

logging

A feature in which errors, service activity, and statistics are written to a log file. The log file provides additional information for an administrator when the error message on the screen is inadequate to understand the failure. The log file, by way of the error stack, shows the state of the software at various layers.

See also **tracing**.

loopback test

A connection from the server back to itself. Performing a successful loopback verifies that Oracle Net is functioning on the database server.

map

Files used by the **Network Information Service (NIS)** `ypserv` program to handle name requests.

Microsoft Active Directory

An LDAP-compliant directory server included with the Windows 2000 Server. It stores information about objects on the network, and makes this information available to users and network administrators. Active Directory also provides access to resources on the network using a single logon process.

Active Directory can be configured as a directory naming method to store service information that clients can access.

names.ora file

A configuration file that contains parameter settings for an Oracle Names server.

Named Pipes protocol

A high-level interface protocol providing interprocess communications between clients and servers using distributed applications. Named Pipes enables client/server conversation over a network using Named Pipes.

naming context

A subtree that resides entirely on one directory server. It is a contiguous subtree, that is, it must begin at an entry that serves as the top of the subtree, and extend downward to either leaf entries or references to subordinate naming contexts. It can range in size from a single entry to the entire **directory information tree (DIT)**.

An **Oracle Context** can be created under a naming context.

naming method

The resolution method used by a client application to resolve a **connect identifier** to a **connect descriptor** when attempting to connect to a database service. Oracle Net provides four naming methods:

- **local naming**
- **directory naming**
- easy connect naming
- **external naming**

net service alias

An alternative name for a **directory naming** object in a directory server. A directory server stores net service aliases for any defined **net service name** or database service. A net service alias entry does not have connect descriptor information. Instead, it only references the location of the object for which it is an alias. When a client requests a directory lookup of a net service alias, the directory determines that the entry is a net service alias and completes the lookup as if it was actually the entry it is referencing.

net service name

A simple name for a service that resolves to a **connect descriptor**. Users initiate a connect request by passing a username and password along with a net service name in a connect string for the service to which they wish to connect:

```
CONNECT username@net_service_name
```

Depending on your needs, net service names can be stored in a variety of places, including:

- Local configuration file, `tnsnames.ora`, on each client
- Directory server
- External naming service, such as **NIS** or **CDS**

network

A group of two or more computers linked together through hardware and software to allow the sharing of data and peripherals.

network administrator

The person who performs network management tasks such as installing, configuring, and testing network components. The administrator typically maintains the configuration files, connect descriptors and service names, aliases, and public and global database links.

network character set

As defined by Oracle, the set of characters acceptable for use as values in keyword-value pairs (that is, in connect descriptors and configuration files). The set includes alphanumeric upper- and lowercase, and some special characters.

Network Information Service (NIS)

Sun Microsystems' Yellow Pages (`yp`) client/server protocol for distributing system configuration data such as user and host names between computers on a network.

Network Interface (NI)

A network layer that provides a generic interface for Oracle clients, servers, or external processes to access Oracle Net functions. The NI layer handles the "break" and "reset" requests for a connection.

network listener

See **listener**.

network object

Any service that can be directly addressed on a network; for example, a listener.

network protocol

See **Oracle protocol support**.

Network Program Interface (NPI)

An interface for server-to-server interactions that performs all of the functions that the **OCI** does for clients, allowing a coordinating server to construct SQL requests for additional servers.

Network Session (NS)

A **session layer** that is used in typical Oracle Net connections to establish and maintain the connection between a client application and a database server.

NI

Network Interface

NIS

See **Network Information Service (NIS)**.

node

A computer or terminal that is part of a network

NPI

See **Network Program Interface (NPI)**.

NR

Network Routing

NS

Network Session. See **Network Session (NS)**.

NT

Network Transport. See **transport**.

object class

In a directory server, a named group of attributes. When you want to assign attributes to an entry, you do so by assigning to that entry the object classes that hold those attributes.

All objects associated with the same object class share the attributes of that object class.

OCI

Oracle Call Interface. See **Oracle Call Interface (OCI)**.

OPI

See **Oracle Program Interface (OPI)**.

Open Systems Interconnection (OSI)

A model of network architecture developed by ISO as a framework for international standards in heterogeneous computer network architecture.

The OSI architecture is split between seven layers, from lowest to highest:

1. Physical layer
2. Data link layer
3. Network layer
4. Transport layer
5. Session layer
6. Presentation layer
7. Application layer

Each layer uses the layer immediately following it and provides a service to the layer preceding.

Oracle Advanced Security

A product that provides a comprehensive suite of security features to protect enterprise networks and securely extends corporate networks to the Internet. Oracle Advanced Security 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 network.

Oracle Call Interface (OCI)

An application programming interface (API) that enables you to create applications that use the native procedures or function calls of a third-generation language to access an Oracle database server and control all phases of SQL statement execution. OCI supports the data types, calling conventions, syntax, and semantics of a number of third-generation languages including C, C++, COBOL and FORTRAN.

Oracle Connection Manager

A router through which a client connection request may be sent either to its next hop or directly to the database server. Clients who route their connection requests through an Oracle Connection Manager can then take advantage of the **session multiplexing**, **access control**, or **protocol conversion** features configured on that Oracle Connection Manager.

Oracle Connection Manager Control utility

A utility included with Oracle Network Services to control various functions, such as starting, stopping, and getting the status of the Oracle Connection Manager.

Oracle Context

A **RDN** of `cn=OracleContext` in a **directory information tree (DIT)** that is located under a **naming context** or an unpublished directory entry. An Oracle Context contains entries for use with Oracle features, such as Oracle Net **directory naming** and **Oracle Advanced Security enterprise user** security. There can be one or more Oracle Contexts in a directory server. **Oracle Internet Directory** automatically creates an

Oracle Context at the root of the DIT structure. This root Oracle Context has a DN of `dn:cn=OracleContext`.

Oracle Enterprise Manager

A separate Oracle product that combines a graphical console, agents, common services, and tools to provide an integrated and comprehensive systems management platform for managing Oracle products.

Oracle Identity Management

An infrastructure enabling deployments to manage centrally and securely all enterprise identities and their access to various applications in the enterprise.

Oracle Internet Directory

A directory server implemented as an application on the Oracle database. It enables retrieval of information about dispersed users and network resources. It combines **Lightweight Directory Access Protocol (LDAP)** Version 3, the open Internet standard directory server access protocol, with the high performance, scalability, robustness, and availability of the Oracle database.

Oracle Net

Communication software that enables a network session from a client application to an Oracle database server. Once a network session is established, Oracle Net acts as a data courier for the client application and the database server. It is responsible for establishing and maintaining the connection between the client application and database server, as well as exchanging messages between them. Oracle Net is able to perform these jobs because it is located on each computer in the network.

Oracle Net Configuration Assistant

A post-installation tool that configures basic network components after installation, including:

- Listener names and protocol addresses
- Naming methods the client will use to resolve **connect identifier**s
- Net service names in a `tnsnames.ora` file
- Directory server usage

Oracle Net Firewall Proxy

Product offered by some firewall vendors that supplies **Oracle Connection Manager** functionality.

Oracle Net foundation layer

A networking communication layer that is responsible for establishing and maintaining the connection between the client application and server, as well as exchanging messages between them.

Oracle Net Listener

A process that resides on the server whose responsibility is to listen for incoming client connection requests and manage the traffic to the server.

When a client requests a network session with a database server, a listener receives the actual request. If the client information matches the listener information, then the listener grants a connection to the database server.

Oracle Net Manager

A tool that combines configuration abilities with component control to provide an integrated environment for configuring and managing Oracle Net Services.

You can use Oracle Net Manager to configure the following network components:

- Naming

  Define **connect identifier**s and map them to **connect descriptor**s to identify the network location and identification of a service. Oracle Net Manager supports configuration of connect descriptors in a local `tnsnames.ora` file or directory server.

- Naming Methods

  Configure the different ways in which connect identifiers are resolved into connect descriptors.

- Listeners

  Create and configure listeners to receive client connections.

Oracle Net Services

A suite of networking components that provide enterprise-wide connectivity solutions in distributed, heterogeneous computing environments. Oracle Net Services is comprised of **Oracle Net**, **listener**, **Oracle Connection Manager**, **Oracle Net Configuration Assistant**, and **Oracle Net Manager**.

Oracle Program Interface (OPI)

A networking layer responsible for responding to each of the possible messages sent by **OCI**. For example, an OCI request to fetch 25 rows would have an OPI response to return the 25 rows once they have been fetched.

Oracle protocol support

A software layer responsible for mapping **Transparent Network Substrate (TNS)** functionality to industry-standard protocols used in the client/server connection.

Oracle Rdb

A database for Digital's 64-bit platforms. Because Oracle Rdb has its own listener, the client interacts with Rdb in the same manner as it does with an Oracle database.

Oracle schema

A set of rules that determine what can be stored in a **directory server**. Oracle has its own schema that is applied to many types of Oracle entries, including Oracle Net Services entries. The Oracle schema for Oracle Net Services' entries includes the attributes the entries may contain.

Oracle System Identifier (SID)

A name that identifies a specific instance of a running pre-release 8.1 Oracle database. For any database, there is at least one instance referencing the database.

For pre-release 8.1 databases, SID is used to identify the database. The SID is included in the connect descriptor of a **tnsnames.ora file** and in the definition of the listener in the **listener.ora file**.

Oracle XML DB

A high-performance XML storage and retrieval technology provided with Oracle database server. It is based on the W3C XML data model.

Oracle Real Application Clusters

An architecture that allows multiple instances to access a shared database of datafiles. Real Application Clusters is also a software component that provides the necessary cluster database scripts, initialization files, and datafiles needed for the Oracle Enterprise Edition and Real Application Clusters.

ORACLE_HOME

An alternate name for the top directory in the Oracle directory hierarchy on some directory-based operating systems.

OSI

See **Open Systems Interconnection (OSI)**.

packet

A block of information sent over the network each time a connection or data transfer is requested. The information contained in packets depends on the type of packet: connect, accept, redirect, data, and so on. Packet information can be useful in troubleshooting.

PMON process

A process monitor database process that performs process recovery when a user process fails. PMON is responsible for cleaning up the cache and freeing resources that the process was using. PMON also checks on dispatcher and server processes and restarts them if they have failed. As a part of **service registration**, PMON registers instance information with the listener.

presentation layer

A networking communication layer that manages the representation of information that application layer entities either communicate or reference in their communication. **Two-Task Common (TTC)** is an example of presentation layer.

private database link

A database link created by one user for his or her exclusive use.

See also **database link** and **public database link**.

profile

A collection of parameters that specifies preferences for enabling and configuring Oracle Net Services' features on the client or server. A profile is stored and implemented through the `sqlnet.ora` file.

protocol

A set of rules that defines how data is transported across the network.

protocol address

An address that identifies the network address of a network object.

When a connection is made, the client and the receiver of the request, such as the **listener** or **Oracle Connection Manager**, are configured with identical protocol addresses. The client uses this address to send the connection request to a particular network object location, and the recipient "listens" for requests on this address. It is important to install the same protocols for the client and the connection recipient, as well as to configure the same addresses.

protocol conversion

A feature of Oracle Connection Manager that enables a client and server with different networking protocols to communicate with each other. This feature replaces functionality previously provided by the Oracle Multi-Protocol Interchange with SQL*Net version 2.

protocol stack

Designates a particular **presentation layer** and **session layer** combination.

proxy server

A server that substitutes for the real server, forwarding client connection requests to the real server or to other proxy servers. Proxy servers provide access control, data and system security, monitoring, and caching.

public database link

A database link created by a DBA on a local database that is accessible to all users on that database.

See also **database link** and **private database link**.

realm Oracle Context

An Oracle Context contained in each **identity management realm**. It stores the following information:

- User naming policy of the identity management realm—that is, how users are named and located
- Mandatory authentication attributes
- Location of groups in the identity management realm
- Privilege assignments for the identity management realm—for example: who has privileges to add more users to the realm.
- Application specific data for that Realm including authorizations

RDBMS

Relational Database Management System

RDN

See **relative distinguished name (RDN)**.

relative distinguished name (RDN)

The local, most granular level entry name. It has no other qualifying entry names that would serve to address the entry uniquely. In the example, `cn=sales,dc=us,dc=acme,dc=com`, `cn=sales` is the RDN.

root Oracle Context

In the **Oracle Identity Management** infrastructure, the The root Oracle Context is an entry in Product_Name containing a pointer to the default **identity management realm** in the infrastructure. It also contains information on how to locate an identity management realm given a simple name of the realm.

RPC

Remote Procedure Call

SDP protocol

Sockets Direct Protocol (SDP).

Secure Sockets Layer (SSL)

An industry standard protocol designed by Netscape Communications Corporation for securing network connections. SSL provides authentication, encryption, and data integrity using public key infrastructure (PKI).

server parameter file

A binary file containing initialization parameter settings that is maintained on the Oracle Database host. You cannot manually edit this file with a text editor. A server parameter file is initially built from a text initialization parameter file by means of the `CREATE SPFILE` statement or created directly with the Database Configuration Assistant.

server process

Database processes that handle a client request on behalf of a database.

service

Work done for others. The database is a service that stores and retrieves data for clients.

service handler

A process that acts a connection point from the listener to the database server. A service handler can be a **dispatcher** or **dedicated server**.

service name

A logical representation of a database, which is the way a database is presented to clients. A database can be presented as multiple services and a service can be implemented as multiple database instances. The service name is a string that is the **global database name**, that is, a name comprising the database name and domain name, entered during installation or database creation. If you are not sure what the global database name is, you can obtain it from the value of the `SERVICE_NAMES` parameter in the initialization parameter file.

The service name is included in the **connect data** part of the **connect descriptor**.

service registration

A feature by which the **PMON process** automatically registers information with a **listener**. Because this information is registered with the listener, the `listener.ora` file does not need to be configured with this static information.

Service registration provides the listener with information about:

- Service names for each running instance of the database
- Instance names of the database
- Service handlers (**dispatcher** or **dedicated server**) available for each instance

  These enable the listener to direct a client request appropriately.
- Dispatcher, instance, and node load information

This load information enables the listener to determine which dispatcher can best handle a client connection request. If all dispatchers are blocked, the listener can spawn a dedicated server for the connection.

session data unit (SDU)

A buffer that Oracle Net uses to place data before transmitting it across the network. Oracle Net sends the data in the buffer either when requested or when it is full.

session layer

A network layer that provides the services needed by the **protocol address** entities that enable them to organize and synchronize their dialogue and manage their data exchange. This layer establishes, manages, and terminates network sessions between the client and server. An example of a session layer is **Network Session (NS)**.

session multiplexing

Combining multiple sessions for transmission over a single network connection in order to conserve the operating system's resources.

shared server

A database server that is configured to allow many user processes to share very few server processes, so the number of users that can be supported is increased. With shared server configuration, many user processes connect to a **dispatcher**. The dispatcher directs multiple incoming network session requests to a common queue. An idle shared server process from a shared pool of server processes picks up a request from the queue. This means that a small pool of server processes can serve a large number of clients. Contrast with **dedicated server**.

shared server process

A process type used with **shared server** configuration.

SID

See **Oracle System Identifier (SID)**.

SID_LIST_*listener_name*

A section of the `listener.ora` file that defines the **Oracle System Identifier (SID)** of the database served by the listener. This section is valid only for version 8.0 Oracle databases, as information for Oracle8*i* or later instances is automatically registered with the listener. Static configuration is also required for other services, such as **external procedure** calls and **Heterogeneous Services**.

single sign-on

The ability for a user to log in to different servers using a single password. This permits the user to authenticate to all servers the user is authorized to access.

sqlnet.ora file

A configuration file for the client or server that specifies:

- Client domain to append to unqualified service names or net service names
- Order of naming methods the client should use when resolving a name
- Logging and tracing features to use
- Route of connections
- External naming parameters
- Oracle Advanced Security parameters

The `sqlnet.ora` file typically resides in `$ORACLE_HOME/network/admin` on UNIX platforms and *`ORACLE_HOME`*`\network\admin` on Windows operating systems.

SSL

See **Secure Sockets Layer (SSL)**.

System Global Area (SGA)

A group of shared memory structures that contain data and control information for an Oracle **instance**.

TCP/IP protocol

Transmission Control Protocol/Internet Protocol. The de facto standard communication protocol used for client/server conversation over a network.

TCP/IP with SSL protocol

A protocol that enables an Oracle application on a client to communicate with remote Oracle databases through the **TCP/IP protocol** and **Secure Sockets Layer (SSL)**.

tick

The amount of time it takes for a message to be sent and processed from the client to the server or from the server to the client

Thin JDBC Driver

Thin JDBC driver is Oracle's Type 4 driver designed for Java applet and Java application developers. The JDBC driver establishes a direct connection to the Oracle database server over Java sockets. Access to the database is assisted with a lightweight implementation of Oracle Net and **Two-Task Common (TTC)**.

TNS

See **Transparent Network Substrate (TNS)**.

tnsnames.ora file

A configuration file that contains maps **net service name**s to **connect descriptor**s. This file is used for the **local naming** method. The `tnsnames.ora` file typically resides in `$ORACLE_HOME/network/admin` on UNIX platforms and *`ORACLE_HOME`*`\network\admin`.

tracing

A facility that writes detailed information about an operation to an output file. The trace facility produces a detailed sequence of statements that describe the events of an operation as they are executed. Administrators use the trace facility for diagnosing an abnormal condition; it is not normally turned on.

See also **logging**.

Transparent Application Failover (TAF)

A runtime failover for high-availability environments, such as Oracle9*i* Real Application Clusters and Oracle Fail Safe, that refers to the failover and re-establishment of application-to-service connections. It enables client applications to automatically reconnect to the database if the connection fails, and, optionally, resume a `SELECT` statement that was in progress. This reconnect happens automatically from within the Oracle Call Interface (OCI) library.

Transparent Network Substrate (TNS)

A foundation technology, built into the **Oracle Net foundation layer** that works with any standard network transport protocol.

transport

A networking layer that maintains end-to-end reliability through data flow control and error recovery methods. The **Oracle Net foundation layer** uses **Oracle protocol support** for the transport layer.

TTC

See **Two-Task Common (TTC)**.

Two-Task Common (TTC)

A **presentation layer** type that is used in a typical Oracle Net connection to provide character set and data type conversion between different character sets or formats on the client and server.

UPI

User Program Interface

virtual circuit

A piece of shared memory used by the **dispatcher** for client database connection requests and replies. The dispatcher places a virtual circuit on a common queue when a request arrives. An idle shared server picks up the virtual circuit from the common queue, services the request, and relinquishes the virtual circuit before attempting to retrieve another virtual circuit from the common queue.

WebDAV protocol

World Wide Web Distributed Authoring and Versioning. A protocol with a set of extensions to the **HTTP protocol** which allows users to manage files on remote Web servers.

Windows NT native authentication

An **authentication method** that enables a client single login access to a Windows NT server and a database running on the server.

Index

Numerics

1521 port, 10-4
  if not using, 10-10

A

absolute naming
  Java Database Connectivity (JDBC)
    OCI drivers, 4-9, 15-5
    Thin drivers, 15-5
  with directory naming, 4-10
access control lists (ACLs)
  described, 4-12
access control to database
  with Oracle Connection Manager, 1-13, 11-7
  with sqlnet.ora parameters, 9-4
ACTION_LIST networking parameter, 11-3, 11-4
ADAPTERS utility, 16-8
Address List Options dialog box, 13-4
ADDRESS networking parameter, 11-3
ADR
  see automatic diagnostic repository
ADR_BASE log parameter, 16-23, 16-24
ADR_BASE_*listener_name* log parameter, 16-23
ADRCI, 16-6
application layer, 5-2
application Web servers, 1-4
architecture
  listener, 5-7
  Oracle Connection Manager, 5-11
  Oracle Net Services stack communication layers, 5-1
attributes
  defined, 4-5
audit trail
  described, 16-26
  script for using information, 16-26
automatic diagnostic repository, 16-2
auto-starting listeners, 10-14

B

BACKUP networking parameter, 13-15
Bequeath protocol, 3-10
BEQUEATH_DETACH networking parameter, 9-7
buffer flushing, described, 7-10

C

CDS
  see Cell Directory Service
Cell Directory Services
  external naming, 3-14, 9-3
Cell Directory Services (CDS) external naming
  configuring, 8-25
Cell Name field in Oracle Net Manager, 9-8
centralized naming, 4-1
  establishing a connection with, 4-1
CHANGE_PASSWORD command, 10-8
Check TCP/IP client access rights option in Oracle Net Manager, 9-4
class of secure transports parameters
  See COST parameters
client configuration
  connect-time failover, 3-7
  default domains, 9-2
  load balancing requests among listeners, 3-8, 13-3
  local naming, 8-13
  log files, 16-23
  net service names, 2-3 to 2-5
  Oracle Connection Manager address, 11-4
  Oracle Rdb connections, 13-28
  shared server, 12-4
  sqlnet.log file, 16-23
  trace files, 16-34
  Transparent Application Failover (TAF), 3-7
client connections
  methods, 15-6
  syntax, 15-4
client load balancing
  configuring, 13-3
  described, 3-8
Client Registration ID field in Oracle Net Manager, 9-6
client testing
  connectivity, 15-9
  from 3GL, 15-7
  from applications, 15-7
  from Oracle Net Manager, 15-12
  from SQL, 15-7
  from the operating system, 15-7

client troubleshooting, 16-8
Clients allowed to access field in Oracle Net Manager, 9-4
Clients excluded from access field in Oracle Net Manager, 9-4
CMADMIN (Connection Manager Administration), 5-11, 11-6
CMAN DISPATCHERS parameter
See also DISPATCHERS parameter, 11-6
cman.ora file
log parameters
ADR_BASE, 16-24
DIAG_ADR_ENABLED, 16-24
EVENT_GROUP, 16-24
LOG_DIRECTORY, 16-24
LOG_LEVEL, 16-24
parameters
ACTION_LIST, 11-3, 11-4
ADDRESS, 11-3
PARAMETER_LIST, 11-4
RULE_LIST, 11-3, 11-7
TRACE_TIMESTAMP, 16-38
setting up, 11-2, 11-3
trace parameters
TRACE_DIRECTORY, 16-38
TRACE_FILELEN, 16-38
TRACE_FILENO, 16-38
TRACE_LEVEL, 16-38
TRACE_TIMESTAMP, 16-38
CMGW (Connection Manager Gateway), 5-11
configuring
access control, 9-4, 11-7
clients
client load balancing, 13-4
connect-time failover, 3-7, 13-3
default domains, 9-2
local naming, 8-13
naming methods in profile, 9-3
net service names, 2-3 to 2-5
session data unit (SDU), 14-2
Transparent Application Failover (TAF), 3-7
connection load balancing, 13-7 to 13-12
connection pooling, 12-2
connection requests in a profile, 9-4
connect-request timeouts, 14-8
database servers
access control, 9-4
connect-request timeouts, 14-8
session data unit (SDU), 14-2, 14-7
shared server, 12-1
directory naming, 8-14 to 8-25
easy connect naming, 8-3 to 8-8
external naming, 8-25 to 8-28
external procedure connections, 13-21
Heterogeneous Services connections, 13-26
instance role, 13-18
listeners, 10-1 to 10-19
connect-request timeouts, 14-8
directory naming, 8-18
external procedures, 13-22
host naming, 8-7
local naming, 8-13
Oracle JServer, 10-4
queue size, 10-5
service information, 10-6
session data unit (SDU), 14-2
local naming, 8-8 to 8-13
localized management, 4-1
multiple protocol addresses, 13-1
naming methods, 8-1 to 8-28
net service aliases, 8-17
net service names
directory naming, 8-14
external naming, 8-25
local naming, 8-9
network domain, default, 9-2
Oracle Advanced Security, 9-8
Oracle Rdb database connections, 13-27
primary and secondary instances, 13-18
protocol addresses, 10-3
quick reference, 6-8 to 6-10
SDP protocol, 14-6
servers
connect-request timeouts, 14-8
session data unit (SDU), 14-2, 14-7
shared server, 12-1
service registration, 11-6
session data unit (SDU) size, 13-6
session multiplexing, 11-7
connect identifiers, 15-4
connect strings, 15-1
connection broker, 3-11
connection load balancing, 13-7
configuring
dedicated server, 13-10
shared server, 13-8
described, 3-8
service registration, 10-2
connection pooling
advantages relative to session multiplexing, 7-3
configuring, 12-2
using with shared server, 12-2
Connection Time Out field in Oracle Net Manager, 9-6
connections, 3-6
adjusting listener queue size to avoid errors, 7-6, 7-9, 10-5
bequeath, 3-10
concurrent, increasing number of, 10-5
connect strings, entering, 15-1, 15-4
dedicated servers, 9-4
directory naming, 8-19
external procedures, 13-21
Heterogeneous Services, 13-26
host naming, 8-8
local naming, 8-13
methods, 15-6
NIS external naming, 8-25
Oracle Connection Manager, 9-4
Oracle Rdb databases, 13-27

startup of components, 15-1
CONNECTIONS (CON or CONN) attribute, 12-2
connect-time failover
and GLOBAL_DNAME parameter, 10-6
configuring, 13-3
service registration, 10-2
COST parameters, 10-8

D

data link layer, 5-2
data transfer, maximizing, 7-10
Database Configuration Assistant
directory naming, 4-7
shared server configuration, 12-2
database resident connection pooling, 3-11
database server configuration
access control, 9-4
allocating resources with shared server, 12-3
connect-request timeouts, 14-8
local naming, 8-9
log files, 16-23
loopback tests, 16-7
shared server, 12-1
starting the database, 2-3, 15-2
trace files, 16-34
database server connections
methods, 15-6
syntax, 15-4
database server testing, 15-8
dead connection timeout. See terminated connection timeout, 9-6
dedicated connection, 3-11
dedicated server
configuring, 12-4
dedicated servers
bequeathed sessions, 3-10
connect descriptor configuration, 3-6
defined, 1-8
described, 3-10 to 3-11
difference with shared server, 1-8
routing connections, 9-4
denial-of-service attacks, 14-8
DESCRIPTION parameter, 8-8
DIAG_ADR_ENABLED log parameter, 16-23, 16-24
DIAG_ADR_ENABLED_*listener_name* log parameter, 16-23
diagnosing. See troubleshooting
direct hand-off
described, 3-9
events in listener.log, 16-29
directory configuration
adding users to the OracleNetAdmins group, 8-20
exporting
net service names from a tnsnames.ora file, 8-21
migrating
net service names from a tnsnames.ora file, 8-21
directory information tree (DIT), defined, 4-4
directory naming, 16-16
absolute naming, 4-10
advantages and disadvantages, 7-9
architecture, 4-4
authentication methods
native, 4-11
simple, 4-11
SSL, 4-11
strong, 4-11
configuring, 8-14 to 8-25
connecting to database services, 4-9
connecting with, 8-19
connections
using an entry's absolute name, 4-10
using an entry's relative name, 4-9
Database Configuration Assistant, 4-7
establishing a connection with, 8-19
exporting
net service names from a tnsnames.ora file, 8-21
Java Database Connectivity (JDBC)
OCI drivers, 4-9, 15-5
Thin drivers, 4-9, 15-5
Java Database Connectivity (JDBC) Thin drivers, 4-4, 4-10
ldapwrite tool, 16-16
listener configuration for, 8-18
migrating
net service names from a tnsnames.ora file, 8-21
object classes, 4-14
objects
database services, 4-4
net service aliases, 4-5
net service names, 4-4
Oracle Context, 4-5
Oracle Net Manager, 4-7
Oracle schema, 4-14
OracleContextAdmins group, 4-8, 4-12
OracleDBCreators group, 4-8, 4-12
OracleNetAdmins group, 4-13, 8-15, 8-16, 8-17, 8-20
overview, 4-3
recommended for, 7-9
security, 4-11
troubleshooting, 16-16
with relative naming, 4-9
Directory Server Migration Wizard, 6-4, 8-23
directory servers
attributes, 4-5
directory information tree (DIT), 4-4
distinguished name (DN), 4-5
entry, 4-4
ldapwrite tool, 16-16
Microsoft Active Directory, 4-14
Oracle Context, 4-5
overview, 1-6
performance, 4-11
relative distinguished name (RDN), 4-5

security, 4-11
Directory Usage configuration option in Oracle Net Configuration Assistant, 6-7
Disable Out-of-Band Break option in Oracle Net Manager, 9-7
DISABLE_OOB networking parameter, 9-7
dispatchers, 1-8, 5-9
described, 1-8, 3-9, 5-9
DISPATCHERS (DIS or DISP) attribute, 12-2
DISPATCHERS initialization parameter, 12-1
configuring connection pooling, 12-2
CONNECTIONS attribute, 12-2
DISPATCHERS attribute, 12-2
LISTENER attribute, 10-11, 10-12, 12-2
MULTIPLEX attribute, 12-2
POOL attribute, 12-2
SERVICE attribute, 12-2
SESSIONS attribute, 12-2
TICKS attribute, 12-2
DISPATCHERS parameter
MULTIPLEX attribute, 11-7
PROTOCOL attribute, 11-7
distinguished name (DN), defined, 4-5
duties of a network administrator, 6-8 to 6-10

E

easy connect naming, 2-3, 3-13
advantages and disadvantages, 7-9
configuring, 8-3 to 8-8
recommended for, 7-9
syntax examples, 8-5
entry, defined, 4-4
Environment field in Oracle Net Manager, 13-24
ENVS networking parameter, 13-24
error messages
contacting Oracle Support Services, 16-58
in trace file, 16-41
ORA-1017, 16-7
ORA-1034, 16-7
ORA-12154, 16-11
ORA-12170, 14-8, 16-12
ORA-12203
sample error stack, 16-21
ORA-12514, 16-13
ORA-12520, 16-14
ORA-12521, 16-14
ORA-12525, 14-8, 16-14
ORA-12526, 5-8
ORA-12527, 5-8
ORA-12528, 5-8
ORA-12533, 16-15
ORA-12535, 9-5
ORA-12547, 14-8
ORA-12608, 9-5
ORA-12637, 14-8
resolving, 16-1 to 16-58
using log file to track, 16-25
error stack
described, 16-20
entries in log files, 16-26
sample, 16-21
typical layers in, 16-20
EVENT_GROUP log parameter, 16-24
exporting
net service names from a tnsnames.ora file to a directory, 8-21
external naming
advantages and disadvantages, 7-9
Cell Directory Services, 3-14, 9-3
Cell Directory Services (CDS), 8-25
configuring, 8-25 to 8-28
network information service, 3-14
NIS, 8-25, 9-3
recommended for, 7-9
external procedures
configuring connections to, 13-21
described, 13-21
extproc agent, 13-21
extproc agent, 13-21
EXTPROC_DLLS environment variable, 13-24

F

failover
connect-time, 3-7, 13-3
Transparent Application Failover (TAF), 3-7, 13-13
FAILOVER networking parameter, 13-3, 13-10, 13-12
FAILOVER_MODE networking parameter, 13-13, 13-15
FTP presentation, 5-6
dedicated server configuration, 5-10
shared server configuration, 5-10
FTP protocol, 1-4

G

global database name
configuring on the listener, 10-10
defined, 8-2
described, 10-10
Global Database Name field in Oracle Net Manager, 10-6, 13-6, 13-28
global parameter
defined, 11-4
GLOBAL_DBNAME networking parameter, 10-6
GLOBAL_NAME networking parameter, 13-6, 13-28

H

Heterogeneous Services
configuring connections to, 13-26
described, 13-26
host naming
connecting with, 8-8
establishing a connection with, 8-8
listener configuration for, 8-7
requirements, 8-6
HS networking parameter, 13-6
HTTP presentation, 5-6

dedicated server configuration, 5-10
shared server configuration, 5-10
HTTP protocol, 1-3

I

INBOUND_CONNECT_TIMEPUT_*listener_name* networking parameter, 14-8
Infiniband network communication
configuring SDP protocol support, 14-6
initialization parameter file
DISPATCHERS parameter, 12-1
INSTANCE_NAME parameter, 3-4
LOCAL_LISTENER parameter, 10-4, 10-11
REMOTE_LISTENER parameter, 10-12
SERVICE_NAMES parameter, 3-2, 8-2
installation
default configuration
listeners, 10-2
local naming, 8-8
profiles, 9-1
Instance Name field in Oracle Net Manager, 13-5
instance role configuration, 13-18
connections in TAF, 13-20
connections to primary and secondary instances, 13-19
connections to specific instances, 13-19
INSTANCE_NAME networking parameter, 3-4, 8-2, 13-5, 13-18

J

Java Database Connectivity (JDBC)
drivers, 1-2
OCI drivers, 5-5
absolute naming support, 4-9, 15-5
relative naming support, 4-9
Thin drivers, 5-5
absolute naming support, 15-5
directory naming support, 4-4, 4-10
relative naming support, 4-9
JavaNet, 5-6
JavaNet layer, 1-4
JavaTTC, 5-6
JDBC. See Java Database Connectivity (JDBC)

L

ldapwrite tool, 16-16
LISTENER (LIS or LIST) attribute, 10-11, 10-12, 12-2
Listener configuration option in Oracle Net Configuration Assistant, 6-7
Listener Control utility
commands
SERVICES, 2-3, 10-17
SET PASSWORD, 15-2
START, 2-2, 15-2
STATUS, 10-15, 15-2
STOP, 15-2
starting a listener, 10-15
stopping a listener, 10-15
using, 6-7
listener.log file, 16-22
listener.ora file
described, 4-2
log parameters
ADR_BASE_*listener_name*, 16-23
DIAG_ADR_ENABLED_*listener_name*, 16-23
LOG_DIRECTORY_*listener_name*, 16-23
LOG_FILE_*listener_name*, 16-23
parameters
ENVS, 13-24
GLOBAL_DBNAME, 10-6
INBOUND_CONNECT_TIMEOUT_*listener_name*, 14-8
ORACLE_HOME, 10-6, 13-24, 13-26
PASSWORDS_*listener_name*, 10-8
PROGRAM, 13-24, 13-26
SID_NAME, 10-6, 13-24, 13-26
trace parameters
TRACE_DIRECTORY_*listener_name*, 16-37
TRACE_FILE_*listener_name*, 16-37
TRACE_FILELEN_*listener_name*, 16-37
TRACE_FILENO_*listener_name*, 16-37
TRACE_LEVEL_*listener_name*, 16-37
TRACE_TIMESTAMP_*listener_name*, 16-37
listeners, 3-5, 13-26
adjusting queue size for, 10-5
auto-starting, 10-14
client load balancing, 3-8
configuring, 10-1 to 10-19
address list, 13-1
directory naming method, 8-18
external procedures, 13-22
global database name, 10-10
host naming method, 8-7
local naming method, 8-13
multihomed hosts, 10-4
nondefault address, 10-10
Oracle JServer access, 10-4
Oracle System Identifier, 10-6
protocol addresses, 10-3
service information, 10-6
session data unit (SDU), 14-2
SID, 10-6
connection load balancing, 3-8, 13-7
connect-time failover, 3-7
default address, 10-10
default configuration, 10-2
direct hand-off, 3-9
handling concurrent connections, 10-5
increasing queue size, 10-5
log files, 10-19, 16-23, 16-26
audit trail, 16-26
direct hand-off event information, 16-29
service registration event information, 16-27
monitoring, 10-15, 10-17, 10-19
multihomed hosts, 10-4
multiple, 10-3
multiple addresses, 13-1
passwords, setting, 10-7

queue size, 7-6, 7-9
redirect connections, 3-9
security
connect-request timeouts, 14-8
password usage, 10-7
See Oracle Net Listener
starting, 2-3, 15-2, 15-3
testing configuration, 15-8
trace files, 16-36
Transparent Application Failover (TAF), 3-7
version requirements for database, 10-1
listener.trc file, 16-33
load balancing
client, 3-8
connection, 3-8, 13-7
LOAD_BALANCE networking parameter, 13-4
LOCAL environment variable, 15-4
local naming
advantages and disadvantages, 7-9
client configuration, 8-13
configuring, 8-8 to 8-13
connecting with, 8-13
database server configuration, 8-9
default configuration, 8-8
establishing a connection with, 8-13
listener configuration for, 8-13
recommended for, 7-9
Local Net Service Name configuration option in Oracle Net Configuration Assistant, 6-7
LOCAL registry entry, 15-4
LOCAL_LISTENER initialization parameter, 10-4, 10-11, 12-2
log files, 10-19, 16-23, 16-26
default names for, 16-22
listener.log, 16-22, 16-23
Oracle Connection Manager, 16-24, 16-30 to 16-32
sqlnet.log, 16-22
sqlnet.log for clients and database servers, 16-23
using to track errors, 16-25
log parameters
cman.ora
ADR_BASE, 16-24
DIAG_ADR_ENABLED, 16-24
EVENT_GROUP, 16-24
LOG_DIRECTORY, 16-24
LOG_LEVEL, 16-24
listener.ora
ADR_BASE_*listener_name*, 16-23
DIAG_ADR_ENABLED_*listener_name*, 16-23
LOG_DIRECTORY_*listener_name*, 16-23
LOG_FILE_*listener_name*, 16-23
sqlnet.ora
ADR_BASE, 16-23
DIAG_ADR_ENABLED, 16-23
LOG_DIRECTORY_CLIENT, 16-23
LOG_DIRECTORY_SERVER, 16-23
LOG_FILE_CLIENT, 16-23
LOG_FILE_SERVER, 16-23
LOG_DIRECTORY log parameter, 16-24
LOG_DIRECTORY_CLIENT log parameter, 16-23
LOG_DIRECTORY_*listener_name* log parameter, 16-23
LOG_DIRECTORY_SERVER log parameter, 16-23
LOG_FILE_CLIENT log parameter, 16-23
LOG_FILE_*listener_name* log parameter, 16-23
LOG_FILE_SERVER log parameter, 16-23
LOG_LEVEL log parameter, 16-24
Logon Authentication Protocol Version field in Oracle Net Manager, 9-7
loopback test, 16-7

M

maximizing data transfer, by adjusting SDU size, 7-10
Meta Map field in Oracle Net Manager, 9-8
METHOD networking parameter, 13-16
Microsoft Active Directory, 4-14
migrating
net service names from a tnsnames.ora file to a directory, 8-21
multihomed hosts, 10-4
multiple addresses, 13-3
configuring client load balancing, 13-4
configuring connect-time failover, 13-4
multiple listeners, 10-3
multiple protocol addresses, 13-1
MULTIPLEX (MUL or MULT) attribute, 11-7, 12-2

N

Named Pipes protocol
described, 5-5
NAMES.DCE.PREFIX networking parameter, 9-8
NAMES.DEFAULT_DOMAIN networking parameter, 9-2
NAMES.DIRECTORY_PATH networking parameter, 9-4
cds, 9-3
ezconnect, 9-3
hostname, 9-3
ldap, 9-3
nis, 9-3
tnsnames, 9-3
names.log file, 16-22
NAMES.NIS.META_MAP networking parameter, 9-8
names.trc file, 16-33
naming method, 9-3
naming methods, 3-14
Cell Directory Services, 3-14
Cell Directory Services (CDS), 8-25
centralized, 4-1
choosing, 7-5, 7-8
described, 3-13
directory naming, 8-14 to 8-25
easy connect naming, 8-3 to 8-8
external naming, 8-25 to 8-27
local naming, 8-8 to 8-13
localized, 4-1

NIS, 8-25
overview, 1-6
Naming Methods configuration option in Oracle Net Configuration Assistant, 6-7
net service aliases
configuring, 8-17
described, 4-5
directory naming, 4-5
uses of, 4-6
Net Service Name Wizard, 6-4, 8-11, 11-5
net service names
adding an address, 13-1
configuring, 2-3 to 2-5
directory naming, 8-14
external naming, 8-25
local naming, 8-9
multiple addresses, 13-1, 13-3
prioritizing naming methods, 9-3
testing with TNSPING, 15-10
network administrator duties, 6-8 to 6-10
Network Authentication (NA)
layer in error stacks, 16-21
network availability, determining, 2-1
network configuration
centralized management, 4-1
localized management, 4-1
network domain, default configuring, 9-2
network information service
See NIS
Network Interface (NI)
layer in error stacks, 16-21
network layer, 5-2
network performance, improving
by adjusting SDU size, 7-10
client load balancing, 3-8
listener queue size, 7-6, 7-9
network planning
session data unit (SDU) size, 7-10
Network Session (NS), layer in error stacks, 16-21
Network Transport (NT), layer in error stacks, 16-21
networking configuration files
cman.ora file, 4-1
listener.ora file, 4-2
sqlnet.ora file, 4-2
tnsnames.ora file, 4-2
networking planning
internal networks
availability, 7-4
client load balancing, 7-5
connection pooling, 7-3
connect-time failover, 7-5
JDBC drivers, 7-5
listener queue size, 7-6
naming methods, 7-5
protocol conversion, 7-6
scalability, 7-2
security, 7-5
session data unit (SDU) size, 7-6
session multiplexing, 7-3
tuning and performance, 7-5
Internet networks
access control, 7-8
availability, 7-7
connect-request timeouts, 7-8
JDBC drivers, 7-8
naming methods, 7-7
scalability, 7-7
security, 7-8
tuning and performance, 7-8
NIS external naming, 3-14, 9-3
configuring, 8-25
connecting with, 8-25
establishing a connection with, 8-25
maps, 8-26
nodes, described, 5-3

O

object classes
described, 4-14
orclDBServer, 4-14
orclNetAddress, 4-14
orclNetAddressList, 4-14
orclNetDescription, 4-14
orclNetDescriptionList, 4-14
orclNetService, 4-14
orclNetServiceAlias, 4-14
Open Systems Interconnection (OSI)
application layer, 5-2
data link layer, 5-2
described, 5-2
network layer, 5-2
Oracle Net foundation layer, 5-3
physical layer, 5-2
presentation layer, 5-2
session layer, 5-2
transport layer, 5-2
ORA-1017 error messages, 16-7
ORA-1034 error messages, 16-7
ORA-12154 error message, 16-11
ORA-12170 error message, 14-8, 16-12
ORA-12203 error message
sample error stack, 16-21
ORA-12514 error message, 16-13
ORA-12520 error message, 16-14
ORA-12521 error message, 16-14
ORA-12525 error message, 14-8, 16-14
ORA-12526 error messages, 5-8
ORA-12527 error messages, 5-8
ORA-12528 error messages, 5-8
ORA-12533 error message, 16-15
ORA-12535 error message, 9-5
ORA-12547 error message, 14-8
ORA-12637 error message, 14-8
Oracle Advanced Security
configuring with Oracle Net Manager, 9-8
overview, 1-16
Oracle Call Interface (OCI) layer, described, 5-3
Oracle Clusterware
notification information, 16-29

Oracle Connection Manager
architecture, 5-11
CMADMIN process, 5-11
configuring
access control, 11-7
clients, 11-4, 11-5
database server, 11-5, 11-6
Oracle Connection Manager computer, 11-2 to 11-4
protocol address for Oracle Connection Manager, 11-4
service registration, 11-6
session multiplexing, 11-7
gateway process, 5-11
listener, 5-11
log files, 16-23, 16-24
names, 16-22
understanding, 16-30 to 16-32
overview, 1-15
protocol address, 11-3
routing connections, 9-4
session multiplexing, 5-11
starting, 15-3
testing, 15-8
trace files, 16-37
configuring, 16-38
names, 16-33
Oracle Connection Manager Control utility
commands
ADMINISTER, 15-3
EXIT, 15-3
STARTUP, 15-3
using, 6-8
Oracle Context
defined, 4-5
Oracle Home Directory field in Oracle Net Manager, 10-6, 13-24
Oracle JServer connections, 10-4
Oracle Net
buffers, 7-10
defined, 1-13
Oracle Net foundation layer, 1-14
Oracle protocol support, 1-14
overview, 1-1 to 1-16
scalability features, 1-7
understanding, 1-1 to 1-16
Oracle Net Configuration Assistant
described, 6-6
Directory Usage configuration option, 6-7
listener configuration, 10-3
Listener configuration option, 6-7
local naming method, 8-12
Local Net Service Name configuration option, 6-7
Naming Methods configuration option, 6-7
net service names, 8-12
OracleContextAdmins group, 4-12
OracleDBCreators group, 4-8, 4-12
OracleNetAdmins group, 4-13
servers
listener configuration, 6-7
starting, 6-6
Oracle Net foundation layer, 1-14, 5-3
Oracle Net Listener
described, 1-14
See listeners
starting, 2-2
Oracle Net Manager
adding addresses, 13-1
Address List Options dialog box, 13-4
clients
client load balancing, 13-3
connect-time failover, 13-3
default network domains, 9-2
local naming method, 8-9, 8-11
Oracle Connection Manager, 11-4
described, 6-2
directory naming, 4-7
Directory Server Migration Wizard, 8-23
external procedure connections, 13-21
Heterogeneous Services connections, 13-26
Instance Name field, 13-5
listeners
Environment field, 13-24
Global Database Name field, 10-6, 13-6
Oracle Home Directory field, 10-6, 13-24
Program Name field, 13-24
protocol addresses, 10-3
SID field, 10-6, 13-24
static service information, 10-6
local naming method, 8-9, 8-11
multiple address options, 13-3
navigating, 6-3
net service aliases, 8-17
Net Service Name Wizard, 8-11, 11-5
net service names, 8-9, 8-11
Oracle Rdb Database field, 13-6
Oracle Rdb databases, 13-28
Global Database Name field, 13-28
Rdb Database field, 13-28
Type of Service field, 13-28
profiles, 9-8
advanced options, 9-5
Cell Name field, 9-8
Check TCP/IP client access rights option, 9-4
Client Registration ID field, 9-6
Clients allowed to access field, 9-4
Clients excluded from access field, 9-4
Connection Time Out field, 9-6
Disable Out-of-Band Break option, 9-7
Logon Authentication Protocol Version field, 9-7
Meta Map field, 9-8
Receive operation Time Out field, 9-5
Send operation Time Out field, 9-5
TNS Time Out Value option, 9-6
Total Receive Buffer field, 9-6
Total Send Buffer field, 9-6
Turn Off UNIX Signal Handling option, 9-7
routing connection requests, 9-4
Session Data Unit (SDU) field in Oracle Net

Manager, 13-6
specifying naming methods, 9-3
starting, 6-3
testing
client configuration, 15-12
server configuration, 15-8
Type of Service field, 13-6
Use for Heterogeneous Services option, 13-6, 13-27
Use Oracle8*i* Release 8.0 Compatible Identification option, 13-5
wizards, 6-3 to 6-5
Oracle Net Services
components
Oracle Connection Manager, 1-15
Oracle Net, 1-13
Oracle Net Listener, 1-14
described, 1-13
Oracle protocol support
described, 1-14, 5-4
Named Pipes, 5-5
TCP/IP, 5-4
TCP/IP with SSL, 5-4
Oracle Rdb database
configuring for connection to, 13-27
described, 13-27
Oracle Rdb Database field in Oracle Net Manager, 13-6
Oracle schema
described, 4-14
Oracle Support Services, contacting, 16-58
Oracle System Identifier, configuring on the listener, 10-6
ORACLE_HOME networking parameter, 10-6, 13-24, 13-26
Oracle9*i* Real Application Clusters
connect-time failover, 3-7, 13-3
FAILOVER networking parameter, 13-3
FAILOVER_MODE networking parameter, 13-15
Transparent Application Failover (TAF), 13-13
OracleContextAdmins group, 4-8, 4-12
OracleDBCreators group, 4-8, 4-12
Oracle*HOME_NAME*CMan service, 15-4
Oracle*HOME_NAME*TNSListener service, 15-2
OracleNetAdmins group, 4-13, 8-15, 8-16, 8-17, 8-20
orclDBServer object class, 4-14
orclNetAddress object class, 4-14
orclNetAddressList object class, 4-14
orclNetDescription object class, 4-14
orclNetDescriptionList object class, 4-14
orclNetService object class, 4-14
orclNetServiceAlias object class, 4-14
OSI. See Open Systems Interconnect (OSI)

P

packets
examining trace data, 16-47, 16-54
types of, 16-40
PARAMETER_LIST networking parameter, 11-4
parameters
global, 11-4
rule-level, 11-4
PASSWORDS_*listener_name* parameter, 10-8
physical layer, 5-2
planning
internal networks
availability, 7-4
connection pooling, 7-3
connect-time failover, 7-5
JDBC drivers, 7-5
listener queue size, 7-6
naming methods, 7-5
protocol conversion, 7-6
scalability, 7-2
security, 7-5
session data unit (SDU) size, 7-6
session multiplexing, 7-3
tuning and performance, 7-5
Internet networks
access control, 7-8
availability, 7-7
connect-request timeouts, 7-8
JDBC drivers, 7-8
naming methods, 7-7
scalability, 7-7
security, 7-8
tuning and performance, 7-8
session data unit (SDU) size, 7-10
PMON process, 10-2, 11-6
POOL (POO) attribute, 12-2
port 1521
if not using, 10-10
ports
privileged, 10-4
presentation layer, 5-2
FTP, 5-6
HTTP, 5-6
JavaTTC, 5-6
Two-Task Common (TTC), 5-3
WebDAV, 5-6
primary and secondary instances, 13-18
privileged ports, 10-4
profiles (sqlnet.ora)
configuring
advanced options, 9-5
default domains, 9-2
default configuration, 9-1
naming methods, specifying, 9-3
routing connection requests, 9-4
Program Name field in Oracle Net Manager, 13-24
PROGRAM networking parameter, 13-24, 13-26
PROTOCOL (PRO or PROT) attribute, 11-7
protocol address, 3-5
protocols
FTP, 1-4
HTTP, 1-3, 1-4
Named Pipes, 5-5
Oracle support for, 1-14
TCP/IP, 5-4

TCP/IP with SSL, 5-4
WebDAV, 1-4
proxy server, 11-1

Q

queue size, 7-6, 7-9, 10-5
QUEUESIZE parameter, 10-5
for adjusting listener queue size, 7-6, 7-9, 10-5

R

randomizing requests among listeners, 3-8
Rdb Database field, 13-28
RDB_DATABASE networking parameter, 13-6, 13-28
Receive operation Time field in Oracle Net Manager, 9-5
redirect connection, 3-9
relative distinguished name (RDN), 4-5
relative naming
directory naming, 4-9
Java Database Connectivity (JDBC)
OCI drivers, 4-9
Thin drivers, 4-9
resolving
errors. See troubleshooting
routing connections, 9-4
RULE_LIST networking parameter, 11-3, 11-7
rule-level parameter
defined, 11-4

S

scalability, of networks, 7-2
SDP protocol
configuring, 14-6
SDU
See session data unit (SDU)
security
database server
access control configuration, 9-4
connect-request timeouts, 14-8
internal networks, 7-5
Internet networks, 7-8
listeners
connect-request timeouts, 14-8
password usage, 10-7
Send operation Time field in Oracle Net Manager, 9-5
server configuration
access control, 9-4
allocating resources with shared server, 12-3
connect-request timeouts, 14-8
local naming, 8-9
log files, 16-23
loopback tests, 16-7
shared server, 12-1
starting, 2-3
starting the database, 15-2
trace files, 16-34
server connections
methods, 15-6
syntax, 15-4
SERVER networking parameter, 3-6
server testing, 15-8
server troubleshooting, 16-7
servers
access control, 9-4
SERVICE (SER or SERV) attribute, 12-2
service handlers
dedicated servers, 3-10 to 3-11
dispatchers, 3-9
service name
configuring, 8-2
described, 3-1, 3-2
service registration
benefits, 10-2
configuring, 10-2, 11-6
connection load balancing, 3-8, 10-2, 13-7
connect-time failover, 10-2
defined, 3-6
events in listener.log, 16-27
service_died listener log event, 16-28
service_register listener log event, 16-27
service_update listener log event, 16-28
service_died listener log event, 16-28
SERVICE_NAME networking parameter, 8-2
SERVICE_NAMES initialization parameter, 3-2, 8-2
service_register listener log event, 16-27
service_update listener log event, 16-28
SERVICES command, 10-17
of Listener Control utility, 2-3
session data unit (SDU), 7-10, 13-6
adjusting to improve network performance, 7-10
configuring, 14-1
Session Data Unit (SDU) Size field in Oracle Net Manager, 13-6
session layer, 5-2
session multiplexing, 1-10, 5-11, 11-7
advantages relative to connection pooling, 7-3
SESSIONS (SES or SESS) attribute, 12-2
SET PASSWORD command
of Listener Control utility, 15-2
shared server
allocating resources, 12-3
compared with dedicated server, 1-7
configuring, 12-4
connect descriptor configuration parameters, 3-6
connection load balancing, 3-8, 13-7
defined, 1-7
described, 5-9
dispatchers, 1-8, 3-9, 5-9
using with connection pooling, 12-2
virtual circuits, 5-9
SID field in Oracle Net Manager, 10-6, 13-24
SID, configuring on the listener, 10-6
SID_LIST_*listener_name* parameter
external procedures, 13-23
Oracle Enterprise Manager requirements, 10-6
SID_NAME networking parameter, 10-6, 13-24,

13-26
simple authentication for directory naming, 4-11
SOURCE_ROUTE networking parameter, 13-3, 13-4
SQLNET.ALLOWED_LOGON_VERSION networking parameter, 9-7
SQLNET.CLIENT_REGISTRATION networking parameter, 9-6
SQLNET.EXPIRE_TIME networking parameter, 9-6
SQLNET.INBOUND_CONNECT_TIMEOUT networking parameter, 9-6, 14-8
sqlnet.log file, 16-22
sqlnet.ora file
  described, 4-2
  log parameters
    ADR_BASE, 16-23
    DIAG_ADR_ENABLED, 16-23
    LOG_DIRECTORY_CLIENT, 16-23
    LOG_DIRECTORY_SERVER, 16-23
    LOG_FILE_CLIENT, 16-23
    LOG_FILE_SERVER, 16-23
  parameters
    NAMES.DCE.PREFIX, 9-8
    NAMES.DEFAULT_DOMAIN, 9-2
    NAMES.DIRECTORY_PATH, 9-3
    NAMES.NIS.META_MAP, 9-8
    SQLNET.INBOUND_CONNECT_TIMEOUT, 14-8
    TCP.EXCLUDED_NODES, 9-4
    TCP.INVITED_NODES, 9-4
    TCP.VALIDNODE_CHECKING, 9-4
  trace parameters
    TNSPING.TRACE_DIRECTORY, 16-36
    TNSPING.TRACE_LEVEL, 16-36
    TRACE_DIRECTORY_CLIENT, 16-34
    TRACE_DIRECTORY_SERVER, 16-34
    TRACE_FILE_CLIENT, 16-34
    TRACE_FILE_SERVER, 16-34
    TRACE_FILELEN_CLIENT, 16-34
    TRACE_FILELEN_SERVER, 16-34
    TRACE_FILEN0_CLIENT, 16-34
    TRACE_FILENO_SERVER, 16-35
    TRACE_LEVEL_CLIENT, 16-35
    TRACE_LEVEL_SERVER, 16-35
    TRACE_TIMESTAMP_CLIENT, 16-36
    TRACE_TIMESTAMP_SERVER, 16-36
    TRACE_UNIQUE_CLIENT, 16-36
SQLNET.RECV_BUF_SIZE networking parameter, 9-6
SQLNET.RECV_TIMEOUT networking parameter, 9-5
SQLNET.SEND_BUF_SIZE networking parameter, 9-6
SQLNET.SEND_TIMEOUT networking parameter, 9-5
sqlnet.trc file, 16-33
SSL authentication for directory naming, 4-11
START command
  of Listener Control utility, 2-2, 10-15, 15-2
starting
  database server, 2-3
  database servers, 15-3
  databases, 2-2, 15-2
  listeners, 2-3, 15-2, 15-3
  Oracle Connection Manager, 15-3
  Oracle Net Configuration Assistant, 6-6
  Oracle Net Listener, 2-2
  Oracle Net Manager, 6-3
  Oracle Net Services components, 15-1
STATUS command
  of Listener Control utility, 10-15, 15-2
STOP command
  of Listener Control utility, 10-14, 10-15, 15-2
strong authentication for directory naming, 4-11
svr_*pid*.trc file, 16-33
syntax
  for connect identifiers, 15-4
  for Listener Control utility, 6-7
  for Oracle Connection Manager Control utility, 6-8

T

TAF. See Transparent Application Failover (TAF)
TCP.EXCLUDED_NODES networking parameter, 9-4
TCP.INVITED_NODES networking parameter, 9-4
TCP/IP protocol
  described, 5-4
TCP/IP with SSL protocol
  described, 5-4
TCP.VALIDNODE_CHECKING networking parameter, 9-4
terminated connection detection
  configuring, 9-6
  limitations, 9-6
testing
  client configuration
    from 3GL, 15-7
    from applications, 15-7
    from Oracle Net Manager, 15-12
    from SQL, 15-7
    from the operating system, 15-7
    with TCROUTE, 15-11
    with TNSPING, 15-9
  listener configuration, 15-8
  network connectivity, 15-9
  Oracle Connection Manager, 15-8
  server configuration, 15-8
  with control utilities, 6-7
TICKS (TIC or TICK) attribute, 12-2
TNS. See Transparent Network Substrate (TNS)
TNS Time Out Value option in Oracle Net Manager, 9-6
TNS_ADMIN environment variable, 4-2
TNS-12154 error
  troubleshooting on UNIX, 16-18
tnsnames.ora file
  described, 4-2
  exporting entries to directory server, 8-21
  migrating entries to directory server, 8-21

parameters
BACKUP parameter, 13-15
FAILOVER, 13-3
FAILOVER_MODE, 13-15
GLOBAL_NAME, 13-6
HS, 13-6
INSTANCE_NAME, 13-5, 13-18
LOAD_BALANCE, 13-3, 13-4
METHOD, 13-16
RDB_DATABASE, 13-6
SDU, 13-6
SOURCE_ROUTE, 13-3, 13-4
TYPE, 13-15
TYPE_OF_SERVICE, 13-6
TNSPING utility, 15-9
compared to TRCROUTE utility, 15-11
TNSPING.TRACE_DIRECTORY trace parameter, 16-36
TNSPING.TRACE_LEVEL trace parameter, 16-36
tnsping.trc file, 16-33
Total Receive Buffer field in Oracle Net Manager, 9-6
Total Send Buffer field in Oracle Net Manager, 9-6
Trace Assistant, 16-6
examining trace files with, 16-44
functions of, 16-44
option reference, 16-44
trace data for IDs, 16-54
trace data for packets, 16-47
trace data statistics, 16-57
trace files
analyzing with Trace Assistant, 16-44
default names for, 16-33
error message information, 16-41
examining with Trace Assistant, 16-44
listener.trc, 16-33, 16-36
sqlnet.trc, 16-33
sqlnet.trc for clients, 16-34
svr_*pid*.trc, 16-33
svr_*pid*.trc for servers, 16-34
tnsping.trc, 16-33
trace parameters
cman.ora
TRACE_DIRECTORY, 16-38
TRACE_FILELEN, 16-38
TRACE_FILENO, 16-38
TRACE_LEVEL, 16-38
TRACE_TIMESTAMP, 16-38
listener.ora
TRACE_DIRECTORY_*listener_name*, 16-37
TRACE_FILE_*listener_name*, 16-37
TRACE_FILELEN_*listener_name*, 16-37
TRACE_FILENO_*listener_name*, 16-37
TRACE_LEVEL_*listener_name*, 16-37
TRACE_TIMESTAMP_*listener_name*, 16-37
sqlnet.ora
TNSPING.TRACE_DIRECTORY, 16-36
TNSPING.TRACE_LEVEL, 16-36
TRACE_DIRECTORY_CLIENT, 16-34
TRACE_DIRECTORY_SERVER, 16-34
TRACE_FILE_CLIENT, 16-34
TRACE_FILE_SERVER, 16-34
TRACE_FILELEN_CLIENT, 16-34
TRACE_FILELEN_SERVER, 16-34
TRACE_FILEN0_CLIENT, 16-34
TRACE_FILENO_SERVER, 16-35
TRACE_LEVEL_CLIENT, 16-35
TRACE_LEVEL_SERVER, 16-35
TRACE_TIMESTAMP_CLIENT, 16-36
TRACE_TIMESTAMP_SERVER, 16-36
TRACE_UNIQUE_CLIENT, 16-36
TRACE_DIRECTORY trace parameter, 16-38
TRACE_DIRECTORY_CLIENT trace parameter, 16-34
TRACE_DIRECTORY_*listener_name* trace parameter, 16-37
TRACE_DIRECTORY_SERVER trace parameter, 16-34
TRACE_FILE_CLIENT trace parameter, 16-34
TRACE_FILE_*listener_name* trace parameter, 16-37
TRACE_FILE_SERVER trace parameter, 16-34
TRACE_FILELEN trace parameter, 16-38
TRACE_FILELEN_CLIENT trace parameter, 16-34
TRACE_FILELEN_*listener_name* trace parameter, 16-37
TRACE_FILELEN_SERVER trace parameter, 16-34
TRACE_FILEN0_CLIENT trace parameter, 16-34
TRACE_FILENO trace parameter, 16-38
TRACE_FILENO_*listener_name* trace parameter, 16-37
TRACE_FILENO_SERVER trace parameter, 16-35
TRACE_LEVEL trace parameter, 16-38
TRACE_LEVEL_CLIENT trace parameter, 16-35
TRACE_LEVEL_*listener_name* trace parameter, 16-37
TRACE_LEVEL_SERVER trace parameter, 16-35
TRACE_TIMESTAMP networking parameter, 16-38
TRACE_TIMESTAMP trace parameter, 16-38
TRACE_TIMESTAMP_CLIENT trace parameter, 16-36
TRACE_TIMESTAMP_*listener_name* trace parameter, 16-37
TRACE_TIMESTAMP_SERVER trace parameter, 16-36
TRACE_UNIQUE_CLIENT trace parameter, 16-36
Transparent Application Failover (TAF)
and GLOBAL_DBNAME parameter, 10-6
configuring, 13-13
GLOBAL_DBNAME networking parameter in listener.ora, 10-6, 13-16
overview, 3-7
with instance role, 13-20
Transparent Network Substrate (TNS)
benefits, 5-3
described, 5-3
transport layer, 5-2
TRCROUTE utility
described, 15-11
troubleshooting, 16-1 to 16-58
client, 16-8
contacting Oracle Support Services, 16-58

log files, 16-20
loopback tests, 16-7
questions, 16-17
server, 16-7
trace files, 16-20
TTC. See Two-Task Common (TTC)
Turn Off UNIX Signal Handling option in Oracle Net Manager, 9-7
TWO_TASK environment variable, 15-4
Two-Task Common (TTC) presentation
dedicated server configurations, 5-10
described, 5-3
shared server configurations, 5-10
TYPE networking parameter, 13-15
Type of Service field in Oracle Net Manager, 13-6, 13-28
TYPE_OF_SERVICE networking parameter, 13-6, 13-28

U

Use for Heterogeneous Services option in Oracle Net Manager, 13-6, 13-27
Use Options Compatible with Net8 8.0 Clients option, 13-4
Use Oracle8*i* Release 8.0 Compatible Identification option, 13-5

V

V$SESSION table, 13-18
virtual circuits, 5-9

W

WebDAV presentation, 5-6
dedicated server configurations, 5-10
shared server configurations, 5-10
WebDAV protocol, 1-4
Windows NT services
Oracle*HOME_NAME*CMan service, 15-4
Oracle*HOME_NAME*TNSListener service, 15-2
wizards
Directory Server Migration, 6-4
Net Service Name, 6-4
Oracle Net Manager, 6-3 to 6-5

Y

ypserv program, 8-25