Net8

Administrator's Guide

Release 8.1.6

December 1999

Part No. A76933-01

Net8 Administrator’s Guide, Release 8.1.6

Part No. A76933-01

Primary Author: Deborah Steiner

Contributors: Pierre Baudin, Toby Close, Santanu Datta, Harvey Eneman, Shuvayu Kanjila, Cynthia Kibbe, Scot McKinley, Sergio Mendiola, Ed Miner, Peter Povinec, Radhika Shah, Richard Smith, Juliet Tran, Sandy Venning, and Norman Woo

Contents

Send Us Your Comments xv

Preface xvii

Part I Net8 Overview and Concepts

1 Introduction to Net8

Net8 Overview 1-2
Oracle Connectivity Overview 1-5
Configuration Overview 1-6
Naming Method Configuration 1-6
Listener Configuration on the Server 1-8

2 Net8 Concepts

Networking Environments 2-2
Distributed Processing 2-2
Distributed Databases 2-2
Connect Operations 2-3
Connecting to Servers 2-3
Disconnecting from Servers 2-6
Exception Operations 2-7
Net8 Connection Models 2-7
Multi-Threaded Server Model 2-7
Dedicated Server Model 2-13

Client Connection Options 2-20
Client Load Balancing for Multiple Listeners 2-20
Connect-Time Failover for Multiple Listeners 2-21
Transparent Application Failover for High Availability 2-21
Service Registration to the Listener 2-22
Benefits 2-23
Registration with the Default Local Listener 2-24
Registration with a Non-Default Listener 2-24
Registration with Remote Listeners 2-25
Connection Load Balancing 2-26
Configuration 2-26
Database Identification by Service Name Rather than SID 2-32
Connect Descriptors Using SERVICE_NAME and INSTANCE_NAME 2-32
Multiple Instances Belonging to the Same Service 2-33
Single Instance Belonging to Multiple Services 2-34
Migration 2-34
Service Resolution through Naming Methods 2-34
Local Naming 2-35
Directory Naming 2-35
Oracle Names 2-36
Host Naming 2-36
External Naming 2-37
Net8 and an LDAP-Compliant Directory Server 2-38
Overview 2-38
Directory Naming Overview 2-40
Directory Entries 2-42
Adding or Modifying Entries in the Directory 2-43
Client Connections Using Directory Naming 2-44
Specifying Absolute Names 2-46
Net8 Naming and Directory Design 2-50

3 Net8 Architecture

Transparent Network Substrate (TNS) 3-2
Overview of Stack Communications 3-2
Stack Communications in a Typical Net8 Client/Server Environment 3-5
Net8 Client-Side Interaction 3-5
Server-Side Interaction 3-9
Server-to-Server Interaction 3-9
Stack Communications for Clients Connecting to a Directory Server 3-10
Stack Communications for JDBC Clients 3-11
Stack Communications for Oracle8*i* JServer Clients 3-16

4 Net8 Products and Tools

Net8 Client 4-2
Net8 Server 4-2
Oracle Protocols 4-2
TCP/IP 4-2
TCP/IP with SSL 4-2
SPX 4-3
Named Pipes 4-3
LU6.2 4-4
Bequeath 4-4
Oracle Connection Manager 4-5
Oracle Connection Manager Processes 4-5
Connection Concentration 4-6
Net8 Access Control 4-8
Multi-Protocol Support 4-8
Oracle Names 4-10
Overview 4-10
Administrative Regions 4-12
Domains 4-12
Oracle Names As Data Repository 4-13
Data Stored in an Oracle Names Server 4-14
Organizing and Naming Network Components 4-16
Using Multiple Regions to Decentralize Administrative Responsibilities 4-18
Differences Between Versions of Oracle Names 4-21
Understanding Discovery 4-24

External Naming Services 4-24
Authentication Methods 4-25
Oracle Advanced Security 4-25
Network Security 4-26
Single Sign-On 4-26
Distributed Computing Environment Integration 4-27
Administration Tools 4-27
Net8 Configuration Assistant 4-27
Net8 Assistant 4-28
Control Utilities 4-28

5 Planning Your Network

Planning Overview 5-2
Defining Your Network Layout 5-2
Resolving Names to Connect Descriptors 5-3
Improving Large Network Performance 5-5
Using Shared Servers or Prespawned Dedicated Servers 5-5
Increasing Listener Queuesize 5-6
Enabling Connection Pooling 5-6
Enabling Connection Concentration 5-7
Enabling Client Load Balancing 5-8
Adjusting the Session Data Unit (SDU) Size for Data Transfer Optimization 5-8
Persistent Buffer Flushing for TCP/IP 5-9
Planning Summary 5-10

Part II Configuration

6 Configuring Naming Methods

Configuration Overview 6-2
Understanding Connect Descriptors 6-2
Configuration Models 6-4
Naming Methods 6-4
Configuring the Local Naming Method 6-5
Configuring the tnsnames.ora File During Installation 6-6
Configuring the tnsnames.ora File After Installation 6-6
Configuring the Directory Naming Method 6-16
Directory Naming Configuration Steps 6-17
Modifying Connectivity Information for Database Service Entries 6-29
Adding Users to and Removing Users from the OracleNetAdmins Group 6-31
Exporting Net Service Names from a tnsnames.ora File 6-32
Exporting Network Objects from an Oracle Names Server 6-35
Configuring the Oracle Names Method 6-47
Task 1: Consider Oracle Names Options 6-47
Task 2: Install Necessary Components 6-48
Task 3: Create an Oracle Names Server 6-49
Task 4: Configure Clients and Database Servers To Use Oracle Names Servers 6-60
Task 5: Configure the Listener 6-64
Task 6: Register Data with the Oracle Names Server 6-65
Task 7: (Optional) Delegate Domains to Delegated Administrative Regions 6-73
Task 8: Connect to the Database 6-74
Configuring the Host Naming Method 6-74
Task 1: Configure the Listener with the Global Database Name 6-75
Task 2: Configure HOSTNAME as the First Naming Method 6-76
Task 3: Set Up Host Name Resolution Environment 6-78
Task 4: Connect to the Database 6-78
Configuring External Naming Methods 6-79
Configuring NIS 6-79
Configuring NDS 6-83

7 Configuring the Listener

Configuration Overview 7-2
Default Listening Protocol Addresses 7-3
Configuration During Installation 7-3
Configuration with Net8 Assistant 7-5
Re-Starting the Listener After a listener.ora File Modification 7-5
Configuring Listener Protocol Addresses 7-6
Configuring Listening Addresses for Oracle8*i* JServer 7-8
Configuring a Listener that Uses a Non-Default Address 7-10
Configuring the LOCAL_LISTENER Parameter 7-10
Configuring a Naming Method 7-10
Handling Large Volumes of Connection Requests 7-12
Configuring Static Service Information 7-13
Configuring Prespawned Dedicated Servers 7-16

8 Enabling Advanced Net8 Features

Configuring Advanced Network Address and Connect Data Information 8-2
Creating a List of Listener Protocol Addresses 8-2
Configuring Address List Parameters 8-4
Configuring Advanced Connect Data Parameters 8-8
Configuring Transparent Application Failover 8-12
FAILOVER_MODE Parameters 8-13
TAF Implementation 8-14
Configuring Client Attributes for Names Resolution 8-17
Configuring a Default Domain for Clients 8-17
Prioritizing Naming Methods 8-19
Routing Connection Requests 8-21
Configuring Advanced Profile Information 8-23
Configuring Preferred Oracle Names Servers 8-26
Configuring Oracle Connection Manager Features 8-28
Enabling Connection Concentration 8-28
Enabling Multi-Protocol Support 8-32
Enabling Net8 Access Control 8-35
Configuring Clients for Oracle Connection Manager 8-38

Configuring Connections to Non-Oracle Database Services 8-45
Configuring Net8 for External Procedures 8-45
Configuring Net8 for Oracle Heterogeneous Services 8-49
Configuring Net8 for an Oracle Rdb Database 8-52

9 Configuring Multi-Threaded Server

Overview 9-2
Configuring MTS with the MTS_DISPATCHERS Parameter 9-4
Setting the Initial Number of Dispatchers 9-7
Enabling Connection Pooling 9-9
Allocating Resources 9-9
Using MTS on Clients 9-10
Overriding MTS on Clients 9-10

10 Enabling Net8 Enhancements for Programmers

Net8 OPEN 10-2
Net8 OPEN API Function Calls 10-3
Finding the Net8 OPEN Applications Program Interface 10-7
Building Your Own Application 10-7
Configuring the System to Use Your Net8 OPEN Application 10-8
Sample Programs 10-9
Net8 OPEN API Errors 10-9
UNIX Client Programming 10-10
Signal Handler and Alarm Programming 10-11
Bequeath 10-11

Part III Net8 Testing and Troubleshooting

11 Establishing a Connection and Testing the Network

Connecting to a Database 11-2
Using Net8 Control Utilities 11-3
Net8 Component Startup Overview 11-4
Entering a Connect String 11-10
Connect Identifier Syntax Characteristics 11-10
Initiating Connections 11-11

Testing the Network 11-13
Testing an Oracle Names Server 11-13
Testing Network Objects Using Net8 Assistant or NAMESCTL Utility 11-14
Testing a Listener 11-15
Testing Oracle Connection Manager 11-15
Testing Configuration on the Server 11-15
Testing Network Connectivity from the Client 11-17

12 Troubleshooting

Diagnosing Net8 12-2
Server Diagnostics 12-3
Client Diagnostics 12-4
Resolving the Most Common Error Messages 12-7
Troubleshooting Tips from the Field 12-14
Questions to Ask When Troubleshooting 12-15
Troubleshooting Network Problems Using Log and Trace Files 12-16
Logging Error Information 12-16
Error Stacks 12-17
Log File Names 12-19
Understanding and Setting Log Parameters 12-19
Setting Log Settings During Runtime of Control Utilities 12-23
Using Log Files 12-24
Understanding Listener Log Files 12-25
Understanding Oracle Connection Manager Logs 12-29
Tracing Error Information 12-33
Trace File Names 12-34
Understanding and Setting Trace Parameters 12-34
Setting Trace Settings During Runtime of Control Utilities 12-42
Evaluating Net8 Traces 12-43
Using the Trace Assistant to Examine Your Trace Files 12-46
Contacting Oracle Customer Support 12-59

Part IV Reference

A Control Utilities

Operating Modes A-2
Listener Control Utility (LSNRCTL) A-3
SET and SHOW Commands A-3
Security A-3
LSNRCTL Commands A-4
Oracle Names Control Utility (NAMESCTL) A-26
SET and SHOW Commands A-26
Security A-26
NAMESCTL's Distributed Operation A-27
Confirmation Mode in NAMESCTL A-27
NAMESCTL Commands A-28
Oracle Connection Manager Control Utility (CMCTL) A-79
SET and SHOW Commands A-80
Shutdown Commands A-80
CMCTL Commands A-81

B Protocol Addresses

ADDRESSes and ADDRESS_LISTs B-2
Protocol Parameters B-3
Recommended Port Numbers B-4
Port Number Limitations B-5

C Configuration Parameters

Syntax Rules for Configuration Files C-2
Further Syntax Rules for Configuration Files C-3
Network Character Set C-4
Character Set C-5
Profile Parameters (sqlnet.ora) C-5

Local Naming Parameters (tnsnames.ora) C-31
General Syntax C-31
Multiple Descriptions C-32
Multiple Address Lists C-32
Changed Functionality of Client Load Balancing and DESCRIPTION_LISTs C-33
Examples C-35
DESCRIPTIONS and DESCRIPTION_LISTs C-37
ADDRESSes and ADDRESS_LISTs C-38
Optional Parameters for Lists C-39
CONNECT_DATA Section C-42
Listener Parameters (listener.ora) C-48
Listener Address Section C-49
SID_LIST_*listener_name* Static Service Section C-50
Control Parameters C-54
Oracle Names Parameters (names.ora) C-59
Oracle Connection Manager Parameters (cman.ora) C-70
Protocol-Specific Parameters (protocol.ora) C-77
Directory Server Access Parameters (ldap.ora) C-78
Configuration File Changes in Release 8.1 .... C-79
Obsolete Parameters for Release 8.1 .... C-80
Configuration File Syntax Changes C-81
tnsnames.ora Changes C-86
cman.ora Changes C-88

D LDAP Schema for Net8

Structural Object Classes D-2
orclDBServer D-2
orclNetService D-2
orclNetDescription D-3
orclNetDescriptionList D-3
orclNetAddress D-4
orclNetAddressList D-4
Attributes D-5

Glossary

Index

List of Figures

1–1 Basic Client-to-Server Connection 1-2
1–2 Listener in a Connection Request 1-3
1–3 Client to Server Connections through Oracle Connection Manager 1-4
1–4 Client Using a Naming Method to Resolve a Connect Identifier 1-7
2–1 Listener In a Typical Net8 Connection 2-5
2–2 MTS Connection Model 2-8
2–3 Direct Hand-Off to a Dispatcher 2-10
2–4 Redirected Connection to a Dispatcher 2-11
2–5 Connection Pooling 2-12
2–6 Dedicated Server Model 2-13
2–7 Redirected Connection to a Dedicated Server Process 2-15
2–8 Connections To Prespawned Dedicated Servers 2-16
2–9 Redirected Connection to a Prespawned Dedicated Server Process 2-18
2–10 Service Registration 2-22
2–11 Load Balancing Environment 2-28
2–12 Load Balancing Example 2-31
2–13 Client Using a Directory to Resolve a Connect Identifier 2-39
2–14 Database Service in a Directory 2-40
2–15 Domain Component DIT 2-42
2–16 X.500 Style DIT 2-42
2–17 Creating Entries in the Directory with Applications 2-43
2–18 Relative Naming 2-45
2–19 Absolute Name for X.500 Namespaces 2-47
2–20 Absolute Name for Domain Component Namespaces 2-49
2–21 Directory User Groups 2-52
3–1 OSI Communications Stack 3-3
3–2 Typical Communications Stack in an Oracle Environment 3-6
3–3 Typical Client vs. Net8 LDAP Client Communications Stack 3-10
3–4 Typical Client vs. JDBC Client Communications Stack 3-12
3–5 Java Client/Server Communications Stack in a JDBC Environment 3-15
3–6 Typical Net8 Client vs. Oracle8i JServer Client Communication Stack 3-16
3–7 Client/Server Communications Stack in an Oracle8i JServer Environment 3-17
4–1 Connection Concentration through Oracle Connection Manager 4-7
4–2 Multi-Protocol Support Through Oracle Connection Manager 4-9
4–3 Oracle Names 4-11
4–4 Single Domain Naming Model 4-16
4–5 Hierarchical Naming Model 4-17
4–6 Delegated Administrative Regions 4-20
6–1 Single Domain Export to Same DIT Node 6-37
6–2 Root and Subdomain Domain Export to Similar DIT 6-38

6–3 Single Domain Export to One DIT Node 6-40
6–4 Root and Subdomain Domain Export to Dissimilar DIT 6-41
6–5 Root and Subdomain Domain Export to One DIT Node 6-43
6–6 Multiple Domain Export to Two Subtrees 6-45
9–1 MTS Architecture 9-3
10–1 Net8 OPEN 10-2
12–1 Network Products and Error Stack Components 12-17

Send Us Your Comments

Net8 Administrator's Guide, Release 8.1.6

Part No. A76933-01

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

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

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

- E-mail - infodev@us.oracle.com
- FAX - (650) 506-7228. Attn: Information Development
- Postal service:
  Oracle Corporation
  Server Technologies Documentation Manager
  500 Oracle Parkway, 4OP12
  Redwood Shores, CA 94065
  USA

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

____

____

____

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

Preface

Net8 Administrator's Guide provides the information you need to understand and use the Net8 product and its related applications.

Note: This guide contains information that describes the Net8 features and functionality within the Oracle8*i* and the Oracle8*i* Enterprise Edition products. Oracle8*i* and the Oracle8*i* Enterprise Edition have the same basic features. However, several advanced Net8 features are available only with Oracle8*i* Enterprise Edition. For example, to use Oracle Connection Manager, you must have Oracle8*i* Enterprise Edition. For information about the differences between Oracle8*i* and the Oracle8*i* Enterprise Edition and the features and options that are available to you, please refer to *Getting to Know Oracle8i*.

This Preface includes the following topics:

- Audience
- How this Book is Organized
- What's New in Oracle8i
- Related Documents
- Conventions
- Your Comments Are Welcome

Audience

The information in this guide is intended primarily for network or database administrators (DBAs) responsible for making connection to services, such as the Oracle8*i* database, and network configuration. This guide is also provided for anyone who wants to understand how Net8 works.

How this Book is Organized

This guide is divided into the following parts, chapters, and appendices:

Part I: Net8 Overview and Concepts	
Chapter 1, "Introduction to Net8"	Introduces Net8
Chapter 2, "Net8 Concepts"	Describes multi-threaded server and dedicated server connection models, client connection features, listener features, naming methods, and Net8 support of LDAP-compliant directory services
Chapter 3, "Net8 Architecture"	Describes the Net8 architecture
Chapter 4, "Net8 Products and Tools"	Describes Net8 products and Net8 administration tools
Chapter 5, "Planning Your Network"	Describes considerations for planning a network using Net8. It explains the relationships of the Net8 products, and options for better managing your future network
Part II: Configuration	
Chapter 6, "Configuring Naming Methods"	Describes how to configure the network using various Net8 methods
Chapter 7, "Configuring the Listener"	Describes how to configure the listener
Chapter 8, "Enabling Advanced Net8 Features"	Describes how to configure advanced and optional Net8 connection features, such as Oracle Connection Manager, client load balancing, connect-time failover, heterogeneous services, external procedures, and Oracle Rdb databases
Chapter 9, "Configuring Multi-Threaded Server"	Describes how to tune the components of the multi-threaded server architecture
Chapter 10, "Enabling Net8 Enhancements for Programmers"	Describes Net8 enhancements for programmers. This includes a review of Net8 Open, UNIX signal handling, and the bequeath adapter.

Part III: Net8 Testing and Troubleshooting

Chapter 11, "Establishing a Connection and Testing the Network"	Describes how to start Net8 components, establish a connection, and test a connection
Chapter 12, "Troubleshooting"	Describes procedures to troubleshoot Net8. This includes information on tracing and logging

Part IV: Reference

Appendix A, "Control Utilities"	Describes all commands for Net8 control utilities including the Listener Control Utility (LSNRCTL), Oracle Names Control Utility (NAMESCTL), and Oracle Connection Manager Control Utility (CMCTL)
Appendix B, "Protocol Addresses"	Describes how to configure protocol addresses
Appendix C, "Configuration Parameters"	Lists and describes configuration parameters for the networking configuration files
Appendix D, "LDAP Schema for Net8"	Describes the Net8 object classes and attributes stored in LDAP-compliant directory service schema

What's New in Oracle8*i*

This section briefly lists the new Oracle8*i* features and provides pointers to other chapters in the document where you can get additional information. New features include:

- Directory Naming
- New Listener Features
- New Client Connections Features
- New Products
- Oracle8i Database Features that Affect Net8

Directory Naming

Network information can now be stored in a centralized LDAP-compliant directory service, including Oracle Internet Directory, Microsoft's Active Directory, and Novell Directory Services.

> **See Also:** "Net8 and an LDAP-Compliant Directory Server" on page 2-38

New Listener Features

Feature	See Also
Service Naming—Oracle8*i* database services are identified by a service name rather than an Oracle System Identifier (SID). This impacts the way connect descriptors are defined.	"Database Identification by Service Name Rather than SID" on page 2-32
Direct Hand-Off—The listener has the ability to hand of requests directly to an MTS dispatcher, without issuing a redirect message back to the client.	"Service Registration to the Listener" on page 2-22
Service Registration—Oracle8*i* database instances register information with the listener during database startup. This eliminates the need to configure the `listener.ora` file with service information.	"Multi-Threaded Server Model" on page 2-7
Connection Load Balancing—The listener is able to balance the number of active connections among various instances and MTS dispatchers for the same service.	"Connection Load Balancing" on page 2-26

New Client Connections Features

Feature	More Information
Client Load Balancing—When more than one listener supports a service, a client can randomize requests to the various listeners.	"Client Load Balancing for Multiple Listeners" on page 2-20
Connect-Time Failover—When more than one listener supports a service, a client can be configured to fail over the client request to a different listener if the first listener fails.	"Connect-Time Failover for Multiple Listeners" on page 2-21

New Oracle Connection Manager Features

Feature	More Information
The Oracle Connection Manager Control Utility (CMCTL) has been enhanced with several new commands that allow for greater control.	"Oracle Connection Manager Control Utility (CMCTL)" on page A-79

New Products

Feature	More Information
Net8 Configuration Assistant—A post-installation tool that performs basic configuration. After installation, it automatically configures default configuration files.	"Net8 Configuration Assistant" on page 4-27 and operating-system installation guides
TCP/IP with Secure Sockets Layer (SSL) protocol—A protocol for client/server authentication over a network using TCP/IP and the Secure Sockets Layer (SSL).	"TCP/IP with SSL" on page 4-2

Oracle8*i* Database Features that Affect Net8

Feature	More Information
Database services are identified by service name and instance name rather than Oracle System Identifier (SID).	"Database Identification by Service Name Rather than SID" on page 2-32

Related Documents

- *Oracle8i Concepts*
- *Oracle8i Reference*
- *Oracle8i Distributed Database Systems*
- *Oracle8i Designing and Tuning for Performance*
- *Oracle Enterprise Manager Administrator's Guide*
- *Oracle Advanced Security Administrator's Guide*

Conventions

The following conventions are used in this book:

Convention	Meaning
UPPERCASE	Calls attention to SQL commands, keywords, and initialization parameters.
bold	Boldface test indicates a term defined in the glossary.
`lowercase courier`	Indicates file names, directories and commands.
`constant width`	This typeface is used for user input and code examples.
Italic	Terms in italics indicate a variable or special emphasis. Italic is also used for book titles.
$ORACLE_HOME on UNIX and *ORACLE_HOME* on Windows platforms	In this Optimal Flexible Architecture (OFA)-compliant release, all subdirectories are no longer under a top level `ORACLE_HOME` directory. There can be an `ORACLE_BASE` directory, which may contain multiple Oracle home directories.
`< >`	Angle brackets enclose user-supplied names.
`[ ]`	Brackets enclose a choice of optional items from which you can choose one or none.
`{ }`	Curly brakets enclose required items.
`...`	Horizontal ellipsis points in code samples mean that parts of the sample have been omitted.

Your Comments Are Welcome

We value and appreciate your comment as an Oracle user and reader of our manuals. As we write, revise, and evaluate our documentation, your opinions are the most important feedback we receive.

You can send comments and suggestions about this reference to the Information Development department at the following e-mail address:

infodev@us.oracle.com

If you prefer, you can send letters or faxes containing your comments to:

Server Technologies Documentation Manager
Oracle Corporation
500 Oracle Parkway, 4OP12
Redwood Shores, CA 94065
Fax: (650) 506-7228 Attn.: Information Development

Part I

Net8 Overview and Concepts

Part I provides an overview of Net8 concepts, products, and tools. It contains the following chapters:

- Chapter 1, "Introduction to Net8"
- Chapter 2, "Net8 Concepts"
- Chapter 3, "Net8 Architecture"
- Chapter 4, "Net8 Products and Tools"
- Chapter 5, "Planning Your Network"

1

Introduction to Net8

This chapter introduces Net8 and provides an overview of its main applications, features, and functionality. It includes the following sections:

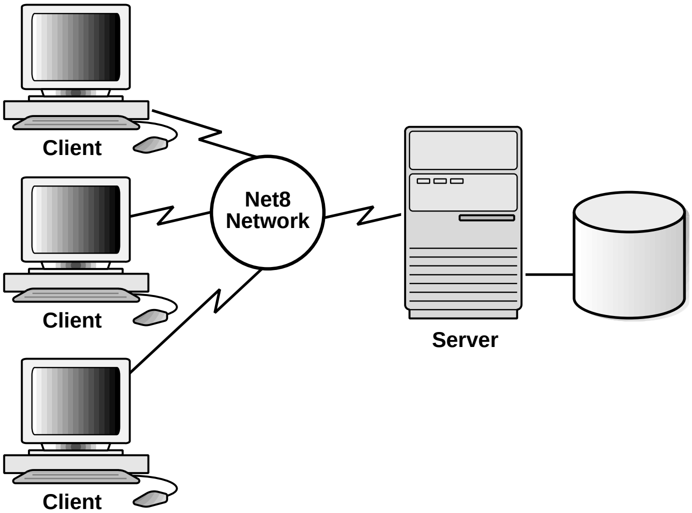
- Net8 Overview
- Oracle Connectivity Overview
- Configuration Overview

Net8 Overview

Net8 enables services and their applications to reside on different computers and communicate as peer applications. The main function of Net8 is to establish network sessions and transfer data between a client machine and a server or between two servers. Net8 is located on each machine in the network. Once a network session is established, Net8 acts as a data courier for the client and the server.

Figure 1–1 shows a client-to-server connection.

Figure 1–1 Basic Client-to-Server Connection

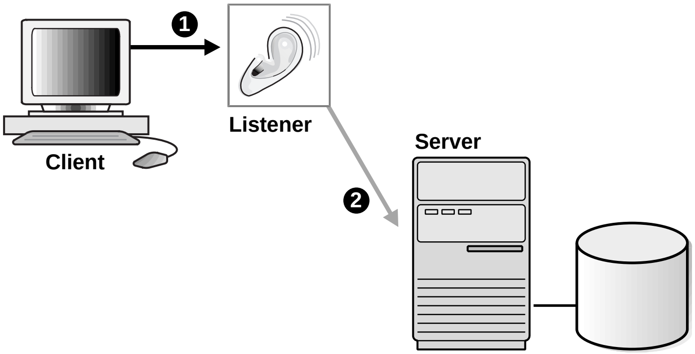


Network sessions are established with the help of a **listener**. The listener is a separate process that resides on the server. The listener receives incoming client connection requests and hands these requests to the server.

The listener brokers the client request, handing off the request to the server. Every time a client or server acting as a client requests a network session with a server, a listener receives the actual request.

Figure 1–2 shows a listener in a connection request.

Figure 1–2 Listener in a Connection Request



For environments in which a large number of connections must access the same service, Net8 offers a connection routing process called **Oracle Connection Manager** that usually resides on a machine separate from the client or server.

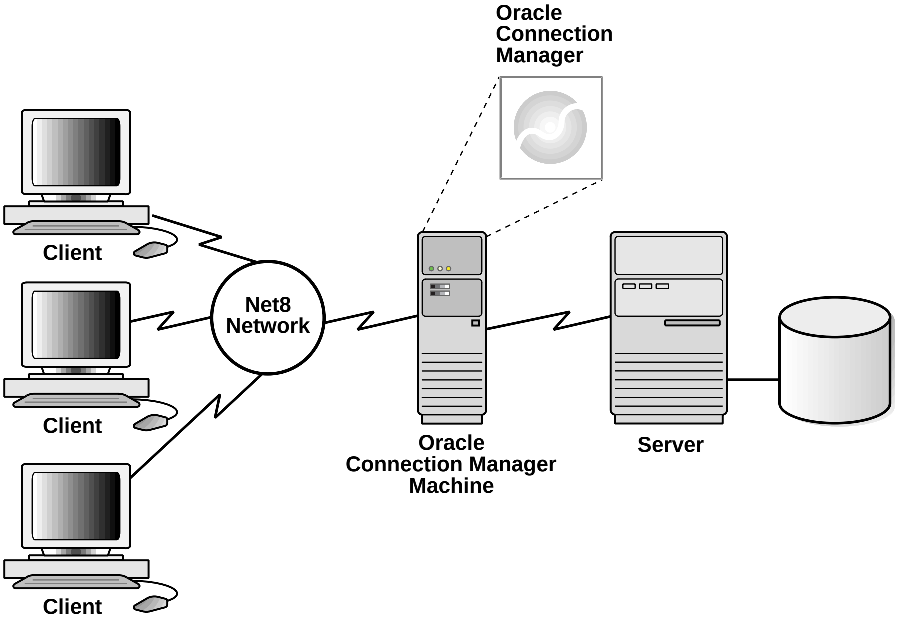
Net8 offers connection routing for environments where:

- Large numbers of users need to access a single service through a single protocol
- Different protocols are installed on the client and server, making a typical connection fail

  Typical Net8 connections require the client and server to have the same protocol installed.
- Control of client access to designated servers in a TCP/IP environment is required

Figure 1–3 shows how client connections are routed to Oracle Connection Manager, which resides on a separate machine in the network.

Figure 1–3 Client to Server Connections through Oracle Connection Manager



See Also: "Oracle Connection Manager" on page 4-5 for product information

Oracle Connectivity Overview

Net8 allows connections to various services, such as Oracle databases and non-Oracle databases. Because an Oracle database is the most common service in an Oracle environment, this section focuses on database services.

When a user connects to a database service from across the network, a **connect descriptor** containing network information about the destination service is passed to the listener.

A connect descriptor contains the:

- Network route to the service, including the location of the listener through a **protocol address**
- Service name that is typically the **global database name**—a name comprised of the database name and database domain—used to identify the database service

Connecting with a complete connect descriptor creates a lengthy **connect string**, as shown in a connection to a database service named `sales.us.acme.com` in the following example:

```
CONNECT
scott/tiger@(description=(address=(protocol=tcp)(host=sales-server)(port=1521))
(connect_data=(service_name=sales.us.acme.com)))
```

To avoid a lengthy connect string, a connect descriptor can be mapped to a **connect identifier**. A connect identifier can be a **net service name** or the actual name of the service. This information is then stored in at least one **naming method**. Clients need only to use the connect identifier in the connect string. For example, if a net service name called `sales` is mapped to the connect descriptor used in the example above, clients can use the following connect string:

```
CONNECT scott/tiger@sales
```

During a connection request, the client contacts a naming method to resolve `sales` to a connect descriptor. The client then forwards the request to the listener address specified in the connect descriptor.

The listener, through a protocol, accepts the client connection. It compares the client information with the information it has received from the database, as well as information it has stored in its own configuration file, `listener.ora`. If the information matches, a connection is granted.

Configuration Overview

Configuring the network for connections consists of two main tasks described in the following sections:

- Naming Method Configuration
- Listener Configuration on the Server

Naming Method Configuration

A naming method must be configured to map connect identifiers to connect descriptors.

Net8 supports several categories of naming methods, including:

- Local Naming

  Net service names can be stored in a local configuration file called `tnsnames.ora` on each client in the network. Because the `tnsnames.ora` file can be configured on individual clients, it allows you to fine tune for a particular client's needs.

- Directory Naming

  Service addresses and net service names can be stored in a Lightweight Directory Access Protocol (LDAP)-compliant directory server.

- Oracle Names

  Service addresses and net service names can be stored in a system of Oracle Names server. Oracle Names is an Oracle-specific name service that maintains a central store of service addresses.

- Host Naming

  Service names can be stored using an existing IP address translation mechanism.

- External Naming

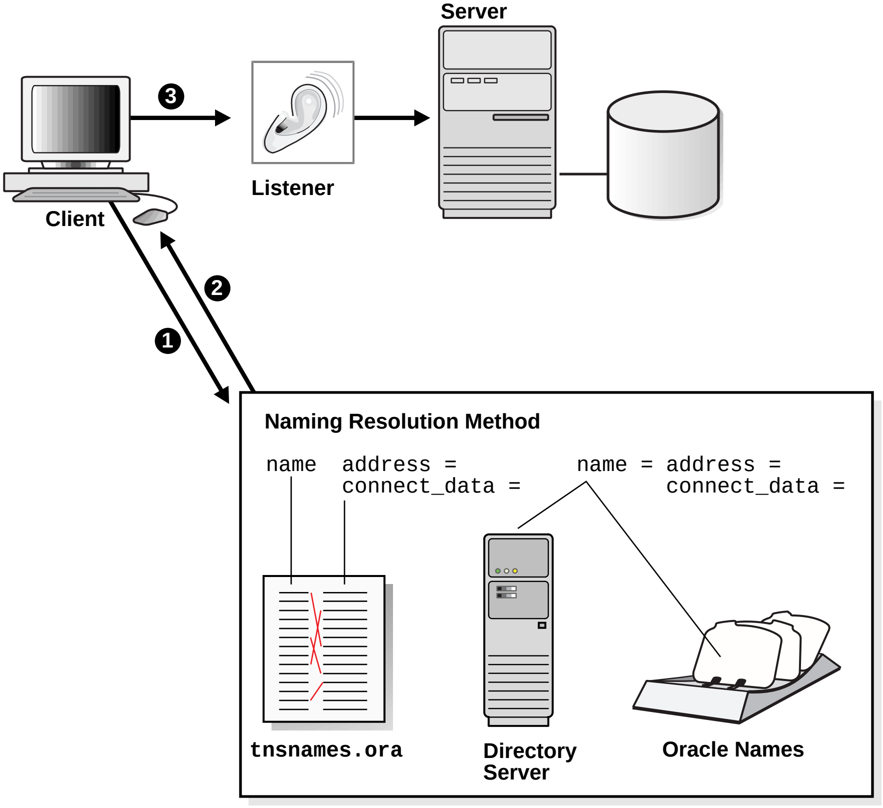
  If a third-party naming service is already configured in your environment, Net8 supports storing service address information in it to avoid additional configuration.

Once a naming method is configured, clients must be enabled to access the naming method.

Figure 1–4 shows a client resolving a connect identifier through a `tnsnames.ora` file, directory server, and Oracle Names server:

1. The client contacts a naming method to resolve a connect identifier to a connect descriptor.
2. The naming method resolves the connect identifier to a connect descriptor.
3. The client forwards the request with the connect descriptor to the listener address.

Figure 1–4 Client Using a Naming Method to Resolve a Connect Identifier



See Also: Chapter 6 for configuration details

Listener Configuration on the Server

A listener is configured with one or more listening protocol addresses and information about the destination service.

Protocol addresses are configured in the listener configuration file, `listener.ora`. Service information may or may not be configured in the `listener.ora` file:

- An Oracle8*i* database automatically registers certain information with the listener, such as its service name(s), instance name(s), service handlers, and load information. This feature, called service registration, does not require configuration in the `listener.ora` file.
- Other services, including an Oracle7 or Oracle8 release 8.0 database and the Oracle Enterprise Manager management tools, require service configuration in the `listener.ora` file.

See Also: Chapter 7 for configuration details

2

Net8 Concepts

This chapter provides an overview of how clients use Net8 in connections to servers. This chapter includes the following sections:

- Networking Environments
- Connect Operations
- Exception Operations
- Net8 Connection Models
- Client Connection Options
- Service Registration to the Listener
- Connection Load Balancing
- Database Identification by Service Name Rather than SID
- Service Resolution through Naming Methods
- Net8 and an LDAP-Compliant Directory Server

Networking Environments

Oracle networking environments are based on two concepts:

- Distributed Processing
- Distributed Databases

Distributed Processing

Oracle databases and client applications operate in what is known as a distributed processing environment. Distributed or cooperative processing involves interaction between two or more computers to complete a single data transaction. Applications, such as an Oracle tool, act as clients requesting data to accomplish a specific operation. Database servers store and provide the data.

In a typical network configuration, clients and servers may exist as separate logical entities on separate physical machines. This configuration enables for a division of labor where resources are allocated efficiently between a client workstation and the server machine. Clients normally reside on desktop computers with just enough memory to execute user friendly applications, while a server has more memory, disk storage, and processing power to execute and administer the database.

Distributed Databases

This type of client-server architecture also enables you to distribute databases across a network. A distributed database is a network of databases stored on multiple computers that appears to the user as a single logical database. Distributed database servers are connected by a database link, or path from one database to another. One server uses a database link to query and modify information on a second server as needed, thereby acting as a client to the second server.

Connect Operations

Net8 supports two types of connect operations:

- Connecting to Servers
- Disconnecting from Servers

Connecting to Servers

Users initiate a connect request by passing a user name and password along with an identifier for the service to which they want to connect. The identifier, called a **connect identifier**, identifies the following:

- The destination service, and
- The path, or network route access to that service

A connect identifier is specified in several different ways. One of the most common ways is through use of a **net service name**, another name for the service, which maps to a **connect descriptor**. A connect descriptor contains destination service and network route information. The destination service is indicated by using its **service name** for Oracle8*i* databases or its **Oracle System Identifier (SID)** for Oracle8 release 8.0 or Oracle7 databases. The network route provides, at a minimum, the location of the listener through use of a network address.

Shown here is a net service name called `sales` mapped to a connect descriptor:

```
sales=
(description=
  (address=(protocol=tcp)(host=sales-server)(port=1521))
  (connect_data=
    (service_name=sales.us.acme.com))
```

The listener, located on `sales-server`, listens with the TCP/IP protocol on port 1521 for connection requests for a database service called `sales.us.acme.com`.

Clients can use the complete connect descriptor or the connect identifier in a connect string. The following examples demonstrate one connect string that uses a complete connect descriptor and another connect string that uses the `sales` net service name.

```
CONNECT
scott/tiger@(description=(address=(protocol=tcp)(host=sales-server1)(port=1521))
(connect_data=(service_name=sales.us.acme.com))


CONNECT scott/tiger@sales
```

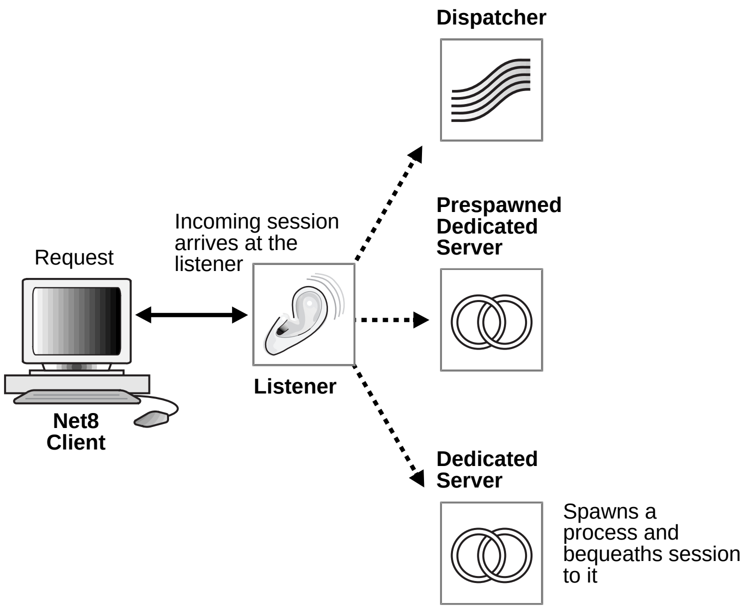
When a net service name is used, connection processing takes place by first mapping `sales` to the connect descriptor. The connect descriptor is used to locate the listener and is then passed to the listener to locate the database. It is the job of the listener to determine what kind of **service handler** can best service the client's request. A service handler can be a multi-threaded server **dispatcher**, **dedicated server**, or **prespawned dedicated server**.

See Also:

- "Database Identification by Service Name Rather than SID" on page 2-32 for an overview of service name and SID
- "Net8 Connection Models" on page 2-7 for information about service handlers

Figure 2–1 shows the role of the listener in a typical Net8 connection.

Figure 2–1 Listener In a Typical Net8 Connection


Dispatcher
Prespawned Dedicated Server
Incoming session arrives at the listener
Request
Listener
Net8 Client
Dedicated Server
Spawns a process and bequeaths session to it

Based on configuration, the listener performs one of the following actions:

- Hands the 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 to the dispatcher or dedicated server process.
- Spawns a dedicated server process and passes the client connection to the dedicated server process.

Disconnecting from Servers

Requests to disconnect from the server can be initiated as described in the following sections:

- User Initiated Disconnect
- Additional Connection Request
- Abnormal Connection Termination
- Timer Initiated Disconnect or Dead Connection Detection

User Initiated Disconnect

A user can request a disconnection from the server when a client-server transaction completes. A server can also disconnect from a second server when all server-server data transfers have been completed and no need for the link remains.

Additional Connection Request

If a client application is connected to a server and requires access to another user account on the same or on another server, most Oracle tools first disconnect the application from the server to which it is currently connected. Once the disconnection is completed, a connection request to the new user account on the appropriate server is initiated.

Abnormal Connection Termination

Other components occasionally disconnect or abort communications without notifying Net8. In this event, Net8 recognizes the failure during its next data operation, and cleans up client and server operations, effectively disconnecting the current operation.

Timer Initiated Disconnect or Dead Connection Detection

Timer initiated disconnect, or dead connection detection enables Net8 to identify connections that have been left hanging by the abnormal termination of a client. This feature minimizes the waste of resources by connections that are no longer valid. It also automatically forces a database rollback of uncommitted transactions and locks held by the user of the broken connection.

On a connection with dead connection detection enabled, a small probe packet is sent from server to client at a user-defined interval (usually several minutes). If the connection is invalid, the send operation generates an error and the server process terminates the connection.

Exception Operations

Net8 supports three types of exception operations:

- Initiate a break over the connection
- Reset a connection for synchronization after a break
- Test the condition of the connection for incoming break

The user controls only one of these three operations, that is, the initiation of a break. When the user presses the Interrupt key (Ctrl-C on some machines), the application calls this operation. Additionally, the database can initiate a break to the client if an abnormal operation occurs, such as during an attempt to load a row of invalid data using SQL*Loader.

Net8 Connection Models

Connections between clients and servers are established using two different models:

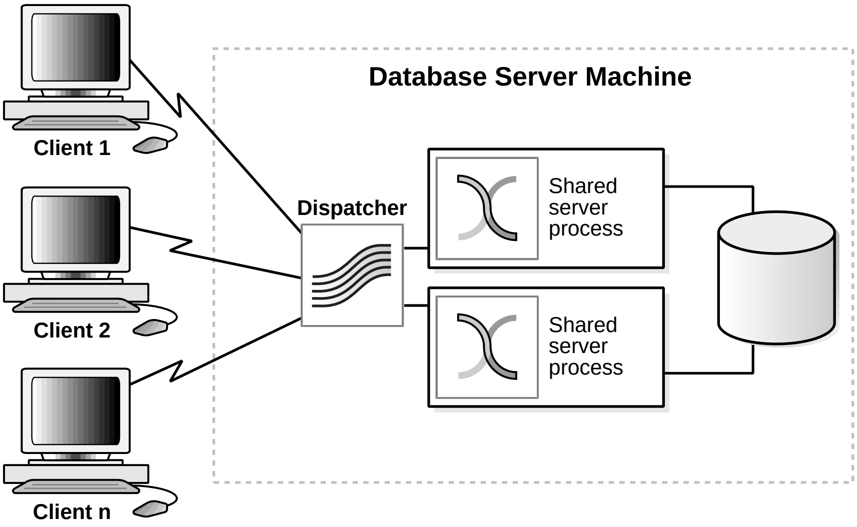
- Multi-Threaded Server Model—The listener passes the connection to the dispatcher or redirects the clients to connect through a dispatcher.
- Dedicated Server Model—The listener starts a dedicated server and passes the connection to the dedicated server or redirects the clients to the dedicated server.

Multi-Threaded Server Model

In a **multi-threaded server (MTS)** configuration, client user processes connect to a 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. A significant advantage of the MTS model over the dedicated server model is the reduction of the use of system resources, enabling the support of an increased number of users.

Figure 2–2 shows client connection requests being routed to one dispatcher.

Figure 2–2 MTS Connection Model



When a database has been configured with MTS, incoming network connection requests are always routed to the dispatcher unless either the network session specifically requests a dedicated server or no dispatchers are available. The sequence of events that occurs with the dispatcher is as follows:

1. The listener is started and listens on either a default address or the addresses specified in its configuration file, `listener.ora`.
2. The database **instance** is started. Dispatchers start according to the configuration parameters in the **initialization parameter file**. Each dispatcher then listens on the address assigned to it.
3. Each dispatcher's address is registered with the listener through **service registration**. It can take up to a minute before the dispatchers are registered with the listener.

   When the listener is not listening on its default address, the listener's network name can be specified in the initialization parameter file. The name can resolve to more than one such address if multiple listeners are used.

   If Step 1 is performed before Step 2, there can be a delay as the server attempts to connect to the listener.

The listener and the dispatcher are now ready to receive incoming network sessions.

> **Note:** You can check which dispatcher addresses have registered with the listener by issuing a SERVICES command from the Listener Control Utility. For more information, see "SERVICES" on page A-9.

4. The client connects to the listener using the listener address in the connect descriptor.

   If connection requests come in a time frame where no dispatchers are registered, these requests can be handled by a dedicated server (configured in the `listener.ora` file). If you want a client to always a use dispatcher, the client can be configured with (SERVER=SHARED) in the connect data portion of the connect descriptor, as shown in the following example connect descriptor. With this configuration, if no shared servers are available, the client connection request fails.

   ```
   sales=
   (description=
     (address=(protocol=tcp)(host=sales1-server)(port=1521))
     (connect_data=
       (service_name=sales.us.acme.com)
       (server=shared)))
   ```

   See Also: "Using MTS on Clients" on page 9-10 for information about MTS configuration on clients

5. The listener receives the connect request, and determines if the client's request can be serviced. If not, the listener refuses the network connection request and resumes at Step 8.

6. If the client's request is valid, the listener performs one of the following:

   - The listener hands the connection request directly to a dispatcher.
   - The listener issues a redirect message to the client containing the network address of a dispatcher. The client then dissolves the network session to the listener, and establishes a network session to the dispatcher using the network address provided in the redirect message.

Direct hand-off to a dispatcher is supported if the dispatcher is local to the listener for the following protocols:

- IPC protocol
- TCP/IP protocol

A redirected connection is used if the dispatcher is remote.

7. The dispatcher updates the listener with significant load information to allow the listener to balance the incoming network connection requests across dispatchers.
8. The listener resumes listening for incoming network sessions.

Figure 2–3 shows the listener handing the connection request directly to the dispatcher.

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

Figure 2–3 Direct Hand-Off to a Dispatcher

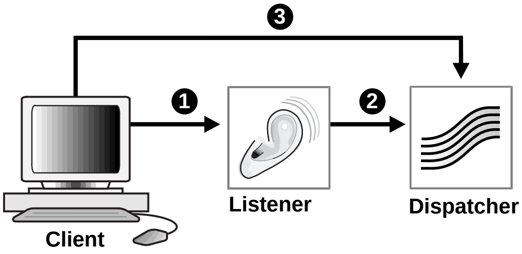
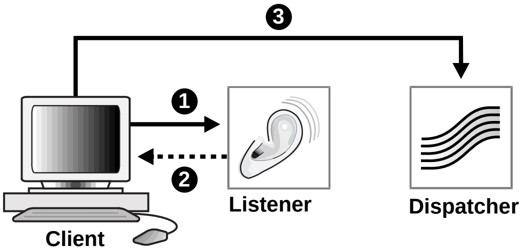


Figure 2–4 shows the role of a dispatcher in a redirected connection.

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 2–4 Redirected Connection to a Dispatcher



Configuration

To enable MTS, the MTS_DISPATCHERS parameter must be set in the initialization parameter file.

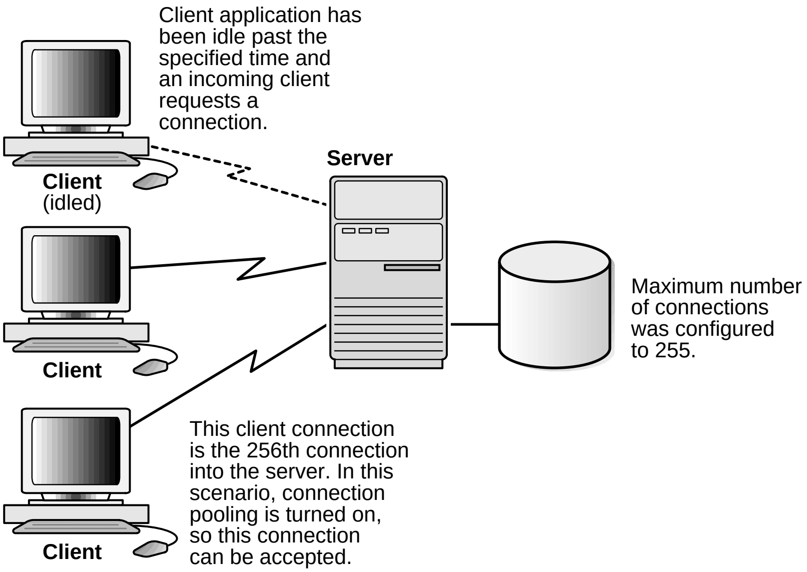
> **See Also:** Chapter 9 for information about configuring the MTS_DISPATCHERS parameter

Connection Pooling

Connection pooling is a resource utilization and user scalability feature of MTS that enables you to maximize the number of network sessions over a limited number of connections to a dispatcher. This is achieved by sharing or pooling a set of connections.

Figure 2–5 shows how connection pooling works when the number of connection is set to 255.

Figure 2–5 Connection Pooling


Client application has been idle past the specified time and an incoming client requests a connection.
Server
Client (idled)
Client
Maximum number of connections was configured to 255.
Client
This client connection is the 256th connection into the server. In this scenario, connection pooling is turned on, so this connection can be accepted.

By using a time-out mechanism to temporarily release network connections that have been idle for a specified period of time, connection pooling makes these network connections available for incoming clients, while still maintaining a network session with the idle network connection. When the idle client has more work to do, the network connection is reestablished with the dispatcher.

> **See Also:** "Enabling Connection Pooling" on page 9-9 for configuration information

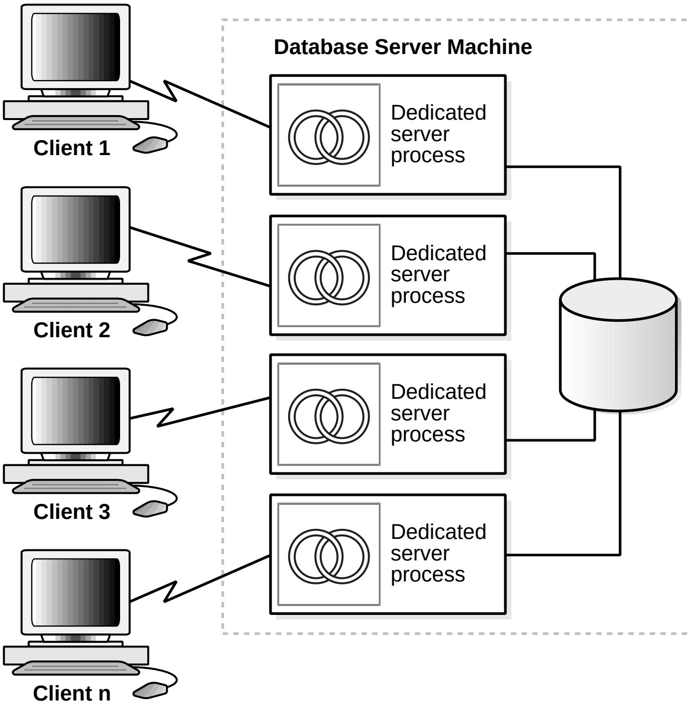
Dedicated Server Model

Alternatively to MTS, Net8 can direct a client connection request to a **dedicated server**. In this model, there is one server process for each client. In order for clients to connect to dedicated servers, the listener and the **instance** must be running on the same machine.

> **Note:** Dedicated server processes require more memory than MTS.

Figure 2–6 shows client requests being routed to dedicated servers.

Figure 2–6 Dedicated Server Model



A client can be configured to override MTS configuration by setting (SERVER=DEDICATED) in the connect data portion of the connect descriptor, as in the following example:

```
sales=
(description=
  (address=(protocol=tcp)(host=sales1-server)(port=1521))
  (connect_data=
    (service_name=sales.us.acme.com)
    (server=dedicated)))
```

> **See Also:** "Overriding MTS on Clients" on page 9-10 for information about overriding MTS configuration on clients

The connection sequence that occurs with a dedicated server is as follows:

1. The listener is started and listens on either a default address or the addresses specified in its configuration file, `listener.ora`.
2. A database instance is started.
3. The listener is updated with the instance information.

   The listener is now ready to receive incoming network connections.

   > **Note:** You can check which dedicated server addresses have registered with the listener by issuing a SERVICES command from the Listener Control Utility. For more information, see "SERVICES" on page A-9.

4. The client connects to the listener using the listener protocol address in the connect descriptor.
5. The listener receives the connect request, performs the connection handshake, and determines if the client's request can be serviced. If not, the listener refuses the network connection and resumes at Step 8.
6. The listener spawns a dedicated server, performing one of the following:
   - The listener passes the client connection to the spawned dedicated server.
   - The dedicated server informs the listener of the address that it is listening on. The listener issues a redirect message back to the client and dissolves the connection. The client connects to the dedicated server directly using the address provided in the redirect message.

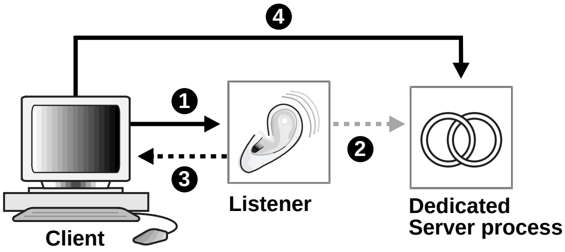
The selection of the option is based on the operating system and the protocol being used.

7. The listener resumes listening for incoming network sessions.

Figure 2–7 shows the role of a dedicated server in a redirected connection.

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

Figure 2–7 Redirected Connection to a Dedicated Server Process



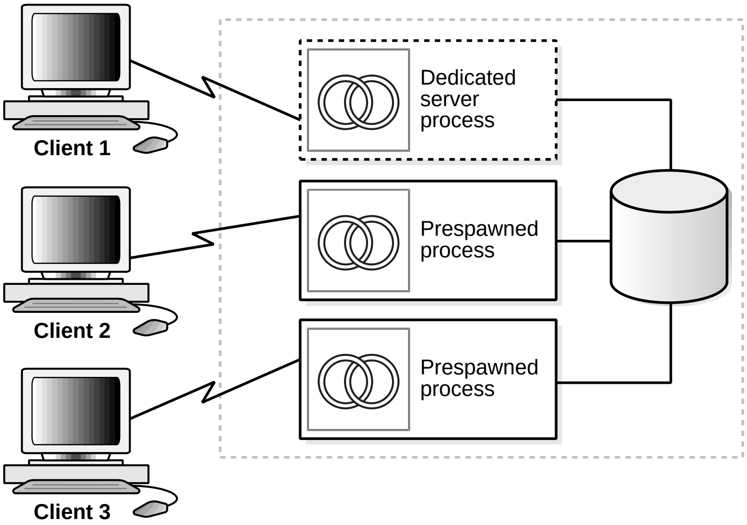
Prespawned Dedicated Servers

Net8 provides the option of automatically creating dedicated server processes before a connect request is received. These processes last for the life of the listener, and can be reused by subsequent connection requests.

Note: Prespawned dedicated servers require SQL*Net version 2.1 or later, and Oracle Server release 7.1 or later.

Figure 2–8 shows client requests being routed to prespawned dedicated servers.

Figure 2–8 Connections To Prespawned Dedicated Servers



The sequence of events that occurs when using prespawned dedicated server processes to service client connection requests is as follows:

1. The listener is started and listens on an address specified in its configuration file, `listener.ora`.
2. The listener spawns a series of dedicated servers until it reaches the specified pool size defined in its configuration file.
3. Each spawned server performs a partial address listen and provides the listener with the address that it is listening on. In a partial address listen, the server process listens, but informs the underlying protocol stack that it has no preference as to the specific address it listens on. As a result, many protocol stacks choose a free listening address and automatically assign this to the requesting server process. The listener initially marks all prespawned servers as idle.
4. The client sends a connect request to the listener.
5. The listener receives the network connection request, and determines if the client's request can be serviced. If not, the listener refuses the network connection and resumes at Step 9.
6. The listener issues a redirect message to the client containing the full network address of one of the prespawned dedicated servers. The listener logs that server process as active.
7. The client dissolves the network connection to the listener and establishes a network connection to the prespawned dedicated server using the address provided in the redirect message.
8. The listener spawns another server process to replace the active prespawned dedicated server, provided a value called PRESPAWN_MAX in the `listener.ora` file is greater than the total number of existing prespawned dedicated server processes (active or idle) and that the POOL_SIZE (number of idle prespawn processes) has not been reached.
9. The listener continues listening for incoming network sessions.

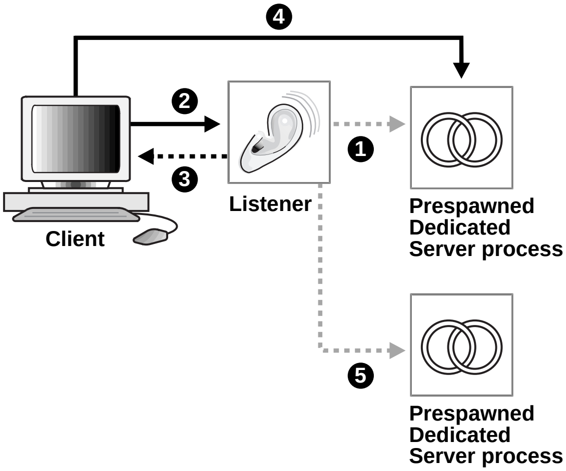
The above sequence of events continues until the maximum prespawn limit is reached. At this point the listener stops prespawning new dedicated server processes.

When client disconnects, the prespawned dedicated server process associated with the client returns to the idle pool and waits a specified length of time to be assigned to another client. If no client is handed to the prespawned server before the timeout expires, the prespawned server shuts down.

Figure 2–9 shows a listener spawning a prespawned dedicated server before the connection request and redirecting the connection back to the client.

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

Figure 2–9 Redirected Connection to a Prespawned Dedicated Server Process


4
2
3
1
5
Client
Listener
Prespawned Dedicated Server process
Prespawned Dedicated Server process

Configuration

The use of prespawned dedicated servers requires specification of parameters in the listener configuration file.

> **See Also:** "Configuring Prespawned Dedicated Servers" on page 7-16 for configuration information

Bequeathed Network Sessions Directly to Dedicated Servers

If the client and server exist on the same machine, a client connection can be bequeathed (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 using the Bequeath protocol.

This happens automatically if an application is used to start the database on the same machine as the database, as in the following:

```
SQLPLUS

SQL> connect internal/password
```

Client Connection Options

Client connections can be customized with the options described in the following sections:

- Client Load Balancing for Multiple Listeners
- Connect-Time Failover for Multiple Listeners
- Transparent Application Failover for High Availability

Client Load Balancing for Multiple Listeners

When more than one listener supports a service, a client can randomize requests to the various listeners. This feature, called **client load balancing**, distributes the load so as not to overburden a single listener.

Configuration

To enable your clients to choose from listeners at random, configure a connect descriptor with multiple listening addresses, and set the LOAD_BALANCE parameter to on, as in the following:

```
sales=
 (description=
  (load_balance=on)
  (address=(protocol=tcp)(host=sales1-server)(port=1521))
  (address=(protocol=tcp)(host=sales2-server)(port=1521))
  (connect_data=
   (service_name=sales.acme.com)))
```

Client load balancing can be used in conjunction with connect-time failover.

> **See Also:** "Configuring Address List Parameters" on page 8-4 for configuration information

Connect-Time Failover for Multiple Listeners

When more than one listener supports a service, a client can be configured to failover the client request to a different listener if the first listener fails. Reconnection attempts continue until the client successfully connects to a listener.

Configuration

To control how the client executes these connection attempts, configure a connect descriptor with multiple listening addresses and set the FAILOVER parameter to on, as in the following:

```
sales=
 (description=
  (failover=on)
  (address=(protocol=tcp)(host=sales1-server)(port=1521))
  (address=(protocol=tcp)(host=sales2-server)(port=1521))
  (connect_data=
   (service_name=sales.acme.com)))
```

Connect-time failover can be used in conjunction with client load balancing.

> **See Also:** See "Configuring Address List Parameters" on page 8-4 for configuration details

Transparent Application Failover for High Availability

Transparent Application Failover (TAF) is a runtime failover for high-availability environments, such as Oracle Parallel Server 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.

> **See Also:** *Oracle8i Concepts*

Configuration

To enable TAF, configure a connect descriptor with the FAILOVER_MODE parameter in the connect data portion of the connect descriptor.

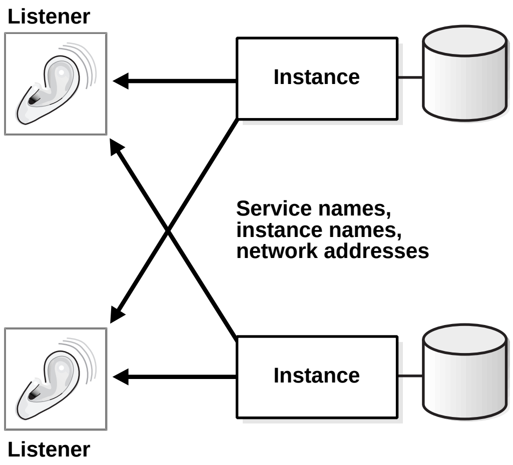
> **See Also:** "Configuring Transparent Application Failover" on page 8-12 for configuration details

Service Registration to the Listener

The **PMON process**—an instance background process—registers instance information with a listener, as well as the current state and load of the instance and MTS dispatchers. This registration process is called **service registration**. The registered information enables the listener to forward client connection requests to the appropriate service handler.

Figure 2–10 shows instances registering information with listeners.

Figure 2–10 Service Registration



Service registration provides the listener with information about:

- Service names for each running instance of the database
- Instance names of the database
- Service handlers (dispatchers and dedicated servers) available for each instance

  This enables the listener to direct a client's request appropriately.
- Dispatcher, instance, and node load information

  This load information enables the listener to determine which dispatcher can best handle a client connection's request. If all dispatchers are blocked, the listener can spawn a dedicated server for the connection.

When an instance is started, initialization parameters are read from the initialization parameter file. By default, PMON registers information with a listener on the local machine at a well-known address.

If the listener is not up when an instance starts, PMON is not able to register information. However, it attempts to connect to the listener periodically. If a listener receives an incoming request before the respective instance has been registered, the listener rejects the request.

Benefits

Service registration offers the following benefits:

Benefit	Description
Configuration	Reduces the need for the `listener.ora` file. **Note:** The SID_LIST_*listener_name* parameter that specifies information about the databases served by the listener is still required if you are using Oracle Enterprise Manager to manage the database.
Connect-Time Failover for MTS	Because service registration enables the listener to always know if an instance is up, this facilitates automatic failover of the client connect request to a different instance if one instance is down. In a static configuration model, a listener would start a dedicated server on receiving a client request. The server would later find out that the instance is not up yet, causing an "`Oracle not available`" error message. **See Also:** "Connect-Time Failover for Multiple Listeners" on page 2-21

Benefit	Description
Connection Load Balancing	Service registration enables the listener to forward client connect requests to the least loaded dispatcher. This enables the listener to balance the load across the service handlers and nodes. In an Oracle Parallel Server environment, service registration is also able to detect the least loaded instance. **See Also:** "Connection Load Balancing" on page 2-26

Registration with the Default Local Listener

By default, PMON registers with the local listener on server at the default local address of TCP/IP, port 1521.

Registration with a Non-Default Listener

If you want PMON to register with a local listener on the server that does not use TCP/IP, port 1521, configure the LOCAL_LISTENER parameter in the initialization parameter file to locate the local listener. The LOCAL_LISTENER parameter should be configured as follows:

```
local_listener=listener_alias
```

listener_alias is then resolved through a naming method, such as the `tnsnames.ora` file on the server.

For example, the LOCAL_LISTENER parameter can be set as follows:

```
local_listener=listener1
```

`listener1` can be resolved through a local `tnsnames.ora` file as follows:

```
listener1=
(description=
  (address=(protocol=tcp)(host=sales-server)(port= 1421)))
```

> **See Also:** "Configuring the LOCAL_LISTENER Parameter" on page 7-10 for information about setting LOCAL_LISTENER

Registration with Remote Listeners

Registration to remote listeners, such as in the case of Oracle Parallel Server, can be configured for MTS environments by using the LISTENER attribute of the MTS_DISPATCHERS parameter. The LISTENER attribute should be set as follows:

```
mts_dispatchers="(protocol=tcp)(listener=listener_alias)"
```

`listener_alias` is resolved through a naming method, such as `tnsnames.ora` file.

For example, the MTS_DISPATCHERS parameter can be set as follows in the initialization parameter file:

```
mts_dispatchers="(protocol=tcp)(listener=listeners_sales)"
```

`listeners_sales` can be then resolved through a local `tnsnames.ora` file as follows:

```
listeners_sales=
 (description=
  (address_list=
   (address=(protocol=tcp)(host=sales1-server)(port= 1521))
   (address=(protocol= tcp)(host=sales2-server)(port=1521))))
```

Configuration

To enable service registration, the initialization parameter file must 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
```

See Also:

- *Oracle8i Reference* for further information about the SERVICE_NAMES and INSTANCE_NAME parameters
- "Configuring a Listener that Uses a Non-Default Address" on page 7-10 for further information about the LOCAL_LISTENER parameter

Connection Load Balancing

Connection load balancing improves connection performance by balancing the number of active connections among multiple dispatchers. In an Oracle Parallel Server environment, connection load balancing also has the capability to balance the number of active connections among multiple instances.

Because PMON 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 which dispatcher to send the incoming client request to. Thus, when a listener gets an incoming request, it selects the dispatcher in the following order: a) least loaded node, b) least loaded instance, and c) least loaded dispatcher for that instance.

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

Configuration

To enable connection load balancing, the initialization parameter file must contain the following parameters:

- SERVICE_NAMES for the database service name
- INSTANCE_NAME for instance name
- MTS_DISPATCHERS to enable MTS and specify the dispatchers used by the instance

For example:

```
service_names=sales.us.acme.com
instance_name=sales
mts_dispatchers="(protocol=tcp)(dispatchers=2)(listener=listeners_sales)"
```

An Oracle Parallel Server 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 MTS_DISPATCHERS parameter.

See Also:

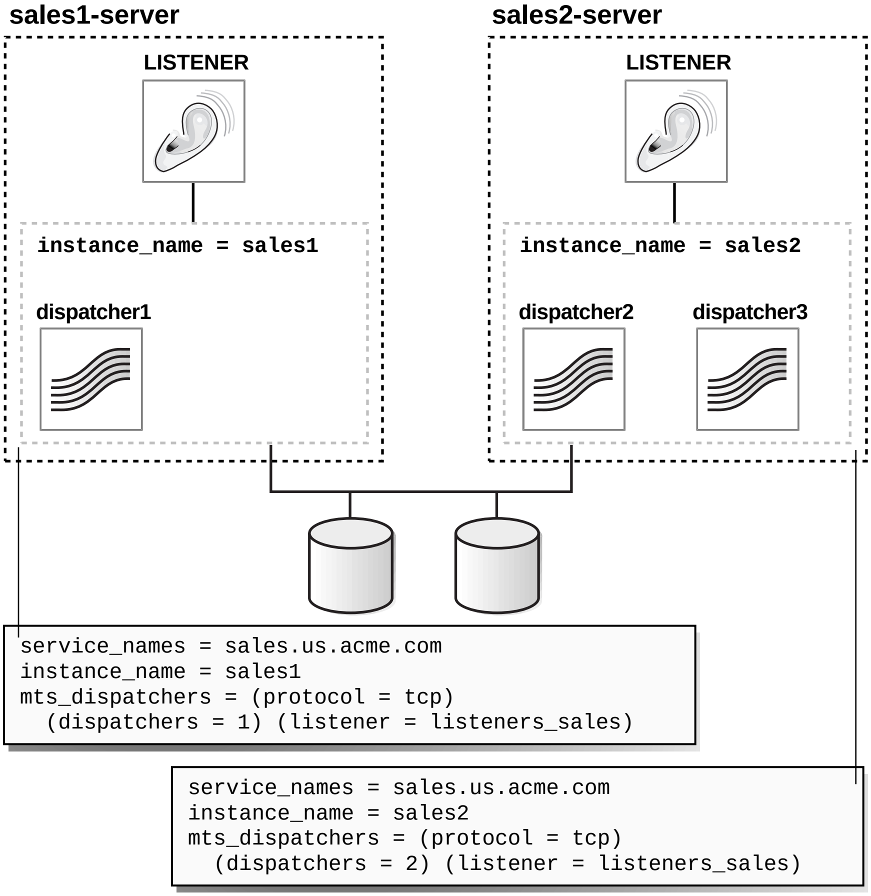
- "Registration with Remote Listeners" on page 2-25 for complete information about cross registration
- *Oracle8i Reference* for complete information about the SERVICE_NAMES and INSTANCE_NAME parameters
- Chapter 9 for complete information about the LISTENER attribute

---

Note: For optimum connection load balancing results, the instances that belong to the same database service should be on equivalent hardware and software configurations.

Example Figure 2–11 shows an Oracle Parallel Server database with two instances, `sales1` and `sales2,` of the same service, `sales.us.acme.com`. The instances `sales1` and `sales2` reside on machines `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 MTS_DISPATCHER parameter has been configured to allow for service registration of information to both listeners.

Figure 2–11 Load Balancing Environment



```
service_names = sales.us.acme.com
instance_name = sales1
mts_dispatchers = (protocol = tcp)
  (dispatchers = 1) (listener = listeners_sales)
```

```
service_names = sales.us.acme.com
instance_name = sales2
mts_dispatchers = (protocol = tcp)
  (dispatchers = 2) (listener = listeners_sales)
```

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_list=
   (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 2–12 on page 2-31:

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, and the following load information is registered:

	dispatcher1	dispatcher2	dispatcher 3
Number of Connections	200	100	200

	sales1	sales2
Number of Connections	200	300

	sales1-server	sales2-server
1 Minute Load Average	600	400

The number of connections for sales1 (200) is the same as that of its only dispatcher, dispatcher1. However, 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. However, 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.

2. The client sends a connect request.

   A connect descriptor is configured to try each listener 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.acme.com)))
   ```

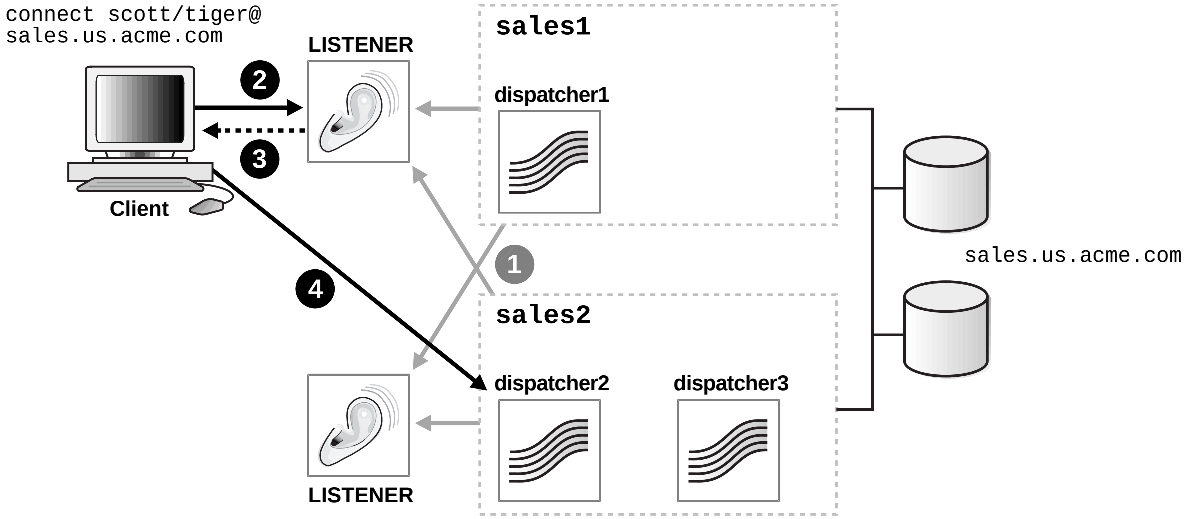
   The listener on `sales1-server` was randomly chosen to receive the client connect request.

   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 chooses `sales2-server` over `sales1-server`.

3. The listener compares the load on dispatchers `dispatcher2` and `dispatcher3`. Since `dispatcher2` is less loaded than `dispatcher3`, the listener redirects the client connect request to `dispatcher2`.

4. The client connects directly to `dispatcher2`.

Figure 2–12 Load Balancing Example



Database Identification by Service Name Rather than SID

An Oracle database can span multiple instances over multiple computers, making the database service a distributed service.

Prior to Oracle8*i*, an Oracle database service was identified by an **Oracle System Identifier (SID)**, a name that identifies a specific instance of a database. Clients connected to a database instance by specifying the SID in the connect descriptor. This naming scheme did not distinguish services from instances.

In Oracle8*i*, a new naming scheme has been implemented. Because an Oracle database can span multiple computers, both the service as a whole and each of its instances are specified.

A database service is now identified by its **service name** with the SERVICE_NAMES parameter in the initialization parameter file. SERVICE_NAMES specifies the name of the highest-level view of Oracle database service, which can span instances and nodes. SERVICE_NAMES is defaulted to the **global database name**, a name comprised of the database name (DB_NAME) and domain name (DB_DOMAIN).

Database instances are identified by an instance name with the INSTANCE_NAME parameter in the initialization parameter file. INSTANCE_NAME corresponds to the SID of the instance.

This section includes the following topics:

- Connect Descriptors Using SERVICE_NAME and INSTANCE_NAME
- Multiple Instances Belonging to the Same Service
- Single Instance Belonging to Multiple Services

Connect Descriptors Using SERVICE_NAME and INSTANCE_NAME

Connect descriptors used by clients should be configured with the SERVICE_NAME (without an S) parameter to connect to an Oracle8*i* database. For example, the following connect descriptor contains the address of a listener located on `sales1-server` listening for connection requests for a database service called `sales.us.acme.com`:

```
sales=
 (description=
  (address=(protocol=tcp)(host=sales1-server)(port=1521))
  (connect_data=
   (service_name=sales.us.acme.com)))
```

Optionally, the connect descriptor can be configured with the INSTANCE_NAME parameter (in addition to the SERVICE_NAME parameter) to connect to a particular instance of the database. This can be useful if you have an Oracle Parallel Server with multiple instances.

For example, the following connect descriptor contains the address of a listener located on `sales1-server` listening for connection requests for an instance called `sales1` associated with the `sales.us.acme.com` database:

```
sales=
 (description=
  (address=(protocol=tcp)(host=sales1-server)(port=1521))
  (connect_data=
   (instance_name=sales1)
   (service_name=sales.us.acme.com)))
```

Multiple Instances Belonging to the Same Service

Multiple database instances can run on different machines, as is the case for Oracle Parallel Server environments, and can access the same data, but they belong to the same service.

In the case of multiple instances, the listener performs connection load balancing to find the least loaded instance and least loaded dispatcher. Clients that specify the SERVICE_NAME parameter in the connect descriptor are directed to the instance and dispatcher with the least load.

If using a specific instance is required, clients can also specify the INSTANCE_NAME of a particular instance in the connect descriptor. This way, clients are connected to the instance as long as the instance belongs to the specified service.

Single Instance Belonging to Multiple Services

A database service can be represented by multiple service entries of the SERVICE_NAMES parameter in the initialization parameter file. This feature enables the following:

- A single instance to be identified in a number of different ways by different clients
- A database administrator to limit resources allocated or reserved for servicing clients requesting one of these services

For example, the initialization parameter file can be configured with:

```
service_names=sales1.us.acme.com, sales2.us.acme.com
```

With this configuration, a pool of MTS dispatchers can be allocated exclusively for clients requesting `sales1.us.acme.com`. Other clients, requesting `sales2.us.acme.com`, can be forced to use another dispatcher to handle their requests. This way, mission critical requests can be provided with more resources and a higher priority.

> **See Also:** "Allocating Resources" on page 9-9 for an example of resource allocation

Migration

The migration from the old naming scheme to the new one should be done as soon as possible to take full advantage of current functionality. New features in the future will not permit the use of SID in the client connect descriptor.

> **See Also:** see *Oracle8i Migration*

Service Resolution through Naming Methods

Connect descriptors are resolved through one or more **naming methods**. A naming method refers to the resolution method used by Net8 to resolve a name to a network description. Net8 provides the following naming methods:

- Local Naming
- Directory Naming
- Oracle Names
- Host Naming
- External Naming

Local Naming

Local naming resolves a net service name to a network address by using information configured in a local naming configuration file called `tnsnames.ora`.

> **See Also:** "Configuring the Local Naming Method" on page 6-5 for configuration details

Establishing a Connection Using Local Naming

The process for establishing a client session using local naming is as follows:

1. The client initiates a connect request providing a net service name.
2. The net service name is resolved to a connect descriptor configured in a `tnsnames.ora` file
3. The client makes the connect request to the address provided in the connect descriptor.
4. A listener receives the request and directs it to the appropriate server.
5. The connection is accepted by the server.

Directory Naming

With **directory naming**, clients can use network information stored in a centralized LDAP-compliant directory server to access a database service.

> **See Also:**
>
> - "Net8 and an LDAP-Compliant Directory Server" on page 2-38 for conceptual information
> - "Configuring the Directory Naming Method" on page 6-16 for configuration information

Establishing a Connection Using Directory Naming

The process for establishing a client session using directory naming is as follows:

1. The client initiates a connect request providing a connect identifier.
2. The connect identifier is resolved to a connect descriptor by a directory server. This information is returned to the client.
3. The client makes the connect request to the address provided in the connect descriptor.

4. A listener receives the request and directs it to the appropriate server.
5. The connection is accepted by the server.

Oracle Names

Oracle Names uses Oracle proprietary software to store the names and addresses of all database services on a network. Clients wishing to connect to a server direct their connect requests to an Oracle Names server. Oracle Names servers resolve the name to a network address and return that information to the client.

See Also:

- "Oracle Names" on page 4-10 for product information
- "Configuring the Oracle Names Method" on page 6-47 for configuration details

Establishing a Connection Using Oracle Names

The process for establishing a client session using the Oracle Names option is as follows:

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

Host Naming

Host naming enables users to connect to an Oracle server by using a host name alias. Host names are mapped to the server's global database name in an existing names resolution service, such as Domain Name System (DNS), Network Information Service (NIS), or a centrally-maintained set of `/etc/hosts` files.

The connection is established by using the default TCP/IP port for the listener, 1521. Multiple databases per node and database location transparency are supported through matching global database names with host name aliases.

Host naming eliminates the need for a `tnsnames.ora` file in environments where simple database connectivity is desired. It is not appropriate for environments that use **Oracle Connection Manager**, are not using TCP/IP, or require Transparent Application Failover.

> **See Also:** "Configuring the Host Naming Method" on page 6-74 for configuration details

Establishing a Connection Using Host Naming

The process for establishing a client session using the host naming option is as follows:

1. The client initiates a connect request providing a TCP/IP host name alias.
2. Host naming resolves the host name or alias by generating a network address using the net service name as both the TCP/IP host name and the service name. The TCP/IP port defaults to 1521.
3. Net8 sends the connect request to the address created.
4. A listener, listening at registered TCP/IP port 1521, receives the request and establishes a connection to the appropriate server.
5. The connection is accepted by the appropriate server.

External Naming

External naming refers to the method of resolving a net service name to a network address by using a supported non-Oracle naming service. External naming resolves net service names stored in customers' external non-Oracle naming services. They include the following:

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

> **See Also:** "Configuring External Naming Methods" on page 6-79 for configuration details

Establishing a Connection Using External Naming

The process for establishing a client session using external naming is as follows:

1. The client initiates a connect request providing a net service name.
2. An external naming method forwards the request to an external naming system that resolves the net service name to a network address. The address is returned to the client.
3. Net8 sends the connect request to the address provided.
4. A listener receives the request and directs it to the appropriate database.
5. The connection is accepted by the server.

Net8 and an LDAP-Compliant Directory Server

This section contains the following topics:

- Overview
- Directory Naming Overview
- Directory Entries
- Adding or Modifying Entries in the Directory
- Client Connections Using Directory Naming
- Specifying Absolute Names
- Net8 Naming and Directory Design

Overview

Today, network information is stored in multiple systems and in multiple directory formats. With new requirements for Internet computing and new e-business technologies, there is a growing need for a common repository infrastructure to serve as a foundation for management and configuration of all data and resources. Such a common infrastructure reduces the cost of managing and configuring resources in heterogeneous networks.

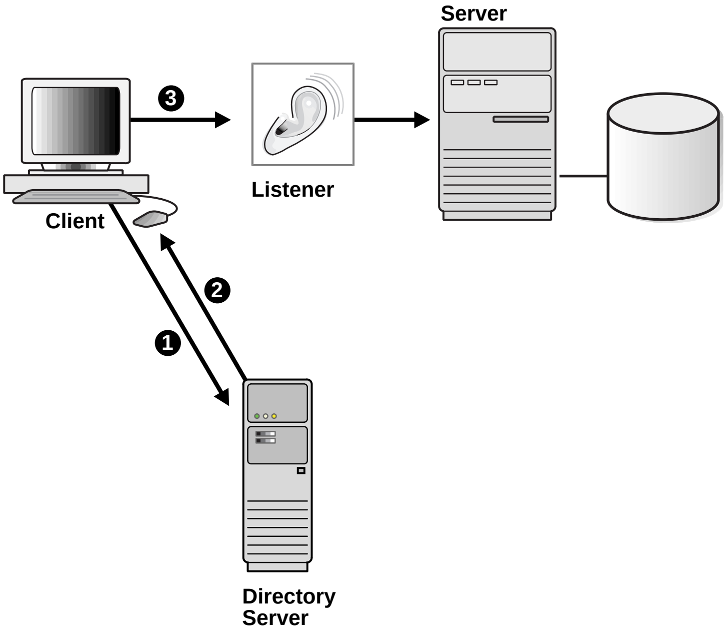
Support of LDAP-compliant directory servers provides a centralized vehicle for managing and configuring a distributed Oracle network. The directory can act as a central repository for all data on database network components, user and corporate policies, and user authentication and security, thus replacing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

Net8 uses a centralized directory server as one of the primary methods for storage of **connect identifiers**. Clients, configured to access the directory, use the connect identifiers in their connect string. The directory resolves the connect identifier to a connect descriptor that is passed back to the client.

Figure 2–13 shows a client resolving a connect identifier through a directory.

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

Figure 2–13 Client Using a Directory to Resolve a Connect Identifier



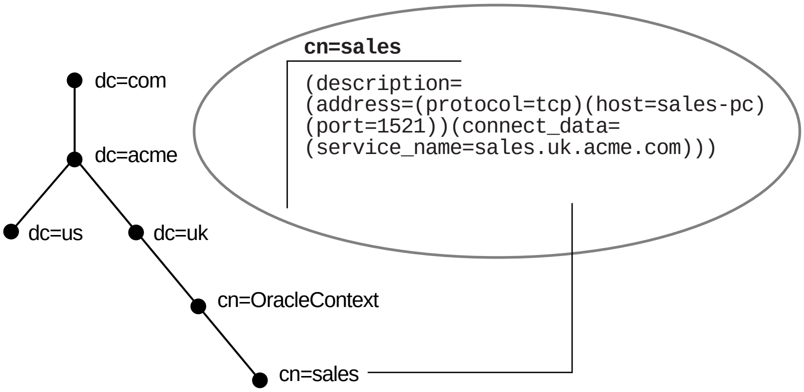
Net8 supports the following directories:

- Oracle Internet Directory
- Microsoft’s Active Directory
- Novell Directory Services

Directory Naming Overview

To understand how Net8 uses a directory, consider Figure 2–14.

Figure 2–14   Database Service in a Directory


dc=com
dc=acme
dc=us
dc=uk
cn=OracleContext
cn=sales
cn=sales
(description=
(address=(protocol=tcp)(host=sales-pc)
(port=1521))(connect_data=
(service_name=sales.uk.acme.com)))

Each node in the tree is an **entry**. The branch on the right represents the entry for a database service called `sales`, which resides under a hierarchical domain structure of `dc=uk,dc=acme,dc=com`.

This hierarchy of entries is called the **Directory Information Tree (DIT)**. Each entry is uniquely identified by a **Distinguished Name (DN)**. The DN tells you exactly where the entry resides in the directory's hierarchy. The DN for `sales` is `(dn:cn=sales,cn=OracleContext,dc=uk,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 `sales`, the RDN is `cn=sales`. A RDN is made up of a set of **attributes**. For example, in `cn=sales`, `cn` is one of the entry's attributes. The attribute, along with its value, uniquely identifies the entry.

Common attributes referred to in this chapter include:

Attribute	Description
commonName, cn	Common name of an entry
country, c	Name of country
domain component, dc	Domain component

Attribute	Description
organization, o	Name of organization
organizationalUnitName, ou	Name of a unit within an organization

Notice that `sales` resides under `cn=OracleContext`. `cn=OracleContext` is a special entry in the directory called an **Oracle Context** that contains various Oracle entries to support directory naming and **enterprise user** security.

> **See Also:** *Oracle Advanced Security Administrator's Guide* for further information about enterprise user security

In this example, the Oracle Context contains an entry for the `sales` database service.

During directory access configuration, completed with Net8 Configuration Assistant during or after installation, an **administrative context** is selected. The administrative context is a directory entry that contains an Oracle Context (`cn=OracleContext`). The client uses this administrative context as the default place to locate connect identifiers in the directory.

The administrative context affects the connect string. For example, if a client needs to access the `sales` entry frequently, a reasonable administrative context would be `dc=uk,dc=acme,dc=com`. Note that `cn=OracleContext` is not needed. This enables the client to use the following connect string:

```
CONNECT scott/tiger@sales
```

`sales` is relative to the Oracle Context.

In the case where a client's administrative context does not match the administrative context where the service is located, the connect string must specify an entry's absolute name, as described in "Client Connections Using Directory Naming" on page 2-44.

Directory Entries

DITs can be structured in many ways, using:

- Existing Domain Name Space (DNS)
- Internet naming scheme
- Geographical and organization lines

Figure 2–15 shows a DIT structured according to DNS domain components.

Figure 2–15 Domain Component DIT

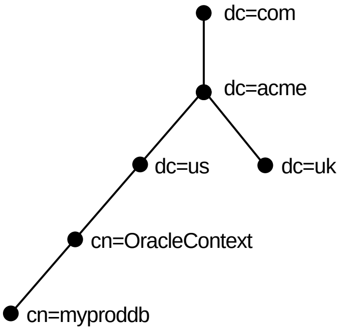
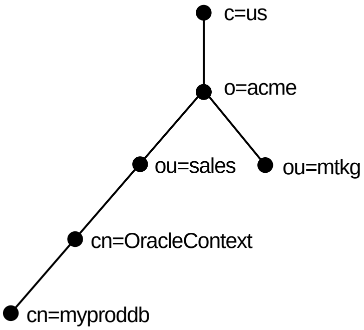


Figure 2–16 shows a DIT structured according to country, organization, and organizational units. This structure is commonly referred to as an X.500 DIT.

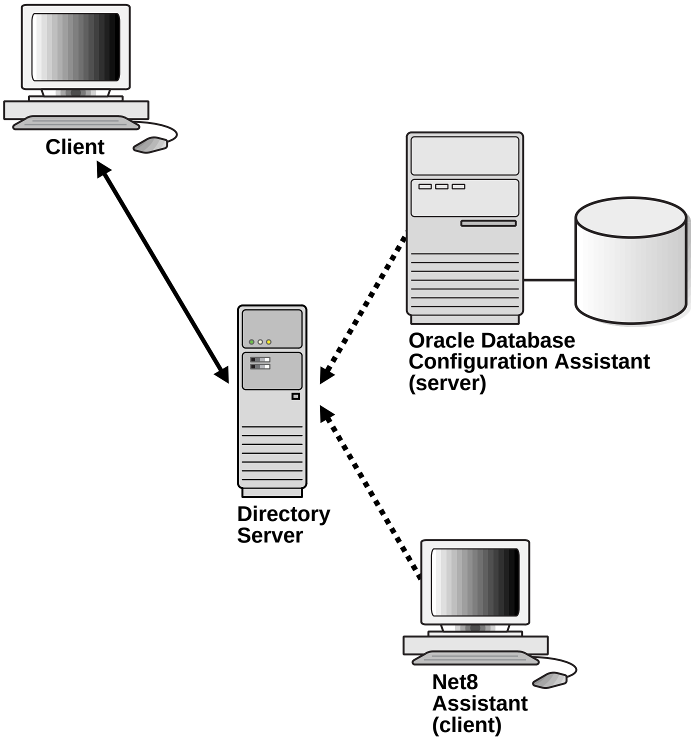
Figure 2–16 X.500 Style DIT



Adding or Modifying Entries in the Directory

A database service entry is created with **Oracle Database Configuration Assistant** during installation; net service name entries can be created with **Net8 Assistant**. Net8 Assistant can also be used to modify the Net8 attributes of the database service entry and the net service name entries. Figure 2–17 shows how Oracle Database Configuration Assistant and Net8 Assistant interface with the directory.

Figure 2–17 Creating Entries in the Directory with Applications



Clients configured for directory access, as described in "Client Connections Using Directory Naming" on page 2-44, can use entries created by these applications to connect to a database.

In order to add entries with these application, an Oracle Context (`cn=OracleContext`) must be created. The Oracle Context is created during directory access configuration on the server with **Net8 Configuration Assistant**.

The administrator using Oracle Database Configuration Assistant must be a member of the OracleDBCreators group (cn=OracleDBCreators,cn=Oraclecontext) to create a database service entry. Likewise, the administrator using Net8 Assistant to add net service names or Net8 attributes for database service or net service names entries must be a member of the OracleNetAdmins group (cn=OracleNetAdmins,cn=OracleContext). The directory user that created the Oracle Context is automatically added to both these groups, making this user the Oracle Context administrator. Other users can be added to this group by the directory administrator.

Client Connections Using Directory Naming

Most clients only need to perform name lookups in the directory. To perform a lookup, the following minimum requirements must be met:

- The client has anonymous authentication with the directory.
- The client has read access to Net8 entries located in the Oracle Context
- The client is configured for directory service access.

Net8 Configuration Assistant typically performs the necessary directory access configuration during client installation and stores the following information in a read-only ldap.ora file:

- Type of directory
- Location of the directory
- Administrative context from which to look up, create, and modify connect identifiers

The client reads this file to locate the directory and Net8 entries to access.

> **See Also:** "Task 1: Configure Directory Access on Server and Clients" on page 6-17 for information about configuring directory access

In the same way they might use other naming methods, clients can make connections to a database using one of the two types of connect identifiers that can stored in the directory. Database service and net service entries can be referred to by their common names, or they can require additional directory location information in the connect string. How the connect identifier is specified in a connect string depends on how the client's administrative context is configured.

With directory naming, there are two ways in which to identify an entry:

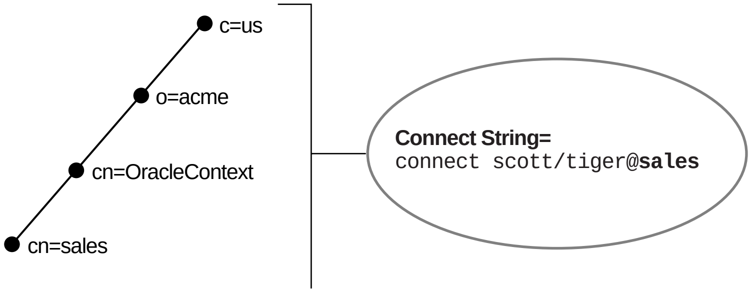
- Using the Entry's Relative Name
- Using the Entry's Absolute Name

Using the Entry's Relative Name

The following example demonstrates using an entry's relative name, whereby the service can be referred to by its common name. A relative name can be used if the client is configured with an administrative context that matches the administrative context for the entry in the directory.

Consider a directory 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 2–18. If the client is configured with an administrative context of `o=acme,c=us`, a connect string that uses `sales` can be used.

Figure 2–18 Relative Naming



Using the Entry's Absolute Name

Consider the same directory structure as Figure 2–18 on page 2-45, but with a client configured with an administrative context of dc=uk,dc=acme,dc=com.

Because the client is configured with an administrative context that does not match the location of sales in the directory, a connect string that uses sales does not work. Instead, the client must specifically identify the location of sales. sales which can be identified in two ways:

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

  ```
  CONNECT scott/tiger@cn=sales,cn=OracleContext,o=acme,c=us
  ```

- The entry can be referred to by its absolute name, a name that includes the name of the object and its location in the directory, much the way an absolute path is specified, for example:

  ```
  CONNECT scott/tiger@sales.acme.us
  ```

Oracle Corporation recommends using an entry's absolute name rather than the entry's DN.

> **See Also:** "Specifying Absolute Names" for further information about absolute names

Specifying Absolute Names

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 administrative context 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`

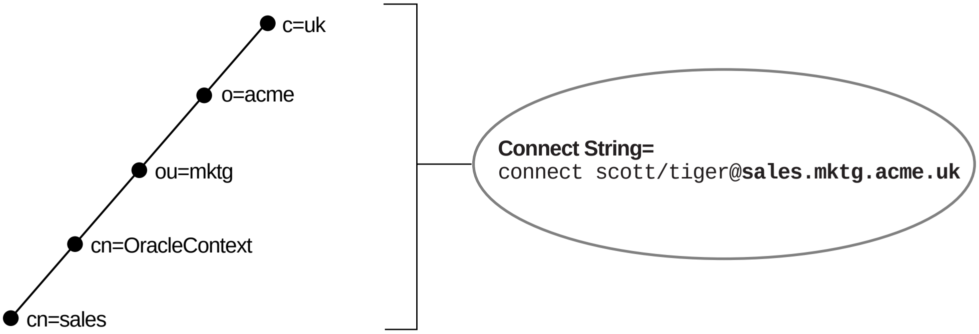
`cn[.ou].o.c`

where `[cn]` represents the Net8 entry.

For example, consider a client that has been configured with an administrative context of `ou=acctg,o=acme,c=us`.

The directory contains an entry for a database called `sales` with a DN of `(dn:cn=sales,cn=OracleContext,ou=mktg,o=acme,c=uk)`. In this scenario, the client requires a connect identifier of `sales.mktg.acme.uk` (`cn.ou.o.c`). Figure 2–19 depicts this example.

Figure 2–19 Absolute Name for X.500 Namespaces



Absolute Names for Domain Component Namespaces

For domain component namespaces, the administrative context 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]*
```

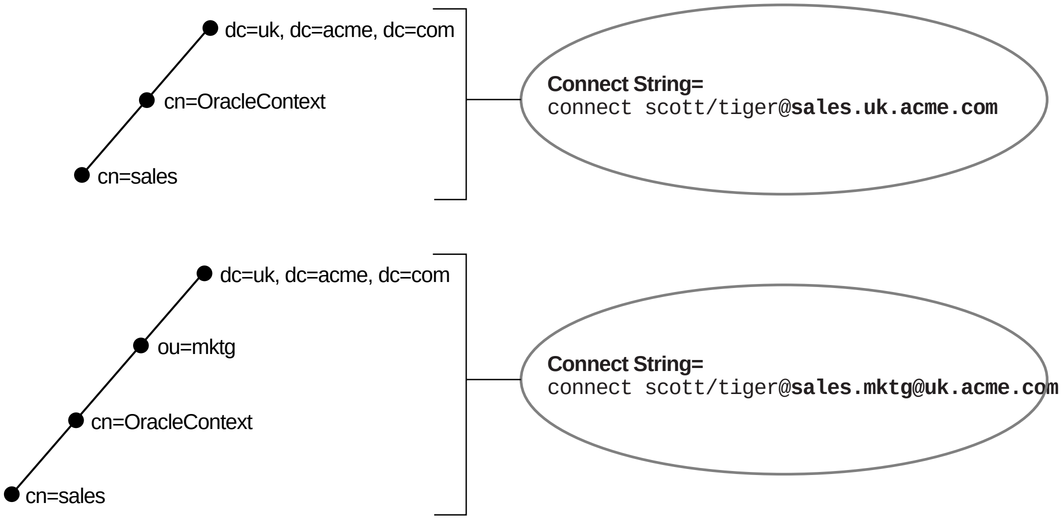
where [cn] represents the Net8 entry.

Example 1 Consider a client that has been configured with an administrative context of dc=us,dc=acme,dc=com.

The directory contains an entry for a database called sales with a DN of (dn:cn=sales,cn=OracleContext,dc=uk,dc=acme,dc=com). In this scenario, the client requires a connect identifier of sales.uk.acme.com (cn.dc.dc.dc). Figure 2–20 on page 2-49 depicts this example.

Example 2 Consider the same administrative context as Example 1. The directory contains an entry for a database called sales with a DN of (dn:cn=sales,cn=OracleContext,ou=mktg,dc=uk,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@uk.acme.com (cn.ou@dc.dc.dc). Figure 2–20 on page 2-49 depicts this example.

Figure 2–20   Absolute Name for Domain Component Namespaces



Net8 Naming and Directory Design

If you are responsible for designing directories with Net8 usage, consider the following issues:

- Performance
- Security
- Schema

Performance

Connect identifiers are stored in a directory for all clients to access. Net8 uses a directory lookup for the requested name, resulting in heavy use of the directory.

During a 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. Otherwise, the database connect time is affected. Users can begin to notice slow connect time if the lookup takes more than one second.

Performance problems can be resolved by the following:

- Topology changes
- Replication

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

Security

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

- Authentication Methods
- Access Control Lists

Authentication Methods Clients that perform typical lookups for information in the directory typically use anonymous authentication.

Clients that add or modify entries in a directory, with Oracle Database Configuration Assistant or Net8 Assistant, must authenticate with the directory. Only authenticated users with proper privileges can modify entries. One of the following authentication methods can be used:

Authentication Method	Description
Native Authentication	The directory uses operating system user credentials.
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.
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.

Access Control Lists Authentication is used in with **Access Control Lists (ACLs)** to make decisions about whether clients can modify or add information in the directory. ACLs are created with the Oracle Context with Net8 Configuration Assistant during directory access configuration on the server.

ACLs specify the following:

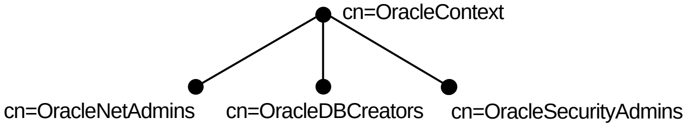
- The entries the user can access
- The authentication method to access the entry
- The access rights, or what the user can do with the object (read / write)

ACLs are established for a group of users. During Oracle Context creation, the following groups are created:

- OracleDBCreators
- OracleNetAdmins
- OracleSecurityAdmins

These groups are created directly under the Oracle Context, as shown in Figure 2–21.

Figure 2–21 Directory User Groups



The user who creates the Oracle Context with Net8 Assistant Configuration Assistant is automatically added as the first member of these groups.

The following table describes ACL requirements for these groups and anonymous users as they relate to Net8 entries in the directory:

Group	ACL Requirements
OracleNetAdmins group users	Members of OracleNetAdmins (`cn=OracleOracleNetAdmins,cn=OracleContext`) have `create`, `modify`, and `read` access to Net8 objects and attributes. Net8 Assistant 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:** "Adding Users to and Removing Users from the OracleNetAdmins Group" on page 6-31 to add users to the OracleNetAdmins group

Group	ACL Requirements
Anonymous users	All Net8 attributes and objects in the directory have `read` access for the anonymous user. Read access of these object for anonymous is also be applied to the Oracle Context. This enables anonymous users to browse entries contained with the `cn=OracleContext` RDN. This does not include objects used for enterprise user security. Net8 Assistant Configuration Assistant sets up this access right during client installation.
OracleDBCreators group users	Members of OracleDBCreators (`cn=OracleDBCreators,cn=OracleContext`) have `create` and `read` access to database service objects and attributes.Net8 Assistant 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 Security Manager. **See Also:** *Oracle Advanced Security Administrator's Guide* for further information about the OracleDBCreators group
OracleSecurityAdmins group users	Members of OracleSecurityAdmins (`cn=OracleOracleSecurityAdmins,cn=OracleContext`) have `create`, `modify`, and `read` access for enterprise user security. Net8 Assistant Configuration Assistant sets up these access rights 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 Security Manager. **See Also:** *Oracle Advanced Security Administrator's Guide* for further information about the OracleSecurityAdmins group

Schema

Directories must be populated with the correct version of the Oracle schema for Net8 clients to look up network information for a service. The Oracle schema defines the type of objects, called **object classes**, that can be stored in the directory and their attributes. The Oracle schema supports the following object classes for Net8 lookups.

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

orclDbServer and orclNetService use the following object classes:

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

See Also: Appendix D for further information about these object classes

3

Net8 Architecture

This chapter describes the Net8 architecture. This chapter includes the following sections:

- Transparent Network Substrate (TNS)
- Overview of Stack Communications
- Stack Communications in a Typical Net8 Client/Server Environment
- Stack Communications for Clients Connecting to a Directory Server
- Stack Communications for JDBC Clients
- Stack Communications for Oracle8i JServer Clients

Transparent Network Substrate (TNS)

Transparent Application Failover (TAF) is a foundation technology that is built into Net8 providing a single, common interface to all industry-standard protocols.

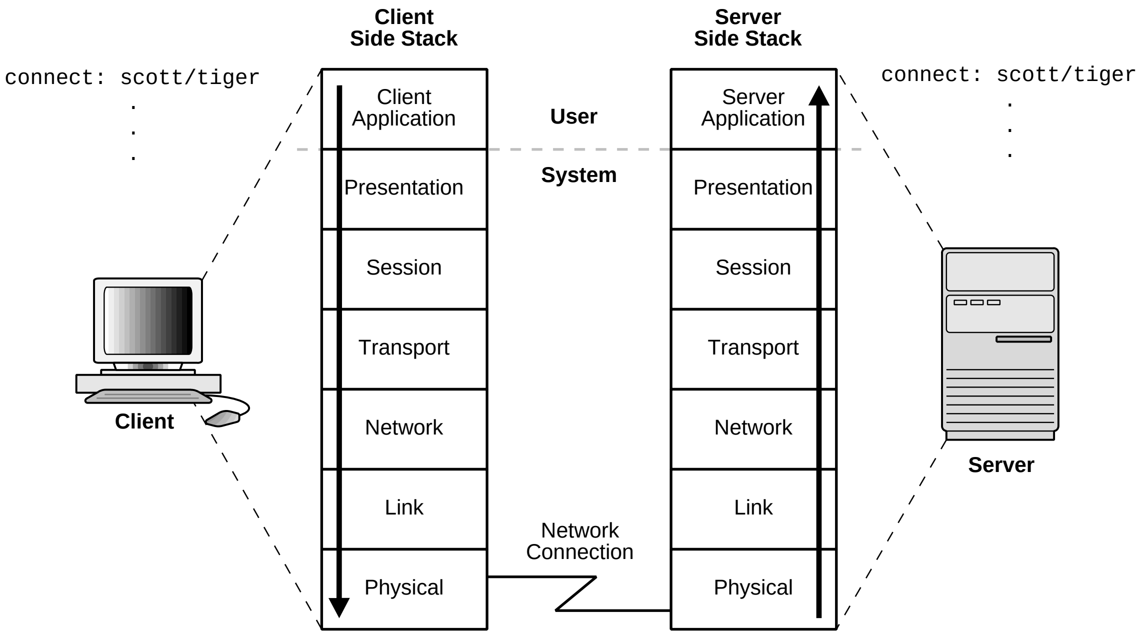
With TNS, peer-to-peer application connectivity is possible. In a peer-to-peer architecture, two or more computers (called **nodes** when they are employed in a networking environment) can communicate with each other directly, without the need for any intermediary devices. In a peer-to-peer system, a node can be both a client and a server.

Overview of Stack Communications

The concept of distributed processing relies on the ability of computers separated by both design and physical location to communicate and interact with each other. This is accomplished through a process known as stack communications.

Stack communications can be explained by referencing 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. Figure 3–1 on page 3-3 shows a typical OSI protocol communications stack.

Figure 3–1 OSI Communications Stack



Information descends through layers on the client side where it is packaged for transport across a network medium in such a manner that it can be translated and understood by corresponding layers on the server side.

A typical OSI protocol communications stack contains seven such layers:

Client Application The OSI layer closest to the user, and as such is dependent on the functionality requested by the user. For example, in a database environment, a Forms application may attempt to initiate communication in order to access data from a server.

Presentation Ensures that data is represented in a format that the application and session layers can accommodate. This includes keeping track of syntax and semantics of the data transferred between the client and server. If necessary, the presentation layer translates between multiple data representation formats by using a common data format.

Session Establishes, manages, and terminates network sessions between the client and server. This is a virtual pipe that carries data requests and responses. The session layer manages whether the data traffic can go in both directions at the same time (referred to as asynchronous), or in only one direction at a time (referred to as synchronous).

Transport Implements the data transport ensuring that the data is transported reliably.

Network Ensures that the data transport is routed through optimal paths through a series of interconnected subnetworks.

Link Provides reliable transit of data across a physical link.

Physical Defines the electrical, mechanical, and procedural specifications for activating, maintaining and deactivating the physical link between client and server.

Stack Communications in a Typical Net8 Client/Server Environment

Stack communications allow Oracle clients and servers to share, modify, and manipulate data between themselves. The layers in a typical Oracle communications stack are similar to those of a standard OSI communications stack. This section covers the following topics:

- Net8 Client-Side Interaction
- Server-Side Interaction

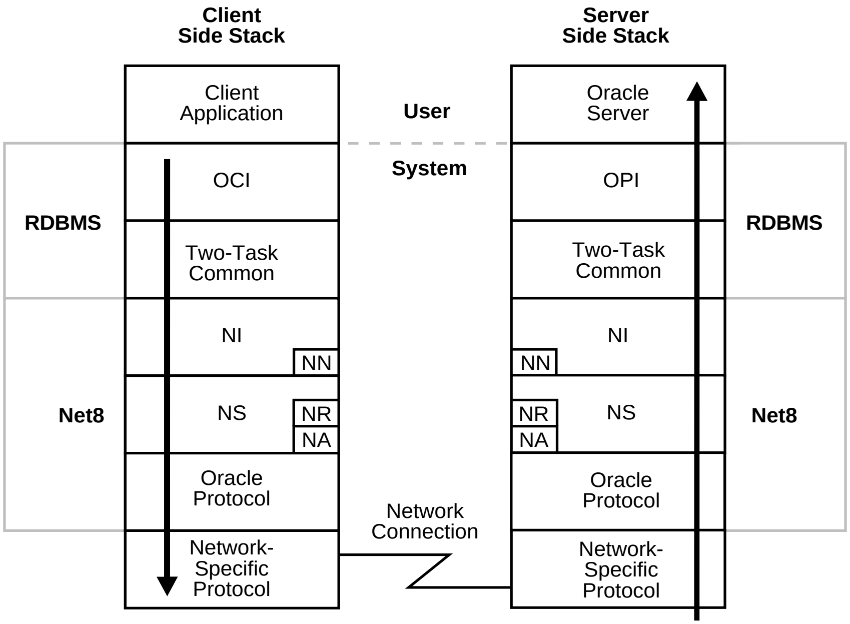
Net8 Client-Side Interaction

In an Oracle client-server transaction, information passes through the following layers:

- Client Application
- Oracle Call Interface (OCI)
- Two-Task Common (TTC)
- Net8
- Network-Specific Protocols

Figure 3–2 shows a client-server communications stack in an Oracle networking environment.

Figure 3–2 Typical Communications Stack in an Oracle Environment



Client Application

Oracle client applications provide all user-oriented activities, such as character or graphical user display, screen control, data presentation, and application flow. The application identifies database operations to send to the server and passes them through to the Oracle Call Interface (OCI).

Oracle Call Interface (OCI)

Oracle Call Interface (OCI) code contains all the information required to initiate a SQL dialog between the client and the server. It defines calls to the server to:

- Parse SQL statements for syntax validation
- Open a cursor for the SQL statement
- Bind client application variables into the server shared memory
- Describe the contents of the fields being returned based on the values in the server’s data dictionary
- Execute SQL statements within the cursor memory space
- Fetch one or more rows of data into the client application
- Close the cursor

The client application uses a combination of these calls to request activity within the server. OCI calls can be combined into a single message to the server, or they may be processed one at a time through multiple messages to the server, depending on the nature of the client application. Oracle products attempt to minimize the number of messages sent to the server by combining many OCI calls into a single message to the server. When a call is performed, control is passed to Net8 to establish the connection and transmit the request to the server.

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

Two-Task Common (TTC)

Two-Task Common (TTC) is Oracle’s implementation of the presentation layer. TTC provides character set and data type conversion between different character sets or formats on the client and server. This layer is optimized on a per connection basis to perform conversion only when required

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.

Net8

Net8 provides all the session and transport layer functionality in an Oracle communication stack. It is responsible for establishing and maintaining the connection between the client application and server, as well as exchanging messages between them. Net8 is also responsible for mapping session functionality into industry-standard protocols.

Net8 has three component layers that facilitate session and transport layer functionality:

Component	Description
Network Interface (NI)	This layer provides a generic interface for Oracle clients, servers, or external processes to access Net8 functions. The NI layer handles the "break" and "reset" requests for a connection. NI uses Network Naming (NN) to resolve names to connect descriptors.
Network Session (NS)	This layer receives requests from NI, and settles all generic machine-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). NS uses Network Route (NR) to route the network session to the destination and Network Authentication (NA) to negotiate any authentication requirements with the destination.
Oracle Protocols	Oracle protocols are Oracle’s implementation of the transport layer. Oracle protocols are responsible for mapping NS functionality to industry-standard protocols used in the client-server connection. Each protocol is responsible for mapping the equivalent functions between NS and a specific protocol. Oracle protocols include: ▪ TCP/IP ▪ TCP/IP with SSL ▪ SPX ▪ Named Pipes ▪ LU6.2

Network-Specific Protocols

All Oracle software in the client-server connection process requires an existing network protocol stack to make the machine-level connection between the two machines for the transport layer. The network protocol is responsible only for getting the data from the client machine to the server machine, at which point the data is passed to the server-side Oracle protocol.

Server-Side Interaction

Information passed from a client application across a network protocol is received by a similar communications stack on the server side. The process stack on the server side is the reverse of the process stack on the client side, with information ascending through communication layers. The one operation unique to the server side is the act of receiving the initial connection through the listener.

The following components above the network session layer are different from those on the client side:

- Oracle Program Interface (OPI)
- Oracle Server

Oracle Program Interface (OPI)

Oracle Program Interface (OPI) performs a complementary function to that of the OCI. It is responsible for responding to each of the possible messages sent by the 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 Server

The Oracle Server side of the connection is responsible for receiving requests from the client OCI code and resolving SQL statements on behalf of the client application. Once received, a request is processed and the resulting data is passed to the OPI for responses to be formatted and returned to the client application.

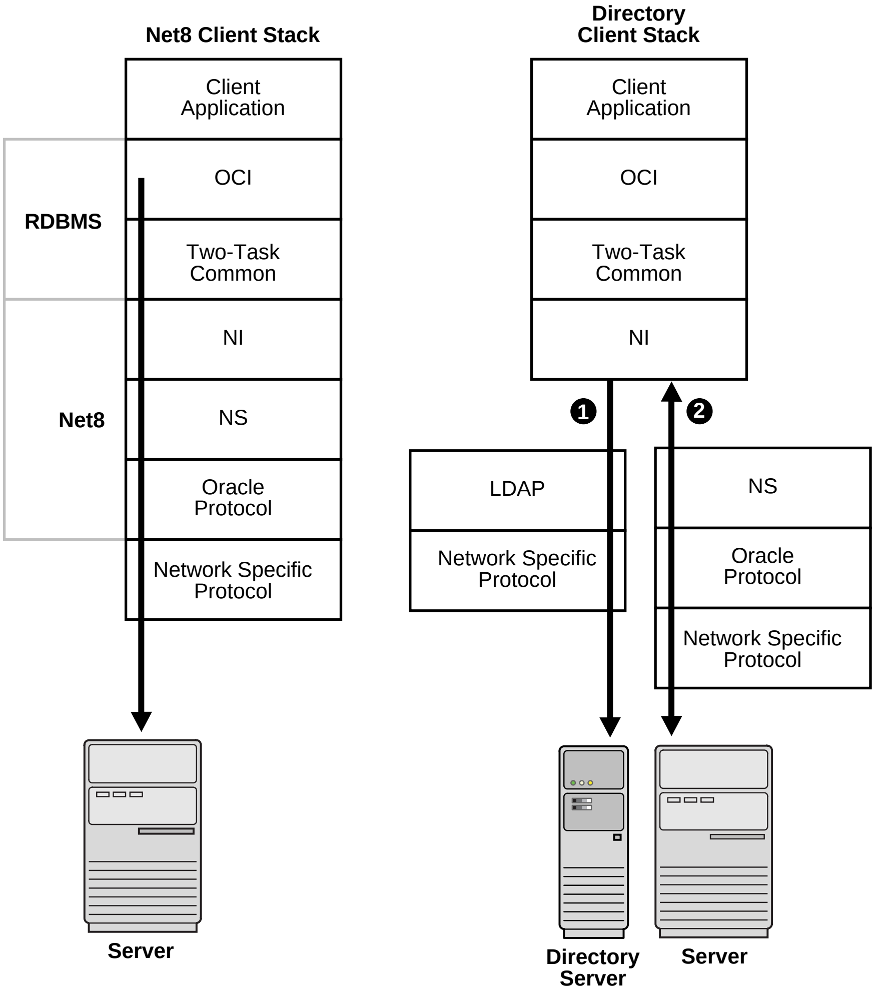
Server-to-Server Interaction

When two servers communicate to complete a distributed transaction, the server acts as a client application. The server has its own version of OCI, called the **Network Program Interface (NPI)**. The NPI interface performs all of the functions that the OCI does for clients, allowing a coordinating server to construct SQL requests for additional servers.

Stack Communications for Clients Connecting to a Directory Server

Clients that access a LDAP-compliant directory server go through an additional LDAP layer to connect to the directory, as shown in Figure 3–3. The directory sends the information back to the client through the traditional client stack.

Figure 3–3 Typical Client vs. Net8 LDAP Client Communications Stack



Stack Communications for JDBC Clients

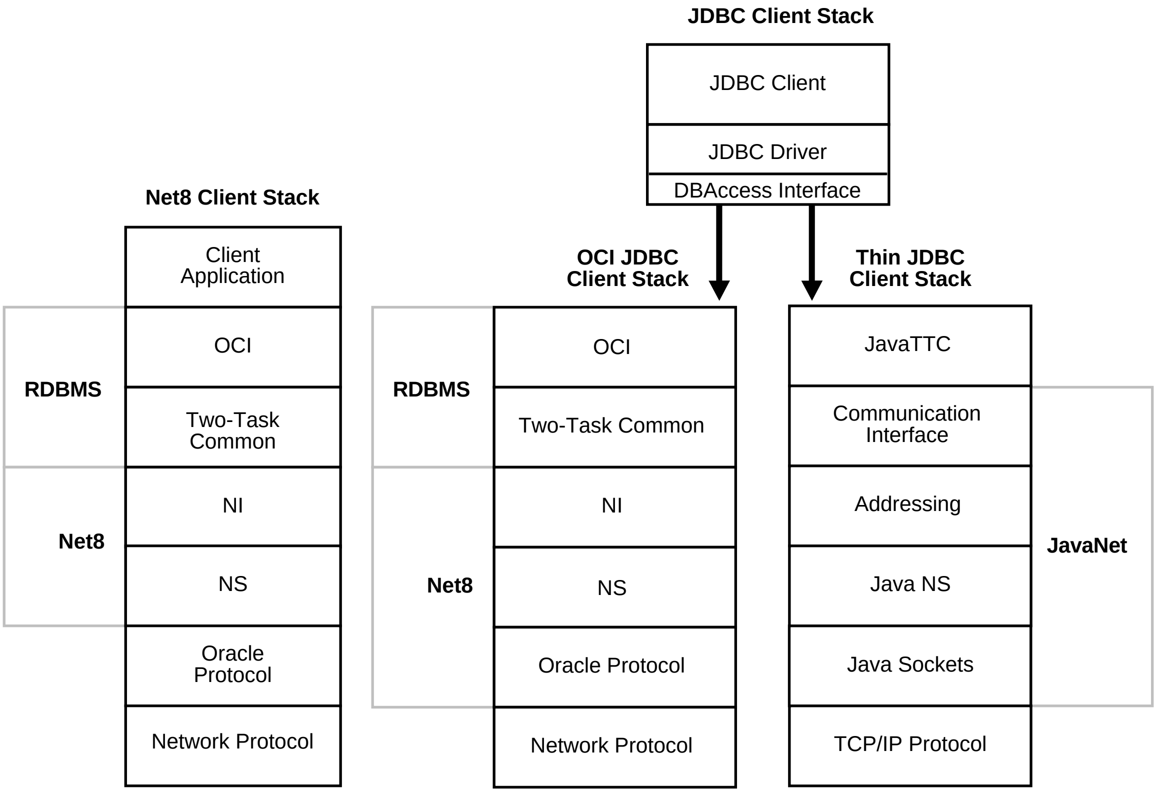
Oracle's Java Database Connectivity (JDBC) Drivers provide Java programmers access to an Oracle database. Oracle offers two JDBC drivers.

- JDBC/OCI is Oracle's Level 2 JDBC driver. It is targeted to client-server Java application programmers and Java-based middle-tier developers. The JDBC/OCI driver converts JDBC invocations to calls to the Oracle Call Interface (OCI) which are then sent over Net8 to the Oracle database server.
- Thin JDBC driver is Oracle's Level 4 driver. It is 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 Two-Task Common (TTC) and Net8.

See Also: *Oracle8i JDBC Developer's Guide and Reference*

The layers in a JDBC client communications stack are similar to those of a standard Oracle communications stack, as shown in Figure 3–4:

Figure 3–4 Typical Client vs. JDBC Client Communications Stack



Note: The server-side stack is the same as Figure 3–2 on page 3-6.

The JDBC / OCI client stack is similar to a typical Net8 Client stack. The Thin JDBC client stack has similar components implemented differently. In a Java client applet-server transaction, information passes through the following layers:

- DBAccess Interface
- JavaTTC
- JavaNet
- TCP / IP Protocol

DBAccess Interface

The Java DBAccess Interface layer enables the client to initiate a SQL session much like Oracle Call Interface (OCI) does in a standard client communication stack.

JavaTTC

JavaTTC provides a subset version of the Two-Task Common (TTC) implementation necessary for exchanges of information between the Java client and the database. JavaTTC is responsible for:

- Negotiating protocol version and data type
- Determining whether conversions between the Java client character set and server character set are necessary
- SQL statement execution

JavaNet

JavaNet provides a communication infrastructure so the JDBC client can connect to an Oracle database. This information is equivalent to NI, NR/NN/NA, and TNS in Net8.

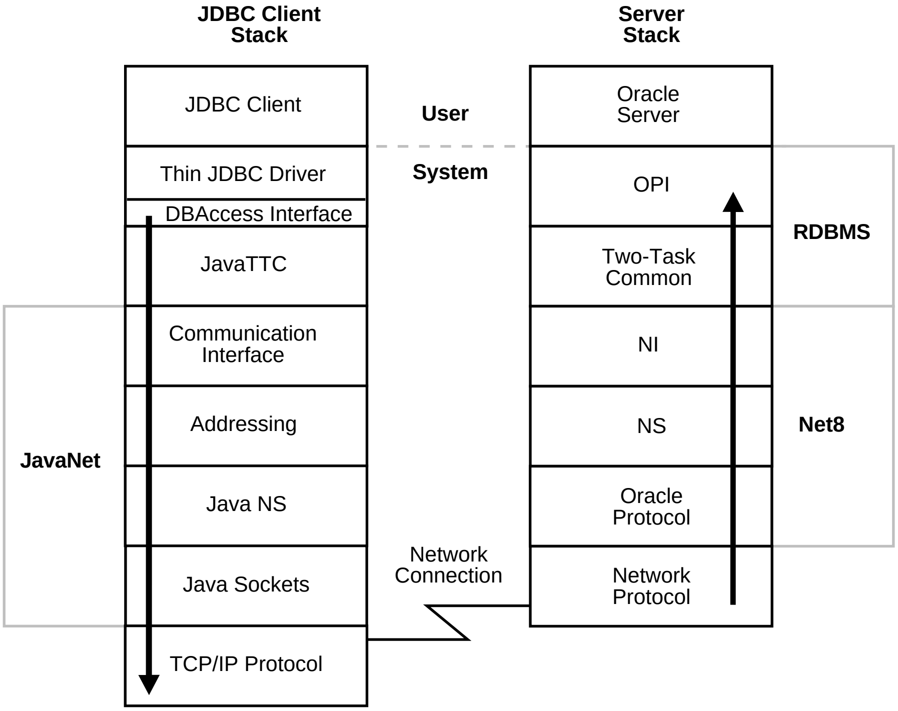
- Communication Interface - Provides an interface between JavaTTC and other networking layers much like NI does in a standard client communication stack.
- Addressing - Processes Net8 destination addresses much like NN does in a standard client communication stack.
- Java NS - Provides a common interface to TCP/IP, and settles all generic machine-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).
- Java Sockets - Provides a connection between Java NS and TCP/IP.

TCP/IP Protocol

TCP/IP Protocol - Java NS will only run on top of TCP/IP, because Java sockets use TCP/IP.

Figure 3–5 shows a communication stack between a Java client applet and an Oracle database.

Figure 3–5 Java Client/Server Communications Stack in a JDBC Environment



Note: JDBC clients can be configured to use directory naming, as described in *Oracle8i JDBC Developer's Guide and Reference*

Stack Communications for Oracle8*i* JServer Clients

Oracle8*i* has Java support with **Oracle8*i* JServer**. Oracle8*i* JServer includes support for Java stored procedures, JDBC, SQLJ, Common Object Request Broker Architecture (CORBA), and Enterprise JavaBeans (EJBs).

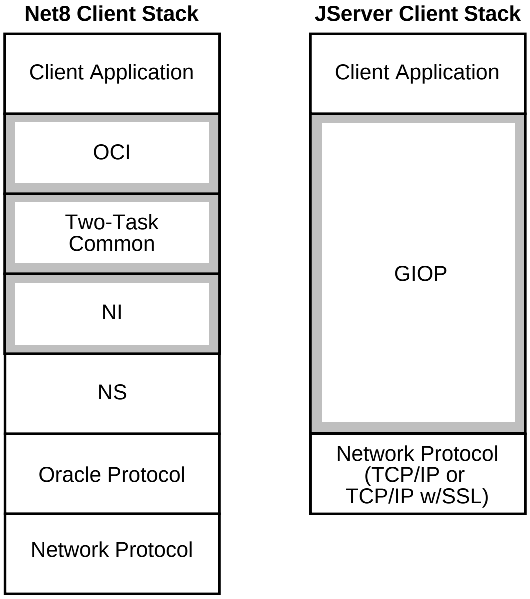
> **See Also:** *Oracle8i Java Developer's Guide* for an overview of Oracle8*i* JServer

Oracle8*i* JServer supports the General Inter-Orb Protocol (GIOP) presentation for Java Virtual Machine (VM) requests. GIOP is used by clients accessing Enterprise Java Beans and CORBA Servers in the Java VM.

EJB and CORBA clients use a different communication stack than a typical Net8 client stack, as shown in Figure 3–6. Differences include:

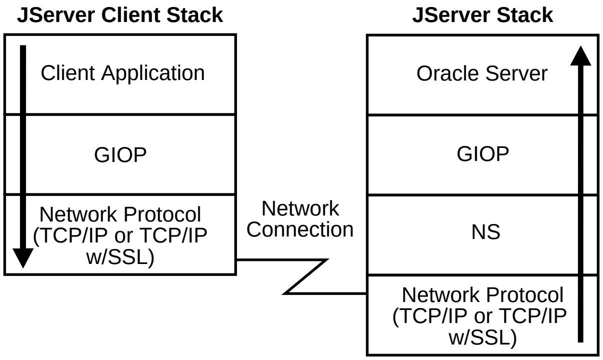
- GIOP as the presentation layer
- No session layer

Figure 3–6 Typical Net8 Client vs. Oracle8i JServer Client Communication Stack



The server side does not require many of the Net8 communication layers needed in a typical Net8 connection. Instead, the server side only requires a network protocol of TCP/IP and an Oracle protocol of TCP/IP or TCP/IP with SSL. The only component of Net8 required is Network Session (NS). Figure 3–7 shows a communication stack between a client and the Oracle8*i* JServer option in the Oracle database.

Figure 3–7 Client/Server Communications Stack in an Oracle8i JServer Environment



See Also:

- *Oracle8i Enterprise JavaBeans and CORBA Developer's Guide* for information on configuring client connections to EJB and CORBA applications
- *Oracle8i Java Stored Procedures Developer's Guide* for information on configuring client connections to Java stored procedures

4

Net8 Products and Tools

This chapter introduces Net8, and provides an overview of its main applications, features, and functionality. It contains the following sections:

- Net8 Client
- Net8 Server
- Oracle Protocols
- Oracle Connection Manager
- Oracle Names
- External Naming Services
- Authentication Methods
- Oracle Advanced Security
- Administration Tools

Note: The availability of the products is dependent on the installation type and the operating system. See your operating-system-specific documentation for further information.

Net8 Client

Net8 Client enables client connections to databases across a network. A client-side application sends a request to Net8 to be transported across the network to the server.

Net8 Server

Net8 Server enables the listener, through a protocol, to accept connections from client applications on the network.

Oracle Protocols

Oracle supports the following protocols:

- TCP/IP
- TCP/IP with SSL
- SPX
- Named Pipes
- LU6.2
- Bequeath

TCP/IP

The Transmission Control Protocol/Internet Protocol (TCP/IP) is the de facto standard Ethernet protocol used for client/server conversation over a network. TCP/IP enables an Oracle application on a client to communicate with remote Oracle databases through TCP/IP (if the Oracle database is running on a host system that supports network communication using TCP/IP).

TCP/IP with SSL

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 (if the Oracle database is running on a host system that supports network communication using 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 Net8 connection to the server, SSL performs a handshake between the two (using the certificate). During the handshake the following processes occur:

- The client and 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 server may then send its own certificate to the client in a message encrypted with the client's public key. The 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 server's certificate bears the certificate authority's signature.
- If required, the client may send the user's certificate to the Oracle server. The certificate ensures that the user's information is correct and that the public key actually belongs to that user.

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

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

SPX

The Sequenced Packet Exchange (SPX) protocol enables client/server conversation over a network using SPX/IPX. This combination of Oracle products enables an Oracle application on a client to communicate with remote Oracle databases through SPX/IPX (if the Oracle database is running on a host system that supports network communication using SPX/IPX). This protocol is predominantly used in Novell Netware environments.

Named Pipes

The **Named Pipes protocol** is a high-level interface providing interprocess communications between clients and servers (distributed applications). One process (the server side of the application) creates the pipe, and the other process (the client side) 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 conversation 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).

LU6.2

The Logical Unit Type 6.2 (LU6.2) protocol is part of the IBM Advanced Program-to-Program Communication (APPC) architecture.

APPC is the IBM peer-to-peer (program-to-program) protocol for a System Network Architecture (SNA) network. SNA is an IBM reference model similar to the Open Systems Interconnect (OSI) model of the International Standards Organization (ISO).

APPC architecture lets the client and host communicate over an SNA network without forcing the client to emulate a terminal (as in terminal-to-host protocols). APPC architecture enables peer-to-peer communication; the client can initiate communication with the server.

An SNA network with the LU6.2 and Physical Unit Type 2.1 (PU2.1) protocols provides APPC. The LU6.2 protocol defines a session between two application programs; LU6.2 is a product-independent LU-type.

LU6.2 enables an Oracle application on a PC to communicate with an Oracle database. This communication occurs over an SNA network with the Oracle database on a host system that supports APPC.

Bequeath

The **Bequeath protocol** enables clients that exist on the same machine as the server to retrieve information from the database without using the listener. The Bequeath protocol internally spawns a dedicated server process for each client applications. In a sense, it does the same operation that a remote network listener does for your connection, yet locally.

Bequeath is used for local connections where an Oracle client application, such as SQL*Plus, communicates with an Oracle server running on the same machine

Oracle Connection Manager

Note: Oracle Connection Manager is available for installation with Oracle8*i* Enterprise Edition.

Oracle Connection Manager is 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 a Connection Manager can then take advantage of the connection concentration, Net8 access control, or multi-protocol support features configured on that Connection Manager.

Oracle Connection Manager Processes

Oracle Connection Manager listens for incoming requests from clients and initiates connect requests to destination services. Oracle Connection Manager performs these tasks with the help of two processes:

Process	Description
CMGW (Oracle Connection Manager Gateway Process)	A gateway process acting as a hub for Oracle Connection Manager. This process is responsible for the following: ■ Registering with the CMADMIN process ■ Listening for incoming connection requests. By default it listens on TCP/IP with port 1630. ■ Initiating connection requests to listeners for clients ■ Relaying data between the client and server ■ Answering requests initiated by Oracle Connection Manager Control Utility (CMCTL)

Process	Description
CMADMIN (Oracle Connection Manager Administrative Process)	CMADMIN is a multi-threaded process that is responsible for all administrative issues of Oracle Connection Manager. This process is responsible for the following: ■ Processing the CMGW registration ■ Identifying all listeners serving at least one database instance ■ Registering source route address information about the CMGW and listeners ■ Monitoring registered listeners with the Oracle Names server, then source route addresses ■ Locating Oracle Names servers ■ Maintaining address information in the Oracle Names server for the SQL*Net 2.*x* and Net8.*x* clients ■ Periodically sending a request to the Oracle Names server to update its cache of available services. ■ Answering requests initiated by CMCTL

See Also: "Oracle Connection Manager Control Utility (CMCTL)" on page A-79 for information on using CMCTL

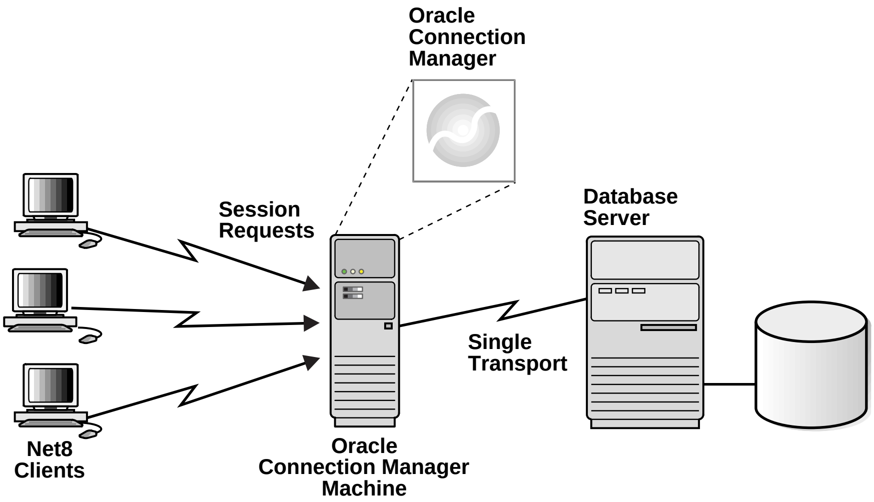
Connection Concentration

Oracle Connection Manager enables you to multiplex or funnel multiple client network sessions through a single transport protocol connection to a multi-threaded server destination.

Concentration reduces the demand on resources needed to maintain multiple connections between two processes by enabling the server to use fewer connection end points for incoming requests. This enables you to increase the total number of network sessions that a server can handle. By using multiple Connection Managers, it is possible for thousands of concurrent users to connect to a server.

Figure 4–1 shows how connection concentration works.

Figure 4–1 Connection Concentration through Oracle Connection Manager



See Also: "Enabling Connection Concentration" on page 8-28 for configuration information

Net8 Access Control

Oracle Connection Manager also includes a feature which you can use to control client access to designated servers in a TCP/IP environment. By specifying certain filtering rules you may allow or restrict specific clients access to a server based on the following criteria:

- Source host name(s) or IP address(es) for clients
- Destination host name(s) or IP address(es) for servers
- Destination database service name

> **See Also:** "Enabling Net8 Access Control" on page 8-35 for configuration information

Net8 Firewall Proxy

Some firewall vendors also offer Net8 Firewall Proxy, which is installed on firewalls requiring an application proxy. Net8 Firewall Proxy has the same access control functionality as Oracle Connection Manager.

> **Note:** Ask your firewall vendor if Net8 Firewall Proxy is supported.

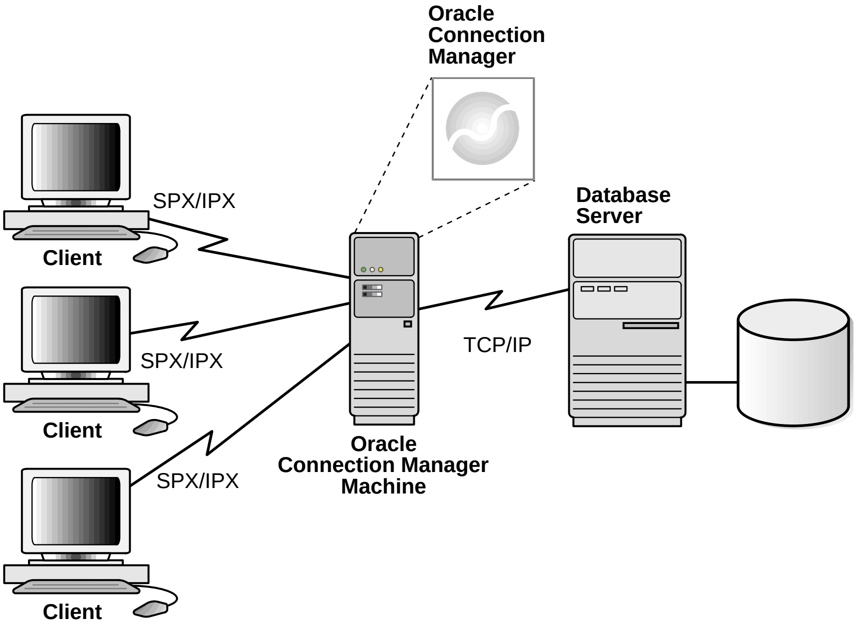
Multi-Protocol Support

Oracle Connection Manager also provides multiple protocol support enabling 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.

Net8 can traverse as many networking protocol stacks as can be installed and supported. In fact, the number of networking protocols supported is limited only by those restrictions imposed by the specific node's hardware, memory, and operating system.

Figure 4–2 shows how a client in an SPX network can route its network session to a server over a TCP/IP transport through Oracle Connection Manager.

Figure 4–2 Multi-Protocol Support Through Oracle Connection Manager



See Also: "Enabling Multi-Protocol Support" on page 8-32 for configuration information

Oracle Names

Oracle Names is a distributed naming service developed for Oracle environments to help simplify the setup and administration of global, client-server computing networks.

This section covers topics in the following sections:

- Overview
- Administrative Regions
- Domains
- Oracle Names As Data Repository
- Data Stored in an Oracle Names Server
- Organizing and Naming Network Components
- Using Multiple Regions to Decentralize Administrative Responsibilities
- Differences Between Versions of Oracle Names
- Understanding Discovery

Overview

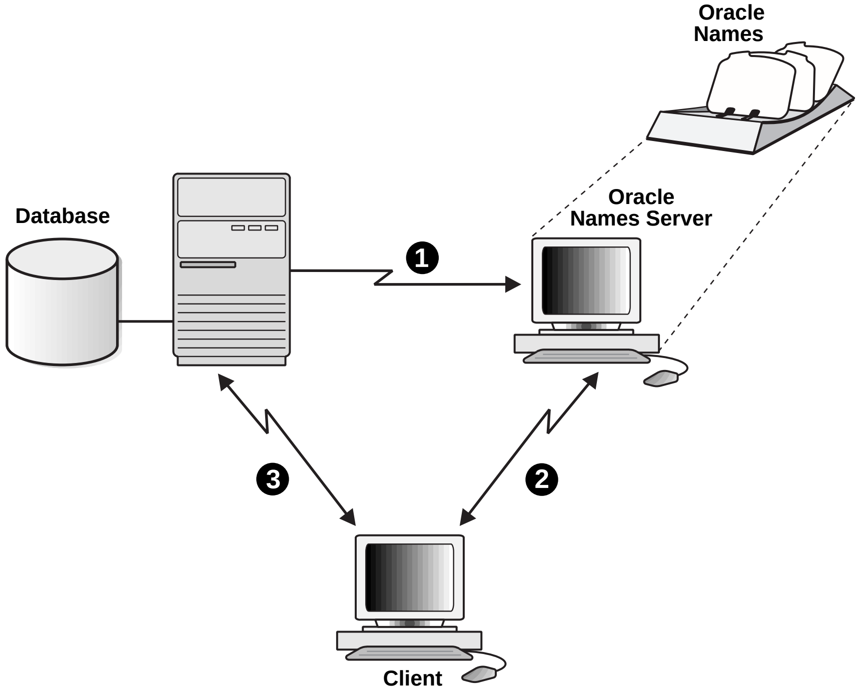
Oracle Names establishes and maintains an integrated system of **Oracle Names servers** which work together like a directory service. The system stores addresses for all the services on a network and makes them available to clients wishing to make a connection.

Much like a caller who uses directory assistance to locate a telephone number, clients configured to use Oracle Names refer their connection requests to an Oracle Names server. The Oracle Names server attempts to resolve the service name provided by the client to a network address. If the Oracle Names server finds the network address, it then returns that information to the client. The client can then use that address to connect to the service.

Figure 4–3 shows how Oracle Names works to help establish a connection between a client and server:

1. A database registers its service with an Oracle Names server.
2. A client seeks to locate a service on the network by contacting an Oracle Names server to retrieve the network address.
3. The client then transparently connects to the service.

Figure 4–3 Oracle Names



See Also: "Configuring the Oracle Names Method" on page 6-47 for configuration information

Administrative Regions

Most networks have one central point of administration, that is, one **administrative region**. An administrative region consists of a collection of Oracle Names servers that administer services in a network. All connect information is stored in a single data repository, which has the authority to interpret a service name. All Oracle Names servers within an administrative region query information from this data repository. If the administrative region uses a database for storage, there is one database per administrative region. There can be any number of Oracle Names servers. Oracle Names provides support for one or more administrative regions.

Most enterprise environments with multiple data centers and many Oracle instances will probably choose to take advantage of multiple administrative regions. This enables each data center to independently define and manage the services in its own environment. At the same time, all service addresses are continuously available to all of the clients in the whole environment. Oracle Names servers transparently forward name resolution requests from clients in foreign administrative regions to the proper Oracle Names server.

Domains

A **domain** is a logical group of machines and network services. Within each domain all names must be unique, but across domains simple names can be repeated.

An administrative region contains one or more domains used to divide administrative responsibilities.

Network domains are similar to file directories used by many operating systems in that they are hierarchical. Unlike file systems however, network domains may or may not correspond to any physical arrangement of databases or other objects in a network. They are simply names spaces developed to prevent name space conflicts.

> **Note:** Although they appear similar, the domains of an Oracle network are completely independent of Domain Name System (DNS) name spaces. For convenience, you may choose to mirror the DNS directory structure in your Oracle network.

Oracle Names As Data Repository

Data in Oracle Names servers is updated through continuous replication between all the Oracle Names servers in the region, or by writing to and reading from a common Oracle database.

For smaller workgroup environments where all of the services are registered dynamically, administrators may configure Oracle Names servers to replicate data continuously among themselves. When a listener registers a new service, information about that service is immediately passed along to other Oracle Names servers in the administrative region.

Alternatively, administrators in large environments normally want to store their registration data in an Oracle database, called the **region database**. A region database consists of tables that store Oracle Names information. If the Oracle Names servers are configured to use an Oracle database as a repository, all service registrations are written to the database. Each Oracle Names server in a given administrative region periodically polls the region database for updated registrations. In this way, new registrations are communicated in a timely manner to all of the Oracle Names servers in a given administrative region. At the same time, it relieves Oracle Names servers of the necessity to communicate directly with each other, and it provides better reliability.

Data Stored in an Oracle Names Server

Below is a description of the types of data stored in an Oracle Names server.

Data	Description
Global database names and addresses	The Oracle Names server retrieves information about the database, including the global database name (database name and domain) and address, from the listener. The address is configured in the `listener.ora` file, and the global database name is registered during database startup or statically configured in the `listener.ora` file. You do not need to register this information.
Other Oracle Names server names and addresses	An Oracle Names server stores the names and addresses of all other Oracle Names servers in the same administrative region. If there is more than one administrative region in a network, the Oracle Names server stores the name and address of at least one Oracle Names server in the root administrative region and each of the immediate sub-regions. You do not need to register this information.
Net service names	If you register net service names with the NAMESCTL control utility or Net8 Assistant, an Oracle Names server stores them. An Oracle Names server also stores gateways to non-Oracle databases and Oracle RDB databases.

Data	Description
Global database links	**Database links** allow a database to communicate with another database. The name of a database link is the same as the global database name of the database to which the link points. Typically, only one database link should exist per database. The following types of database links can be created: ■ 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. ■ A **global database link** in Oracle Names server so anyone in the network can use it. Because Oracle Names retrieves the global database name from the listener, a global database link that is the global database name is automatically registered with the Oracle Names server. Therefore, you do not need to register this information. User name and password credentials for the global database link may be registered with the Oracle Names server using Net8 Assistant.These global database links may be supplemented with link qualifiers defined through the Net8 Assistant. Global database links may be superseded with private and public database links created by individual users. **See Also:** *Oracle8i Distributed Database Systems*
Aliases	An Oracle Names server stores aliases or alternative service names for any defined net service name, database service, or global database link. Aliases may be registered with the Oracle Names server using either the NAMESCTL control utility or Net8 Assistant.
Oracle Connection Managers	An Oracle Names server stores the names and listening addresses of all Oracle Connection Managers on the network. You do not need to register this information.

Organizing and Naming Network Components

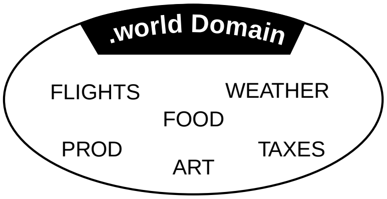
When you use Oracle Names, objects such as databases in a networked environment need to be named in a way as to ensure that they are unique within the network. There are two basic models for naming objects in a network:

- Single Domain Model
- Hierarchical Naming Model

Single Domain Model

The use of the single domain naming model is useful if your network is small, and there is no duplication of names. Figure 4–4 shows a typical flat naming structure using a single domain name, .WORLD.

Figure 4–4 Single Domain Naming Model



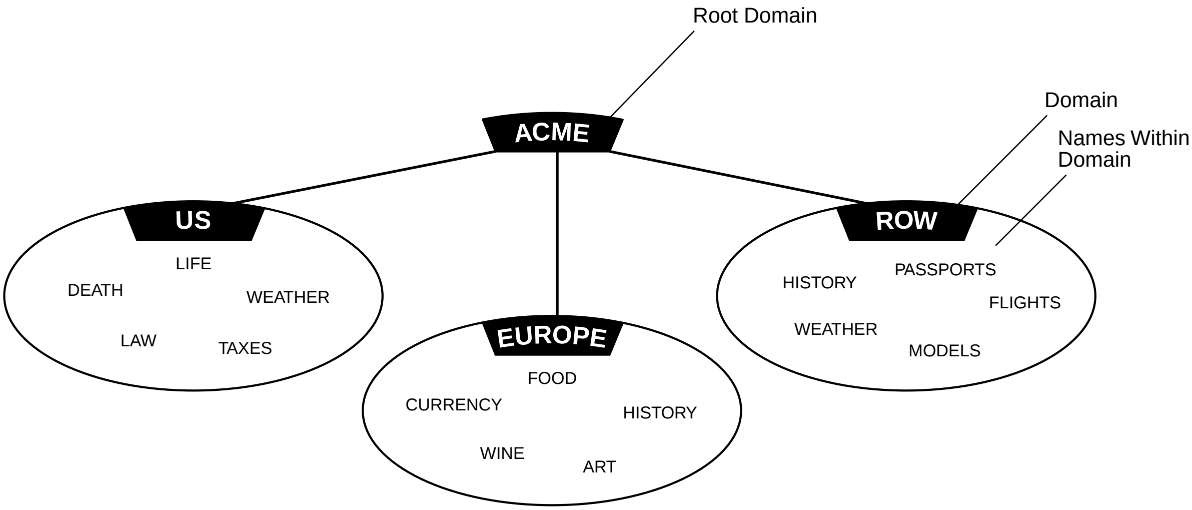
In this environment, database service names are automatically appended with a .WORLD extension (for example, PROD.WORLD, FLIGHTS.WORLD, and so forth).

Hierarchical Naming Model

Hierarchical naming models divide names into a hierarchical structure to allow for future growth or greater naming autonomy. This type of naming model enables more than one database with the same simple name in different domains.

Figure 4–5 shows a hierarchical structure of domains including the (ROOT) domain, ACME domain, US.ACME, EUROPE.ACME, and ROW.ACME (Rest of World) domains.

Figure 4–5 Hierarchical Naming Model



Notice in Figure 4–5 both WEATHER and HISTORY are repeated, but the names remain unique (that is, HISTORY.ROW.ACME and HISTORY.EUROPE.ACME).

Default Domains The default domain is the domain within which most of the client's name requests are conducted. This is usually the domain in which the client resides, though it could also be another domain from which the client most often requests services. A client can request a network service within its default domain using the service's simple, unqualified name, that is, without specifying a domain name. If a user requests a name without a "." character in it, the default domain name is automatically appended to the database service or database link name requested.

For example, a client is configured with a default domain of EUROPE.ACME.COM. When it makes a request for the service name "WINE" in Figure 4–5, the default domain name EUROPE.ACME.COM is appended to the requested name so that the name becomes WINE.EUROPE.ACME.COM.

Multiple Domains Multiple domains are related hierarchically to a root domain (the highest-level domain in the hierarchy) in a series of parent-child relationships. For example, under the root might be several domains, one of which is called COM. Under the COM domain might be several more domains, one of which is ACME. Under the ACME domain might be several domains, such as US, EUROPE, and so forth.

Note: In previous releases of SQL*Net and Oracle Names, a network with only one domain would by default be called ".world". This is no longer a requirement with Net8 and Oracle Names version 8. You may, however, want to keep the same convention to be backward compatible, as well as to avoid having to rename all your databases.

Using Multiple Regions to Decentralize Administrative Responsibilities

If you are using Oracle Names and your network is large or widely distributed geographically, you may choose to have multiple administrative regions. For example, if your network includes both the United States and Europe, you might want to have administrative decisions about the network made locally. To subdivide, you must delegate regions and domains from a parent to a child or subregion.

To delegate administrative regions, you must use a hierarchical naming model with each administrative region controlling one or more different domains.

Networks with multiple administrative regions are composed of one **root administrative region** and one or more **delegated administrative region**.

Root Administrative Regions

The root administrative region contains the **root domain**. The root administrative region contains the following information:

- Oracle Names servers in the root region
- Domains that are administered in this region. This is always at least the root domain, and can include other domains.
- Delegated administrative region Oracle Names servers: The domains and Oracle Names server addresses in any alternate regions which act as direct child regions of the root.
- Data definitions for the root region: All of the database service names, database links and aliases associated with the root administrative region.

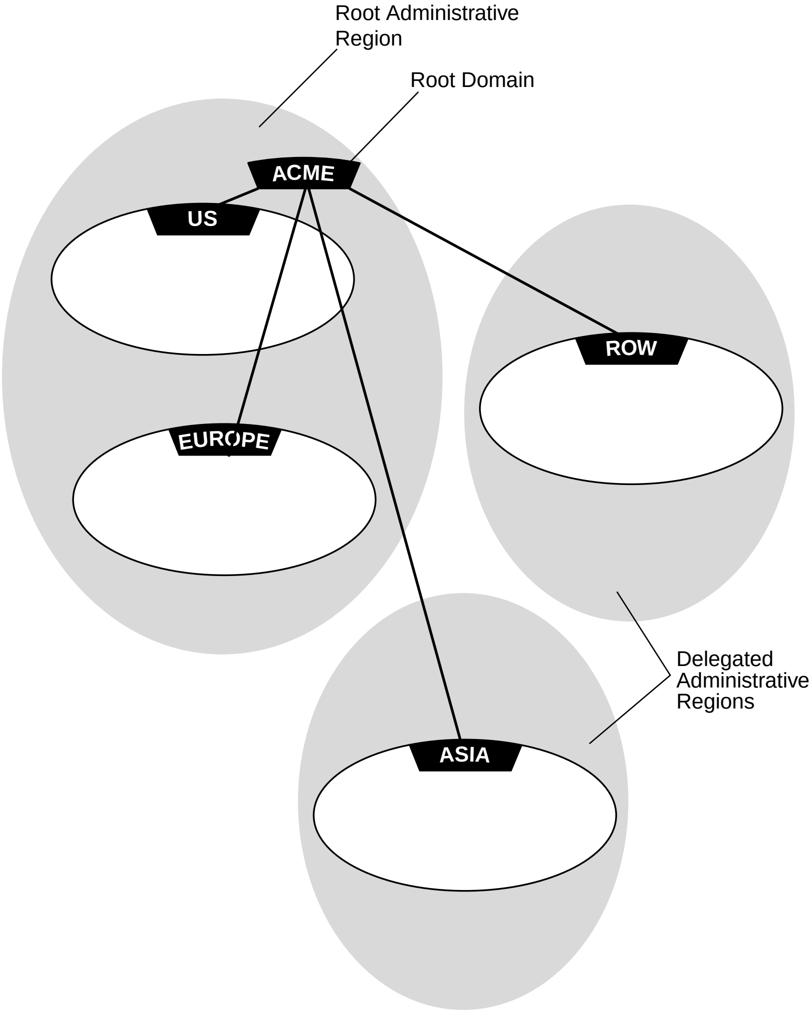
Delegated Administrative Regions

Administrative regions can be delegated from the top of the hierarchy down to other domains in the naming model. For example, a network with ten domains can have between one and ten administrative regions.

All administrative regions other than the root are hierarchically delegated directly or indirectly from it.

Figure 4–6 shows a network with five domains and three administrative regions: the root, ACME, and two delegated regions (ROW, ASIA).

Figure 4–6 Delegated Administrative Regions



Delegated Administrative Regions Below Root

All administrative regions below the root are considered delegated administrative regions. Delegated administrative regions receive administrative responsibilities for a domain from other regions, such a the root administrative region. A delegated administrative region contains the following information:

- All Oracle Names servers and domains in the region
- Domains and Oracle Names server addresses in any of this administrative region's child regions
- Addresses of the Oracle Names servers in the root region. Having this data enables Oracle Names servers in delegated regions to contact any other region (through the root region)
- Data definitions: All of the database service names, database links, and aliases for all of the domains in this local (delegated) administrative region

Differences Between Versions of Oracle Names

There are significant differences between this version of Oracle Names and earlier versions:

- Oracle Names version 1
- Oracle Names Version 2
- Oracle Names Version 8 (this release)

Oracle Names version 1

In Oracle Names version 1, administrators configured Oracle Names servers using Oracle Network Manager and stored all topology data in a database. All the Oracle Names servers in a region shared the same information because they accessed the same database.

The clients had a list of **preferred Oracle Names servers** specified in the `sqlnet.ora` file. This list was created by the user, listing the order of preferred Oracle Names to contact. The first Oracle Names server in the list would be contacted first by a client.

Preferred Oracle Names servers may still be configured.

> **See Also:** "Configuring Preferred Oracle Names Servers" on page 8-26 for configuration information

Oracle Names Version 2

In Oracle Names version 2, the administrator could choose between continuing Oracle Names Server configuration as in version 1, or using the Dynamic Discovery Option. The Dynamic Discovery Option was recommended only for a network with a single region and single DNS domain. The Dynamic Discovery Option uses **well-known Oracle Names servers**, which are precise names hard-coded into DNS or the `hosts` file on both the Oracle Names Server and its clients:

The well-known host names for TCP connections	The well-known computer Names for Named Pipes connections[1]	The well-known service name for an SPX connection
oranamesrvr0	ORANAMESRVR0	oranamesrvr
oranamesrvr1	ORANAMESRVR1	
oranamesrvr2	ORANAMESRVR2	
oranamesrvr3	ORANAMESRVR3	
oranamesrvr4	ORANAMESRVR4	

[1] Well-known Names Server names for Named Pipes must be in all uppercase.

Oracle Names servers then become available at these well-known addresses, so that clients do not need to be told, by way of a preferred Oracle Names server list, where to find an Oracle Names server.

If the Dynamic Discovery Option was chosen, each Oracle Names server automatically replicated its data to all other well-known Oracle Names servers in the administrative region. Listeners were configured to register themselves with well-known Oracle Names servers.

> **See Also:** *Oracle Names Administrator's Guide*, Release 2.0, for configuration information

Oracle Names Version 8 (this release)

In Oracle Names version 8, the administrator may choose between continuing Oracle Names Server configuration as in version 1 or version 2, or using the new functionality. Oracle Names version 8 incorporates version 2 Dynamic Discovery Option features without the constraints of a single region and single domain. The main features of Oracle Names version 8 include:

- A service can register itself with any Oracle Names server it can find, and its name and address are made available to all Oracle Names server in the region. Similarly, if an administrator manually registers a service to any Oracle Names server, that service information is available to all other Oracle Names Servers. The address information is shared in one of two ways:
  - Region Database: If an Oracle database is used as the registration repository, the registration information is stored in the database, and from there is accessible to all the Oracle Names Servers.
  - Service Replication: In service replication, service information is stored in an Oracle Names server's cache and is instantly replicated to the caches of all other Oracle Names servers.
- A list of Oracle Names servers to contact is created on a client through a discovery process rather than manual configuration, as described in "Understanding Discovery" on page 4-24.
- A client-side process may be run to create a client cache version of the data stored in Oracle Names servers, including database addresses, Oracle Names server addresses, as well as other service information. As a result, the client does not have to contact an Oracle Names server for address information, reducing address look up time.
- Oracle Names version 8 does not require a database to hold topology information. However, an administrator may choose to use one, and it is recommended.

> **See Also:** "Configuring the Oracle Names Method" on page 6-47 for configuration information

Understanding Discovery

A list of Oracle Names servers is created that enables a client or another Oracle Names server to contact an Oracle Names server. The process of creating the list is called discovery.

When a client tries to discover an Oracle Names server with the NAMESCTL utility or Net8 Assistant, one Oracle Names server is found first. Once the client finds an Oracle Names server, it pings all other Oracle Names servers in the region. A list of Oracle Names servers is then created on the client and saved to `.sdns.ora` on UNIX and `sdns.ora` on Windows platforms. This list is sorted in order of response time.

Discovery searches for the first Oracle Names server in the following order:

1. A preferred Oracle Names server in the `sqlnet.ora` file
2. A well-known Oracle Names server
3. A local Oracle Names server configured with TCP/IP on port 1575

If the client is unable to find the first Oracle Names using the above methods:

- You are prompted to specify the address of an Oracle Names server if you are using Net8 Assistant.
- The NAMESCTL control utility will fail. You must re-run the command with the address specified.

External Naming Services

Net8 offers the external naming methods:

- **Novell Directory Services (NDS)** External Naming

  NDS external naming enables you to use external NDS naming conventions to connect to an Oracle database on a Novell NDS-enabled network.

- **Network Information Service (NIS)** External Naming

  Organizations and corporations already using Network Information Service (NIS) as part of their system infrastructure have the option to store Oracle service aliases and addresses in NIS, using NIS External Naming.

- **Cell Directory Services (CDS)** External Naming

  CDS external naming enables you to transparently use Oracle tools and applications to access Oracle8*i* databases in a Distributed Computing Environment (DCE) environment.

  > **See Also:** "Configuring External Naming Methods" on page 6-79 for NDS and NIS configuration information

Authentication Methods

Net8 offers the following authentication methods:

- **Windows NT native authentication**

  Enables a client single login access to an Windows NT server and an Oracle database.

- **NDS authentication**

  Enables a client single login access to a multi-server and multi-database network under a single Novell Directory Services (NDS) directory tree.

  > **See Also:**
  >
  > - *Oracle8i Administrator's Guide for Windows NT* for configuration information about Windows NT native authentication
  > - Oracle Novell documentation for configuration information about NDS authentication

  Other authentication methods are available with Oracle Advanced Security, as described in "Single Sign-On" on page 4-26.

Oracle Advanced Security

Oracle Advanced Security, a separately licensable option, consists of the following components:

- Network Security
- Single Sign-On
- Distributed Computing Environment Integration

> **See Also:** *Oracle Advanced Security Administrator's Guide* for complete details on these features

Network Security

This Oracle network data encryption and checksumming service ensures secure transmission of data over networks. Network Security uses encryption and checksumming engines from RSA Data Security, Incorporated.

The following algorithms are supported:

Encryption

- DES_40 (Domestic and Export versions)
- RC4_40 (Domestic and Export versions)
- RC4_56 (Domestic and Export versions)
- RC4_128 (Domestic version)

Checksumming

- MD5

Single Sign-On

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

Centralized, secure authentication services allow you to have high confidence in the identity of users, clients, and servers in distributed environments. Network authentication services can also provide the benefit of single sign–on for users.

The following authentication methods are supported:

- CyberSAFE
- Identix (Biometric)
- Kerberos
- RADIUS
- SecurID
- SSL

Distributed Computing Environment Integration

Distributed Computing Environment (DCE) Integration enables users to transparently use Oracle tools and applications to access Oracle8*i* databases in a DCE environment. The Oracle DCE Integration product consists of two major components:

- DCE Communications/Security method
- DCE CDS (Cell Directory Service) Naming method

Administration Tools

This section introduces the administration tools available with Net8. These tools include:

- Net8 Configuration Assistant
- Net8 Assistant
- Control Utilities

Net8 Configuration Assistant

Net8 Configuration Assistant enables you to configure basic network components.

Net8 Configuration Assistant runs automatically after software installation. as described in your Oracle installation guide. It can be used on either the client or server.

It may be also be run in stand-alone mode to configure naming methods usage, the listener, net service names in the `tnsnames.ora` file, and directory server access.

To start Net8 Configuration Assistant:

- On UNIX, run `netca` from `$ORACLE_HOME/bin`.
- On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Configuration Assistant.

See Also: Net8 Configuration Assistant online help

Net8 Assistant

Net8 Assistant is a graphical user interface tool that combines configuration abilities with component control to provide an integrated environment for configuring and managing Net8. It can be used on either the client or server.

You can use Net8 Assistant to configure the following network components:

- Naming—Define simple names, connect identifiers, and map them to connect descriptors to identify the network location and identification of a service. Net8 Assistant supports configuration of connect descriptors in local `tnsnames.ora` files, centralized LDAP-compliant directory service, or an Oracle Names 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.

If an Oracle Names server is configured, you can start, stop, tune, or gather statistics for it with Net8 Assistant.

To start Net8 Assistant:

- On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
- On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

See Also: Net8 Assistant online help

Control Utilities

Net8 provides control utilities to control listeners, Oracle Names servers, and Oracle Connection Managers. These utilities include:

- LSNRCTL Control Utility
- CMCTL Control Utility
- NAMESCTL Control Utility

LSNRCTL Control Utility

The Listener Control Utility (LSNRCTL) manages the listener. The general syntax of the LSNRCTL is as follows:

```
LSNRCTL command [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.

LSNRCTL contains several types of commands:

- Operational commands such as START, STOP, and so forth
- Modifier commands for a running listener in the form of SET *command*
- Informational commands, such as STATUS, SHOW *command*
- Command utility operational commands such as EXIT, QUIT, and HELP

> **See Also:** "Listener Control Utility (LSNRCTL)" on page A-3

CMCTL Control Utility

The Oracle Connection Manager Control Utility (CMCTL) is a tool that you run from the operating system prompt to start and control Oracle Connection Manager. The general syntax of CMCTL is as follows:

```
CMCTL command [process_type]
```

where *process_type* is the name of the process that the command is being executed on. The choices are:

- `cman` (both CMGW gateway and CMADMIN administrative processes)
- `adm` (only the CMADMIN process)
- `cm` (only the CMGW process)

The CMCTL utility contains several types of commands:

- Operational commands such as START, STOP, SHUTDOWN and so forth
- Modifier commands for a running Oracle Connection Manager in the form of SET *command*
- Informational commands, such as STATUS, STATS, and SHOW *command*
- Command utility operational commands, such as EXIT, QUIT, and HELP

> **See Also:** "Oracle Connection Manager Control Utility (CMCTL)" on page A-79

NAMESCTL Control Utility

The Oracle Names Control Utility (NAMESCTL) is a tool that you run from the operating system prompt to start and control Oracle Names servers. The general syntax of the NAMESCTL is as follows:

```
NAMESCTL command
```

NAMESCTL contains several types of commands:

- Operational commands such as START, STOP, RESTART, and so forth.
- Modifier commands for a running Oracle Connection Manager in the form of SET *command*
- Informational commands, such as STATUS, SHOW *command*, and PING
- Command utility operational commands, such as EXIT, QUIT, and HELP

> **See Also:** "Oracle Names Control Utility (NAMESCTL)" on page A-26

5

Planning Your Network

Net8 provides a variety of options to help you design and manage networks that are both flexible and easy to use. With Net8's 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 Net8. It explains the relationships of the network products, and options for expanding and better managing your future network. It includes the following sections:

- Planning Overview
- Defining Your Network Layout
- Resolving Names to Connect Descriptors
- Improving Large Network Performance
- Planning Summary

Planning Overview

Take the time to review and plan your network before you configure it. As you are planning your Oracle network, remember to keep future needs in mind as well as present requirements. Some of the more important decisions which you will need to make regarding your network include:

- Defining Your Network Layout
- Resolving Names to Connect Descriptors
- Improving Large Network Performance

Defining Your Network Layout

The following checklist is provided to help you outline the main components of your network.

1. Define from the outset what it is you hope to accomplish with your network.
2. Research the functionality required by your client applications, then assess the resources that are available to meet those requirements.
3. Determine which machines or nodes are best suited for client or server applications.
4. Select a networking protocol which best suits your existing or future networking requirements. Protocols are available for most of the major protocols on many platforms. You may be able to choose a single transport level protocol that works well on all the components in your network. Your network may involve clients or servers operating over more than one protocol.
5. If you decide to use multiple protocols on your network, determine which nodes are best suited to install Oracle Connection Manager. Your choice of nodes will be determined by the networking protocols you have chosen as well as the machine’s capacity to handle anticipated traffic.

It helps sometimes to draw a picture of your network layout displaying the logical as well as physical relationships between networking components.

Resolving Names to Connect Descriptors

Once you have defined your network layout, you will need to decide how best to configure and manage your network implementation. One of the first and most important decisions that you will need to make is choosing a naming method.

The following table summarizes the relative advantages and disadvantages of each naming method and provides recommendations for using them in your network.

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.

Naming Method	Advantages/Disadvantages	Recommended for:
Oracle Names	**Advantage**: Centralizes network names and addresses in a single place, facilitating administration of name changes and updates. For example, whenever a change is made to an existing server or a new server is added to the network, the change is made only once on one Oracle Names server. This eliminates the need for an administrator to make changes to what potentially could be hundreds or even thousands of clients. **Disadvantages**: ■ Oracle Names stores network names and addresses for Oracle services only. ■ Requires additional setup and administration of Oracle Names servers.	Large, complex networks (over 20 databases) that change on a frequent basis.
Host Naming	**Advantages**: ■ Requires minimal user configuration. The user may provide only the name of the 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. ■ The host name is resolved through an IP address translation mechanism such as Domain Name Services (DNS), Network Information Services (NIS), or a centrally maintained TCP/IP `hosts` file. ■ No Oracle Connection Manager features are requested.
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-naming services that cannot be administered using Net8 products	Networks with existing name services.

Improving Large Network Performance

You may improve the performance of large networks by implementing any of the following:

- Using Shared Servers or Prespawned Dedicated Servers
- Increasing Listener Queuesize
- Enabling Connection Pooling
- Enabling Connection Concentration
- Enabling Client Load Balancing
- Adjusting the Session Data Unit (SDU) Size for Data Transfer Optimization
- Persistent Buffer Flushing for TCP / IP

Using Shared Servers or Prespawned Dedicated Servers

If you expect your network to receive a lot of connection traffic, you can use the listener to manage these requests by redirecting them to either **multi-threaded server (MTS)** shared servers or prespawned dedicated servers.

The following table summarizes the relative advantages of each, and provides recommendations for using them in your network.

Type	Advantages	Recommended for:
MTS Shared servers	■ Utilizes network resources more efficiently than a dedicated server, thus increasing the throughput and performance of your sessions ■ Enables you to minimize the memory and processing resources needed on the server side as the number of sessions to the database increases	Networks where MTS is supported, or where the creation of a new server process is slow and resource-intensive.
Prestarted or prespawned dedicated servers	■ Reduces connect time by eliminating the need to create a dedicated server for each new connection request ■ Provides better use of allocated memory and system resources by recycling servers for use by other connections without having to shut down and recreate a server process	Networks where MTS is not supported, or where the creation of a new server process is slow and resource-intensive.

See Also:

- "Multi-Threaded Server Model" on page 2-7 for MTS conceptual information
- Chapter 9, "Configuring Multi-Threaded Server"
- "Prespawned Dedicated Servers" on page 2-16 for conceptual information
- "Configuring Prespawned Dedicated Servers" on page 7-16 for configuration information

Increasing Listener Queuesize

If you anticipate receiving a large number of connection requests for a listening process (such as a listener, Oracle Connection Manager or Oracle Names) over TCP/IP, Net8 enables you to configure the listening queue to be higher than the system default.

See Also: "Handling Large Volumes of Connection Requests" on page 7-12 for information about configuring the existing queuesize

Enabling Connection Pooling

Connection pooling is a resource utilization feature that enables you to maximize the number of physical network connections to a multi-threaded server. This is achieved by sharing or pooling a dispatcher’s set of connections among multiple client processes.

By using a time-out mechanism to temporarily release transport connections that have been idle for a specified period of time, connection pooling makes these physical connections available for incoming clients, while still maintaining a logical session with the previous idle connection. When the idle client has more work to do, the physical connection is reestablished with the dispatcher.

This feature only works when multi-threaded server is configured.

See Also:

- "Connection Pooling" on page 2-12 for conceptual information
- "Using MTS on Clients" on page 9-10 for configuration details

Enabling Connection Concentration

Connection concentration is a feature that is available through Oracle Connection Manager. It enables you take advantage of Oracle Connection Manager's ability to multiplex or funnel multiple client sessions over a single transport to a multi-threaded server. Like connection pooling, connection concentration optimizes network resources and increases the number of client-server sessions that are possible across a fixed number of physical server ports. Unlike connection pooling, connection concentration maintains the transport connection.

See Also:

- "Oracle Connection Manager" on page 4-5 for conceptual information
- "Enabling Connection Concentration" on page 8-28 for configuration information

Using Connection Pooling and Connection Concentration

The following table summarizes the relative advantages of using connection pooling and connection concentration and provides recommendations for using them in your network.

Feature	Advantages	Recommended for:
Connection Pooling	■ Limits the number of network resources used per process ■ Maximizes the number of client/server sessions over a limited number of physical connections ■ Optimizes resource utilization	Networks where many clients run interactive "high think/search time" applications such as messaging and OLAP

Feature	Advantages	Recommended for:
Connection Concentration	▪ Supports large client populations. ▪ 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.	Networks where "continuous" connectivity is required.

Enabling Client Load Balancing

When more than one listener supports a service, a client can be configured to randomize requests to the various listeners. The randomization serves to distribute the load so as not to overburden a single listener. By balancing the load, you can improve connection performance.

> **See Also:** "Configuring Address List Parameters" on page 8-4 for configuration details

Adjusting the Session Data Unit (SDU) 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 Net8 uses to place data before transmitting it across the network. Net8 sends the data in the buffer either when requested or when it is full.

The following table outlines considerations when modifying the size of the SDU may or may not be appropriate.

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 ■ Your packet size is consistently the same ■ Large amounts of data are returned	■ Your application can be tuned to account for the delays ■ 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

Note: The SDU size should be set as a multiple of the normal transport frame size. Since the normal Ethernet frame size is 1500, the most efficient SDU size over an Ethernet protocol should be a multiple of 1500.

If you are using either connection pooling or connection concentration, keep in mind that these features require an additional 16 bytes per transport. For more information about the protocol frame size, refer to your operating-system-specific documentation.

See Also: "Configuring Advanced Connect Data Parameters" on page 8-8 for configuration details

Persistent Buffer Flushing for TCP/IP

Under certain conditions in some applications using TCP/IP, Net8 packets may not get flushed immediately to the network. Most often, this behavior occurs when large amounts of data are streamed from one end to another. The implementation of TCP/IP itself is the reason for the lack of flushing, and can cause unacceptable delays. To remedy this problem, you can specify no delays in the buffer flushing process.

See Also: "TCP.NODELAY" on page C-78 for information about this parameter

Planning Summary

The table below summarizes many of the options you may have chosen as you planned your network.

Subject	Options
Network Layout	■ Single or Multiple Protocols
Net Service Name Resolution	■ Local Naming ■ Directory Naming ■ Oracle Names ■ Host Naming ■ External Naming
Connection Request Management	■ Shared Servers ■ Prespawned Dedicated Servers
Network Performance	■ Increasing Listener queuesize ■ Enabling Client Load Balancing ■ Enabling Connection Pooling ■ Enabling Client Load Balancing ■ Adjusting the SDU Size ■ Persistent Buffer Flushing

Part II

Configuration

Part II describes how to set up and configure Net8. Part II contains the following chapters:

- Chapter 6, "Configuring Naming Methods"
- Chapter 7, "Configuring the Listener"
- Chapter 8, "Enabling Advanced Net8 Features"
- Chapter 9, "Configuring Multi-Threaded Server"
- Chapter 10, "Enabling Net8 Enhancements for Programmers"

6

Configuring Naming Methods

This chapter describes how to configure naming methods. This chapter includes the following sections:

- Configuration Overview
- Understanding Connect Descriptors
- Configuration Models
- Configuring the Local Naming Method
- Configuring the Directory Naming Method
- Configuring the Oracle Names Method
- Configuring the Host Naming Method
- Configuring External Naming Methods

Configuration Overview

To connect to a service, clients use a simple name, called a **connect identifier** in their connect string to connect to a service, rather than a complete **connect descriptor**. The connect descriptor contains:

- Network route to the service, including the location of the listener through a protocol address
- Service name of an Oracle release 8.1 database or **Oracle System Identifier (SID)** of an Oracle release 8.0 or version 7 database

A connect identifier can be **net service name** (a simple name for a service) or the actual name of the service. A connect identifier is resolved to a connect descriptor by a **naming method** specified for the client.

Once the connect identifier is resolved, the client then forwards the connection request to the listener. The listener compares the client information with the information it has received from the database service, as well as information it has stored in its own configuration file, `listener.ora`. If the information matches, a connection is granted.

Naming method configuration consists of creating connect descriptors for services and configuring clients to access the naming method.

Understanding Connect Descriptors

A connect descriptor is comprised of one or more protocol addresses of the listener and connect data information for the destination service. The following example shows a typical connect descriptor:

```
sales=
 (description=
  (address= (protocol=tcp)(host=sales-server)(port=1521))
  (connect_data=
     (service_name=sales.us.acme.com)))
```

The ADDRESS portion contains the listener protocol address, and the CONNECT_ DATA portion 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 for an Oracle release 8.1 database service, you must identify the service with the SERVICE_NAME parameter. Optionally, you may 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 (with an S) and INSTANCE_NAME parameters in the **initialization parameter file**. The SERVICE_NAMES parameter in the initialization parameter file is typically the **global database name**, a name comprised of 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 is defaulted to the SID entered during installation or database creation.

> **See Also:** "Database Identification by Service Name Rather than SID" on page 2-32

When creating a connect a descriptor for an Oracle release 8.0 or version 7 database, you must identify the service with the SID parameter. The value for the SID should match the ORACLE_SID environment variable or the registry entry. The following example shows a connect descriptor for an Oracle release 8.0 database with a SID of `sales`:

```
sales=
 (description=
  (address= (protocol=tcp)(host=sales-server)(port=1521)
  (connect_data=
     (sid=sales)))
```

Configuration Models

Net8 configuration is based upon one of two models:

Network Configuration Model	Suitable When...	Naming Method
Localized management	Network addresses are mapped in a tnsnames.ora file on each machine	Local Naming Host Naming
Centralized management	Networks where an Oracle Names server or directory performs network address resolution	Directory Naming Oracle Names External Naming

Naming Methods

Net8 supports the following naming methods:

Naming Method	Description
Local Naming	Stores net service names 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 6-5
Directory Naming	Stores services and net service names in a centralized LDAP-compliant directory service **See Also**: "Configuring the Directory Naming Method" on page 6-16
Oracle Names	Stores services and net service names in an Oracle Names server. **See Also**: "Configuring the Oracle Names Method" on page 6-47
Host Naming	Enables users to connect to an Oracle server by using a host name alias. Host names are mapped to the server's global database name in an existing names resolution service, such as Domain Name System (DNS), Network Information Service (NIS), or a centrally-maintained set of /etc/hosts files. No client configuration is required to take advantage of this feature. This method is recommended for simple TCP/IP environments. **See Also**: "Configuring the Host Naming Method" on page 6-74
External Naming	Stores service information in an a third-party naming service **See Also**: "Configuring External Naming Methods" on page 6-79

See Also: "Service Resolution through Naming Methods" on page 2-34

Configuring the Local Naming Method

With the local naming method, net service names are added to the tnsnames.ora file. Each net service name is mapped to a connect descriptor. An example of a tnsnames.ora file's basic syntax for a net service name mapped to a connect descriptor is shown in the following:

```
sales=
(description=
  (address=(protocol=tcp)(host=sales-server)(port=1521))
  (connect_data=
     (service_name=sales.us.acme.com)))
```

In the this example, sales is the net service name that is mapped to the connect descriptor contained in DESCRIPTION. DESCRIPTION contains the listener address and the destination database service identified.

Local naming configuration can be configured during or after installation, as described in the following sections:

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

Configuring the tnsnames.ora File During Installation

Net8 Configuration Assistant enables you to configure net service names for clients. It is launched by the Oracle Universal Installer after software installation. The configuration varies depending on the installation mode:

- Typical or Minimal Installation
- Custom Installation

Typical or Minimal Installation

Net8 Configuration Assistant prompts you to configure net service names in the `tnsnames.ora` file to connect to an Oracle database service.

Custom Installation

Net8 Configuration Assistant prompts you to select naming methods to use. If Local is selected, Net8 Configuration Assistant prompts you to configure net service names in a `tnsnames.ora` file to connect to an Oracle database services.

Configuring the tnsnames.ora File After Installation

Net service names can be added to the `tnsnames.ora` file at any time. 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

Task 1: Configure Net Service Names

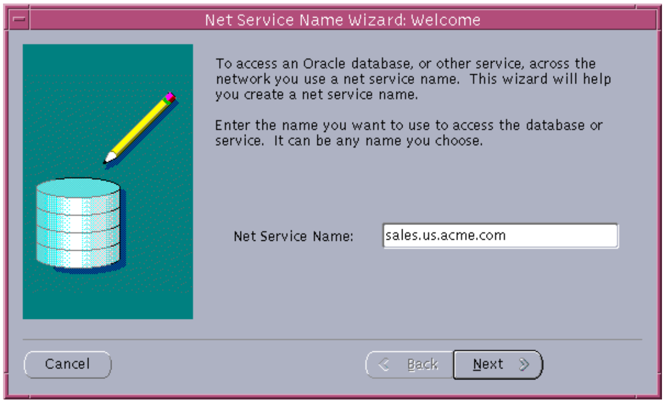
To configure with the local naming method, use either one of the following tools:

- Net8 Assistant
- Net8 Configuration Assistant

Net8 Assistant To configure net service names in the `tnsnames.ora` file with **Net8 Assistant**:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Service Naming.
3. Click "+" from the toolbar, or choose Create from the Edit menu.

   The Net Service Name Wizard starts.
4. Enter any name in the Net Service Name field, then click Next:



Net Service Name Wizard: Welcome

To access an Oracle database, or other service, across the network you use a net service name. This wizard will help you create a net service name.

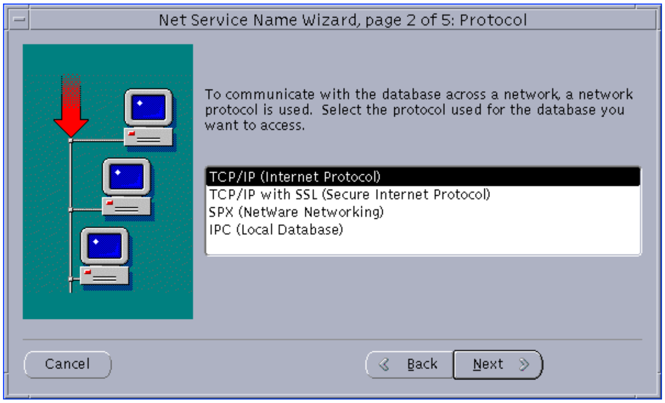
Enter the name you want to use to access the database or service. It can be any name you choose.

Net Service Name: sales.us.acme.com

The net service name can be qualified with the client's domain. The net service name is automatically domain qualified if the NAMES.DEFAULT_DOMAIN parameter is specified in the `sqlnet.ora` file.

> **See Also:** "Configuring a Default Domain for Clients" on page 8-17

5. Select the protocol on which the listener is configured to listen. Note that this protocol must also installed on the client. Click Next:

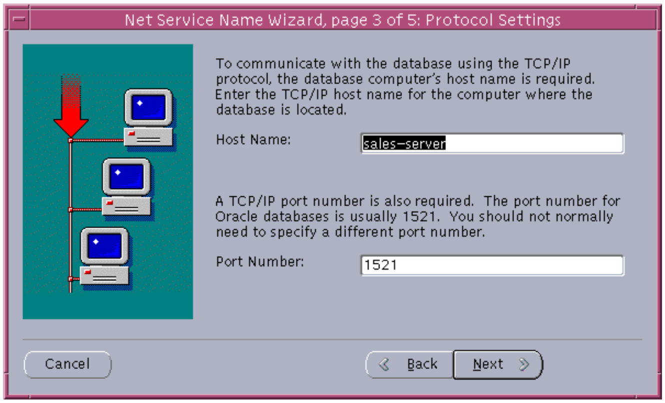


Net Service Name Wizard, page 2 of 5: Protocol

To communicate with the database across a network, a network protocol is used. Select the protocol used for the database you want to access.

- TCP/IP (Internet Protocol)
- TCP/IP with SSL (Secure Internet Protocol)
- SPX (NetWare Networking)
- IPC (Local Database)

6. Enter the appropriate protocol parameters for the chosen protocol in the fields provided, then click Next:



Net Service Name Wizard, page 3 of 5: Protocol Settings

To communicate with the database using the TCP/IP protocol, the database computer's host name is required. Enter the TCP/IP host name for the computer where the database is located.

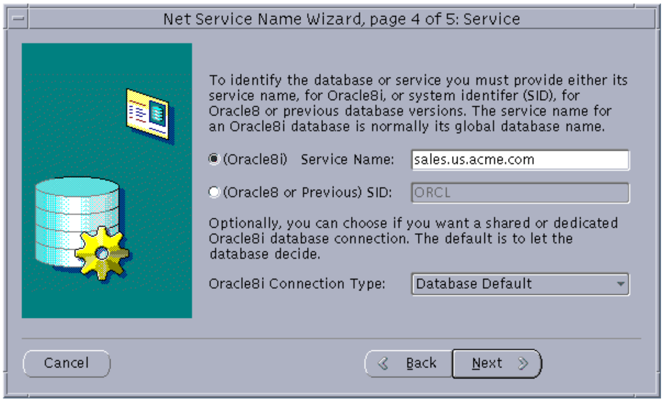
Host Name: sales–server

A TCP/IP port number is also required. The port number for Oracle databases is usually 1521. You should not normally need to specify a different port number.

Port Number: 1521

See Also: Appendix B for protocol parameter settings

7. Select a release, enter a destination service, then click Next:



Net Service Name Wizard, page 4 of 5: Service

To identify the database or service you must provide either its service name, for Oracle8i, or system identifer (SID), for Oracle8 or previous database versions. The service name for an Oracle8i database is normally its global database name.

● (Oracle8i) Service Name: sales.us.acme.com

○ (Oracle8 or Previous) SID: ORCL

Optionally, you can choose if you want a shared or dedicated Oracle8i database connection. The default is to let the database decide.

Oracle8i Connection Type: Database Default

If the destination service is Oracle release 8.1, click (Oracle8i), and enter a service name in the Service Name field. If destination service is Oracle release 8.0 or version 7 database, click (Oracle8 or Previous), and enter an Oracle System Identifier for an instance in the Database SID field.

> **See Also:** "Understanding Connect Descriptors" on page 6-2 for further information about the service name string to use

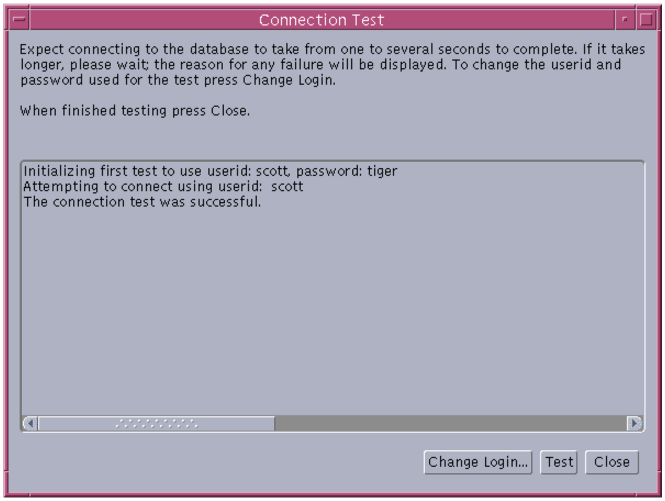
Optionally, you can select a database connection type from the Oracle8i Connection Type list for this net service name. Oracle Corporation recommends you use the default setting of Database Default. If **multi-threaded server (MTS)** is configured in the initialization parameter file, you can select Dedicated Server to force the listener to spawn a dedicated server, bypassing MTS configuration. If MTS is configured in the initialization parameter file and you want to guarantee the connection always uses MTS, select Shared Server.

> **See Also:** Chapter 9 for further information about MTS configuration

8. Click Test to verify that the net service name works, then click Next.

   Testing assumes the database and listener are running. If they are not, see "Using Net8 Control Utilities" on page 11-3 to start components, or click Next to disregard testing and continue to Step 10.

A successful test results in the following message:



Expect connecting to the database to take from one to several seconds to complete. If it takes longer, please wait; the reason for any failure will be displayed. To change the userid and password used for the test press Change Login.

When finished testing press Close.

```
Initializing first test to use userid: scott, password: tiger
Attempting to connect using userid:  scott
The connection test was successful.
```

If the test was not successful:

- Ensure that the database and listener are running, then click Test.
- Click Change Login to change the user name and password for the connection, then click Test.

9. Click Close to dismiss the Connect Test dialog box.
10. Click Finish to save your configuration and dismiss Net Service Name Wizard.

See Also:

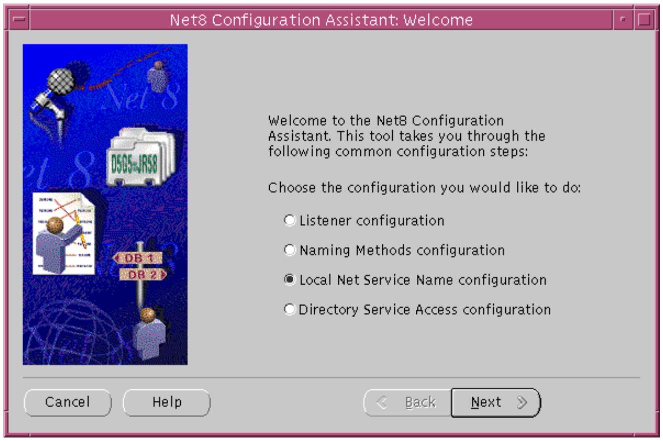
- "Creating a List of Listener Protocol Addresses" on page 8-2 to configure multiple protocol addresses
- "Configuring Advanced Connect Data Parameters" on page 8-8 to configure additional connect data options

Net8 Configuration Assistant To configure net service names in the `tnsnames.ora` file with Net8 Configuration Assistant:

1. Start Net8 Configuration Assistant:

   - On UNIX, run `netca` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Configuration Assistant.

   The Welcome page appears:



Net8 Configuration Assistant: Welcome

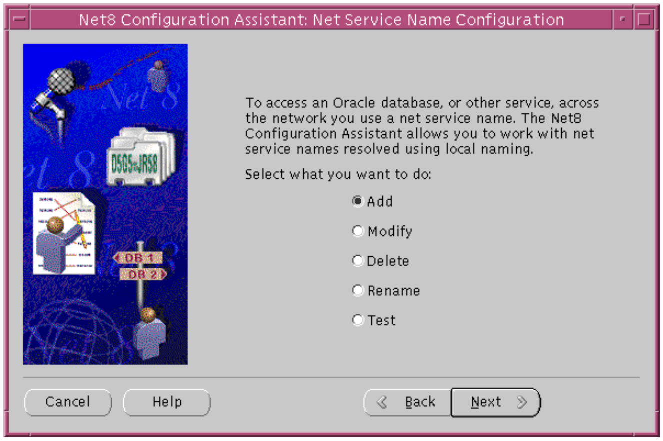
Welcome to the Net8 Configuration Assistant. This tool takes you through the following common configuration steps:

Choose the configuration you would like to do:

○ Listener configuration
○ Naming Methods configuration
● Local Net Service Name configuration
○ Directory Service Access configuration

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

The Net Service Name Configuration page appears:



Net8 Configuration Assistant: Net Service Name Configuration

To access an Oracle database, or other service, across the network you use a net service name. The Net8 Configuration Assistant allows you to work with net service names resolved using local naming.

Select what you want to do:

● Add

○ Modify

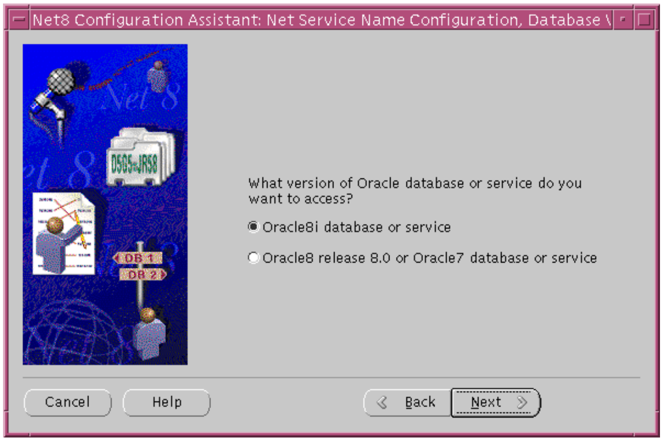
○ Delete

○ Rename

○ Test

3. Click Add, then Next.

   The Net Service Name Configuration, Database Version page appears:



Net8 Configuration Assistant: Net Service Name Configuration, Database \

What version of Oracle database or service do you want to access?

● Oracle8i database or service

○ Oracle8 release 8.0 or Oracle7 database or service

4. Select a release, then click Next:

   If the destination service is an Oracle8 release 8.1 database, click "Oracle8i database or service". If destination service is an Oracle8 release 8.0 or version 7 database, click "Oracle8 release 8.0 or Oracle7 database or service".

5. Follow the prompts in the wizard and online help to complete net service name creation.

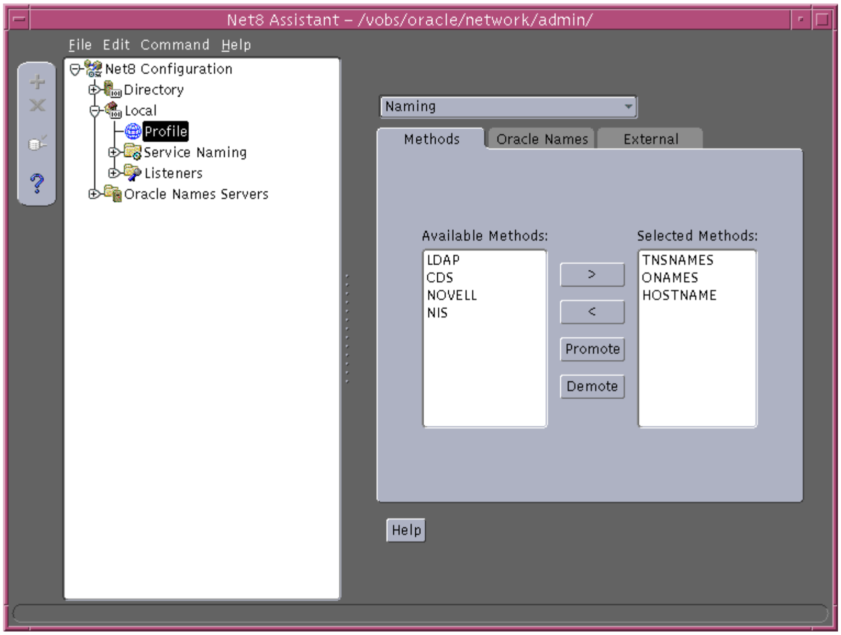
Task 2: Configure TNSNAMES as the First Naming Method

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 Net8 uses to resolve connect identifiers to connect descriptors.

To specify local naming as the first naming method:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Profile.
3. From the list in the right pane, select Naming.
4. Click the Methods tab.
5. Select TNSNAMES from the Available Methods list, then click the right-arrow button.

6. Select TNSNAMES in the Selected Methods list, 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, onames, hostname)
```

Task 3: Distribute Configuration

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

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 7 for listener configuration details

Task 5: Connect to the Database

Clients can connect to the database using the following syntax:

```
CONNECT username/password@net_service_name
```

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, Microsoft's Active Directory, or Novell Directory Services. 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 **Oracle Database Configuration Assistant** during installation; net service name entries can be created with **Net8 Assistant**. Net8 Assistant can also be used to modify Net8 attributes of a database service entry and the net service name entries.

Client configured to access the directory can use these entries to connect to the database.

This section discusses the topics: in the following sections:

- Directory Naming Configuration Steps
- Modifying Connectivity Information for Database Service Entries
- Adding Users to and Removing Users from the OracleNetAdmins Group
- Exporting Net Service Names from a tnsnames.ora File
- Exporting Network Objects from an Oracle Names Server

Directory Naming Configuration Steps

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

Task 1: Configure Directory Access on Server and Clients

Task 2: Create Net Service Name Entries (Optional)

Task 3: Configure LDAP as the First Naming Method

Task 4: Configure the Listener

Task 1: Configure Directory Access on Server and Clients

Before a database service or net service name can be added to a directory, directory access configuration must be completed. Directory access can be configured during or after installation, as described in the following sections.

- Configuring Directory Access During Installation
- Configuring Directory Access After Installation

Configuring Directory Access During Installation

Net8 Configuration Assistant is launched by Oracle Universal Installer after software installation. It enables you to configure access to a directory. Directory access configuration varies depending on the installation mode, as described in the following sections.

- Custom Installation on the Server
- Client Installation

Custom Installation on the Server After a Custom installation on the server, Net8 Configuration Assistant prompts you to configure access to a directory. Directory access configuration enables:

- Oracle Database Configuration, which runs after Net8 Configuration Assistant, to create a database service entry in the directory
- Net8 Assistant to create net service names in the directory, as well as to modify Net8 attributes of the database service entry and the net service name entries
- The server to look up database service and net service name entries in the directory

Note: Directory access configuration is not available during a Typical or Minimal installation on the server.

During directory access configuration, Net8 Configuration Assistant prompts you to configure the following directory access settings:

- Type of directory
- Location of the directory
- **Administrative context** from which this server can look up, create, and modify connect identifiers

  The administrative context is a directory entry that contains an **Oracle Context** (`cn=OracleContext`). An Oracle Context is the root of a directory subtree under which all Oracle software relevant information is kept.

This information is stored in a read-only `ldap.ora` file that the client reads to locate the directory and Oracle entries to access.

If an Oracle Context does not exist in the directory, Net8 Configuration Assistant prompts you to create it. If the Oracle Context is created successfully, the authenticated user is added to the following groups in the directory:

- OracleDBCreators (`cn=OracleDBCreators,cn=OracleContext`)
- OracleNetAdmins (`cn=OracleNetAdmins,cn=OracleContext`)
- OracleSecurityAdmins (`cn=OracleSecurityAdmins,cn=OracleContext`)

Being a member of the OracleDBCreators and the OracleNetAdmins groups enables this user to use Oracle Database Configuration Assistant to create a database service and Net8 Assistant to create net service names or modify Net8 attributes of database services or net service names. A directory administrator can add other users to these groups.

See Also:

- "Adding Users to and Removing Users from the OracleNetAdmins Group" on page 6-31 to add users to the NetAdmins group
- *Oracle Advanced Security Administrator's Guide* for further information about adding users to the OracleDBCreators group

In addition, Net8 Configuration Assistant verifies that the **Oracle schema** was created. The Oracle schema defines the Oracle entries and their attributes. If the schema does not exist or is an older version, you are prompted to create it.

After Net8 Configuration Assistant completes configuration, Oracle Database Configuration Assistant creates the database. The service name for the database is automatically created under the Oracle Context.

See Also: Oracle installation guide

Client Installation If you choose to configure the directory naming method, Net8 Configuration Assistant prompts you to configure access to a directory. Directory access configuration enables the client to look up connect identifier entries in the directory. If directory access is not configured, the client cannot use directory naming.

Net8 Configuration Assistant typically performs the necessary directory access configuration during client installation, and stores the following in a read-only `ldap.ora` file:

During directory access configuration, Net8 Configuration Assistant prompts you to configure the following directory access settings:

- Type of directory
- Location of the directory
- Administrative context from which this client can look up connect identifiers

This information is stored in a read-only `ldap.ora` file that the client reads to locate the directory and Oracle entries to access.

In addition, Net8 Configuration Assistant verifies that the Oracle schema was installed. If an Oracle Context or the Oracle schema was not configured by the server, you cannot complete directory access configuration on the client.

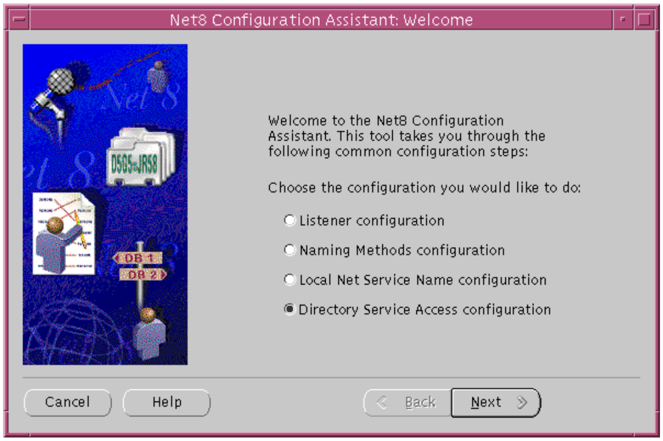
Configuring Directory Access After Installation

If you plan to use the directory naming method, directory access must be configured. Directory access can be configured with Net8 Configuration Assistant at any time.

To configure directory access:

1. Start the Net8 Configuration Assistant:
   - On UNIX, run `netca` at `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Configuration Assistant.

The Welcome page appears:



Net8 Configuration Assistant: Welcome

Welcome to the Net8 Configuration Assistant. This tool takes you through the following common configuration steps:

Choose the configuration you would like to do:

○ Listener configuration

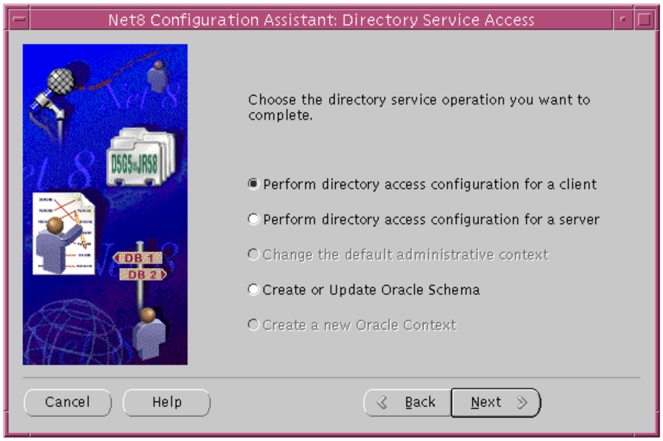
○ Naming Methods configuration

○ Local Net Service Name configuration

● Directory Service Access configuration

2. Select "Directory Service Access configuration", then click Next.

The Directory Service Access page appears:



Choose the directory service operation you want to complete.

● Perform directory access configuration for a client

○ Perform directory access configuration for a server

○ Change the default administrative context

○ Create or Update Oracle Schema

○ Create a new Oracle Context

Option	Description
Perform directory access configuration for client	Click to configure directory access on the clients. This option enables the client to look up directory entries. This option does not enable the client to add or modify entries in the directory. Directory access configuration on the client prompts you to configure the following: ■ Type of directory ■ Location of the directory ■ Administrative context from which this client can look up connect identifiers **Note:** If no Oracle Context or Oracle schema exists, you cannot configure the client for directory access.

Option	Description
Perform directory access configuration for server	Click to configure directory access on the server. This option enables the server to add, modify, and look up directory entries. Directory access configuration on the server prompts you to configure the following: ■ Type of directory ■ Specify the location of the directory ■ Administrative context from which this server can access and create Oracle entries If an Oracle Context does not exist in the directory, you are prompted to create one. Net8 Configuration Assistant also verifies the Oracle schema is created in the directory. If the schema does not exist or is an older version, you are prompted to create it. If the Oracle Context is created successfully, the authenticated user is added to the following groups in the directory: ■ OracleDBCreators (`cn=OracleDBCreators,cn=OracleContext`) ■ OracleNetAdmins (`cn=OracleNetAdmins,cn=OracleContext`) ■ OracleSecurityAdmins (`cn=OracleSecurityAdmins,cn=OracleContext`) Being a member of the OracleDBCreators and the OracleNetAdmins groups enables this user to use Oracle Database Configuration Assistant to create a database service and Net8 Assistant to create net service names or modify Net8 attributes of database services or net service names. A directory administrator can add other users to these groups. **See Also:** ■ "Adding Users to and Removing Users from the OracleNetAdmins Group" on page 6-31 to add users to the OracleNetAdmins group ■ *Oracle Advanced Security Administrator's Guide* for further information about adding users to the OracleDBCreators group
Create or Update Schema	Click to create or update the Oracle schema in the directory.

3. Select the appropriate option, then follow the prompts in the wizard and online help to complete directory access configuration.

Task 2: Create Net Service Name Entries (Optional)

You can configure clients to use a net service name rather than the database service entry created by Oracle Database Configuration Assistant. Net service names are created under the Oracle Context (`cn=OracleContext`).

Note: Only users that are members of the OracleNetAdmins group are allowed to create net service entries in a directory. The authenticated user that created the Oracle Context during directory access on the server is automatically a member of this group. To add or remove users from this group, see "Adding Users to and Removing Users from the OracleNetAdmins Group" on page 6-31.

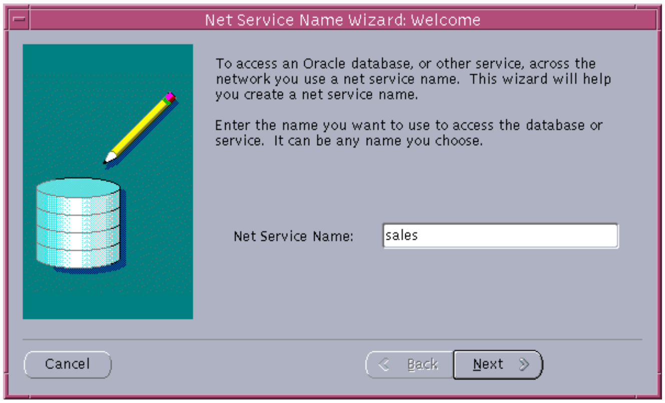
Note: Net service names stored in a `tnsnames.ora` file or an Oracle Names server can be exported into a directory server, as described in "Exporting Net Service Names from a tnsnames.ora File" on page 6-32 and "Exporting Network Objects from an Oracle Names Server" on page 6-35.

To create a net service name in a directory server:

1. Start Net8 Assistant on a machine that has been configured with directory access for a server:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Directory > Service Naming.
3. Click "+" from the toolbar, or choose Create from the Edit menu.

   The Net Service Name Wizard starts.

4. Enter any name in the Net Service Name field, then click Next:



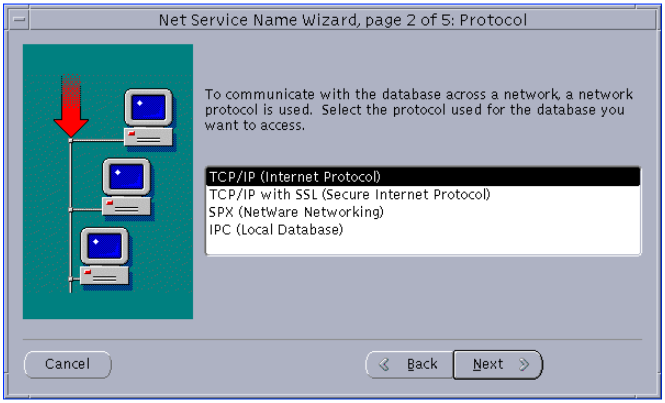
Net Service Name Wizard: Welcome

To access an Oracle database, or other service, across the network you use a net service name. This wizard will help you create a net service name.

Enter the name you want to use to access the database or service. It can be any name you choose.

Net Service Name: sales

5. Select the protocol on which the listener is configured to listen. Note that this protocol must also installed on the client. Click Next:

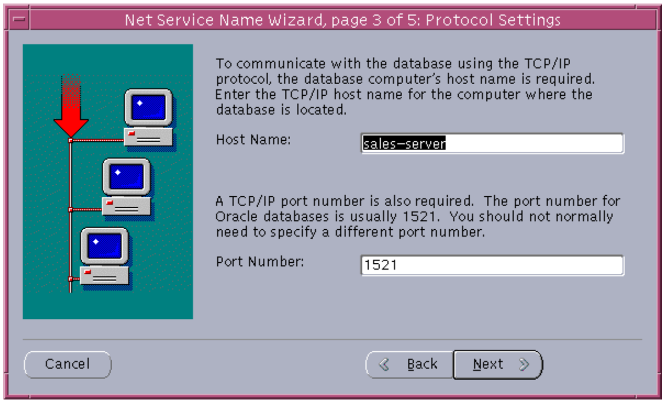


Net Service Name Wizard, page 2 of 5: Protocol

To communicate with the database across a network, a network protocol is used. Select the protocol used for the database you want to access.

TCP/IP (Internet Protocol)
TCP/IP with SSL (Secure Internet Protocol)
SPX (NetWare Networking)
IPC (Local Database)

6. Enter the appropriate protocol parameters for the chosen protocol in the fields provided, then click Next:



Net Service Name Wizard, page 3 of 5: Protocol Settings

To communicate with the database using the TCP/IP protocol, the database computer's host name is required. Enter the TCP/IP host name for the computer where the database is located.

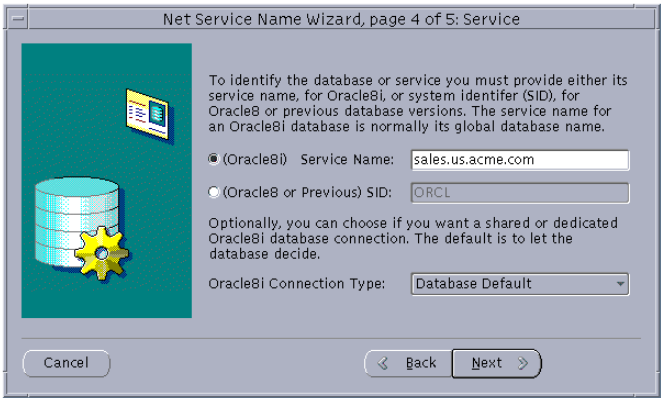
Host Name: sales-server

A TCP/IP port number is also required. The port number for Oracle databases is usually 1521. You should not normally need to specify a different port number.

Port Number: 1521

See Also: Appendix B for protocol parameter settings

7. Select a release, enter a destination service, then click Next:



Net Service Name Wizard, page 4 of 5: Service

To identify the database or service you must provide either its service name, for Oracle8i, or system identifer (SID), for Oracle8 or previous database versions. The service name for an Oracle8i database is normally its global database name.

● (Oracle8i) Service Name: sales.us.acme.com

○ (Oracle8 or Previous) SID: ORCL

Optionally, you can choose if you want a shared or dedicated Oracle8i database connection. The default is to let the database decide.

Oracle8i Connection Type: Database Default

If the destination service is an Oracle release 8.1 database, click (Oracle8i), and enter a service name in the Service Name field. If destination service is Oracle release 8.0 or version 7, click (Oracle8 or Previous), and enter an Oracle System Identifier (SID) in the Database SID field.

See Also: "Understanding Connect Descriptors" on page 6-2 for further information about the service name string to use

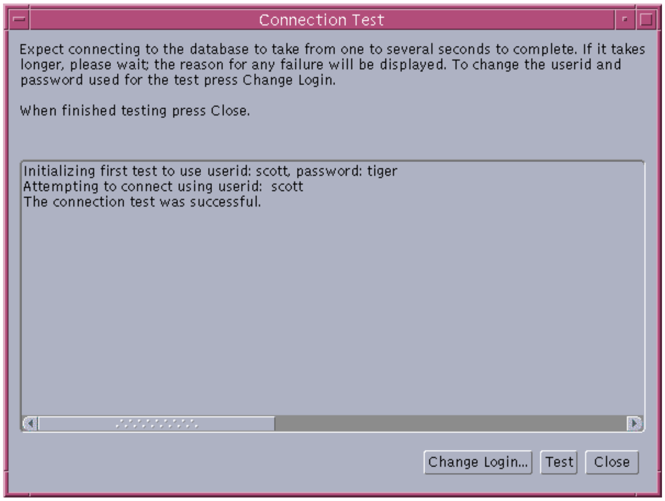
Optionally, you can select a database connection type from the Oracle8i Connection Type list for this net service name. Oracle Corporation recommends you use the default setting of Database Default. If **multi-threaded server (MTS)** is configured in the initialization parameter file, you can select Dedicated Server to force the listener to spawn a dedicated server, bypassing MTS configuration. If MTS is configured in the initialization parameter file and you want to guarantee the connection always uses MTS, select Shared Server.

> **See Also:** Chapter 9 for further information about MTS configuration

8. Click Test to verify that the net service name works, then click Next.

Testing assumes the database and listener are running. If they are not, see "Using Net8 Control Utilities" on page 11-3 to start components, or click Next to disregard testing and continue to Step 10.

A successful test results in "`The connect test was successful.`" message in the Connect Test dialog box:



Expect connecting to the database to take from one to several seconds to complete. If it takes longer, please wait; the reason for any failure will be displayed. To change the userid and password used for the test press Change Login.

When finished testing press Close.

```
Initializing first test to use userid: scott, password: tiger
Attempting to connect using userid:  scott
The connection test was successful.
```

If the test was not successful:

- Ensure that the database and listener are running, then click Test.
- Click Change Login to change the user name and password for the connection, then click Test.

9. Click Close to dismiss the Connect Test dialog box.
10. Click Finish to save your configuration and dismiss Net Service Name Wizard.

See Also:

- "Creating a List of Listener Protocol Addresses" on page 8-2 to configure multiple protocol addresses
- "Configuring Advanced Connect Data Parameters" on page 8-8 to configure additional connect data options

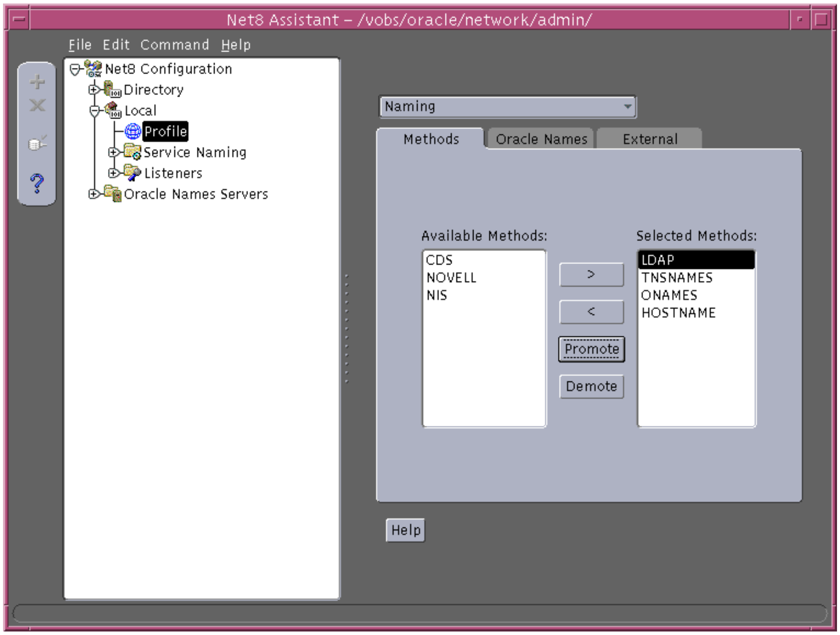
Task 3: Configure LDAP as the First Naming Method

After database services or net service names are configured as entries in the directory, set 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 Net8 can use to resolve connect identifiers to connect descriptors.

To specify local naming as the first naming method:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Profile.
3. From the list in the right pane, select Naming.
4. Click the Methods tab.
5. Select LDAP from the Available Methods list, then click the right-arrow button.

6. Select LDAP in the Selected Methods list, 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, onames, hostname)
   ```

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 7 for listener configuration details

Task 5: Connect to the Database

Clients that are configured with an administrative context that matches the administrative context for the entry the client is looking up can connect to the database using the following syntax:

```
CONNECT username/password@connect_identifier
```

Clients that are configured with an administrative context that does not match the entry's administrative context cannot use the connect identifier in the connect string. Instead, these connections require the entry's distinguished name or its absolute name.

See Also:

- "Client Connections Using Directory Naming" on page 2-44 for absolute name usage
- "Connect Identifier Syntax Characteristics" on page 11-10 for connect identifier syntax rules

Modifying Connectivity Information for Database Service Entries

> **Note:** Only users that are members of the OracleNetAdmins group are allowed to modify Net8 attributes for database service entries. The authenticated user that created the Oracle Context during directory access on the server is automatically a member of this group. To add or remove users from this group, see "Adding Users to and Removing Users from the OracleNetAdmins Group" on page 6-31.

A database service entry is created by Oracle Database Configuration Assistant during database creation after directory access has been configured on the server. The entry is contained under an Oracle Context (`cn=OracleContext`).

A database service entry stored in a directory may not contain any network route information. A network route that includes the location of the listener through a protocol address must be included in the connect descriptor. Otherwise, the client has no way of locating the listener.

Note: Database services stored in an Oracle Names server can be exported into a directory, as described in "Exporting Net Service Names from a tnsnames.ora File" on page 6-32 and "Exporting Network Objects from an Oracle Names Server" on page 6-35.

To create or modify network route information for a database service:

1. Start Net8 Assistant on a machine that has been configured with directory access for a server:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Directory > Service Naming.
3. Select the database service. The right pane displays the current destination service name.
4. In the Address Configuration box, click “+”. A new address tab appears.
5. Select a protocol and enter the appropriate protocol parameter information for the selected protocol.

   See Also: Appendix B for protocol parameter settings

6. In the right pane, click Apply.

Adding Users to and Removing Users from the OracleNetAdmins Group

The user that creates the Oracle Context is a member of the OracleNetAdmins (`cn=OracleNetAdmins,cn=OracleContext`) group. Using directory tools, such as ldapmodify, a directory administrator or the directory user who created the Oracle Context can add users to this group.

To add a user to the OracleNetAdmins group with ldapmodify:

1. Create a **LDAP Data Interchange Format (LDIF)** file that specifies that you want to add a user to the OracleNetAdmins group. You can use the following sample LDIF file with your own settings for the **Distinguished Name (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. Use the following ldapmodify syntax to add the user:

```
ldapmodify -h host -p port -D binddn
-w password -file ldif_file
```

Argument	Description
`-h` *`host`*	Specifies the directory server host
`-p` *`port`*	Specifies the listening TCP/IP port for the directory server. If you do not specify this option, the default port (389) is used.
`-D` *`binddn`*	Specifies the directory administrator or user DN.
`-w` *`password`*	Specifies the password for the directory administrator or directory user.
`-f` *`ldif_file`*	Specifies the input file name

To remove a user from the OracleNetAdmins group with ldapmodify:

1. Create a LDIF file that specifies that you want to add a user to the OracleNetAdmins group. You can use the following sample LDIF file with your own settings for the distinguished name (DN) for `cn=OracleNetAdmins` and the user that you want to add.

```
dn: cn=OracleNetAdmins,cn=OracleContext,...
changetype: modify
delete: uniquemember
uniquemember: <DN of user being delete from group>
```

2. Use the following ldapmodify syntax to add the user:

```
ldapmodify -h host -p port -D binddn
-w password -file ldif_file
```

Exporting Net Service Names from a tnsnames.ora File

If a `tnsnames.ora` file already exists, its net service names can be exported to a directory server. The export is performed for one domain at a time.

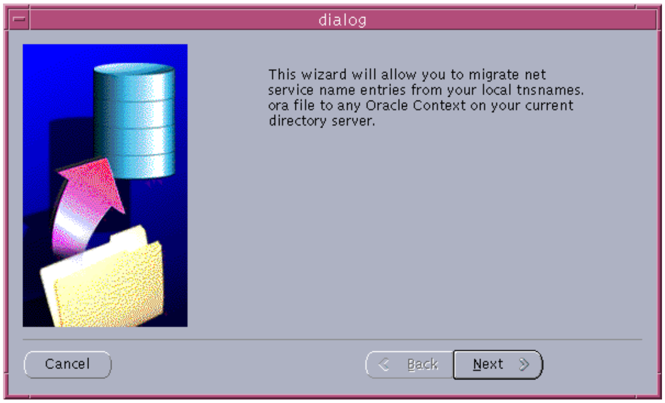
Exporting a `tnsnames.ora` file creates new net service name entries in the currently selected **Oracle Context** (`cn=OracleContext`). If you need to change the Oracle Context, choose Command > Directory Server > Change Oracle Context in Net8 Assistant.

The computer where you intend to perform the export of data must be configured for server directory access.

> **See Also:** "Task 1: Configure Directory Access on Server and Clients" on page 6-17

To export net service names contained in a `tnsnames.ora` file to a directory:

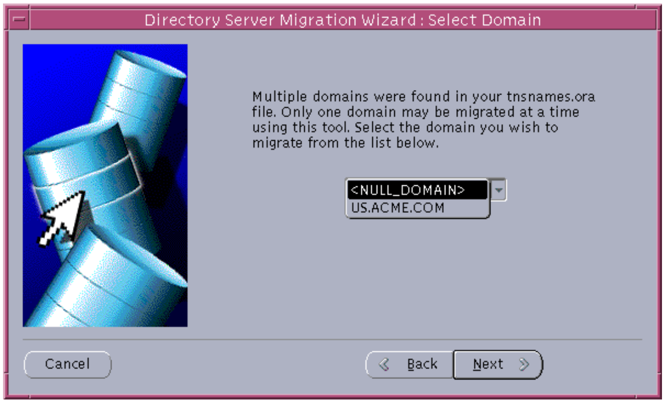
1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. Choose Command > Directory > Import Net Service Names. The Directory Server Migration Wizard starts.



This wizard will allow you to migrate net service name entries from your local tnsnames.ora file to any Oracle Context on your current directory server.

3. Click Next.

If multiple domains were detected, the Select Domain page appears:



Directory Server Migration Wizard : Select Domain

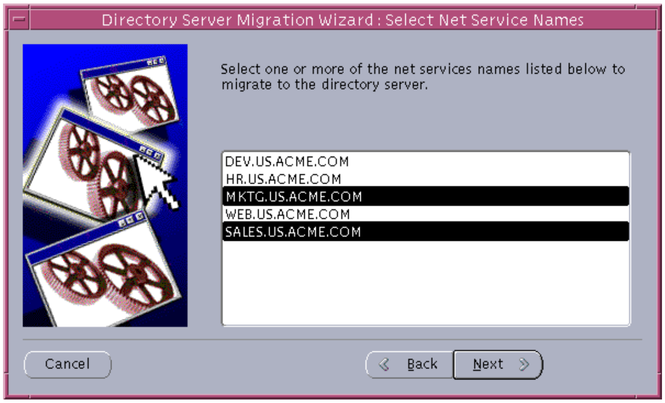
Multiple domains were found in your tnsnames.ora file. Only one domain may be migrated at a time using this tool. Select the domain you wish to migrate from the list below.

<NULL_DOMAIN>
US.ACME.COM

If the net service names are not domain qualified, the Select Net Service Names page appears, as shown in Step 4.

4. From the list, select the network domain whose net service names you want to export.

   The Select Net Service Names page appears:



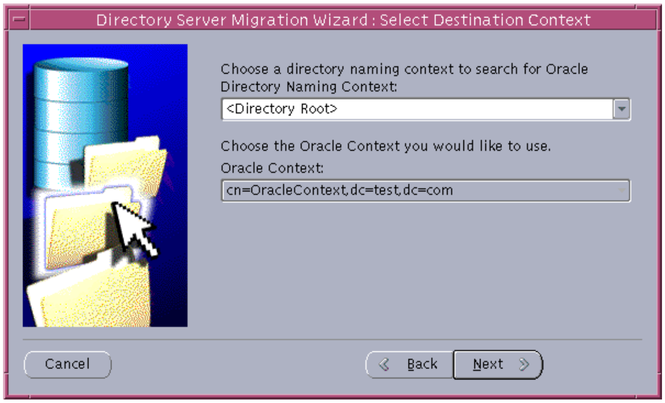
Directory Server Migration Wizard : Select Net Service Names

Select one or more of the net services names listed below to migrate to the directory server.

DEV.US.ACME.COM
HR.US.ACME.COM
MKTG.US.ACME.COM
WEB.US.ACME.COM
SALES.US.ACME.COM

5. Select the net service names from the list to export, then click Next.

The Select Destination Context page appears:


Directory Server Migration Wizard : Select Destination Context

Choose a directory naming context to search for Oracle

Directory Naming Context: <Directory Root>

Choose the Oracle Context you would like to use.

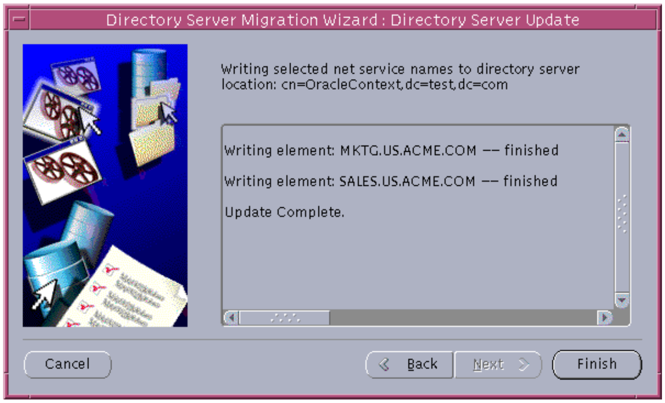
Oracle Context: cn=OracleContext,dc=test,dc=com

6. Select the **directory naming context** that contains the Oracle Context from the Directory Naming list. The directory naming context is part of a directory subtree that contains one or more Oracle Contexts.

7. Select the Oracle Context to which you want to export the selected net service names from the Oracle Context list.

8. Click Next.

   The wizard exports the net service names:



Writing selected net service names to directory server location: cn=OracleContext,dc=test,dc=com

```
Writing element: MKTG.US.ACME.COM -- finished

Writing element: SALES.US.ACME.COM -- finished

Update Complete.
```

9. Click Finish to dismiss the Directory Server Migration Wizard.

Exporting Network Objects from an Oracle Names Server

Database services and net service names stored in an Oracle Names server can be exported to a directory or to an **LDAP Data Interchange Format (LDIF)** file. Data is exported from a specified domain. If the domain has authority for subdomains, their data can also be exported.

> **Prerequisite:** The computer where you intend to perform the export of data must be configured for server directory access, as described in "Task 1: Configure Directory Access on Server and Clients" on page 6-17.

The steps and examples to perform an export of data are explained in the following sections:

- Task 1: Create Structure in Directory
- Task 2: Obtain List of Objects to Export
- Task 3: Export Objects To a Directory Server
- Exporting Domains To a Similar DIT
- Exporting Domains To a Dissimilar DIT

Task 1: Create Structure in Directory

Configure the directory with **Directory Information Tree (DIT)** for the Oracle Names objects. You can replicate the domain structure you currently use with Oracle Names, or you can develop an entirely different structure.

Oracle Corporation recommends creating an Oracle Context (`cn=OracleContext`) to store Net8 objects.

> **See Also:** Chapter 2 for further information about the Oracle Context

Task 2: Obtain List of Objects to Export

Determine the Oracle Names domain structure and the objects within that structure. NAMESCTL offers three commands to help you with this task:

Command	Description
LIST_DOMAINS	Lists all the authoritative domains. **See Also:** LIST_DOMAINS on page A-37
LIST_DELEGATED	Lists all the delegated domains. **See Also:** LIST_DELEGATED on page A-37
LIST_OBJECTS	Lists all the authoritative network objects. **See Also:** LIST_OBJECTS on page A-38

Task 3: Export Objects To a Directory Server

The NAMESCTL utility exports network objects into the directory with the DUMP_LDAP command. This command enables you to export network objects to an LDIF file or directly into a directory.

The syntax to export data to a LDIF file is as follows:

```
NAMESCTL
NAMESCTL> dump_ldap [source] [destination] [options] {-f filename}
```

Note: The generated LDIF file can later be loaded into the directory with the `ldap_modify` command. See your directory's documentation for syntax usage.

The syntax to export data directly to a directory is as follows:

```
NAMESCTL
NAMESCTL> dump_ldap [source] [destination] [options] {-h host} {-p
port} {-D user_dn} {-w password}
```

See Also: DUMP_LDAP on page A-30 for a description of the arguments.

Exporting Domains To a Similar DIT

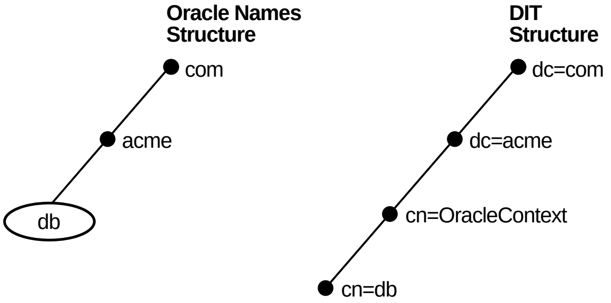
If your directory's DIT structure has been designed with domain components (dc) to match the current Oracle Names structure, review the following examples to understand how to export data:

- Example 1: Exporting Data from a Single Domain to the Same DIT Node
- Example 2: Exporting Data from a Domain Tree to a Similar DIT

Example 1: Exporting Data from a Single Domain to the Same DIT Node

Figure 6–1 shows an Oracle Names domain structure of acme.com. It contains a database service called db. The directory DIT has been designed with domain components that match the Oracle Names structure. With this DIT structure, db can be exported to cn=OracleContext,dc=acme,dc=com:

Figure 6–1 Single Domain Export to the Same DIT Node



Either of the following syntax can be used to export data from Oracle Names to the configured DIT structure:

```
NAMESCTL> dump_ldap acme.com (dn:cn=OracleContext,dc=acme,dc=com) -f sample.ldif
NAMESCTL> dump_ldap acme.com -x -f sample.ldif
```

In the first line of syntax, the destination DN is explicitly specified. It is not necessary to specify the destination DN, as shown in the in the second line of syntax, because the destination DN matches the domain model used in Oracle Names. The -x option is used to pre-append cn=OracleContext to the left of the destination DN. In other words, you can use -x instead of specifying cn=OracleContext in the DN. This way, all objects can be created under a cn=OracleContext RDN in the directory.

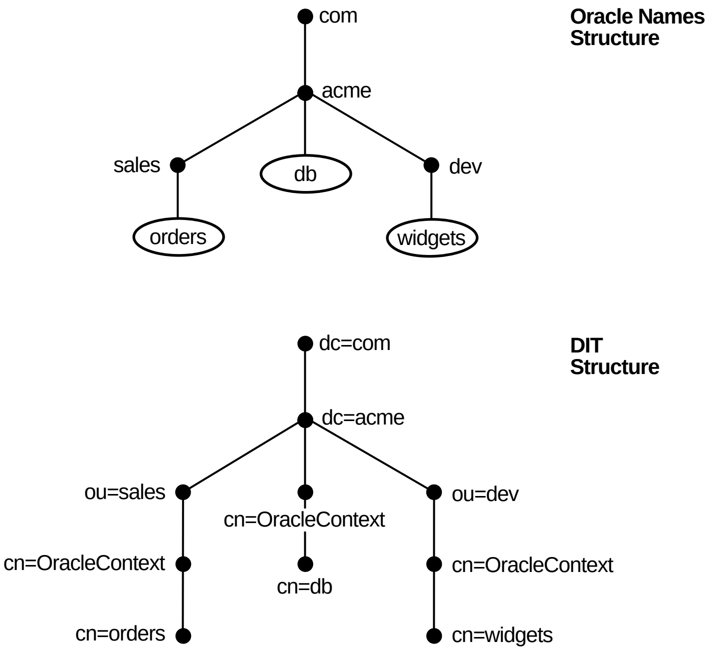
The database service db is exported to cn=OracleContext,dc=acme,dc=com and has a DN location of (dn:cn=db,cn=OracleContext,dc=acme,dc=com).

Example 2: Exporting Data from a Domain Tree to a Similar DIT

Data can be exported from a root domain and its subdomains in Oracle Names to a directory that uses a similar DIT.

Figure 6–2 shows an Oracle Names structure that contains a root domain of `acme.com` and subdomains of `sales.acme.com` and `dev.acme.com`. Database services of `db`, `orders`, and `widgets` reside in `acme.com`, `sales.acme.com` and `dev.acme.com`, respectively. The directory DIT is similar to the Oracle Names structure.

Figure 6–2 Root and Subdomain Domain Export to Similar DIT



The following syntax is used to export data from the acme.com root domain and its subdomains to the configured DIT structure:

```
dump_ldap acme.com -R -x -f sample.ldif
```

The following table shows how database objects in acme.com, sales.acme.com, and dev.acme.com are mapped to DNs in the directory. Because -x is used, all objects are created under cn=OracleContext RDNs in the directory.

Database Object in Oracle Names	New DN in Directory
db.acme.com	dn:cn=db,cn=OracleContext,dc=acme,dc=com
orders.sales.acme.com	dn:cn=orders,cn=OracleContext,dc=sales, dc=acme,dc=com
widgets.dev.acme.com	dn:cn=widgets,cn=OracleContext,dc=dev, dc=acme,dc=com

Exporting Domains To a Dissimilar DIT

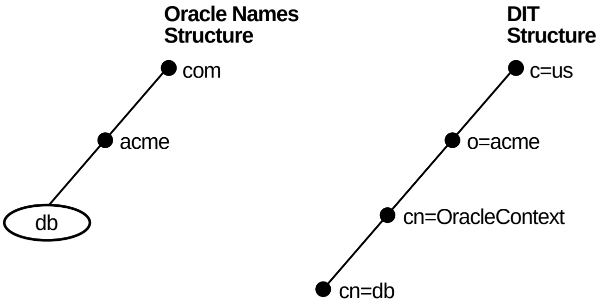
If your directory's DIT structure has been designed with a DIT that is dissimilar to the current Oracle Names domain structure, review the following examples to understand how to export data:

- Example 1: Exporting Data from a Single Domain to a Non-DC Node
- Example 2: Exporting Data from a Domain Tree to a Non-DC DIT
- Example 3: Exporting Data from Multiple Domains to One DIT Node
- Example 4: Reorganizing a Tree Structure During an Export

Example 1: Exporting Data from a Single Domain to a Non-DC Node

Figure 6–3 shows an Oracle Names domain structure of `acme.com`. It contains a database service called `db`. The directory has been designed with a DIT of `cn=OracleContext,o=acme,c=us`.

Figure 6–3 Single Domain Export to One DIT Node



The following syntax can be used to export data from the `acme.com` domain to the configured DIT structure:

```
dump_ldap acme.com (dn:c=us,o-acme) -x -f sample.ldif
```

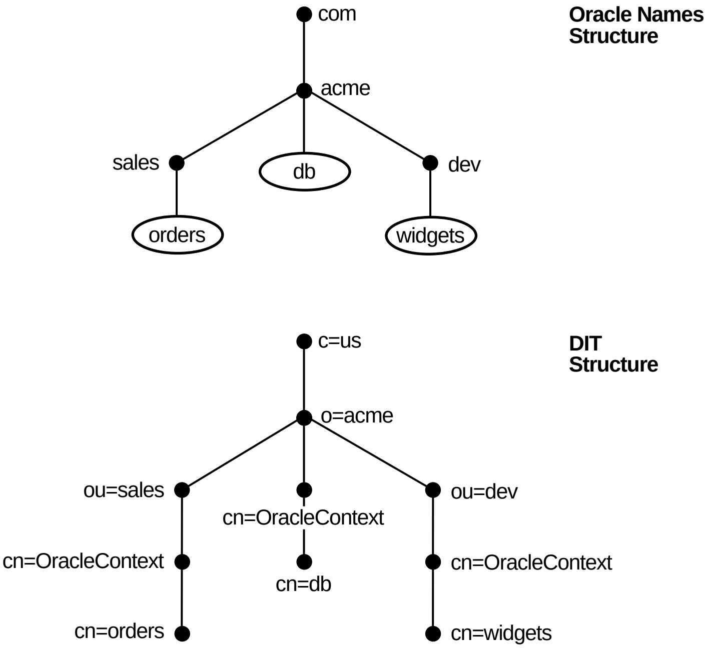
Because the DIT is different from the Oracle Names structure, the destination DN must be explicitly specified. The -x option is used to pre-append `cn=OracleContext` to the left of the destination DN. In other words, you can use -x instead of specifying `cn=OracleContext` in the DN. This way, objects can be created under a `cn=OracleContext` RDN in the directory.

Example 2: Exporting Data from a Domain Tree to a Non-DC DIT

Data can be exported from a root domain and its subdomains in Oracle Names to a directory that uses a dissimilar DIT. RDNs in the destination DN must be wildcarded—that is, specified without a value—for the subdomains.

Figure 6–4 shows an Oracle Names structure that contains a root domain of `acme.com` and subdomains of `sales.acme.com` and `dev.acme.com`. Database services of `db`, `orders`, and `widgets` reside in `acme.com`, `sales.acme.com` and `dev.acme.com`, respectively. The directory DIT has a top-level structure of `o=acme,c=us` that correlates to the `acme.com` in Oracle Names. The subtrees, `ou=sales` and `ou=dev`, correlate to the `sales.acme.com` and `dev.acme.com` subdomains in Oracle Names.

Figure 6–4 Root and Subdomain Domain Export to Dissimilar DIT



The following syntax is used to export data from the acme.com root domain and its subdomains to the configured DIT structure:

```
dump_ldap acme.com (dn:ou,o=acme,c=us) -R -x -f sample.ldif
```

Note that organizationalUnitName (ou) is wildcarded—that is, it contains no value—so that the sales and dev subdomain of acme.com in the source region can be mapped to an ou.

The following table shows how database objects in acme.com, sales.acme.com, and dev.acme.com are mapped to DNs in the directory. Because -x is used, all objects are created under cn=OracleContext RDNs in the directory.

Database Object in Oracle Names	New DN in Directory
db.acme.com	dn:cn=db,cn=OracleContext,o=acme,c=us
orders.sales.acme.com	dn:cn=orders,cn=OracleContext,ou=sales, o=acme,c=us
widgets.dev.acme.com	dn:cn=widgets,cn=OracleContext,ou=dev, o=acme,c=us

If acme.com contained a subdomain of mktg.dept.acme.com, that subdomain's network objects would not exported. This is because the destination DN (dn:ou,o=acme,c=us) only permits one-level subdomains. In order to export objects from mktg.dept.acme.com, the following syntax would be required:

```
dump_ldap acme.com (dn:ou,ou,o=acme,c=us) -x -f sample.ldif
```

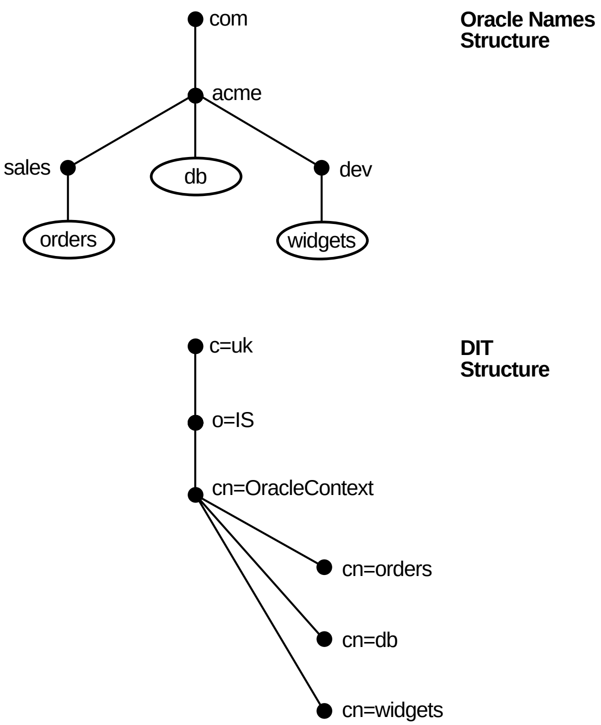
This enables up to two levels of subdomains to be exported. By adding additional wildcards, you can specify any level of depth.

Example 3: Exporting Data from Multiple Domains to One DIT Node

Data can be exported from multiple domains to one node in the destination DIT.

Figure 6–5 shows an Oracle Names structure that contains a root domain of `acme.com` and subdomains of `sales.acme.com` and `dev.acme.com`. Database services of `db`, `orders`, and `widgets` reside in `acme.com`, `sales.acme.com` and `dev.acme.com`, respectively. The directory DIT has a structure of `o=IS,c=uk` that contains no subtrees that correlate to the Oracle Names subdomains.

Figure 6–5 Root and Subdomain Domain Export to One DIT Node



All data can be exported from the root domain and the subdomains to `cn=OracleContext,o=IS,c=uk` in the DIT with the following syntax:

```
dump_ldap acme.com (dn:o=IS,c=uk) -R -x -f sample.ldif
```

The following table shows how database objects in acme.com, sales.acme.com, and dev.acme.com are mapped to DNs in the directory. Because -x is used, all objects are created under cn=OracleContext RDNs in the directory.

Database Object in Oracle Names	New DN in Directory
db.acme.com	dn:cn=db,cn=OracleContext,o=IS,c=uk
orders.sales.acme.com	dn:cn=orders,cn=OracleContext,o=IS,c=uk
widgets.dev.acme.com	dn:cn=widgets,cn=OracleContext,o=IS,c=uk

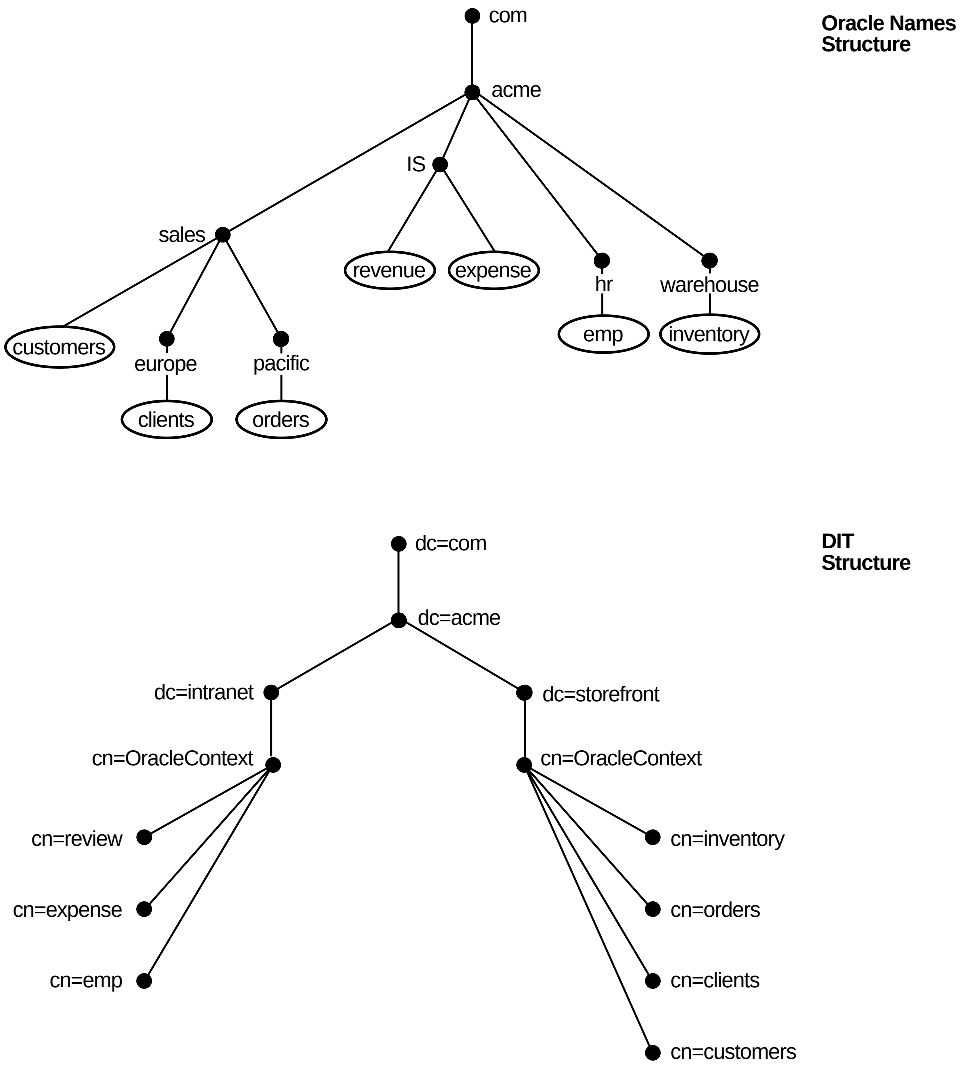
If one of the subdomain contained a database object named db, it would not be exported, because its name would collide with the db object exported from db.acme.com.

Example 4: Reorganizing a Tree Structure During an Export

In the previous examples, you saw how data can be exported to a non-DC DIT and how data can be exported from multiple domains to a one node in the DIT. This example combines these two types of exports to demonstrate how to export data to DIT with a very different structure.

Figure 6–6 on page 6-45 shows an Oracle Names structure that contains a root domain of acme.com and four subdomains, each of which contains at least one database service object. The directory DIT has a top-level structure of dc=acme,dc=com that correlates to the acme.com domain in Oracle Names. The two subtrees, dc=intranet and dc=storefront, are unrelated to the subdomains in Oracle Names.

Figure 6–6 Multiple Domain Export to Two Subtrees



In order to export data from the Oracle Names structure to the DIT, each domain must be exported separately:

```
dump_ldap IS.acme.com (dn:dc=intranet,dc=com,dc=acme) -x -f sample.ldif
dump_ldap hr.acme.com (dn:dc=intranet,dc=com,dc=acme) -x -f sample.ldif
dump_ldap warehouse.acme.com (dn:dc=storefront,dc=com,dc=acme) -x -f sample.ldif
dump_ldap sales.acme.com (dn:dc=storefront,dc=com,dc=acme) -R -x -f sample.ldif
```

The first two dump_ldap commands export database objects to cn=Oraclecontext,dc=intranet,dc=acme,dc=com; the last two dump_ldap commands export database objects to cn=Oraclecontext,dc=storefront,dc=acme,dc=com. -R in the dump_ldap sales.acme.com command enables the database service objects to be exported from sales.acme.com, europe.sales.acme.com, and pacific.sales.acme.com.

The following table shows how database objects in the Oracle Names domains are mapped to DNs in the directory. Because -x is used, all objects are created under cn=OracleContext RDNs in the directory.

Database Object in Oracle Names	New DN in Directory
revenue.IS.acme.com	dn:cn=revenue,cn=OracleContext,dc=intra,dc=acme,dc=com
expense.IS.acme.com	dn:cn=expense,cn=OracleContext,dc=intra,dc=acme,dc=com
emp.hr.acme.com	dn:cn=emp,cn=OracleContext,dc=intra,dc=acme,dc=com
inventory.warehouse.com	dn:cn=inventory,cn=OracleContext,dc=storefront,dc=acme,dc=com
customer.sales.acme.com	dn:cn=customer,cn=OracleContext,dc=storefront,dc=acme,dc=com
clients.europe.sales.acme.com	dn:cn=clients,cn=OracleContext,dc=storefront,dc=acme,dc=com
orders.pacific.sales.acme.com	dn:cn=orders,cn=OracleContext,dc=storefront,dc=acme,dc=com

Configuring the Oracle Names Method

Note: This section describes configuration from a release 8.1 perspective. If you have an existing release 8.0 or release 7.*x* configuration, see *Oracle8i Migration*.

Oracle Names simplifies the setup and administration of global, client/server computing networks. Oracle Names makes network address and database link information available to all nodes throughout the network. Each database server's network address is identified with a simple service name. Client applications then can request a database connection with that name rather than a lengthy address. Oracle Names shields users and applications from changes made to the network infrastructure. It provides for centralized administration of network service names.

Configuring Oracle Names involves the following tasks:

Task 1: Consider Oracle Names Options

Task 2: Install Necessary Components

Task 3: Create an Oracle Names Server

Task 4: Configure Clients and Database Servers To Use Oracle Names Servers

Task 5: Configure the Listener

Task 6: Register Data with the Oracle Names Server

Task 7: (Optional) Delegate Domains to Delegated Administrative Regions

Task 8: Connect to the Database

Task 1: Consider Oracle Names Options

Prior to creating an Oracle Names server, you must decide whether:

- You need service registration data replicated continuously among Oracle Names servers, or want all Oracle Names servers within a region to store their registration data in an Oracle database.
- You need support for one or more **administrative region**. An administrative region is a collection of Oracle Names servers in one or more **domains**, a grouping of network objects, such as databases. Networks with multiple administrative regions must have one **root administrative region** and one or

more **delegated administrative regions**. Delegated administrative regions contain the domains and Oracle Names server addresses in any alternate regions which act as direct child regions of the root.

> **See Also:** "Oracle Names" on page 4-10

Task 2: Install Necessary Components

Ensure that the following are installed:

- Oracle Names on its own machine that is designated as the Oracle Names server
- Net8 Client or SQL*Net Client on the clients
- Net8 Server or SQL*Net Server on the server

> **Important:** The Oracle Universal Installer does not install the `$ORACLE_HOME/network/names` directory on UNIX platforms. This directory is necessary for successful configuration. If it does not exist, manually create it.

Task 3: Create an Oracle Names Server

How an Oracle Names server is created depends upon how you want region data stored. You can have the data:

- Stored in tables, called a region database, in an Oracle database accessible to Oracle Names servers

  When information is added to an Oracle Names server, the information is stored in the database and in the Oracle Names server's cache files. If the database is unavailable, the information in the cache files is used.

- Replicated among Oracle Names servers and checkpointed to disk

The checkpoint files are stored in `$ORACLE_HOME/network/names` on UNIX platforms and *`ORACLE_HOME`*`\network\names` on Windows NT:

File	Description
`ckcfg.ora`	A backup copy of the configuration parameters stored in the ONRS_CONFIG table in the region database
`ckpcch.ora`	Contains all current non-authoritative data which has been retrieved and cached from remote regions and has not expired yet
`ckpreg.ora`	Contains all authoritative data for the region. If the Oracle Names server uses a region database, then this checkpoint file is a copy of the region data in the tables as of the last reload. This data is used when the Oracle Names server starts if the databases is inaccessible. If the server is not using the database, this file is its only persistent storage and is loaded by the Oracle Names server at startup. The Oracle Names server considers the data in the region checkpoint file to be current if there are no other Oracle Names servers in the region. In this case, the file is kept current as of the last update to the region.

The following sections cover both modes, as well as a default Oracle Names server that requires no configuration:

- Default Oracle Names Server
- Create Tables in a Database
- Replicate Data in Checkpointed Files

Whichever method you choose, Oracle Corporation recommends you create more than one Oracle Names server for the network, in case one should go down.

Default Oracle Names Server

An Oracle Names server can run without any configuration. Its name defaults to ONAMES_*host* if its name is configured in the `names.ora` file. The listening address defaults to TCP/IP, port 1575 on the local host, or other well-known addresses for protocols.

> **See Also:** Your Oracle operating-system documentation for default address information

If the NAMES.DOMAINS parameter is not configured in the `names.ora` file, the Oracle Names server assumes authority for the root domain.

If you would like to use this Oracle Names server, proceed to "Task 2: Discover Oracle Names Servers" on page 6-59.

Create Tables in a Database

To store service data in an Oracle database, perform the following tasks:

- Task 1: Configure the Database Server
- Task 2: Create Oracle Names Servers
- Task 3: Start the Oracle Names Server

Task 1: Configure the Database Server To configure the database server:

1. Start the database if it is not currently running. Otherwise, go to Step 2.

   a. Connect to the database as INTERNAL:

   ```
   SQL> CONNECT internal/password
   ```

   where *password* is ORACLE for the INTERNAL user account by default.

   b. Start the database:

   ```
   SQL> STARTUP;
   ```

2. Connect to the database as the SYSTEM user:

   ```
   SQL> CONNECT system/password
   ```

   where *password* is MANAGER for the SYSTEM user account by default.

3. Create a user account and password:

```
SQL> CREATE user user
  identified by password
  default tablespace users
  temporary tablespace temp;
```

4. Run the `namesini.sql` script located in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin\names` on Windows NT platforms. This script creates the tables needed by Oracle Names to store information. Optionally, run `namesupg.sql` to upgrade old tables.

```
SQL> CONNECT user/password
SQL> @oracle_home/network/admin/namesini.sql;
```

Task 2: Create Oracle Names Servers For each machine where Oracle Names is installed and where you want an Oracle Names server, create an Oracle Names server from Net8 Assistant.

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Oracle Names Servers.
3. Choose Edit > Create, or click the "+" button to create a new Oracle Names server.

   The Names Wizard starts.

   The wizard guides you through the creation and configuration process, prompting for:

   - A unique Oracle Name Server name
   - A protocol address for the Oracle Names server

     If you choose TCP/IP, Oracle Corporation recommends using the default and officially registered port of 1575 for the Oracle Names server.
   - A choice to store information in a database or replicate information among Oracle Names servers

     (Click "Use a region database".)

- A protocol address for a database's listener

  If you choose TCP/IP, Oracle Corporation recommends using the default and officially registered port of 1521 for the listener.

- Database user ID, password, and service name or SID
- Whether or not this Oracle Names server is in the root administrative region

If you specify that this Oracle Names server is in the root administrative region, configuration completes.

If you specify that this Oracle Names server is not in the root administrative region, it assumed this Oracle Names server is in a delegated administrative region. The wizard then prompts you for the local administrative region's domain name and the address of an Oracle Names server in the root administrative region before completing.

When the wizard completes, the following message appears:

```
A Names Server, onames_server, has been created with
default settings. Use the Configure Server section to
modify the default configuration.
```

4. Choose File > Save Network Configuration.
5. Repeat Steps 2-4 to create additional Oracle Names servers in a region. Net8 Assistant does not support creation of multiple Oracle Names servers on one machine.

Net8 Assistant creates a `names.ora` file with at least the following settings:

- Oracle Names server name (NAMES.SERVER_NAME)
- Oracle Names server listening protocol address (NAMES.ADDRESSES)
- Database information (NAMES.ADMIN_REGION)

A `names.ora` file is shown next with annotations of content:

```
#Oracle Names server name. The name should include the name of the domain
this Oracle Names server is in.
names.server_name=namesvr2.com

#Oracle Names server listening protocol address
names.addresses=
  (address=(protocol=tcp)(host=namesrv2-pc)(port=1575))
  [(address=...))]
```

```
#Database repository information
names.admin_region=
  (region=
    (description=
      (address=(protocol=tcp)(host=sales-server)(port-1521))
      (connect_data=
        (service_name=sales.us.acme.com))
    (userid=system)
    (password=password)
    (name=local_region)
    (refresh=86400)
    (retry=60)
    (expire=600))
#If an Oracle Names server is in a delegated administrative region, identify
the address of an Oracle Names server in the root administrative region.
names.domain_hints=
  (address=(protocol=tcp)(host=namesrv1-pc)(port=1575))


# Specify the domain controlled by this region and the time to live (TTL).
If this is the root administrative region, you must have NAME= (null) to
identify the root domain properly.
names.domains=
  (domain=
    (name=)
    (min_ttl=86400))
```

See Also: Appendix B for more information about protocol syntax needed for address information

Administering Multiple Domains

If you want the region to administer more than one domain, specify the additional domains in the NAMES.DOMAINS parameter with Net8 Assistant:

1. In the navigator pane, expand Oracle Names Servers.
2. Select the Oracle Names server.
3. From the list in the right pane, select Configure Server.
4. Click the Domains tab.
5. Enter the domain name in the Domain Name field and time-to-live information, then click Add.
6. Repeat Step 5 for each additional domain.
7. Choose File > Save Network Configuration.

In the following example, NAMES.DOMAINS contains a listing for the root, com, oracle.com, and hq.oracle.com domains. All the domain precedent to hq.oracle.com must be defined in order to define hq.oracle.com.

```
names.domains=
 (domain_list=
  (domain=
    (name=)
    (min_ttl=86400))
  (domain=
    (name=com)
    (min_ttl=86400))
  (domain=
    (name=oracle.com)
    (min_ttl=86400))
  (domain=
    (name=hq.oracle.com)
    (min_ttl=86400)))
```

Task 3: Start the Oracle Names Server Start the Oracle Names server using either Net8 Assistant or the NAMESCTL control utility:

Use Net8 Assistant...	Use the NAMESCTL control utility...

1. In the navigator pane, expand Oracle Names Servers.
2. Select the Oracle Names server.
3. From the list in the right pane, select Manage Server.
4. Click the Control tab.
5. Click Start.
6. Click Apply.
The following message appears:
`Server started successfully.`
On Windows NT, a service called Oracle*ORACLE_HOME*Names*names_server* is created.
	From the command line, enter:
	`NAMESCTL`
	If the following error messages appear, ignore them.
	`NNL-00024: warning: no preferred names servers in SQLNET.ORA`
	`NNL-00018: warning: could not contact default name server`
	`NAMESCTL> start`
	The START command of NAMESCTL loads the Oracle Names server into memory and tells it to begin executing. At startup, the Oracle Names server loads its configuration and data.
	On Windows NT, a service called Oracle*ORACLE_HOME*Names*names_server*Service is created

Replicate Data in Checkpointed Files

To replicate data among Oracle Names server in checkpointed files, perform these tasks:

- Task 1: Create an Oracle Names Server
- Task 2: Discover Oracle Names Servers

Task 1: Create an Oracle Names Server For each machine where Oracle Names is installed and where you want an Oracle Names server, create an Oracle Names server from Net8 Assistant:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Oracle Names Servers.
3. Choose Edit > Create, or click the "+" button to create a new Oracle Names server.

   The Names Wizard starts.

   The wizard guides you through the creation and configuration process, prompting you for:

   - A unique Oracle Name Server name
   - A listening protocol address for the Oracle Names server

     If you choose TCP/IP, Oracle Corporation recommends using the default and officially registered port of 1575 for the Oracle Names server.
   - A choice to store information in a database or replicate information among Oracle Names servers

     (Click "Don't use a region database".)
   - To identify if this is the first Oracle Names server in the region

     If this is not the first Oracle Names server in the region, the wizard then prompts you to discover the other Oracle Names servers or to specify the address of another Oracle Names server in the region.
   - To identify if this Oracle Names server is in the root administrative region

   If you specify that this Oracle Names server is in the root administrative region, configuration completes.

If you specify that this Oracle Names server is not in the root administrative region, it assumed this Oracle Names server is in a delegated administrative region. The wizard then prompts you for the local administrative region's domain name and the address of an Oracle Names server in the root administrative region before completing.

When the wizard completes, the following message appears:

```
A Names Server, onames_server, has been created with
default settings. Use the Configure Server section to
modify the default configuration.
```

4. Select Save Network Configuration from the File menu.
5. Repeat Steps 2-4 to create additional Oracle Names servers in a region. Net8 Assistant does not support creation of multiple Oracle Names servers on one machine.

Net8 Assistant creates a `names.ora` file with at least the following settings:

- Oracle Names server name (NAMES.SERVER_NAME)
- Oracle Names listening protocol address (NAMES.ADDRESSES)

A `names.ora` file is shown next with annotations of content:

```
#Oracle Names server name. The name should include the name of the domain
this Oracle Names server is in.
names.server_name=namesvr2.com

#Oracle Names server listening protocol address
names.addresses=
  (address=(protocol=tcp)(host=namesrv2-pc)(port=1575))
  [(address=...))]


#If an Oracle Names server is in a delegated administrative region, identify
the address of an Oracle Names server in the root administrative region.
names.domain_hints=
  (address=(protocol=tcp)(host=namesrv1-pc)(port=1575))


# Specify the domain controlled by this region and the time to live (TTL).
If this is the root administrative region, you must have NAME= (null) to
identify the root domain properly.
names.domains=
  (domain=
    (name=)
    (min_ttl=86400))
```

> **See Also:** Appendix B for more information about protocol syntax needed for address information

Administering Multiple Domains

If you want the region to administer more than one domain, specify the additional domains in the NAMES.DOMAINS parameter with Net8 Assistant:

1. In the navigator pane, expand Oracle Names Servers.
2. Select the Oracle Names server.
3. From the list in the right pane, select Configure Server.
4. Click the Domains tab.
5. Enter the domain name in the Domain Name field and time-to-live information, then click Add.
6. Repeat Step 5 for each additional domain.
7. Choose File > Save Network Configuration.

In the following example, NAMES.DOMAINS contains a listing for the root, `com`, `oracle.com`, and `hq.oracle.com` domains. All the domain precedent to `hq.oracle.com` must be defined in order to define `hq.oracle.com`.

```
names.domains=
 (domain_list=
  (domain=
    (name=)
    (min_ttl=86400))
  (domain=
    (name=com)
    (min_ttl=86400))
  (domain=
    (name=oracle.com)
    (min_ttl=86400))
  (domain=
    (name=hq.oracle.com)
    (min_ttl=86400)))
```

Task 2: Discover Oracle Names Servers After all the Oracle Names servers are created, have each Oracle Names, except the first Oracle Names server in the region, discover the other Oracle Names servers in a region:

Use Net8 Assistant...

1. Start the first Oracle Names server in the region:

   a. In the navigator pane, expand Oracle Names Servers.

   b. Select the Oracle Names server.

   c. From the list in the right pane, select Manage Server.

   d. Click the Control tab.

   e. Click Server Operations, then click Start.

   f. Click Apply to start the Oracle Names server. The following message appears:

   Server started successfully.

2. From the second Oracle Names server, discover the first Oracle Names server.

   a. Choose Command > Discover Oracle Names servers. The following message appears:

   ```
   Discovered Oracle Names Server in the
   region. Please exit the tool and start
   again.
   ```

   The Discover command produces an `.sdns.ora` file in `$ORACLE_HOME/network/names` on UNIX or a `sdns.ora` file in *`ORACLE_HOME`*`\network\names` on Windows platforms. This file contains the names and addresses of the Oracle Names server(s). This file is read to find the addresses of Oracle Names servers.

   If an Oracle Names server does not respond, a dialog prompts you for another Oracle Names server address. If you know the network address of a particular Oracle Names server, enter it.

3. Start the second Oracle Names server, following Step 1.

4. For each Oracle Names server added to the region, repeat steps 2-3.

Use the NAMESCTL control utility...

1. Start the first Oracle Names server in the region:

   ```
   NAMESCTL
   NAMESCTL> start
   ```

2. From the second Oracle Names server, discover the first Oracle Names server.

   ```
   NAMESCTL
   NAMESCTL> reorder_ns
   ```

   The REORDER_NS command produces an `.sdns.ora` file in `$ORACLE_HOME/network/names` on UNIX or a `sdns.ora` file in *`ORACLE_HOME`*`\network\names` on Windows platforms. This file should contain the name and address of the first Oracle Names server.

   If an Oracle Names server cannot be found, and you know the network address of a particular Oracle Names server, enter the address at the prompt. For example:

   ```
   NAMESCTL> reorder_ns
   (address=(protocol=tcp)(host=mail-pc)(port
   =1575))
   ```

3. Start the second Oracle Names server in the region:

   ```
   NAMESCTL> start
   ```

4. Connect to the first Oracle Names server in the region, and query for the second Oracle Names server to make sure it has registered itself.

   ```
   NAMESCTL> query onames_server
   ```

5. For each Oracle Names server added to the region, repeat steps 2-4.

See Also: "Understanding Discovery" on page 4-24

Task 4: Configure Clients and Database Servers To Use Oracle Names Servers

To configure client and database server machines to use Oracle Names servers:

1. Create a list of Oracle Names servers to contact, using one of the following methods:

   - Create a static list of preferred Oracle Names server in the `sqlnet.ora` file, as described in "Configuring Preferred Oracle Names Servers" on page 8-26.
   - Discover Oracle Names servers by dynamically creating an `.sdns.ora` file in `$ORACLE_HOME\network\names` on UNIX an `sdns.ora` file in *`ORACLE_HOME`*`\network\names` on Windows platforms. This file contains the order of names and addresses of the Oracle Names server(s) to contact. Client machines read this file to find the addresses of Oracle Names servers to contact. Use either Net8 Assistant or the NAMESCTL utility.

   Preferred Oracle Names servers take precedence over the Oracle Names server configured in the `.sdns.ora` or `sdns.ora` file.

   > **See Also:** "Understanding Discovery" on page 4-24 for a description of discovery

To create an `.sdns.ora` file or an `sdns.ora` file:

Use Net8 Assistant...	Use the NAMESCTL control utility...

1. Start Net8 Assistant:
-On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
-On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. Choose Command > Discover Oracle Names servers.
The following message appears:
`Discovered Oracle Names Server in the region. Please exit the tool and start again.`
The Discover command produces `.sdns.ora` file in `$ORACLE_HOME/network/names` on UNIX or a `sdns.ora` file in *`ORACLE_HOME`*`\network\names` on Windows platforms. This file contains the names and addresses of the Oracle Names server(s). This file is read to find the addresses of Oracle Names servers.
If an Oracle Names server does not respond, a dialog prompts you for another Oracle Names server address. If you know the network address of a particular Oracle Names server, enter it.
	Discover the other Oracle Names servers:
	`NAMESCTL`
	`NAMESCTL> reorder_ns`
	The REORDER_NS command produces an `.sdns.ora` file in `$ORACLE_HOME/network/names` on UNIX or a `sdns.ora` file in *`ORACLE_HOME`*`\network\names` on Windows platforms. This file contains the names and addresses of the Oracle Names server(s). This file is read to find the addresses of Oracle Names servers.
	If an Oracle Names server cannot be found, and you know the network address of a particular Oracle Names server, enter the address at the prompt. For example:
	`NAMESCTL> reorder_ns`
	`(address=(protocol=tcp)(host=mail-pc)(port`
	`=1575))`

2. If you want Oracle Names to be the first method to reconcile net service names, set the NAMES.DIRECTORY_PATH parameter in the `sqlnet.ora` file in the following manner:

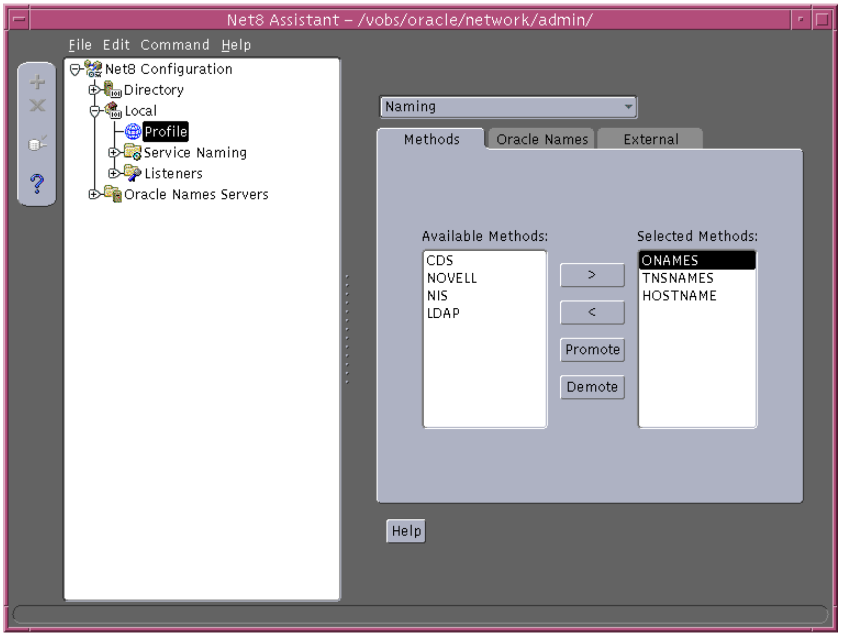
   a. In the navigator pane, expand Local > Profile.

   b. From the list in the right pane, select Naming.

   c. Click the Methods tab.

   d. Select ONAMES from the Available Methods list, then click the right-arrow.

   e. Select ONAMES in the Selected Methods list, then use the Promote button to move the selection to the top of the list:



Available Methods:
CDS
NOVELL
NIS
LDAP

Selected Methods:
ONAMES
TNSNAMES
HOSTNAME

   f. Choose File > Save Network Configuration.

   The `sqlnet.ora` file should contain an entry that lists ONAMES first in the NAMES.DIRECTORY_PATH parameter:

```
names.directory_path=(onames, tnsnames, hostname)
```

3. You may configure additional features for clients that use Oracle Names. These features are listed and described in the following table:

Net8 Assistant Field	sqlnet.ora Parameter	Description
Default Domain	NAMES.DEFAULT_DOMAIN	Indicates the domain name space from which the client most often requests an Oracle Names server. When set, this name is automatically appended to any unqualified name in an Oracle Names request. Net8 sets the default domain to NULL by default.
Maximum Wait Each Attempt	NAMES.INITIAL_RETRY_ TIMEOUT	Specifies how long a client waits for a response from an Oracle Names server before reiterating the request to another Oracle Names server. Net8 waits for 15 seconds by default.
Attempts Per Names Server	NAMES.REQUEST_RETRIES	Specifies the number of times a client attempts to iterate through the list of Oracle Names servers before allowing the operation to fail. Net8 attempts to iterate through the list of Oracle Names servers once before allowing the operation to fail by default.
Maximum Open Connections	NAMES.MAX_OPEN_ CONNECTIONS	Specifies how many connections an Oracle Names client may have open at one time. Net8 enables a client to have 10 connections open at any one time by default. This default value should be sufficient for almost all situations.
Initial Preallocated Requests	NAMES.MESSAGE_POOL_ START_SIZE	Enables you to pre-allocate an initial number of messages in a client's message pool. These messages may be used for future requests to Oracle Names servers. Net8 allocates 10 messages in the pool by default. This default value should be sufficient for almost all situations.

To add or configure these features:

a. In the navigator pane, expand Local > Profile.

b. From the list in the right pane, select Naming.

c. Click the Oracle Names tab.

d. Enter data for the options indicated in the table on the previous page.

e. Choose File > Save Network Configuration.

Task 5: Configure the Listener

Configure the listener with listening database protocol addresses if you are not using your operating system's primary protocol.

See Also: Chapter 7 for listener configuration details

Additionally, set USE_PLUG_AND_PLAY_*listener_name*=ON in the `listener.ora` file. This permits the listener to register database information, such as the service name, SID, and global database name, with Oracle Names servers.

To set the USE_PLUG_AND_PLAY parameter:

1. Start Net8 Assistant:

   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

2. In the navigator pane, expand Local > Listeners

3. Select a listener.

4. From the list in the right pane, select General Parameters.

5. Click the General tab.

6. Click "Register Service with Oracle Names".

7. Choose File > Select Save Network Configuration.

Task 6: Register Data with the Oracle Names Server

As described in the table in "Configuring the Oracle Names Method" on page 6-47, an Oracle Names server stores many kinds of information, including database locations, Oracle Names server locations, net service names, aliases, global database links, and Oracle Connection Manager. Some of this information is automatically registered with the Oracle Names server because the database registers information with the listener. If you would like to store any of the following kinds of information, you must register it using either Net8 Assistant or the NAMESCTL control utility:

- Net Service Names
- Global Database Links Credentials and Database Qualifiers
- Aliases

Registration needs to occur with only one Oracle Names server in an administrative region. The other Oracle Names servers see the information if data is stored in a region database, or the information is propagated to other Oracle Names if you chose not to use a region database.

After registering these kinds of information, you should query them, as described in "Testing Network Objects Using Net8 Assistant or NAMESCTL Utility" on page 11-14.

Net Service Names

Oracle Corporation advises creating a net service name for each service. If a net service name is not created, a user trying to connect must pass the global database name to make a connection. Rather than provide this information to your end users, create a net service name for the service.

Use either Net8 Assistant or the NAMESCTL control utility:

Use Net8 Assistant...	Use the NAMESCTL control utility...

1. Start Net8 Assistant:
-On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
-On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Oracle Names Servers.
3. Select an Oracle Names server.
4. From the list in the right pane, select Manage Data.
5. Click the Net Service Names tab.
6. Click Add.
7. Enter name in the Service Name field. The net service name may be qualified with the domain. The net service name is automatically domain qualified if the domain is specified.
See Also: "Configuring a Default Domain for Clients" on page 8-17
8. Select a protocol the listener is configured to listen on.
9. Enter the appropriate protocol parameters for the chosen protocol in the fields provided.
See Also: Appendix B for more information about protocol syntax needed for address information
Note: Additional addresses can be created by clicking "+" at the bottom of the Address tab.
10. Enter a destination service. If the destination service is an Oracle release 8.1 database, enter the service name in the Service Name field. If destination service is prior to release 8.1, click "Oracle8i Release 8.0 Compatible Identification", then enter its SID in the SID field.
Note: If you want to configure advanced CONNECT_DATA options besides the destination service, click Advanced.
11. Click Execute.
12. Choose File > Save Network Configuration.
	Create and register a net service name with the Oracle Names server:
	`NAMESCTL`
	`NAMESCTL> register` *`net_service_name`* `-d (description=[(address_list=](address= ...)(connect_data=(service_name|sid=)))[)]`
	For example, to register a net service name of `sales` with a default TCP/IP listening address, enter:
	`register sales -d (description=(address=(protocol-tcp)(host=sales-server)(port=1521))(connect_data=(service_name=sales.us.acme.com)))`
	See Also: Appendix B for more information about protocol syntax needed for address information and "CONNECT_DATA Section" on page C-42 for CONNECT_DATA parameters

If you have been using local naming as your configuration method and want to migrate to an Oracle Names configuration, the net service name information used in the `tnsnames.ora` file can uploaded to an Oracle Names server. To load information from a local naming configuration file into an Oracle Names server, use either Net8 Assistant or the NAMESCTL control utility:

Use Net8 Assistant...	**Use the NAMESCTL control utility...**

1. Start Net8 Assistant:
-On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
-On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
	From the command line, enter:
	`NAMESCTL`
	`NAMESCTL> load_tnsnames tnsnames.ora`

2. In the navigator pane, expand Oracle Names Servers.	
3. Select an Oracle Names server.	
4. From the list in the right pane, select Manage Data.	
5. Click the Net Service Names tab.	
6. Click Load.	
7. Enter the path for your current master local naming configuration file in the File field.	
8. Click Execute.	
9. Choose File > Save Network Configuration.	

Global Database Links Credentials and Database Qualifiers

Oracle Corporation recommends having one **global database link** per database. A global database link that is the same as the **global database name** is automatically registered with the Oracle Names server.

For example, if the global database name is `sales.us.acme.com`, a global link with the same name is registered with the Oracle Names server. You verify a global database link is working correctly by performing a SELECT statement on a table, for example:

```
SQL> SELECT * from emp@sales.us.acme.com
```

Unless you want to specify user name and password credentials for a global database link, no additional configuration is required to use the global database link.

To define a user name and password for a global database link, use either Net8 Assistant or the NAMESCTL control utility. Because the NAMESCTL utility is limited in global database link registration functionality, Oracle Corporation recommends you use Net8 Assistant.

Use Net8 Assistant...	Use the NAMESCTL control utility...

1. Start Net8 Assistant:
-On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
-On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Oracle Names Servers.
3. Select an Oracle Names server.
4. From the list in the right pane, select Manage Data.
5. Click the Links tab.
6. Select Add.
7. Enter the global database link in the DB Link Name field.
Note: If the GLOBAL_NAMES parameter has been set to TRUE in the initialization parameter file, the name entered must be the global database name.
8. Enter valid user name and password credentials for the database in the User and Password fields.
9. Click Execute.
10. If you want to use a link qualifier, see "Adding Link Qualifiers" on page 6-70.
11. Choose File > Save Network Configuration.
	Create and register a global database link with the Oracle Names server:
	`NAMESCTL`
	`NAMESCTL> register` *`dblink`* `-d`
	`(description=(address= ...))`
	For example, to register a global database link of `sales.us.acme.com` with a default TCP/IP listening address, enter:
	`register sales.us.acme.com -d (description=(address=(protocol-tcp)(host=sales-pc)(port=1521)))`
	See Also: Appendix B for more information about protocol syntax needed for address information
	The NAMESCTL utility does not permit user name and password credentials.

While one global database link per database is recommend, there is no limit on the number of **link qualifiers** that may be added. Link qualifiers are appended to global database link to provide alternate settings for the database user name and password credentials. For example, a link qualifier of EMP can be appended to `sales.us.acme.com`. A connection to a remote database with the global database link and link qualifier would be:

```
SQL> CONNECT @sales.us.acme.com@emp
```

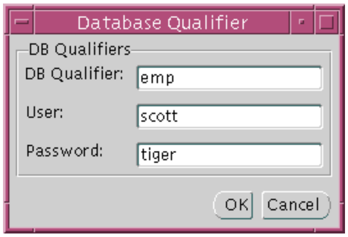
Adding Link Qualifiers

Link qualifiers are primarily for older Oracle database environments where multiple database links are used to get to the different schemas in the database. Rather than use multiple database links, multiple link qualifiers can be created for a database link. Multiple link qualifiers enable the global database links to comply with the GLOBAL_NAMES=TRUE setting in the initialization parameter file, which enables the global database link to be set to the global database name.

To create a global link qualifier for a global database link, use Net8 Assistant. The NAMESCTL utility does not support creation of link qualifiers.

1. Follows steps 1-8 on the procedure on the previous page.
2. Click DB Qualifiers.
3. Click Create Qualifier.

   The Database Qualifier dialog box appears:



Database Qualifier

DB Qualifiers

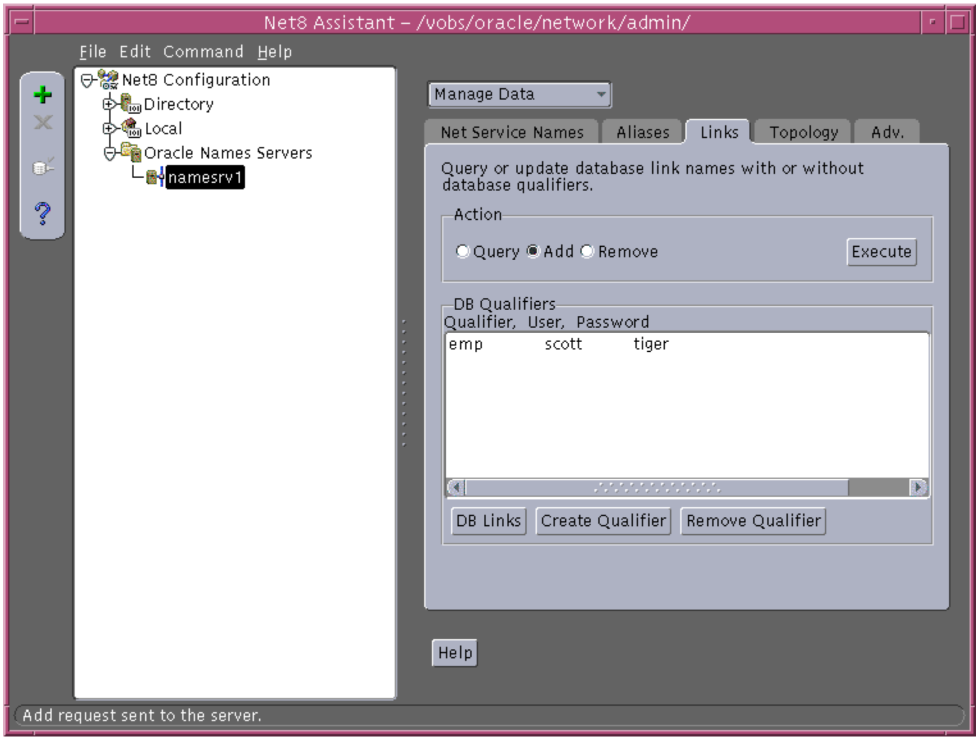
DB Qualifier: emp

User: scott

Password: tiger

4. Enter a name for the link qualifier, and valid user name and password credentials, then click OK.

The link qualifier is added to the DB Qualifiers list box:



Manage Data

Query or update database link names with or without database qualifiers.

Action

○ Query ● Add ○ Remove

DB Qualifiers

Qualifier,	User,	Password
emp	scott	tiger

Add request sent to the server.

5. Repeat Steps 3 and 4 for each link qualifier you want to add.
6. Click Execute.
7. Choose File > Save Network Configuration.

To verify a global database link and link qualifier is working correctly, perform a SELECT statement on a table the user is authorized for:

```
sql> SELECT * from  table@db_link;
sql> SELECT * from  table@db_link@link_qualifier;
```

For example:

```
sql> SELECT * from EMP@sales.us.acme.com;
sql> SELECT * from EMP@sales.us.acme.com@emp;
```

See Also: *Oracle8i Distributed Database Systems*

Aliases

An alias can be used for any type of item stored in the Oracle Names server, such as a Oracle Names server, net service name, or global database link.

To create an alias, use either Net8 Assistant:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Oracle Names Servers.
3. Select an Oracle Names server.
4. From the list in the right pane, select Manage Data.
5. Click the Aliases tab.
6. Select Add.
7. Enter the alias in the Alias field and the actual name of the object in the Canonical Name field.
8. Click Execute.
9. Choose File > Save Network Configuration.

Task 7: (Optional) Delegate Domains to Delegated Administrative Regions

If delegated administrative regions are configured, the upper-level administrative region must delegate sub-domains to the delegated administrative regions. Otherwise, the Oracle Names servers in the upper-level administrative region assumes authority over all sub-domains.

Once a domain is delegated, the Oracle Names servers in the current region forwards subsequent operations to the sub-region where the domain is administered by Oracle Names servers.

To delegate a domain, use either Net8 Assistant or the NAMESCTL control utility.

Use Net8 Assistant...

1. Start Net8 Assistant:

   -On UNIX, run `netasst` from `$ORACLE_HOME/bin`.

   -On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Oracle Names Servers
3. Select an Oracle Names server in the upper-level region.
4. From the list in the right pane, select Manage Data.
5. Click the Topology tab.
6. Click Delegate Domain.
7. Enter the name of the sub-domain in the Domain Name field.
8. Enter the Oracle Names server that serves the domain in the Names Server Name field, and its protocol address in the Names Server Address field.

   See Also: Appendix B for more information about protocol syntax needed for address information
9. Click Execute.
10. Choose File > Save Network Configuration.

Use the NAMESCTL control utility...

Create and register an alias with the Oracle Names server:

```
NAMESCTL
NAMESCTL> delegate_domain domain_name onames_server
[(description=(address_list=] (address= ...)[))]
```

See Also: Appendix B for more information about protocol syntax needed for address information

For example, to delegate domain `us.acme.com` serviced by namesrv3.us.acme.com, enter:

```
delegate_domain us.acme.com
namesrv3.us.acme.com
(address=(protocol-tcp)(host=namesrv3-pc)(port
=1575)))
```

Task 8: Connect to the Database

Clients can connect to the database using the following syntax:

```
CONNECT username/password@net_service_name
CONNECT username/password@database_service
```

Configuring the Host Naming Method

In environments where simple connectivity is desired, host naming can eliminate the need for service name lookup in the `tnsnames.ora` files. However, for large or complex environments where advanced features such as connection pooling, external procedures, or Heterogeneous Services, which require additional connect information, are desired, host naming is not suitable. In these cases, another naming method is recommended.

The host naming method is available for TCP/IP network environments only. When instances register with the listener, the global database name of the server, a name comprised of the database name and domain name. The global database name is equal to a host name on an alias in an existing name resolution service. Clients use this name in their connect string to connect to the database.

Clients may connect to a server using the alias if:

- You are connecting to an Oracle8*i* database service with Net8 Server/Net8 Client software installed
- Your client and server are connecting over a TCP/IP protocol
- All names are resolved through an IP address translation mechanism such as **Domain Name System (DNS)**, or a centrally maintained TCP/IP `hosts` file
- No advanced features like **Oracle Connection Manager** or security are requested or required

To configure the host naming method, perform these tasks:

- Task 1: Configure the Listener with the Global Database Name
- Task 2: Configure HOSTNAME as the First Naming Method
- Task 3: Set Up Host Name Resolution Environment
- Task 4: Connect to the Database

Task 1: Configure the Listener with the Global Database Name

The global database name must be registered with the listener. How this registration takes place depends upon the release of the database.

Oracle8*i* release 8.1

If the database can find the listener, information about the database is automatically registered with the listener, including the global database name. The listener is found if:

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

If the database cannot find the listener, the `listener.ora` file must be configured with the GLOBAL_DBNAME parameter, as described in "Oracle8i Release 8.0 or Previous".

Oracle8*i* Release 8.0 or Previous

If the database is an Oracle8*i* release 8.0 database or Oracle7 database, database information is registered with the listener through the `listener.ora` file. For these database, statically configured the SID_LIST_*listener_name* section to include the GLOBAL_DBNAME parameter, as shown next:

```
sid_list_listener=(sid_list=
(sid_desc=
 (global_dbname=sales.us.acme.com)
 (sid_name=db1)
 (oracle_home=/u01/app/oracle/8.0.5))
```

See Also: "Configuring Static Service Information" on page 7-13

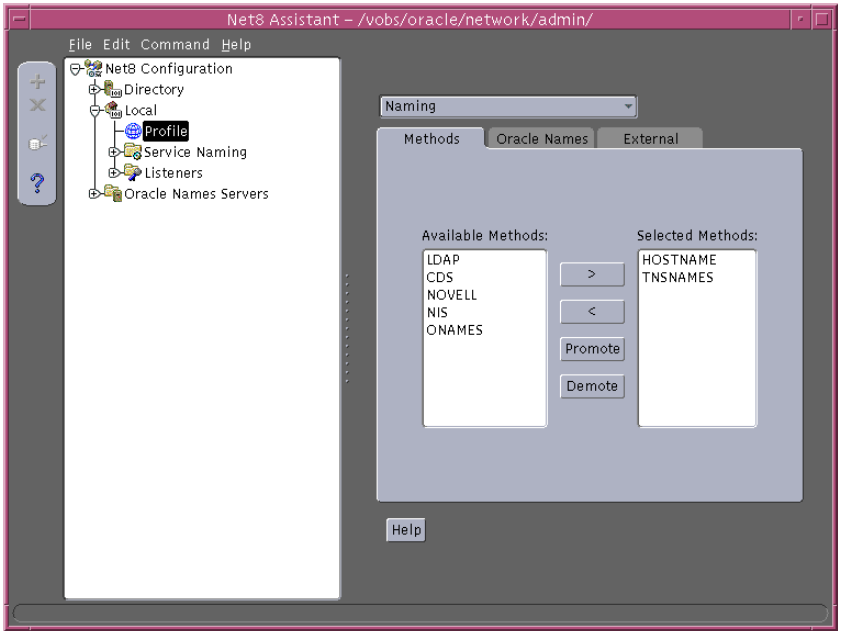
Task 2: Configure HOSTNAME as the First Naming Method

Configure host 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 Net8 can use to resolve connect identifiers to connect descriptors.

To specify host naming as the first naming method:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Profile.
3. From the list in the right pane, select Naming.
4. Click the Methods tab.
5. Choose HOSTNAME from the Available Methods list, then click the right-arrow button.

6. Select HOSTNAME in the Selected Methods list, 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=(hostname, tnsnames)
```

Task 3: Set Up Host Name Resolution Environment

The global database name must be resolved through an IP address translation mechanism, such as DNS, NIS, or a centrally-maintained TCP/IP host file, `/etc/hosts`.

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

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

Note that the domain portion of the global database name must match the network domain.

Task 4: Connect to the Database

Clients can connect to the database using the alias. Using the example in "Task 3: Set Up Host Name Resolution Environment", the connect string, the client can use `sales.us.acme.com` in the connect string:

```
CONNECT username/password@sales.us.acme.com
```

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 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 (NIS)**
- **Novell Directory Services (NDS)**
- **Cell Directory Services (CDS)**

> **See Also:** *Oracle Advanced Security Administrator's Guide* for instructions on configuring CDS

Configuring NIS

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/tiger@payroll
```

(where "payroll" is an Oracle service) NIS External Naming on the node running the client program (or 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 Net8 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 machine that acts as an NIS server runs a program called `ypserv`, which handles name requests. `ypserv` stores different types of data in special files called **maps**. 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 command like the one in the previous section, 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.

System Requirements

NIS External Naming requires SQL*Net 2.2 or greater.

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.

Add 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 6-5.

   > **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 on the Oracle Universal Installer tape or disk.

   For example, run `tns2nis` on the command line with one argument:

   ```
   tns2nis tnsnames.ora
   ```

   `tns2nis` reads the `native.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/native.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. Refer to your NIS documentation for more information.

Note: This step should be performed by the person in charge of NIS administration.

`makedbm` converts the `tnsnames` map into two files that the NIS server can read. The location of these files is platform-specific. Refer to your platform-specific documentation for details.

For example, to generate and install a `tnsnames` map on the Sun Solaris, as root 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
```

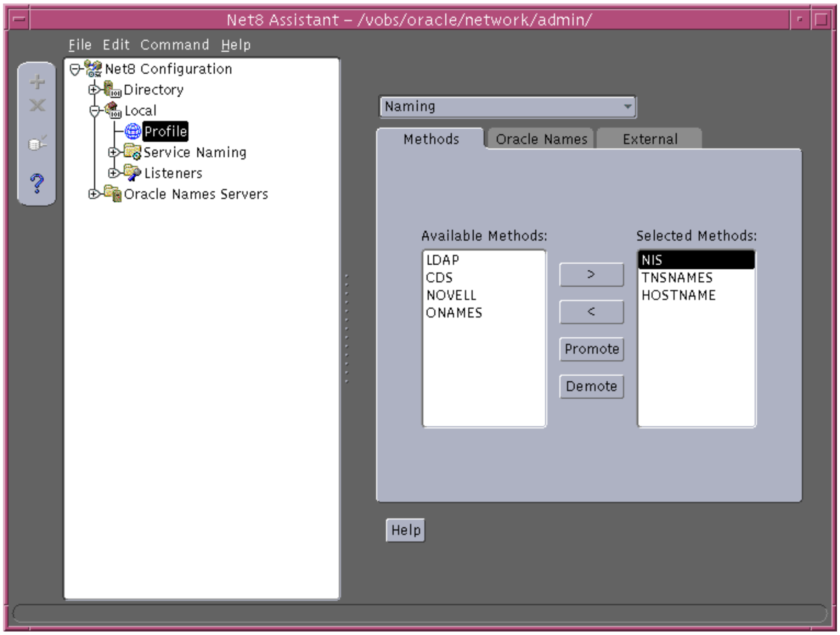
This returns the length of the address (in characters) followed by the address; for example:

```
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 Net8 can use to resolve connect identifiers to connect descriptors.

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Profile.
3. From the list in the right pane, select Naming.
4. Click the Methods tab.
5. Select NIS from the Available Methods list, then click the right-arrow button.
6. Select NIS in the Selected Methods list, then use the Promote button to move the selection to the top of the list:



Net8 Assistant - /vobs/oracle/network/admin/

Naming

Available Methods:
- LDAP
- CDS
- NOVELL
- ONAMES

Selected Methods:
- NIS
- TNSNAMES
- HOSTNAME

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)
   ```

Configuring NDS

NDS external naming enables you to use native NDS naming conventions to connect to an Oracle database on a Novell NDS-enabled network. After the NDS external naming has been installed on clients and servers, users can enter:

```
username/password@.OracleInstance.Context
```

NDS external naming provides network users with the following benefits:

- Enables clients to use simple NDS names (partial or full) when connecting to a database.
- Simplifies the maintenance of the network addresses; one change can affect all clients using NDS external naming.
- Reduces network traffic by eliminating the need for the listener to advertise.

Client Operations

NDS external naming resides on the client workstation and translates the NDS object name into a network address. The client code gets attributes from the NDS tree for the NDS object whose name matches the Oracle database. This name can be a full name or a partial name. If it is a partial name, it is qualified with respect to the current naming context.

Netware Server Operations

There are three aspects to the server-side NDS External Naming:

- Schema Extension
- SAP (Service Advertising Protocol) Disabling
- Database Service and Address Storage in NDS

Schema Extension During the Oracle installation process on Netware 4 and 5 servers the NDS schema is extended to include an object class called "ORACLE:DBInstance". For NDS external naming to function, this class needs an attribute called "ORACLE:TNSAddress". If the class does not exist, it is created and includes the TNSAddress attribute. If the class exists but the TNSAddress attribute does not, the class is modified. The Netware Loadable Module (NLM) that performs this during installation is called ORASCHEM.NLM.

SAP (Service Advertising Protocol) Disabling The SPX protocol for Netware looks for a value in the `config.ora` file called ORACLE_SAP. The value of this parameter is ON or OFF. If this parameter is not specified, the default is ON. This has performance implications for SPX networks. When ORACLE_SAP is ON, the listener advertises its address using SAP (Service Advertising Protocol). When ORACLE_SAP is set to OFF, the listener relies on NDS to deliver information to its clients.

If your network consists entirely of NDS enabled clients—that is, clients using Netware 4 and above—you can get better network performance if you edit the `config.ora` file to set ORACLE_SAP to OFF.

Database Service and Address Storage in NDS When the listener is started, it stores the database address in NDS under the OracleInstance object.

Note: An NDS object for the Oracle database must have already been created.

At that point, the address(es) is (are) accessible to the client from the NDS database

System Requirements

NDS external naming requires SQL*Net 2.2 or later and Oracle 7.2 or later. It can be used with any client running Novell libraries, but requires Netware 4.1 or later on the server.

Configuring Clients

Clients should be configured with the NDS external naming method and the NDS naming context.

To configure clients:

1. Configure NDS as the first method specified in the NAMES.DIRECTORY_PATH parameter in the `sqlnet.ora` file. This parameter specifies the order of naming methods Net8 can use to reconcile services.

   a. Start Net8 Assistant:

      - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.

      - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

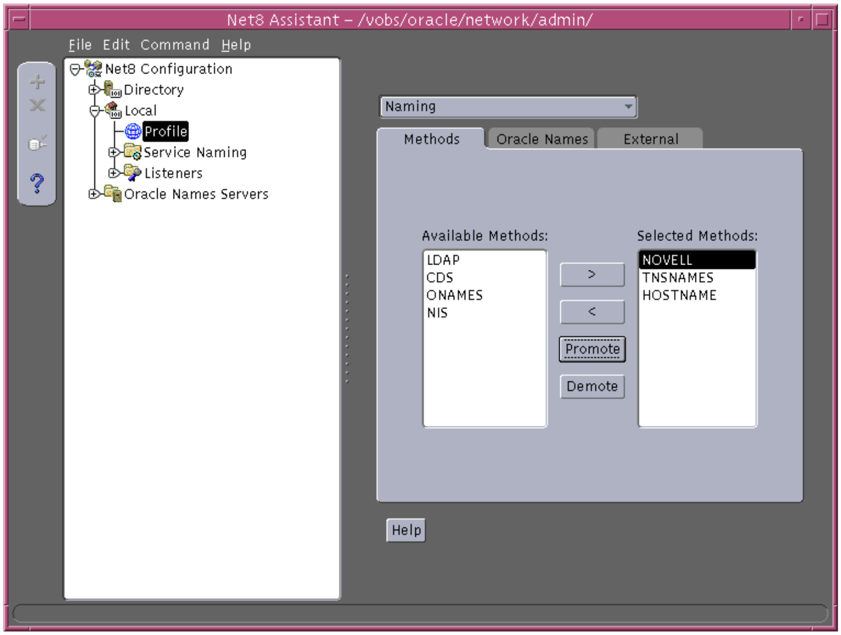
   b. In the navigator pane, expand Local > Profile.

   c. From the list in the right pane, select Naming.

   d. Click the Methods tab.

   e. Select Novell from the Available Methods list, then click the right-arrow button.

f. Select Novell in the Selected Methods list, then use the Promote button to move the selection to the top of the list:



Available Methods:
LDAP
CDS
ONAMES
NIS

Selected Methods:
NOVELL
TNSNAMES
HOSTNAME

2. Configure a typeless or typed name with the NAMES.NDS.NAME_CONTEXT parameter in the `sqlnet.ora` file. This parameter specifies the naming context within the NDS tree where the database object resides. To specify a naming context:

   a. Click the External tab.

   b. Enter a naming context in the Name Context field.

   If the name of the database object is "Payroll.Finance.Oracle" and the naming context is `Finance.Oracle`, only "Payroll" needs to be specified in the connect string. "Payroll" is qualified to ".Payroll.Finance.Oracle". This is an example of a typeless name.

   Note: The leading dot designates this as a full NDS name. If you want to override the name context setting, then you can specify the full NDS name in the connect string by using a leading dot.

If the naming context is `ou=Finance.o=Oracle`, then the connect string would have to include a name of `cn=Payroll.ou=Finance.o=Oracle`. This is an example of a typed name.

The name context in the `sqlnet.ora` file can override the "name context" entry in the `client32net.cfg` file. If no name context is specified in either file, it defaults to [root]. See the Novell client documentation for more information about the `client32net.cfg` file parameters.

Note: The name context cannot contain a leading dot. This can result in an NDS error code of -309 (ERR_EXPECTED_ IDENTIFIER):

```
The parameter being parsed is not typed.
```

3. Choose File > Save Network Configuration.

   The `sqlnet.ora` file updates with the following entries:

   ```
   names.directory_path=(novell, tnsnames, hostname)
   names.nds.name_context=Finance.Oracle
   ```

Configure the Server

To configure the Netware Server, use Net8 Easy Config application, as described in your Novell documentation.

Known Limitations

Following are some known limitations when using the NDS external naming:

- The database address stored in the NDS database cannot be more than 2048 characters in length.
- You cannot use more than one listener per database instance. If you do, the last listener to start overwrites any other TNS address stored in the database object.
- If SID support is enabled on the server, you should not use a NULL SID for any of the database instances. If a NULL SID is used for one of the instances, you cannot connect to it using SQL*Net version 2 or Net8.
- If SID support is not enabled, the last SID specified in the listener's SID_LIST can be the one used. In this case, the SID is transparent to the user and database. The user does not see it and the database ignores it.

Note: SID support is controlled by the following parameter in the `config.ora` file:

```
nw_enable_sid_support=[true|false]
```

7

Configuring the Listener

This chapter describes how to configure the listener. It includes the following sections:

- Configuration Overview
- Re-Starting the Listener After a listener.ora File Modification
- Configuring Listener Protocol Addresses
- Configuring a Listener that Uses a Non-Default Address
- Handling Large Volumes of Connection Requests
- Configuring Static Service Information
- Configuring Prespawned Dedicated Servers

Configuration Overview

The listener is a separate process that resides on the server. It receives incoming client connection requests and manages the traffic of these requests to the server.

A listener is configured with one or more listening protocol addresses and service information about the destination service.

Protocol addresses are configured in the listener configuration file, `listener.ora`. Service information may or may not be configured in the `listener.ora` file:

- An Oracle release 8.1 instance automatically registers information with the listener, such as its service names, instance names, service handlers, and load information. This feature, called **service registration**, does not require configuration in the `listener.ora` file.

> **Note:** If you use Oracle Enterprise Manager to manage an Oracle8*i* database, service configuration is required in the `listener.ora` file. Oracle Enterprise Manager relies on service information in the `listener.ora` file to discover the database and to perform remote administration.

- Other services, including an Oracle7 or Oracle8 release 8.0 database and the Oracle Enterprise Manager management tools, require service configuration in the `listener.ora` file.

Once the listener is configured, the listener may be managed with the Listener Control Utility (LSNRCTL).

> **Note:** A release 8.1.6 listener is required for an Oracle8*i* release 8.1.6 database. Previous version of the listener are not supported for use with an Oracle8*i* release 8.1.6 database. However, it is possible to use a release 8.1.6 listener with previous versions of the Oracle8*i* database.

This section covers the following additional configuration topics:

- Default Listening Protocol Addresses
- Configuration During Installation
- Configuration with Net8 Assistant

Default Listening Protocol Addresses

The listener has a default name of LISTENER and is configured to listen on the following default protocol addresses:

- TCP/IP protocol on port 1521

  ```
  (address=(protocol=tcp)(host=host_name)(port=1521))
  ```

- IPC protocol

  ```
  (address=(protocol=ipc)(key=PNPKEY))
  ```

Clients configured with these addresses can connect to the listener.

Configuration During Installation

Net8 Configuration Assistant is launched by the Oracle Universal Installer after software installation. It enables you to configure the protocol address and service information for a database.

After a Typical or Minimal installation on the server, Net8 Configuration Assistant automatically configures a listener with a name of LISTENER for an Oracle8*i* database using the TCP/IP protocol. After a Custom installation, Net8 Configuration Assistant prompts you to configure a listener name and a protocol address of your choice.

Additionally, a listening address for **external procedures**—functions that can be called from PL/SQL code—using the IPC protocol is automatically configured, regardless of the installation type.

Net8 Configuration Assistant also automatically configures service information for the Oracle8*i* database and the external procedures in the `listener.ora` file.

> **Note:** While service information is not required in the `listener.ora` file for an Oracle release 8.1 database service, Net8 Configuration Assistant assumes that you may use Oracle Enterprise Manager to monitor the database. Oracle Enterprise Manager requires this information to discover the database.

Shown next is a `listener.ora` file sample. The LISTENER entry defines the listening protocol address for a listener named LISTENER, and the SID_LIST_LISTENER entry provides information about the database service, including the global database name, the Oracle home location of the database, and the Oracle System Identifier (SID) of the instance.

```
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=/oracle81)
      (sid_name=sales)))
    (sid_desc=
      (sid_name=plsextproc)
      (oracle_home=/oracle81)
      (program=extproc)))
```

Configuration with Net8 Assistant

If the default or installed configuration is not adequate or the network has older versions of the listener or database, you can use **Net8 Configuration Assistant** to enhance the `listener.ora` configuration.

The following listener configuration file items are addressed in the following sections:

- Re-Starting the Listener After a listener.ora File Modification
- Configuring Listener Protocol Addresses
- Handling Large Volumes of Connection Requests
- Configuring Static Service Information

See Also:

- "Configuring Net8 for External Procedures" on page 8-45 for further information about external procedure configuration
- "Configuring Connections to Non-Oracle Database Services" on page 8-45 for further information about Java option and IIOP client configuration

Re-Starting the Listener After a listener.ora File Modification

Anytime the `listener.ora` is modified, the listener must be stopped and re-started with the LSNRCTL utility.

To start or stop the listener:

```
LSNRCTL
LSNRCTL> start|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.

See Also:

- "Using Net8 Control Utilities" on page 11-3 for further information about starting the listener
- "Listener Control Utility (LSNRCTL)" on page A-3 for a complete listing of all LSNRCTL commands

Configuring Listener Protocol Addresses

To configure other, non-default protocol listener addresses:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Listeners.
3. Select a listener.

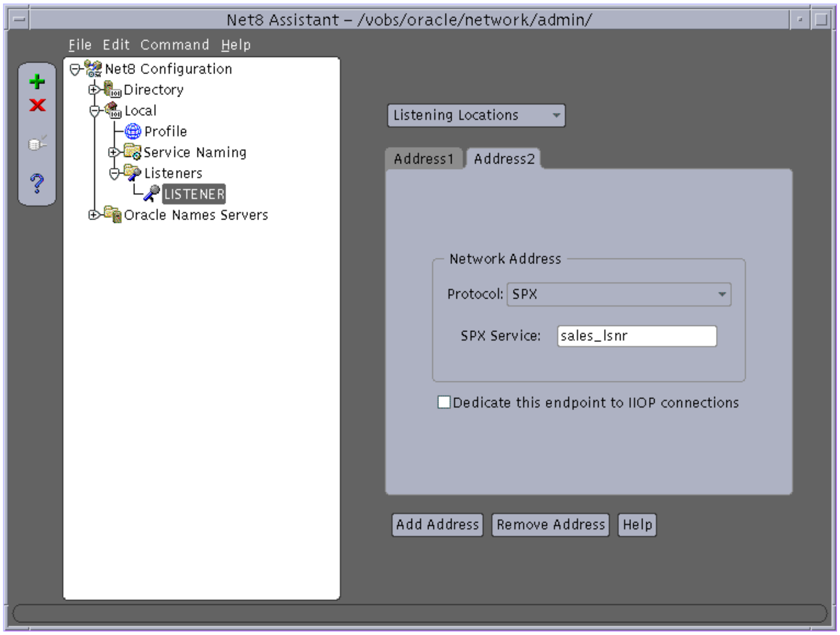
   > **Note:** If a `listener.ora` file has been configured, but the listener has never been started with the LSNRCTL utility, the Listeners folder does not display a listener. If this occurs, exit Net8 Assistant, start the listener, as described in "Using Net8 Control Utilities" on page 11-3, and start Net8 Assistant again.

   If a listener has never been created:

   a. Click "+" from the toolbar or select Create from the Edit menu.

   b. Enter a unique listener name in the Choose Listener Name dialog box.
4. From the list in the right pane, select Listening Locations.

5. Click Add Address. A new address tab appears.



Network Address

Protocol: SPX

SPX Service: sales_lsnr

☐ Dedicate this endpoint to IIOP connections

6. Select a protocol and enter the appropriate protocol parameter information for the selected protocol.

> **See Also:** Appendix B for further information about protocol addresses and parameters

When configuring the listener to listen on TCP/IP, you should choose the default port or 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:** "Configuring a Listener that Uses a Non-Default Address" on page 7-10

If your machine 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 machine in the Host field.

7. Repeat Step 5 and 6 for additional protocols.

8. Choose File > Save Network Configuration.

A `listener.ora` file statically configured with TCP/IP and SPX address follows:

```
listener=
  (address=(protocol= tcp)(host= sales-server)(port= 1521))
  (address=(protocol= spx)(service=orasrvc1))
```

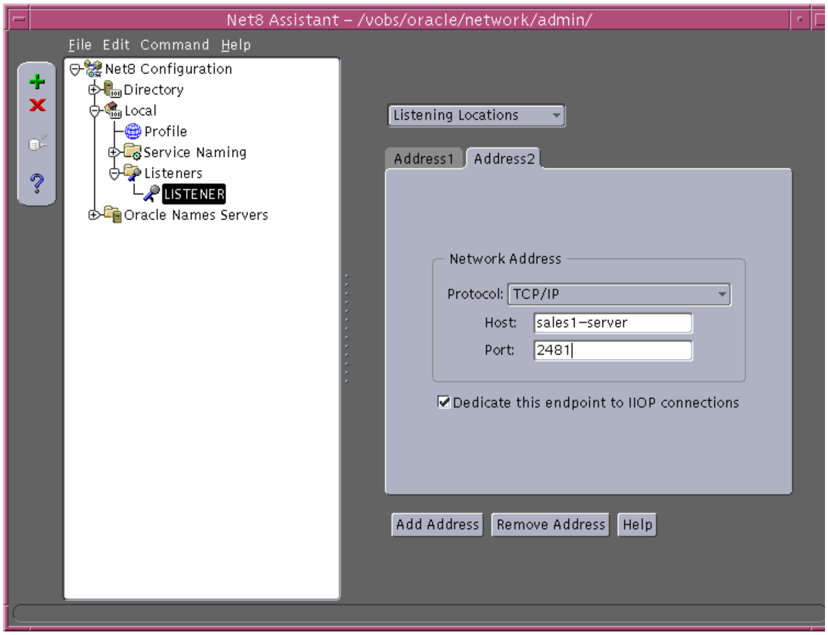
Configuring Listening Addresses for Oracle8*i* JServer

Connections to the **Oracle8*i* JServer** in the database requires the TCP/IP or TCP/IP with SSL protocol addresses

To configure a listening address for Oracle8i JServer:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
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.



Protocol: TCP/IP
Host: sales1–server
Port: 2481

☑ Dedicate this endpoint to IIOP connections

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 if the chosen protocol is TCP/IP in the Port field, or enter port 2482 if the chosen protocol is TCP/IP with SSL in the Port field.

9. Click "Dedicate this endpoint to IIOP connections".

10. Choose File > Save Network Configuration.

    The `listener.ora` file updates with the following:

```
listener=
 (description_list=
  (descriptioin=
   (address=(protocol=tcp)(host=sales1-server)(port=2481))
   (protocol_stack=
     (presentation=giop)
     (session=raw))))
```

See Also:

- *Oracle8i Enterprise JavaBeans and CORBA Developer's Guide* for information on configuring client connections to EJB and CORBA applications
- *Oracle8i Java Stored Procedures Developer's Guide* for information on configuring client connections to Java stored procedures

Configuring a Listener that Uses a Non-Default Address

By default, the **PMON process** registers with the local listener on the default local address of TCP/IP on a port 1521. If you want to register information with another listener, you must configure the LOCAL_LISTENER parameter in the initialization parameter file in order for PMON to register information with it. The value for LOCAL_LISTENER must then be resolved with a net service name entry in the `tnsnames.ora` file or an Oracle Names server.

- Configuring the LOCAL_LISTENER Parameter
- Configuring a Naming Method

Configuring the LOCAL_LISTENER Parameter

The LOCAL_LISTENER parameter should be configured as follows in the initialization parameter file:

```
local_listener=listener_name_alias
```

Configuring a Naming Method

The listener name alias specified for the LOCAL_LISTENER parameter must be resolved through a one of the following naming methods:

- tnsnames.ora File
- Oracle Names Server

tnsnames.ora File

A net service name entry should be created for the listener address without the CONNECT_DATA portion of the connect descriptor. Net8 Assistant does not enable you to configure a `tnsnames.ora` file without the CONNECT_DATA information. Therefore, Oracle Corporation recommends that you modify the `tnsnames.ora` file manually.

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 look like:

```
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. Oracle Connection Manager's SOURCE_ROUTE parameter is supported.

See Also: Chapter 8 for further information about multiple address configuration

Oracle Names Server

An entry for the listener alias name can be made to an Oracle Names server through Net8 Assistant:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Oracle Names Servers.
3. Select the Oracle Names server.
4. From the list in the right pane, select Manage Data.
5. Click the Advanced tab.
6. Click Add, enter the listener alias name in the Name field, A.SMD record type in the Type field, and enter the address in the Value field, for example:

   ```
   (address=(protocol=tcp)(host=sales-server)(port=2481))
   ```

7. Choose File > Save Network Configuration.

Handling Large Volumes of Connection Requests

If you expect the listener to handle large volumes of connection requests, you may specify a queue for the process. This enables the listener to dynamically handle larger numbers of concurrent connection requests.

To specify a queue size for a listener, enter a value to the QUEUESIZE parameter at the end of any listening address:

```
listener=
  (address=
     (protocol= tcp)
     (host= sales-server)
     (port= 1521)
     (queuesize=20))
```

Note: Currently, you can configure only the queue size for listeners operating on TCP/IP and DECnet. The default queue size is system specific. On Solaris, the default queue size is set to 5. The queue size is 5 for Windows NT 4.0 Workstation and 50 for Windows NT 4.0 Server.

Configuring Static Service Information

In order for the listener to accept client requests to an Oracle8 release 8.0 or Oracle7 database, the `listener.ora` file must be configured with information about the database. Static configuration is also required for other services, such as **external procedures** and **Heterogeneous Services**, and some management tools, including Oracle Enterprise Manager.

In Oracle8 release 8.1, the listener uses the dynamic information about the database and instance it has received through service registration before using statically configured information.

The `listener.ora` file can be configured with service parameters described in the following table:

listener.ora File Parameter	Net8 Assistant Field	Description
SID_NAME	SID	The **Oracle System Identifier (SID)** identifies the instance. You can obtain the SID value from the INSTANCE_NAME parameter in the initialization parameter file. **Note:** This setting is required on UNIX and Windows NT.
GLOBAL_DBNAME	Global Database Name	The **global database name** is comprised of the database name and database domain name. You can obtain the GLOBAL_DBNAME value from the SERVICE_NAMES parameter or the DB_NAME and DB_DOMAIN parameters in the initialization parameter file.
ORACLE_HOME	Oracle Home Directory	The Oracle Home Directory identifies the Oracle home location of the database that you are specifying. **Note:** This setting is required on UNIX.

Important: If you are using **connect-time failover** or **Transparent Application Failover (TAF)**, such as in an Oracle Parallel Server environment, do not set the GLOBAL_DBNAME parameter. A statically configured global database name disables these features.

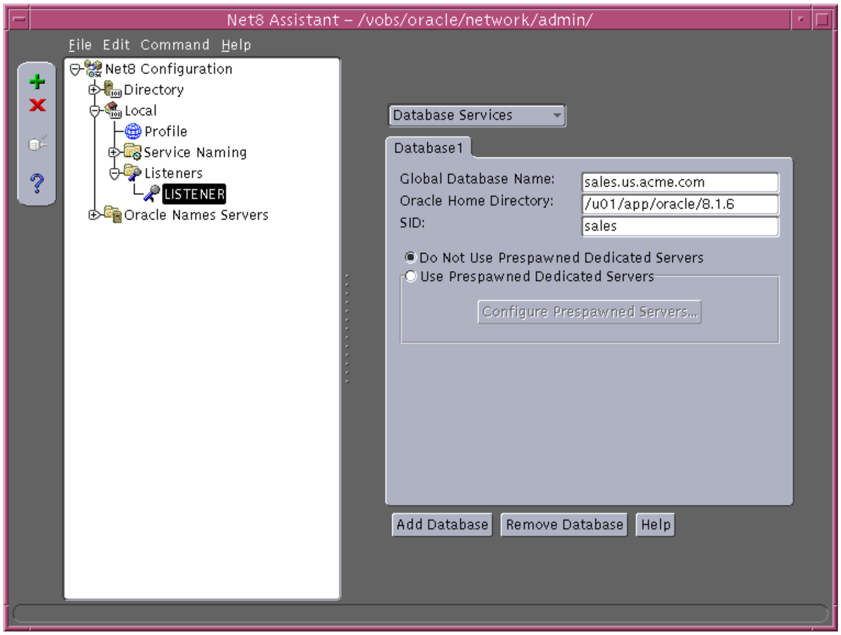
See Also:

- Chapter 8 for more information about statically configuring the listener for external procedures and heterogeneous services
- *Oracle Enterprise Manager Configuration Guide* further information about Oracle Enterprise Manager

To statically configure the listener:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Listeners.
3. Select a listener.
4. From the list in the right pane, select Database Services.

5. Click Add Database. A new database tab appears.



Global Database Name: sales.us.acme.com
Oracle Home Directory: /u01/app/oracle/8.1.6
SID: sales

● Do Not Use Prespawned Dedicated Servers
○ Use Prespawned Dedicated Servers

6. Enter the global database name, location of the Oracle home directory, and the Oracle System Identifier (SID) of the instance in the appropriate fields.

7. Choose File > Save Network Configuration.

An excerpt of a `listener.ora` file statically configured for a database service called `sales.us.acme.com` follows:

```
sid_list_listener=
(sid_list=
 (sid_desc=
  (global_dbname=sales.us.acme.com)
  (sid_name=sales)
  (oracle_home=/u01/app/oracle/8.1.6)))
```

Configuring Prespawned Dedicated Servers

Prespawned dedicated servers are dedicated servers started by the listener at startup before any incoming connection requests. They improve the time it takes to establish a connection on servers where **multi-threaded server (MTS)** is not used. They also use allocated memory and system resources better by recycling server processes for use by other connections without shutting down and recreating a server.

Note: Oracle Corporation recommends that you use MTS rather than prespawned dedicated server to solve performance and scaleability problems.

Note: Prespawned dedicates servers are not available for Windows NT.

The `listener.ora` file can be configured with the prespawned dedicated server parameters described in the following table:

listener.ora File Parameter	Net8 Assistant Field	Description
PRESPAWN_MAX	Maximum Prespawned Servers	The maximum number of prespawned dedicated servers the listener can create. This number must be at least as many as the sum of the pool size for each protocol. Set this value to a large number so that prespawned dedicated servers are always available for new connections.
PROTOCOL	TCP/IP, SPX, Named Pipes, IPC	The protocol on which the listener creates prespawned dedicated servers.

listener.ora File Parameter	Net8 Assistant Field	Description
POOL_SIZE	Number	The number of unused prespawned dedicated servers for the listener to maintain on the selected protocol. Choose a number that is greater than 0 but no greater than the PRESPAWN_MAX value. The value should be about what you expect the average number of connections to be at any given time.
TIMEOUT	Timeout	Time in minutes that an inactive prespawned dedicated server process waits for the next connection. The value should be greater than 0. (A value of 0 enables an inactive shadow process to continue indefinitely, thus wasting machine resources.) Set a short time-out value. The time out is activated only after a prespawned dedicated server process has carried a connection and been disconnected. In other words, prespawned dedicated servers that are waiting for their first connection do not time out.

In Oracle8*i* release 8.1, the listener handles client requests based on information it receives from PMON through service registration rather than static information in the `listener.ora` file. If dynamic information for a particular service name or instance names is not available, the listener falls back to the static information. Because of this, additional configuration is necessary in order to use prespawned dedicated servers.

You can configure some clients to use the prespawned dedicated servers and other clients to use service handlers (**dedicated servers** or MTS **dispatchers**) that have been dynamically registered with the listener. For those clients that are to use prespawned dedicated servers, no special configuration is needed. Clients should use connect descriptors that contain a SERVICE_NAME or a SID parameter value that matches the GLOBAL_DBNAME or the SID_NAME parameter value in the `listener.ora` file.

Clients that are to use dynamically registered service handlers should use connect descriptors that contain a SERVICE_NAME parameter value that is different from the GLOBAL_DBNAME parameter value specified in the `listener.ora`. This service name has to be configured in the initialization parameter file with the SERVICE_NAMES parameter. Once the database is started, the service name is dynamically registered with the listener and the clients can connect to the database.

For an environment with pre-release 8.1 clients, an analogous modification needs to be done for INSTANCE_NAME parameter in the initialization parameter file. It has to be set to a value different from SID_NAME specified in the `listener.ora` file. The INSTANCE_NAME parameter value dynamically registers with the listener along with the SERVICE_NAMES parameter value and service handler information. The fact that this INSTANCE_NAME parameter value is different from the SID_NAME parameter value in the `listener.ora` file ensures that the clients that use connect descriptors containing SID (and matching the SID_NAME value) are redirected to a prespawned dedicated server.

Configuration tasks are described in the following sections:

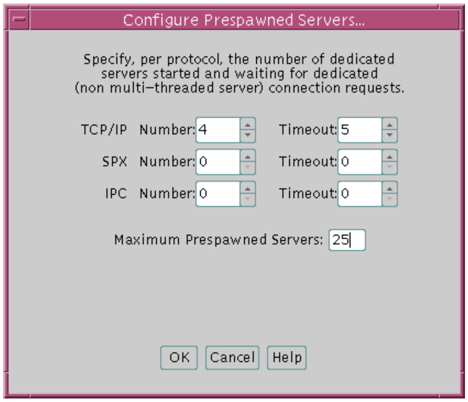
- Configure the listener.ora File with Prespawned Dedicated Server
- Modify Configuration for Clients That Are Not to Use Prespawned Dedicated Servers

Configure the listener.ora File with Prespawned Dedicated Server

To configure prespawned dedicated servers in the `listener.ora` file:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Listeners.
3. Select a listener.
4. From the list in the right pane, select Database Services.
5. Select Use Prespawned Dedicated Server, and click Configure Prespawned Servers.

The Configure Prespawned Servers dialog box appears:



Configure Prespawned Servers...

Specify, per protocol, the number of dedicated servers started and waiting for dedicated (non multi-threaded server) connection requests.

TCP/IP Number: 4 Timeout: 5

SPX Number: 0 Timeout: 0

IPC Number: 0 Timeout: 0

Maximum Prespawned Servers: 25

6. Specify the number of dedicated servers to start for each protocol.
7. Choose File > Save Network Configuration.

An excerpt of a `listener.ora` file configured with prespawned dedicated servers follows:

```
sid_list_listener=
(sid_list=
 (sid_desc=
  (global_dbname=sales.us.acme.com)
  (sid_name=sales)
  (oracle_home=/u01/app/oracle/product/8.1.6)
  (prespawn_max=25)
  (prespawn_list=
   (prespawn_desc=
    (protocol=tcp)
    (pool_size=4)
    (timeout=5)))))
```

Modify Configuration for Clients That Are Not to Use Prespawned Dedicated Servers

If you have clients that use SERVICE_NAME in their connect descriptor, replace the value for the SERVICE_NAMES parameter in the initialization parameter file with another value and connect to the new service name. For example, if the initialization parameter file contains `service_names=sales.us.acme.com`, change `sales.us.acme.com` to `sales2.us.acme.com`. Create a connect descriptor that connects to `sales2.us.acme.com`, as shown in the following:

```
sales=
(description=
  (address=(protocol=tcp)(host=sales-server)(port=1521))
  (connect_data=
    (service_name=sales2.us.acme.com)))
```

If you have clients that use SID in their connect descriptor, replace the value for the INSTANCE_NAME parameter in the initialization parameter file with another value and connect to the new instance. For example, if the initialization parameter file contains `instance_name=sales`, change `sales` to `sales2`. Create a connect descriptor that connects to `sales2`, as shown in the following:

```
sales=
(description=
  (address=(protocol=tcp)(host=sales-server)(port=1521))
  (connect_data=
    (sid=sales2)))
```

8

Enabling Advanced Net8 Features

This chapter describes how to enable advanced features. It includes the following sections:

- Configuring Advanced Network Address and Connect Data Information
- Configuring Transparent Application Failover
- Configuring Client Attributes for Names Resolution
- Configuring Advanced Profile Information
- Configuring Preferred Oracle Names Servers
- Configuring Oracle Connection Manager Features
- 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 listener 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`.

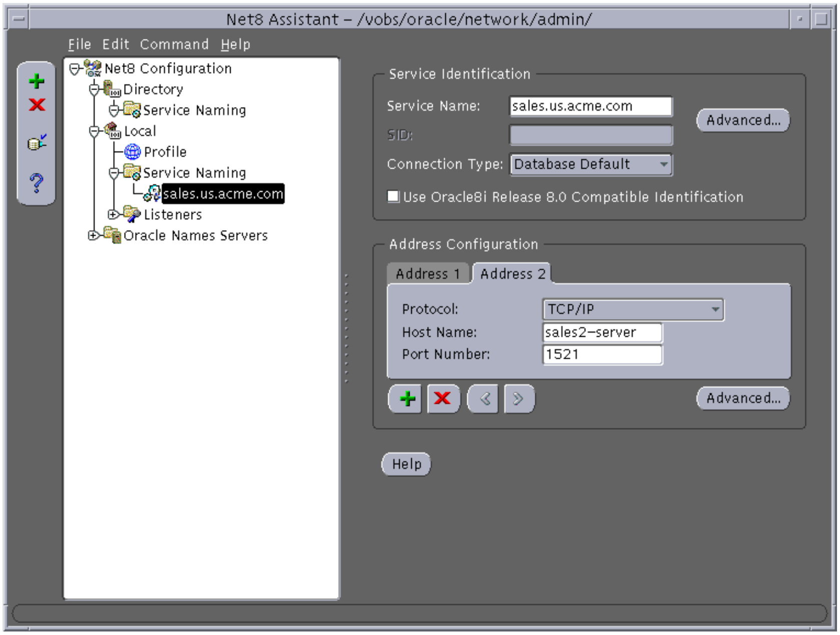
```
sales.us.acme.com=
 (description=
  (address_list=
    (address=(protocol=tcp)(host=sales1-server)(port=1521))
    (address=(protocol=tcp)(host=sales2-server)(port=1521)))
  (connect_data=
    (service_name=sales.us.acme.com)))
```

To add a network protocol address to an existing net service name or database service:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Directory or Local > Service Naming.
3. Select either a 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 "+".

A new address tab appears:



Service Name: sales.us.acme.com

SID:

Connection Type: Database Default

□ Use Oracle8i Release 8.0 Compatible Identification

Protocol: TCP/IP

Host Name: sales2–server

Port Number: 1521

5. Select the protocol and enter appropriate address information.

   See Also: Appendix B for details about protocol address parameters

6. Order the protocol addresses according to where they should be in the 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 8-4 for address list options

7. Click Apply.

Configuring Address List Parameters

When a database service is accessible by multiple listener protocol addresses, it is important to specify the order of how the addresses are to be used. For example, the addresses can be chosen randomly or tried sequentially.

When multiple protocol addresses have been configured for a net service name or database service, the following options may be configured:

Parameter	Description
SOURCE_ROUTE (Source Routing	Instructs Net8 to use each address in the order presented until the destination reached when set to ON. This parameter is required when reaching the destination requires more than one address stop. This feature is typically used to enable Oracle Connection Manager features.
FAILOVER (Connect-Time Failover)	At connect time, instructs Net8 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 Net8 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:** Do not set the GLOBAL_DBNAME parameter in the SID_LIST_*listener_name* section of the `listener.ora` file. A statically configured global database name disables connect-time failover.
LOAD_BALANCE (Client Load Balancing)	When set to ON, instructs Net8 to progress through the list of listener addresses in a random sequence, balancing the load on the various listeners. When set to OFF, instructs Net8 to try the addresses sequentially until one succeeds. Client load balancing is turned ON by default for multiple connect descriptors (DESCRIPTION_LIST).

Note: Connect-time-failover and client-load-balancing parameters are available only for release 8.1 configurations. Source routing is available for both pre-release 8.1 and release 8.0 configurations.

It is not possible to set client load balancing or connect-time failover with source routing. While connect-time failover and client load balancing choose an address from a list, source routing connects to each address in the list sequentially.

Implementation of these parameter depends on the naming method used. The following table describes the tools to configure connect-time failover or client load balancing. Source routing involves other configuration that goes beyond the scope of this section.

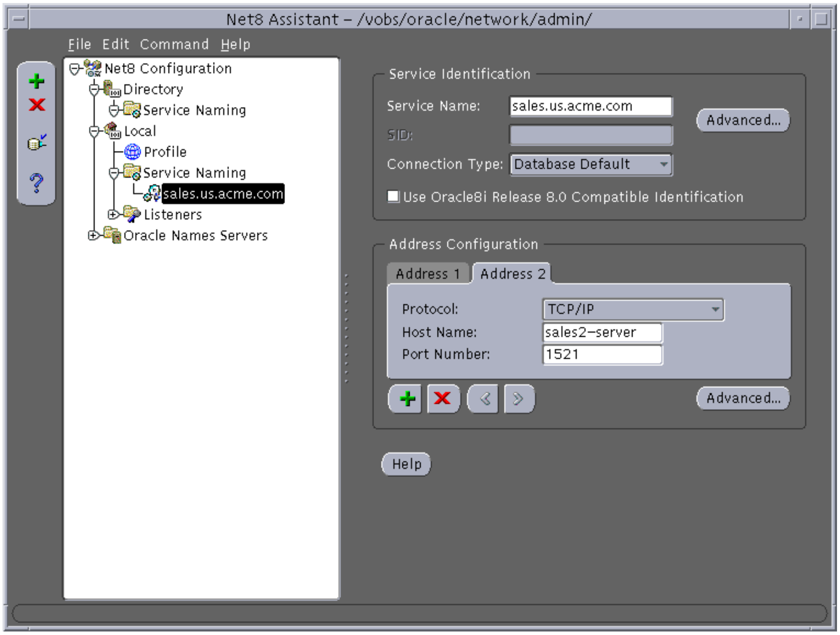
> **See Also:** "Configuring Clients for Oracle Connection Manager" on page 8-38 for more information about configuring clients for source routing

Naming Method	Tool to Configure Multiple Address Options
Local Naming	Net service names stored in a `tnsnames.ora` file may be configured for connect-time failover and client load balancing using Net8 Assistant. See the procedure that follows.
Directory Naming	Net service names and database services stored in a directory may be configured for connect-time failover and client load balancing using Net8 Assistant. See the procedure that follows.
Oracle Names	Net service names stored in an Oracle Names server may be configured for connect-time failover and client load balancing using the following tools: ■ NAMESCTL REGISTER ■ Net8 Assistant **See Also:** "Task 6: Register Data with the Oracle Names Server" on page 6-65

To configure connect-time failover or client load balancing:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Directory or Local > Service Naming.
3. Select either a net service name or a database service.

The right pane displays the current destination service and address list:



Net8 Assistant – /vobs/oracle/network/admin/

Service Identification

Service Name: sales.us.acme.com

SID:

Connection Type: Database Default

□ Use Oracle8i Release 8.0 Compatible Identification

Address Configuration

Address 1 | Address 2

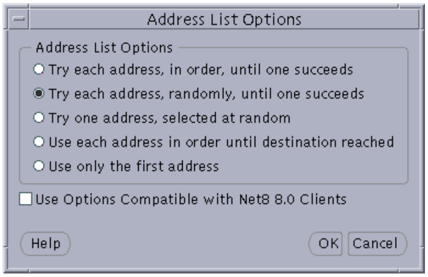
Protocol: TCP/IP

Host Name: sales2–server

Port Number: 1521

4. In the Address Configuration box, click Advanced.

   The Address List Options dialog box appears:



Address List Options

Address List Options

○ Try each address, in order, until one succeeds

● Try each address, randomly, until one succeeds

○ Try one address, selected at random

○ Use each address in order until destination reached

○ Use only the first address

□ Use Options Compatible with Net8 8.0 Clients

The following table describes the options:

Net8 Assistant Option	Parameter

Try each address, in order, until one succeeds	FAILOVER=ON for release 8.1 clients
	SOURCE_ROUTE=OFF for pre-release 8.1 clients.

Try each address, randomly, until one succeeds
Note: This option is not enabled if "Use Options Compatible with Net8 8.0 Clients" is selected.
	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.	LOAD_BALANCE=ON
Use each address in order until destination reached	SOURCE_ROUTE=ON

Use only the first address
Note: This option is not enabled if "Use Options Compatible with Net8 8.0 Clients" is selected.
	LOAD_BALANCE=OFF
	FAILOVER=OFF
	SOURCE_ROUTE=OFF

5. Select one of the address list options, then click OK.
6. In the right pane, click Apply.

In the following example, the `tnsnames.ora` file is configured for client load balancing with the "Try each address, randomly, until one succeeds" option:

```
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)))
```

In the following example, the `tnsnames.ora` file is configured for connect-time failover with the "Try each address, in order, until one succeeds" option:

```
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 portion 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, connect data information may be optionally configured with the following parameters:

tnsnames.ora File Parameter	Net8 Assistant Field/Option	Description
INSTANCE_NAME	Instance Name	Identifies the database instance to access The instance name can be obtained from the INSTANCE_ NAME parameter in the **initialization parameter file**. **Note:** Not enabled if "Use Oracle8i Release 8.0 Compatible Identification" is checked. **See Also:** "Understanding Connect Descriptors" on page 6-2
SDU	Session Data Unit Size	If you want 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:** "Adjusting the Session Data Unit (SDU) Size for Data Transfer Optimization" on page 5-8
HS	Use for Heterogeneous Services	If you want an Oracle8*i* server to access a non-Oracle system, turn this option on. **See Also:** "Configuring Net8 for Oracle Heterogeneous Services" on page 8-49
Oracle Rdb Settings		
RDB_DATABASE)	Oracle RDB Database	Specifies the file name of an Oracle RDB database **See Also:** "Configuring Net8 for an Oracle Rdb Database" on page 8-52
TYPE_OF_SERVICE)	Type of Service	Specifies the type of service to use for an Oracle RDB database **See Also:** "Configuring Net8 for an Oracle Rdb Database" on page 8-52
GLOBAL_NAME)	Global Database Name	Identifies the Oracle Rdb database **See Also:** "Configuring Net8 for an Oracle Rdb Database" on page 8-52

Implementation of these parameter depends on the naming method used. The following table describes the tools to configure connect-time failover or client load balancing. Source routing involves other configuration that goes beyond the scope of this section.

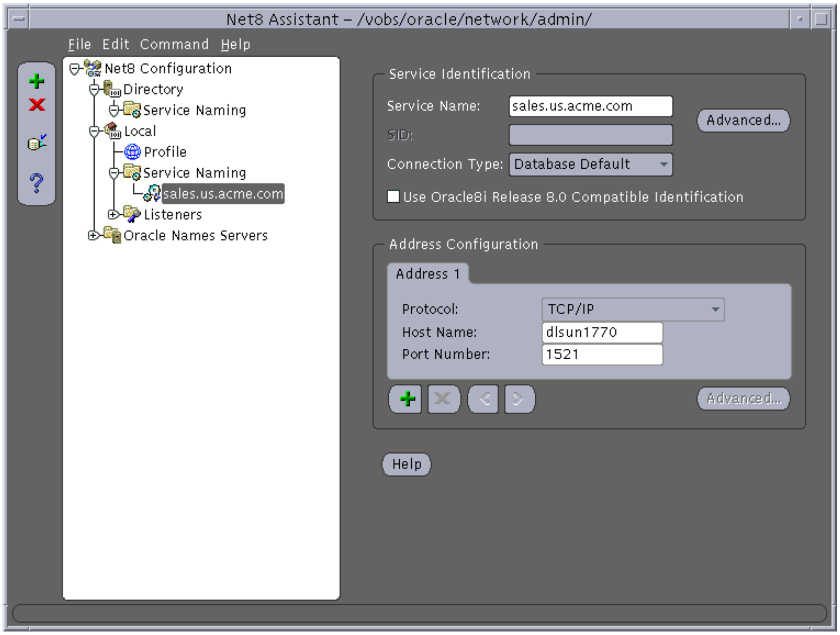
> **See Also:** "Configuring Clients for Oracle Connection Manager" on page 8-38 for more information about configuring clients for source routing

Naming Method	Tool to Configure Multiple Address Options
Local Naming	Net service names stored in a `tnsnames.ora` file may be configured with advanced connect data information using Net8 Assistant. See the procedure that follows.
Directory Naming	Net service names and database services stored in a directory may be configured with advanced connect data information using Net8 Assistant. See the procedure that follows.
Oracle Names	Net service names and database services stored in an Oracle Names server may be configured with advanced connect data information using the following tools: ■ NAMESCTL REGISTER ■ Net8 Assistant **See Also:** See "Task 6: Register Data with the Oracle Names Server" on page 6-65

To configure advanced CONNECT_DATA parameters for either a net service name or a database service:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
   - Network Administration > Net8 Assistant.
2. In the navigator pane, expand Directory or Local > Service Naming.
3. Select either a net service name or a database service.

The right pane displays the current destination service and address list:



Service Name: sales.us.acme.com

SID:

Connection Type: Database Default

□ Use Oracle8i Release 8.0 Compatible Identification

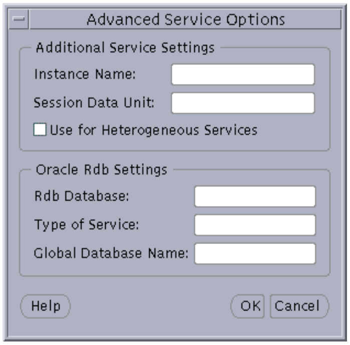
Protocol: TCP/IP

Host Name: dlsun1770

Port Number: 1521

4. In the Service Identification box, click Advanced.

The Advanced Service Options dialog box appears:



5. Enter fields or select options as appropriate, then click OK.
6. In the right pane, click Apply.

Configuring Transparent Application Failover

Transparent application failover (TAF) instructs Net8 to fail over an established connection that has failed 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 portion of the connect descriptor.

This sections covers the following topics:

- FAILOVER_MODE Parameters
- TAF Implementation
- Verification

FAILOVER_MODE Parameters

The FAILOVER_MODE parameter must be included in the CONNECT_DATA portion of a connect descriptor. FAILOVER_MODE may contain the following parameters:

tnsnames.ora file Parameter	Description
BACKUP	Specifies a different net service name for backup connections. A backup should be specified when using PRECONNECT to pre-establish connections.
TYPE **(Required)**	Specifies the type of failover. Three types of Net8 failover functionality are available by default to Oracle Call Interface (OCI) applications: ▪ SESSION: Fails over the session; that is, 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: Enables 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, in which no failover functionality is used. This can also be explicitly specified to prevent failover from happening.
METHOD	Determines how fast failover occurs from the primary node to the backup node: ▪ BASIC: Establishes connections at failover time. This option requires almost no work on the backup server until failover time. ▪ PRECONNECT: Pre-establishes connections. This provides faster failover but requires that the backup instance be able to support all connections from every supported instance.
RETRIES	Specifies the number of times to attempt to connect. If DELAY is specified, RETRIES defaults to five retry attempts.
DELAY	Specifies the amount of time in seconds to wait between connect attempts. If RETRIES is specified, DELAY defaults to one second.

Note: Net8 Assistant 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, TAF can be implemented in a number of ways. Oracle recommends the following methods:

- Implementing TAF with Connect-Time Failover and Client Load Balancing
- Retrying a Connection
- Pre-Establishing a Connection

Implementing TAF with Connect-Time Failover and Client Load Balancing

TAF can be implemented with connect-time failover and client load balancing for multiple addresses. In the following example, Net8 connects randomly to one of the listener addresses on `sales1-server` or `sales2-server`. If the instance fails after the connection, Net8 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))))
```

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, Net8 tries to connect to the listener on `sales1-server`. If the connection attempt fails, Net8 waits 15 seconds before trying to connect again. Net8 attempts to connect 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))))
```

Pre-Establishing a Connection

A backup connection can be pre-established. The initial and backup connections must be explicitly specified. In the following example, Net8 connects to the listener on `sales1-server`. If `sales1-server` fails after the connection, Net8 fails over to `sales2-server`, reserving any SELECT statements in progress.

```
sales.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.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_name=sales2)))
```

Verification

You can query FAILOVER_TYPE, FAILOVER_METHOD, and FAILED_OVER columns in the V$SESSION view to verify that TAF is correctly configured.

> **See Also:** *Oracle8i Reference* for more information about the V$SESSION view

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's `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 or Oracle Names server.

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/tiger@sales
```

In this example, `sales` gets searched as `sales.us.acme.com`.

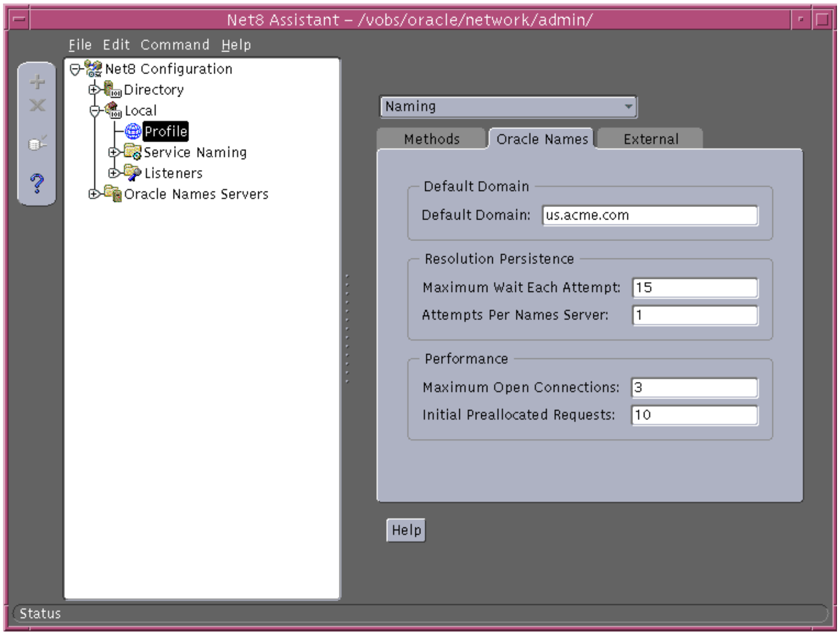
If the connect string includes the domain extension, such as in `CONNECT scott/tiger@sales.acme.com`, the domain is not appended. If a net service name in a `tnsnames.ora` file or Oracle Names server is not domain qualified and this parameter is set, the net service name must be entered with a dot ("."). 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/tiger@sales.
```

Note: During installation, Net8 Configuration Assistant sets the default domain to your client's network domain.

To specify a default domain:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Profile.
3. From the list in the right pane, select Naming.
4. Click the Oracle Names tab:



Default Domain: us.acme.com

Maximum Wait Each Attempt: 15

Attempts Per Names Server: 1

Maximum Open Connections: 3

Initial Preallocated Requests: 10

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:

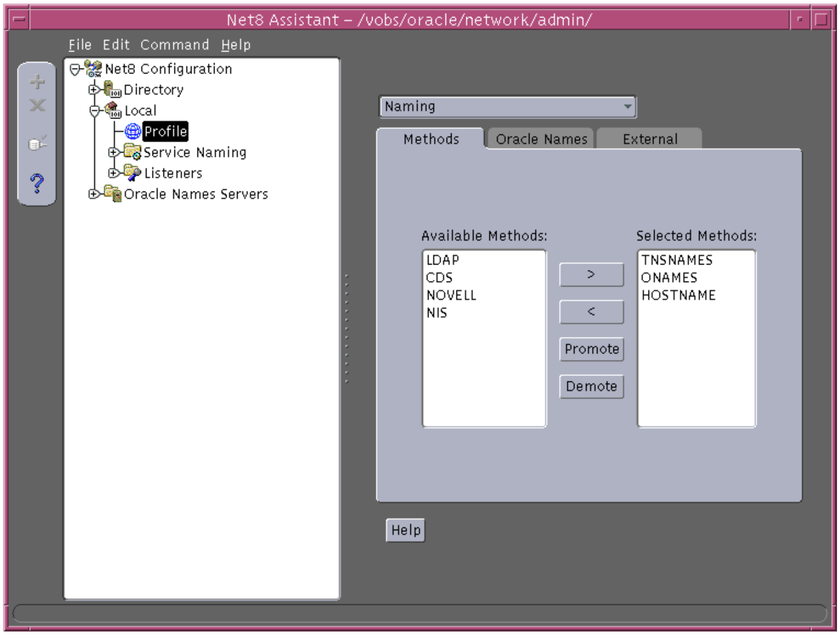
```
names.default_domain=com
```

Prioritizing Naming Methods

After naming methods are configured, as described in Chapter 6, they must be prioritized. The naming method at the top of the list is used first to resolve a net service name. If it cannot resolve the net service name, the second method in the list is used to resolve the net service name.

To specify the order of naming methods:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Profile.
3. From the list in the right pane, select Naming.
4. Click the Methods tab:



Naming

Available Methods:
LDAP
CDS
NOVELL
NIS

Selected Methods:
TNSNAMES
ONAMES
HOSTNAME

Available naming methods are described in the following table:

Naming Method	Description
TNSNAMES	Local Naming Net service names are resolved using the `tnsnames.ora` file which resides on the client.
LDAP	Directory Naming Net service names and database service names are resolved through a directory server.
ONAMES	Oracle Names Net service names and database service names are resolved centrally, through an Oracle Names server.
HOSTNAME	Host Naming Net service names are resolved using the host naming method. Certain criteria must be met to use host naming. **See Also:** "Configuring the Host Naming Method" on page 6-74
CDS	Cell Directory Services This naming method is available with Oracle Advanced Security option. **See Also:** *Oracle Advanced Security Administrator's Guide*
NDS	Novell Directory Services (NDS)
NIS	Network Information Service (NIS)

5. Choose from the Available Methods list which naming method you want to use.
6. Select naming methods from the Available Methods list, then click the right-arrow button.

   The naming method moves to the Selected Methods list.
7. Order the naming methods according to the order in which you want Net8 to try to resolve the net service name or database service name. Select a naming method in the Selected Methods list, and click Promote or Demote to move the selection up or down in the list.
8. Choose File > Save Network Configuration.

   The `sqlnet.ora` file updates with the NAMES.DIRECTORY_PATH parameter:

   ```
   names.directory_path=(onames, tnsnames)
   ```

Routing Connection Requests

Clients and servers acting as clients can be configured so connection requests are directed to a specific process at all times. The `sqlnet.ora` file parameters described in the following table can be used to route connection requests:

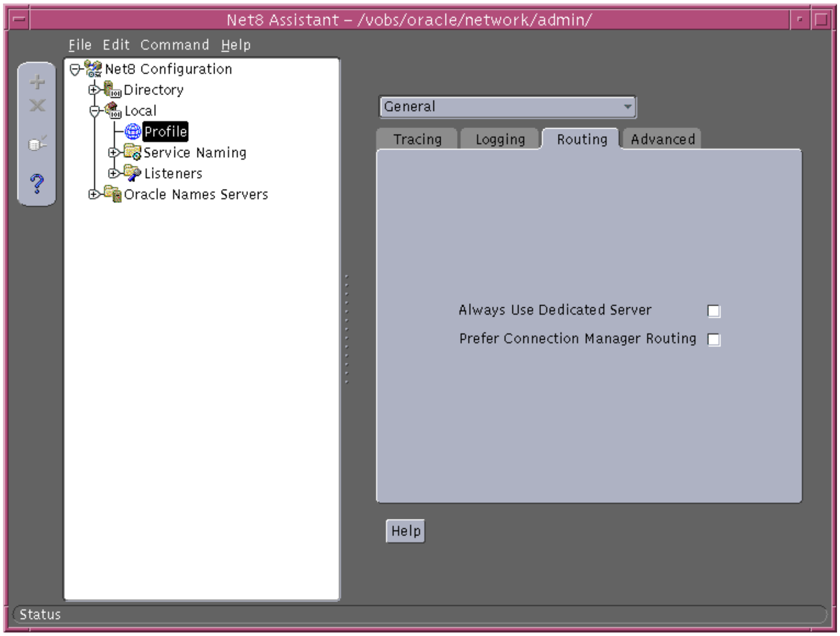
sqlnet.ora File Parameter	Net8 Assistant Option	Description
USE_DEDICATED_SERVER	Always Use Dedicated Server	Forces the listener to spawn a dedicated server for all network session from this client. It does this by adding (SERVER=DEDICATED) to the CONNECT_DATA portion of connect descriptors. This way, connections from this client use dedicated servers, even if **multi-threaded server (MTS)** is configured.
USE_CMAN	Prefer Connection Manager Routing	Instructs the client to an Oracle Connection Manager whenever possible. If no Oracle Connection Manager addresses are available, connections are routed through any available listener address. **Note:** If you are using Oracle Connection Manager with Oracle Names, this option must be set on clients and Oracle Names servers.

Note: While Net8 Assistant displays the Use IPC Address for Client, this feature is not enabled because its corresponding parameter, AUTOMATIC_IPC, has been disabled for release 8.1.

To route connection requests:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle **-** *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Profile.
3. From the list in the right pane, select General.

4. Click the Routing tab:



Net8 Assistant - /vobs/oracle/network/admin/

Always Use Dedicated Server □

Prefer Connection Manager Routing □

5. Click the preferred way you want connection requests routed.

6. Choose File > Save Network Configuration.

Configuring Advanced Profile Information

The `sqlnet.ora` file can be configured with the following advanced parameters:

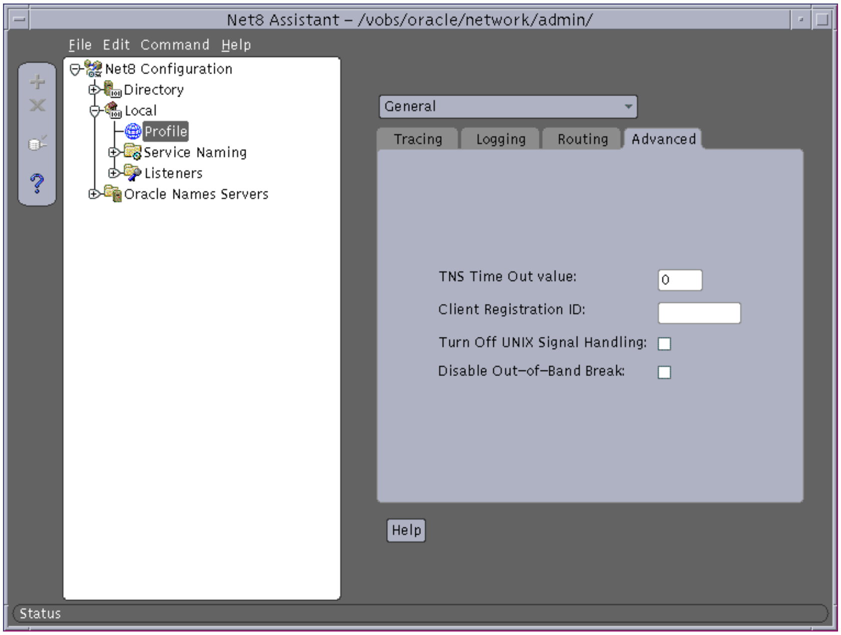
sqlnet.ora File Parameter	Net8 Assistant Field/Option	Description
SQLNET.EXPIRE_TIME	TNS Time Out Value	This option can be used only on the server. Sets a dead connection time-out value. Net8 sends a probe periodically to verify that a client-server connection is still active. This ensures that connections are not left open indefinitely, due to an abnormal client termination. If the probe finds a dead connection, or a connection that is no longer in use, it returns an error, causing the server process to exit. Limitations on using the dead connection detection feature are: ■ Dead connection detection is not allowed on bequeathed connections. ■ Though very small, a probe packet generates additional traffic that may downgrade network performance. ■ The server may need to perform additional processing to distinguish the connection probing event from other events that occur, depending on which operating system is in use. This may also result in downgrading network performance.
SQLNET.CLIENT_ REGISTRATION	Client Registration ID	Sets a unique identifier for this client machine. This identifier is passed to the listener with any connection request. The identifier can be any string up to 128 characters long.
BEQUEATH_DETACH	Turn 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 init process by disabling signal handlers.

sqlnet.ora File Parameter	Net8 Assistant Field/Option	Description
DISABLE_OOB	Disable Out-of-Band Break	If deselected or turned OFF, enables Net8 to send and receive "break" messages using urgent data provided by the underlying protocol. If selected or turned 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 documentation to determine if the protocols you are using supports urgent data requests. TCP/IP is an example of a protocol that supports this feature.

To set advanced features for clients:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Profile.
3. From the list in the right pane, select General.

4. Click the Advanced tab:



TNS Time Out value: 0

Client Registration ID:

Turn Off UNIX Signal Handling: □

Disable Out-of-Band Break: □

5. Enter the values for the option(s) you would like to set.

6. Choose File > Save Network Configuration.

Configuring Preferred Oracle Names Servers

If you are using Oracle Names as a naming method, you can specify the name and address of **preferred Oracle Names servers** to take precedence over any other available Oracle Names server addresses. Net8 routes name resolution requests to each preferred Oracle Names Server until a response is received.

Preferred Oracle Names servers are an alternative to using the Discover Oracle Names Servers command in Net8 Assistant or the REORDER_NS command in the NAMESCTL control utility. These commands create a list of Oracle Names servers based on what is available throughout the network and ranks them in the order of fastest response time. Preferred Oracle Names servers override any other Oracle Names servers found during the discovery process. Once you have initially discovered an Oracle Names server, you may want to delete preferred Oracle Names servers.

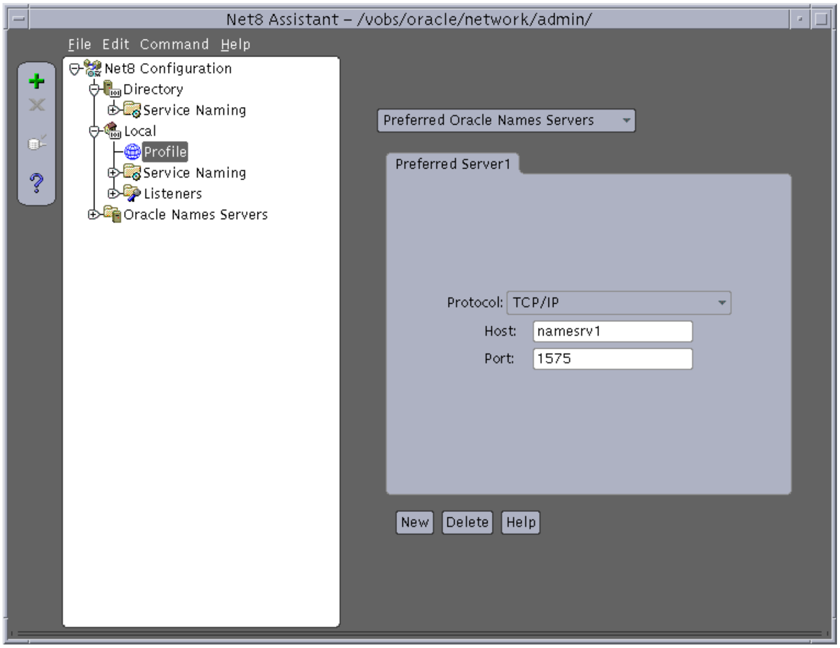
See Also:

- "Differences Between Versions of Oracle Names" on page 4-21 for a description of how Oracle Names server lists are created
- "Task 4: Configure Clients and Database Servers To Use Oracle Names Servers" on page 6-60 for information about discovering Oracle Names servers

To specify a preferred Oracle Names server:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Local > Profile.
3. From the list in the right pane, select Preferred Oracle Names Servers.

4. Click New:



Protocol: TCP/IP
Host: namesrv1
Port: 1575

5. Choose the protocol and enter the requested protocol address information for an Oracle Names server configured on that address.

   See Also: Appendix B for protocol parameter settings

6. Repeat Steps 4 and 5 for each additional Oracle Names server you want added to the list.

7. Choose File > Save Network Configuration.

   The `sqlnet.ora` file updates with the NAMES.PREFERRED_SERVERS parameter:

```
names.preferred_servers=
  (address=(protocol=tcp) (host=namesrv1)(port=1575))
```

> **Note:** The preferred Oracle Names server(s) must match the Oracle Names listening protocol address(es) you configured in the `names.ora` file with the NAMES.ADDRESSES parameter.

Configuring Oracle Connection Manager Features

The following sections describe how to configure Oracle Connection Manager features:

- Enabling Connection Concentration
- Enabling Multi-Protocol Support
- Enabling Net8 Access Control
- Configuring Clients for Oracle Connection Manager

Enabling Connection Concentration

Oracle Connection Manager enables you to take advantage of the ability to multiplex or funnel multiple logical client network sessions through a single transport connection to a multi-threaded server destination. This is accomplished through Oracle Connection Manager's connection concentration feature.

To configure connection concentration, perform these tasks:

Task 1: Configure Oracle Connection Manager

Task 2: Configure the Server

Task 3: Configure Clients

Task 1: Configure Oracle Connection Manager

Oracle Connection Manager accepts client connection requests at the following default listening address:

```
cman=(address=(protocol=tcp)(host=anyhost)(port=1630))
```

Oracle Connection Manager also listens for local and remote administration commands at the following listening address:

```
cman_admin=(address=(protocol=tcp)(host=anyhost)(port=1830))
```

If you do not want to use the default addresses, you must create a `cman.ora` file with the following information:

```
cman=(address=(protocol_address_information))
cman_admin=(address=(protocol_address_information))
```

A `cman.ora` file should exist in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows NT that you can edit.

See Also:

- "Oracle Connection Manager Parameters (cman.ora)" on page C-70 for more information about the `cman.ora` file
- Appendix B for more information about protocol syntax needed for address information

Note: Net8 Assistant does not support configuration of the `cman.ora` file, so changes must be made manually.

Task 2: Configure the Server

Verify that the destination server is configured with **multi-threaded server (MTS)** and that the multiplexing option is turned on. MTS and multiplexing may be turned on using Oracle Database Configuration Assistant or by setting the MTS_DISPATCHERS parameter in the **initialization parameter file** with the PROTOCOL and MULIPLEX attributes. For example:

```
mts_dispatchers="(protocol=tcp) (multiplex=on)"
```

Attribute	Description
PROTOCOL (PRO or PROT)	The network protocol (TCP in the example above) for which the dispatcher generates a listening end point.
MULTIPLEX (MUL or MULT)	Used to enable connection concentration. If 1, ON, YES, TRUE, or BOTH is specified, then connection concentration is enabled for both incoming and outgoing network connections. If IN is specified, the connection concentration is enabled for incoming network connections from the client. If OUT is specified, the connection concentration is enabled for outgoing network connections. If 0, NO, OFF, or FALSE is specified, then connection concentration is disabled for both incoming and outgoing network connections.

See Also: Chapter 9 for more information about configuring MTS

Task 3: Configure Clients

Client support is accomplished by creating two addresses, the first for the Oracle Connection Manager and the second for the listener. Configuration varies according to naming method, as described in the following table:

Naming Method	Procedure
Local Naming and Directory Naming	An address list must be created, where the first address must be for the Oracle Connection Manager, and the second address must be for the listener. In order to have the client connect to the Oracle Connection Manager first and have the Oracle Connection Manager then connect to the listener, the SOURCE_ROUTE parameter must be set. **See Also:** "Configuring Clients for Oracle Connection Manager" on page 8-38 to configure the Oracle Connection Manager and listener addresses
Oracle Names	Oracle Names servers are automatically updated with the listener and Oracle Connection Manager addresses. In order for a client using an Oracle Names server to send requests to Oracle Connection Manager, the USE_CMAN=TRUE parameter must be set in the `sqlnet.ora` file on clients and machines where each Oracle Names server resides. To set this parameter, follow the procedure in "Routing Connection Requests" on page 8-21.

Example 8–1 shows a comparison of a regular `tnsnames.ora` file and a `tnsnames.ora` file with an entry to use the Oracle Connection Manager (Oracle Connection Manager entries are shown in boldface text):

Example 8–1 tnsnames.ora File with and without Oracle Connection Manager

```
sales=
   (description=
      (address=
         (protocol=tcp)
         (host=sales-pc)
         (port=1521))
      (connect_data=
            (service_name=sales,com)))
```

```
sales=
   (description=
      (source_route=yes)
      (address=
         (protocol=tcp)
         (host=cman-pc)
         (port=1630))
      (address=
         (protocol=tcp)
         (host=sales-pc)
         (port=1521))
      (connect_data=
         (service_name=sales.com)))
```

tnsnames.ora File Element	Description
`source_route=yes`	Creates a source route of addresses through an Oracle Connection Manager to the destination database

`(address=`
`   (protocol=tcp)`
`   (host=cman-pc)`
`   (port=1630))`
	This first address is to the Oracle Connection Manager. From here, the Oracle Connection Manager connects to the database service through the listener

Enabling Multi-Protocol Support

> **Note:** If more than one Oracle Connection Manager is used in the connection path, you cannot use Oracle Names to connect clients through it.

Oracle Connection Manager also provides multiple protocol support enabling a client and server with different networking protocols to communicate with each other. An Oracle Connection Manager can listen on any protocol that Oracle supports.

Without multi-protocol support, a client that uses SPX cannot connect to a server that uses TCP/IP. If Oracle Connection Manager is configured for TCP/IP, the client can connect to Oracle Connection Manager using SPX, and Oracle Connection Manager can connect to the server using TCP/IP.

To configure multi-protocol support, performs these tasks:

Task 1: Configure Oracle Connection Manager

Task 2: Configure Clients

Task 1: Configure Oracle Connection Manager

Oracle Connection Manager accepts client connection requests at the following default listening address, and:

```
cman=(address=(protocol=tcp)(host=anyhost)(port=1630))
```

listens for local and remote administration commands at the following listening address:

```
cman_admin=(address=(protocol=tcp)(host=anyhost)(port=1830))
```

If you do not want to use the default addresses, you must create a `cman.ora` file with the following information:

```
cman=(address=(protocol_address_information))
cman_admin=(address=(protocol_address_information))
```

A `cman.ora` file should exist in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows NT that you can modify.

See Also:

- "Oracle Connection Manager Parameters (cman.ora)" on page C-70 for more information about the `cman.ora` file
- Appendix B for more information about protocol syntax needed for address information

Note: Net8 Assistant does not support configuration of the `cman.ora` file, so changes must be made manually.

Task 2: Configure Clients

Client support is accomplished by creating two addresses, the first for the Oracle Connection Manager and the second for the listener. Configuration varies according to naming method, as described in the following table:

Naming Method	Procedure
Local Naming and Directory Naming	An address list must be created, where the first address must be for the Oracle Connection Manager, and the second address must be for the listener. In order to have the client connect to the Oracle Connection Manager first and have the Oracle Connection Manager then connect to the listener, the SOURCE_ROUTE parameter must be set. **See Also:** "Configuring Clients for Oracle Connection Manager" on page 8-38 to configure the Oracle Connection Manager and listener addresses
Oracle Names	Oracle Names servers are automatically updated with the listener and Oracle Connection Manager addresses. In order for a client using an Oracle Names server to send requests to Oracle Connection Manager, the USE_CMAN=TRUE parameter must be set in the `sqlnet.ora` file on clients and machines where each Oracle Names server resides. To set this parameter, follow the procedure in "Routing Connection Requests" on page 8-21.

Example 8–2 depicts a client using SPX to connect to an Oracle Connection Manager, and Oracle Connection Manager using TCP/IP to connect to a database server.

Example 8–2 CMAN Parameter Configured to Use Multi-Protocol Support

```
cman=
   (description=
      (source_route=yes)
      (address=
            (protocol=spx)
            (service=cman))
      (address=
            (protocol=tcp)
            (host=sales-pc)
            (port=1521))
      (connect_data=
            (service_name=sales.com)))
```

tnsnames.ora File Element	Description
source_route=yes	Creates a source route of addresses through an Oracle Connection Manager to the destination database.
(address= (protocol=spx) (service=cman))	This first address is from a client using SPX to the Oracle Connection Manager. From here, the Oracle Connection Manager connects to the database service using TCP/IP through the listener.

Enabling Net8 Access Control

Oracle Connection Manager also includes a feature which you can use to control client access to designated servers in a TCP/IP environment. By specifying certain filtering rules, you may allow or restrict specific clients access to a server.

Use of this feature depends on whether you want to use connection concentration and multi-protocol support features with or without filtering.

This feature requires the release 8.1 Oracle Connection Manager if there are release 8.1 services in the network.

To configure Net8 access control, perform these tasks:

Task 1: Configure Oracle Connection Manager

Task 2: Configure Clients

Task 1: Configure Oracle Connection Manager

To configure the Oracle Connection Manager:

1. Manually create a `cman.ora` file, if one does not already exist. Net8 Assistant does not support configuration of this file.

   A `cman.ora` file should exist in `$ORACLE_HOME/network/admin` on UNIX and *ORACLE_HOME*`\network\admin` on Windows NT that you can edit.

   See Also:

   - "Oracle Connection Manager Parameters (cman.ora)" on page C-70 for more information about the `cman.ora` file
   - Appendix B for more information about protocol syntax needed for address information

2. If you do not want to use the default listening port of 1630 for client connections, add the following:

   ```
   cman=(address=(protocol=tcp)(host=anyhost)(port=port))
   ```

3. If you do not want to use the default listening port of 1830 for administration commands, add the following:

   ```
   cman=(address=(protocol=tcp)(host=anyhost)(port=port))
   ```

4. Add the CMAN_RULES parameter with the following parameters:

```
cman_rules=
 (rule_list=
   (rule=
     (src=shost)
     (dst=dhost)
     (srv=service)
     (act=accept | reject)))
```

Parameter	Description
SRC	Source host name or IP address (client)
DST	Destination host name or IP address (server)
SRV	Service name of the release 8.1 database (obtained from the SERVICE_NAME parameter in the **initialization parameter file**) or SID value of the pre-release 8.1 database (obtained from the ORACLE_SID environment variable or registry value)
ACT	Accept or reject the incoming requests based on the above three parameters.

Multiple RULEs can be defined within the RULE_LIST. The action (ACT) in the first matched RULE is applied to the request. If the CMAN_RULES are not defined, then all connections are permitted.

The following example shows restriction to service `sales.us.acme.com` for a client machine of `client1-pc`, and access to service `db1` for client `144.25.23.45`.

```
cman_rules=
 (rule_list=
   (rule=(src=client1-pc)(dst=sales-pc)(srv=sales.us.acme.com)(act=reject))
    (rule=(src=144.25.23.45)(dst=144.25.187.200)(srv=db1)(act=accept)))
```

Task 2: Configure Clients

Client support is accomplished by creating two addresses, the first for the Oracle Connection Manager and the second for the listener. Configuration varies according to naming method, as described in the following table:

Naming Method	Procedure
Local Naming and Directory Naming	An address list must be created, where the first address must be for the Oracle Connection Manager, and the second address must be for the listener. In order to have the client connect to the Oracle Connection Manager first and have the Oracle Connection Manager then connect to the listener, the SOURCE_ ROUTE parameter must be set. **See Also:** "Configuring Clients for Oracle Connection Manager" on page 8-38 to configure the Oracle Connection Manager and listener addresses
Oracle Names	Oracle Names servers are automatically updated with the listener and Oracle Connection Manager addresses. In order for a client using an Oracle Names server to send requests to Oracle Connection Manager, the USE_CMAN=TRUE parameter must be set in the `sqlnet.ora` file on clients and machines where each Oracle Names server resides. To set this parameter, follow the procedure in "Routing Connection Requests" on page 8-21.

Configuring Clients for Oracle Connection Manager

> **Note:** If more than one Oracle Connection Manager is used in the connection path, you cannot use Oracle Names to connect clients through it.

Configuring the client involves routing client connection requests that you want concentrated to the database server through an Oracle Connection Manager. This is achieved by setting the Oracle Connection Manager and listener protocol addresses in a `tnsnames.ora` file, directory or an Oracle Names server. Take note of the following:

- If using Oracle Names servers, the Oracle Connection Manager automatically updates the addresses in the Oracle Names servers, inserting the address for the Oracle Connection Manager into the existing addresses. In order for Oracle Names servers to send requests to Oracle Connection Manager, USE_CMAN=TRUE must be set in the `sqlnet.ora` file on machines where each Oracle Names server resides. To set this parameter, follow the procedure in "Routing Connection Requests" on page 8-21.
- If using a `tnsnames.ora` file or a directory, the protocol addresses and the SOURCE_ROUTE parameter must be set. To configure the local or directory naming methods for Oracle Connection Manager, perform these tasks:

  Task 1: Configure an Oracle Connection Manager Address

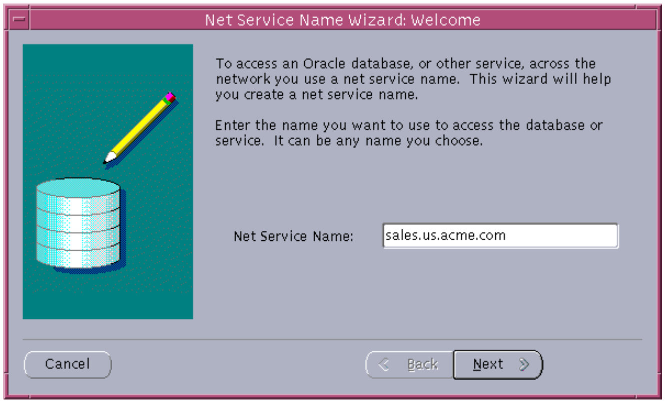
  Task 2: Configure the Listener Address

Task 1: Configure an Oracle Connection Manager Address

To configure an Oracle Connection Manager address:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. In the navigator pane, expand Directory or Local. Then select Service Naming.
3. Click "+" from the toolbar, or select Create from the Edit menu.

   The Net Service Name Wizard starts.
4. Enter any name in the Net Service Name field, then click Next:



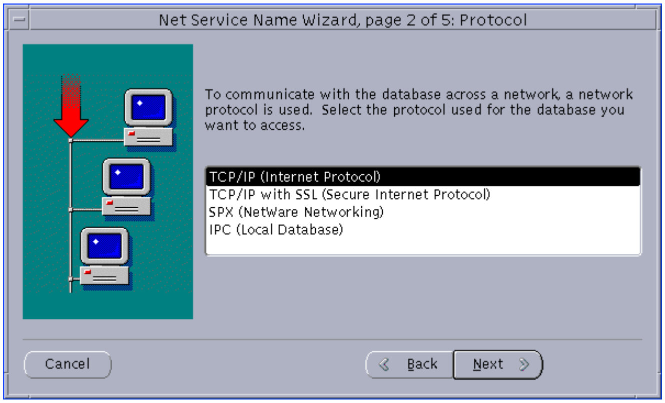
Net Service Name Wizard: Welcome

To access an Oracle database, or other service, across the network you use a net service name. This wizard will help you create a net service name.

Enter the name you want to use to access the database or service. It can be any name you choose.

Net Service Name: sales.us.acme.com

5. Select the protocol the Oracle Connection Manager is configured to listen on, then click Next. By default this protocol is TCP/IP.

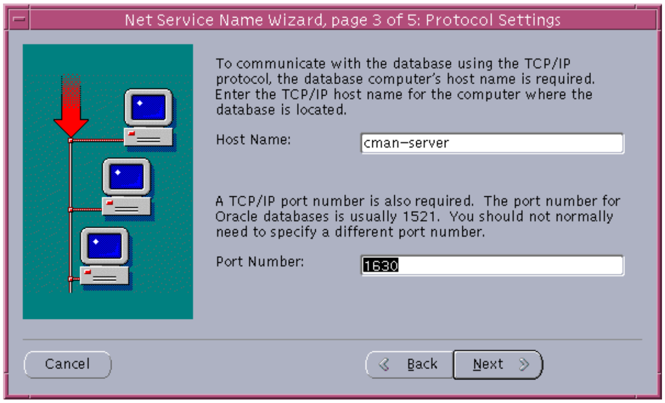


Net Service Name Wizard, page 2 of 5: Protocol

To communicate with the database across a network, a network protocol is used. Select the protocol used for the database you want to access.

- TCP/IP (Internet Protocol)
- TCP/IP with SSL (Secure Internet Protocol)
- SPX (NetWare Networking)
- IPC (Local Database)

6. Enter network protocol information for the Oracle Connection Manager, then click Next. If you are using TCP/IP, the default port is 1630.



Net Service Name Wizard, page 3 of 5: Protocol Settings

To communicate with the database using the TCP/IP protocol, the database computer's host name is required. Enter the TCP/IP host name for the computer where the database is located.

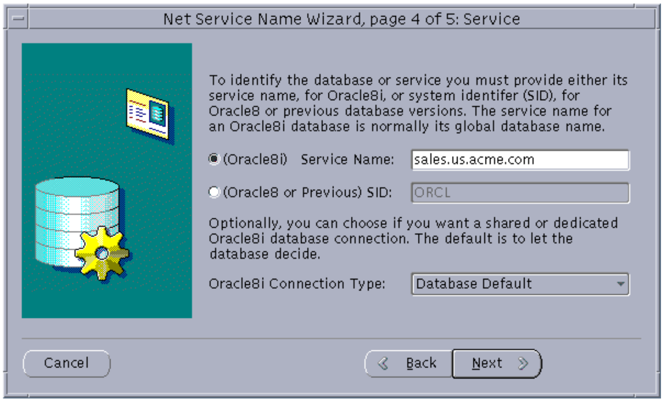
Host Name: cman-server

A TCP/IP port number is also required. The port number for Oracle databases is usually 1521. You should not normally need to specify a different port number.

Port Number: 1630

See Also: Appendix B for protocol parameter settings

7. Select a release, enter a destination service, then click Next:



Net Service Name Wizard, page 4 of 5: Service

To identify the database or service you must provide either its service name, for Oracle8i, or system identifier (SID), for Oracle8 or previous database versions. The service name for an Oracle8i database is normally its global database name.

● (Oracle8i) Service Name: sales.us.acme.com

○ (Oracle8 or Previous) SID: ORCL

Optionally, you can choose if you want a shared or dedicated Oracle8i database connection. The default is to let the database decide.

Oracle8i Connection Type: Database Default

If the destination service is Oracle release 8.1, click (Oracle8i), and enter a service name in the Service Name field. If destination service is Oracle release 8.0 or version 7 database, click (Oracle8 or Previous), and enter an Oracle System Identifier (SID) in the Database SID field.

> **See Also:** "Understanding Connect Descriptors" on page 6-2 for further information about setting the service name string

8. Click Next.

> **Note:** Do not click Test, as a connection cannot be tested at this point.

9. Click Finish to save your configuration and dismiss Net Service Name Wizard.

   The new net service name and the Oracle Connection Manager address is added to the Service Naming folder.

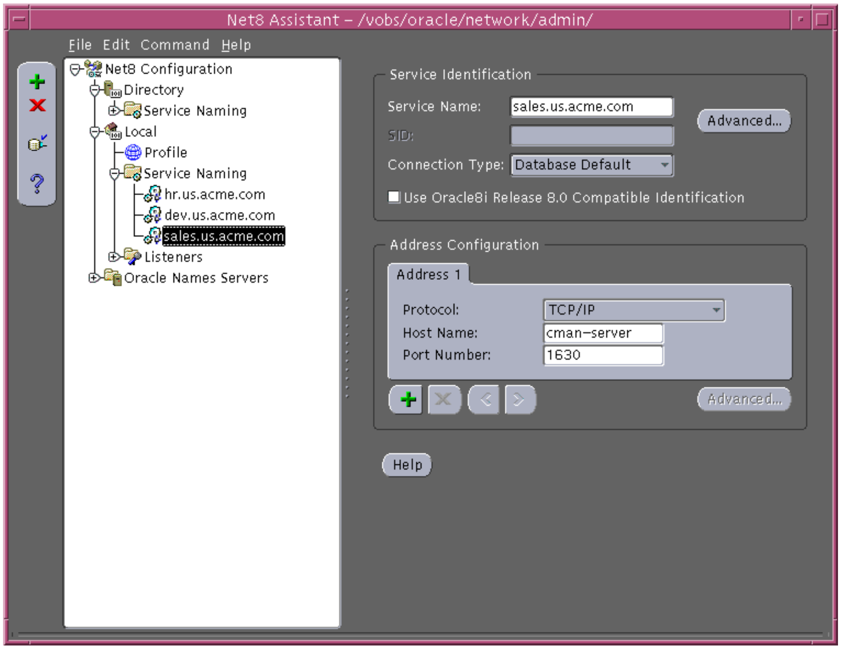
Task 2: Configure the Listener Address

After the Oracle Connection Manager address is specified, create an address for the listener, so that the Oracle Connection Manager can connect to the server.

To configure the listener address:

1. Click the net service name you created in "Task 1: Configure an Oracle Connection Manager Address" on page 8-39.

   Notice the Oracle Connection Manager address is displayed in the Address 1 tab:



Service Identification

Service Name: sales.us.acme.com

SID:

Connection Type: Database Default

□ Use Oracle8i Release 8.0 Compatible Identification

Address Configuration

Address 1

Protocol: TCP/IP

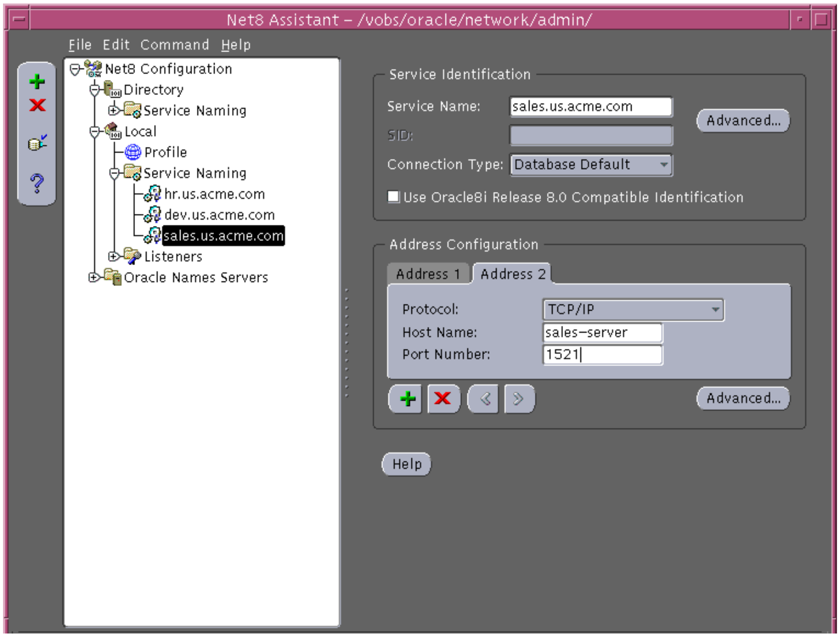
Host Name: cman–server

Port Number: 1630

2. Click "+".

   A new address tab displays.

3. Select the protocol, then enter protocol information for the listener. If you are using TCP/IP, the default port number is 1521:



Service Name: sales.us.acme.com

Connection Type: Database Default

□ Use Oracle8i Release 8.0 Compatible Identification

Protocol: TCP/IP

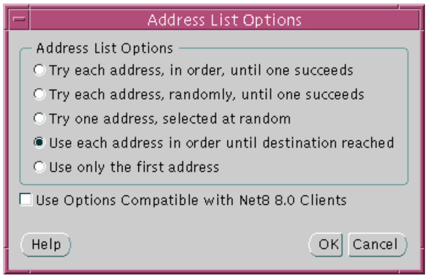
Host Name: sales−server

Port Number: 1521

See Also: Appendix B for protocol parameter settings

4. In the Address Configuration box, click Advanced.

The Address List Options dialog box appears:



Address List Options

○ Try each address, in order, until one succeeds
○ Try each address, randomly, until one succeeds
○ Try one address, selected at random
● Use each address in order until destination reached
○ Use only the first address

□ Use Options Compatible with Net8 8.0 Clients

5. Select "Use each address in order until destination reached", then click OK.

This option sets SOURCE_ROUTE=ON. It tells the client to connect to the first address, the Oracle Connection Manager, and from the first address to the second address, the listener.

> **Note:** There is not an option that supports client load balancing and connect-time failover options with Oracle Connection Manager's source routing feature within the same address list. When SOURCE_ROUTE is set, the first address is for the client connection to an Oracle Connection Manager and the second address is for the Oracle Connection Manager connection to a listener. Client load balancing and connect-time failover are disabled when SOURCE_ROUTE is set.
>
> It is possible to manually configure client load balancing and connect-time failover with source routing for multiple address lists. See "Examples" on page C-35.

6. In the right pane, click Apply.

Configuring Connections to Non-Oracle Database Services

The following sections describes how to configure connections to non-Oracle database services:

- Configuring Net8 for External Procedures
- Configuring Net8 for Oracle Heterogeneous Services
- Configuring Net8 for an Oracle Rdb Database

Configuring Net8 for External Procedures

External procedures are functions or procedures written in a third-generation language (3GL) that can be called from PL/SQL code.

The listener can be configured to listen for external procedure calls. When a PL/SQL or SQL application calls an external procedure, the listener launches a network session-specific process called `extproc`. Through the listener service, PL/SQL passes the following information to `extproc`:

- Shared library name
- External procedure name
- Parameters (if necessary)

`extproc` then loads the shared library and invokes the external procedure.

By default, Net8 Configuration Assistant configures connections to external procedure during server installation. For environments where the connection information do not exist, edit the `listener.ora` file and `tnsnames.ora` files, located in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows platforms, as follows:

1. Configure either a TCP/IP or IPC listening address in the `listener.ora` file. The following procedures describes creating an IPC address:

   a. Start Net8 Assistant:

      -On UNIX, run `netasst` from `$ORACLE_HOME/bin`.

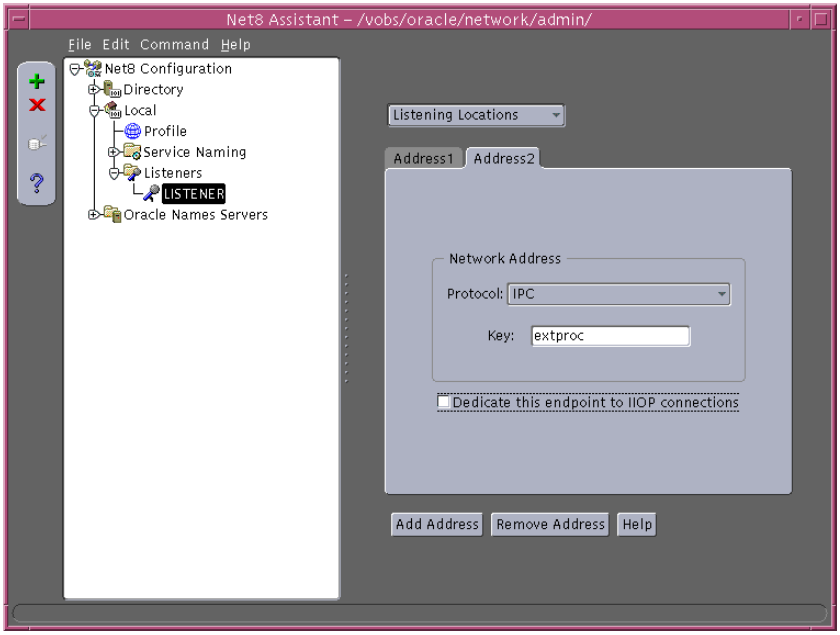
      -On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

   b. In the navigator pane, expand Local > Listeners.

   c. Select a listener.

d. From the list in the right pane, select Listening Locations.

e. Click Add Address. A new address tab appears.



Protocol: IPC

Key: extproc

□ Dedicate this endpoint to IIOP connections

f. Select IPC from the Protocol list, and enter a value for the KEY. Oracle Corporation recommends a KEY value of `extproc`.

Note: If your machine has more than one Oracle home or more than one listener, each listener must specify a unique KEY. Oracle Corporation recommends using `extproc1` for the first listener, `extproc2` for the second listener, and so on.

See Also: "Configuring Listener Protocol Addresses" on page 7-6 for more information about configuring listener protocol addresses

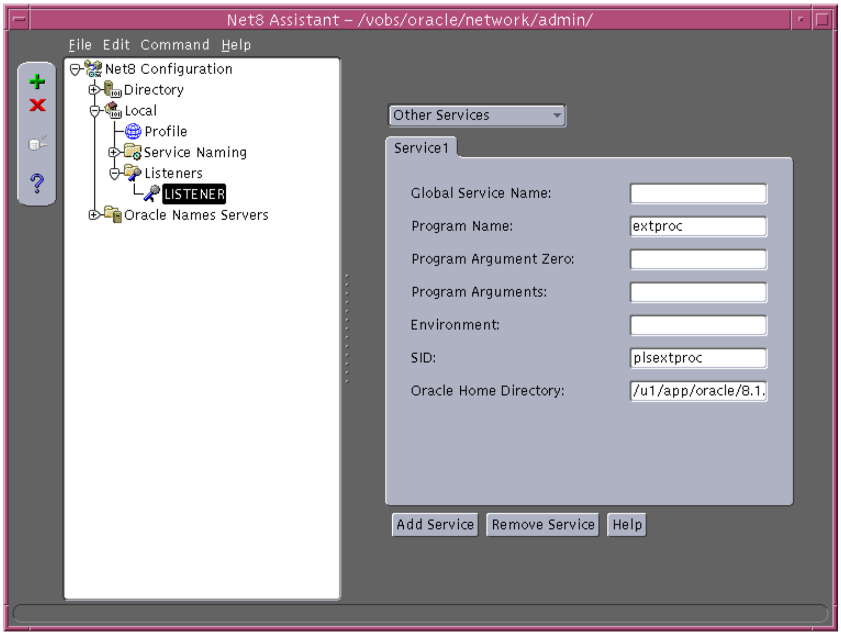
2. Add service information about `extproc` in the `listener.ora` file, including the following parameters:

listener.ora Parameter	Net8 Assistant Field	Description
SID_NAME	SID	Specifies the Oracle System Identifier (SID) for the non-Oracle database
ORACLE_HOME	Oracle Home Directory	Specifies the Oracle home location of the `extproc` program
PROGRAM	Program Name	Specifies the name of the `extproc` executable

To configure the SID_NAME, ORACLE_HOME, and PROGRAM parameters:

a. From the list in the right pane, select Other Services.

b. Click Add Service. A new Service tab appears:



Global Service Name:

Program Name: extproc

Program Argument Zero:

Program Arguments:

Environment:

SID: plsextproc

Oracle Home Directory: /u1/app/oracle/8.1.

c. Enter `extproc` in the Program Name field, a SID, such as `plsextproc`, in the SID field, and the Oracle home where the `extproc` executable resides in the Oracle Home field,

d. Choose File > Save Network Configuration.

The `listener.ora` file updates with external procedures, as shown in the following:

```
listener=
  (address=
     (protocol=ipc)
     (key=extproc))
sid_list_listener
  (sid_list=
    (sid_desc=
     (sid_name=plsextproc)
     (oracle_home=/u1/app/oracle/8.1.6)
     (program=extproc)))
```

3. Create a net service name in the server's `tnsnames.ora` file that matches the information configured in the `listener.ora` file.
4. Based on the `listener.ora` file shown in Step 2, the `tnsnames.ora` file should be configured with the following entry:

```
extproc_connection_data=
 (description=
   (address=(protocol=ipc)(key=extproc))
   (connect_data=
    (sid=plsextproc)))
```

See Also: "Configuring the Local Naming Method" on page 6-5 for more information about creating a net service name

Configuring Net8 for Oracle Heterogeneous Services

Heterogeneous Services are an integrated component within the Oracle server, and provides the generic technology for accessing non-Oracle systems from the Oracle 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 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 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 server starts an agent process through the listener on the gateway. For the Oracle 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 server and spawn Heterogeneous Services agents by configuring the following parameters in the `listener.ora` file:

listener.ora Parameter	Net8 Assistant Field	Description
SID_NAME	SID	Specifies the Oracle System Identifier (SID)
ORACLE_HOME	Oracle Home	Specifies the Oracle home location of the agent executable
PROGRAM	Program Name	Specifies the name the agent executable

To configure the SID, ORACLE_HOME, and PROGRAM parameters:

a. Start Net8 Assistant:

-On UNIX, run `netasst` from `$ORACLE_HOME/bin`.

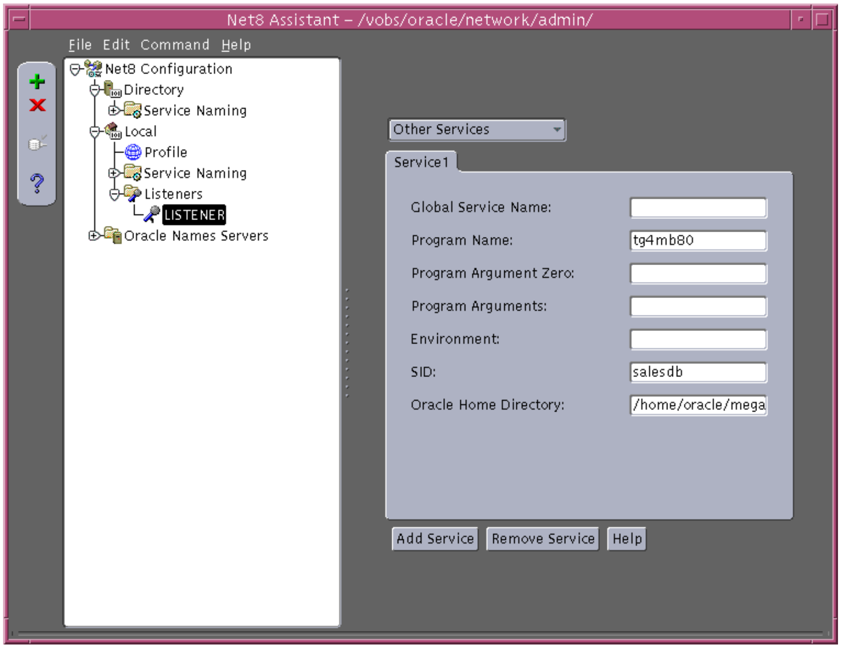
-On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

b. In the navigator pane, expand Local > Listeners.

c. Select a listener.

d. From the list in the right-pane, select Other Services.

e. Click Add Service. A new service tab appears:



Global Service Name:
Program Name: tg4mb80
Program Argument Zero:
Program Arguments:
Environment:
SID: salesdb
Oracle Home Directory: /home/oracle/mega

f. Enter the program name in the Program Names field that gets executed to create a gateway, the Oracle home where the agent executable resides in the Oracle Home field, and the SID or service name of the non-Oracle system in the SID field.

g. Choose File > Save Network Configuration.

The `listener.ora` file updates information about the Heterogeneous Services, as shown in the following:

```
sid_list_listener=
 (sid_list=
  (sid_desc=
   (sid_name=salesdb)
   (oracle_home=/home/oracle/megabase/8.1.6)
   (program=tg4mb80)))
```

See Also: *Oracle8i Distributed Database Systems*

2. On the machine 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. Start Net8 Assistant:

      -On UNIX, run `netasst` from `$ORACLE_HOME/bin`.

      -On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

   b. Create a net service name that can be used for connections from the Oracle server to a non-Oracle system.

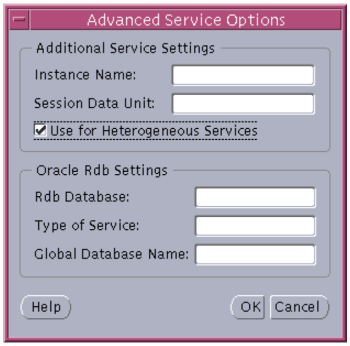
      See Also: "Task 1: Configure Net Service Names" on page 6-6 for net service name creation for local naming

   c. In the navigator pane, expand Local > Service Naming.

   d. Select the net service name.

   e. Click Advanced in the Service Identification group box.

      The Advanced Service Options dialog box appears:



Advanced Service Options

Additional Service Settings

Instance Name:

Session Data Unit:

☑ Use for Heterogeneous Services

Oracle Rdb Settings

Rdb Database:

Type of Service:

Global Database Name:

   f. Click Use for Heterogeneous Service, then click OK.

   g. Choose File > Save Network Configuration.

The `tnsnames.ora` file updates with the new net service name configured for Heterogeneous Services, as shown in the following:

```
megabase6_sales=
 (description=
  (address=(protocol=tcp)(host=dlsun206)(port=1521))
  (connect_data=
    (service_name=sales6)
    (hs=ok)))
```

Configuring Net8 for an Oracle Rdb Database

Oracle Rdb is a database for Digital's 64-bit platforms. With Net8, Oracle Rdb servers appears the same way to clients as Oracle8*i* databases. Because Oracle Rdb has its own listener, the client interacts with Rdb in the same manner as it does with Oracle7.

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 the following table:

tnsnames.ora Parameter	Net8 Assistant Field	Description
RDB_DATABASE	RDB Database	Specifies the file name of an Oracle RDB database. Embed this parameter in the CONNECT_DATA section
TYPE_OF_SERVICE	Type of Service	Specifies 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 an Oracle Rdb and Oracle database, and you want the application to randomly choose (load balance). Embed this parameter in the DESCRIPTION.
GLOBAL_NAME	Global Database Name	An optional feature, it identifies the Oracle Rdb database. Embed this parameter in the CONNECT_DATA section.

See Also: Oracle Rdb documentation

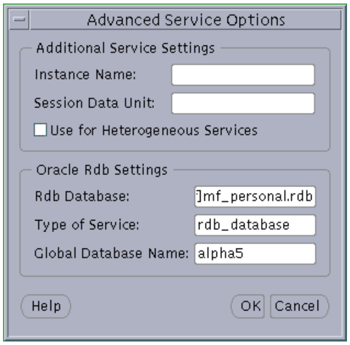
To configure a client for an Oracle Rdb database, use Net8 Assistant:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. Create a net service name that can be used for connections from the Oracle8*i* server to a non-Oracle system.

   See Also: "Task 1: Configure Net Service Names" on page 6-6 for net service name creation for local naming

3. In the navigator pane, expand Directory or Local > Service Naming.
4. Select the net service name.
5. Click Advanced in the Service Identification group box.

   The Advanced Service Options dialog box appears:



Advanced Service Options

Additional Service Settings

Instance Name:

Session Data Unit:

□ Use for Heterogeneous Services

Oracle Rdb Settings

Rdb Database:]mf_personal.rdb

Type of Service: rdb_database

Global Database Name: alpha5

6. Enter the file name of an Oracle Rdb database in the Rdb Database field.
7. 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, then click OK.

8. Choose File > Save Network Configuration.

   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)
      (rbd_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 or Oracle database service:

```
alpha5=
 (description_list=
  (description=
    (address=...)
    (connect_data=
     (service_name=generic)
     (rbd_database=[.mf]mf_personnel.rdb)
     (global_name=alpha5)))
 (description=
   (address=...)
    (connect_data=
     (service_name=sales.com))
    (type_of_service=oracle8_database))
```

   See Also: Oracle Rdb documentation

9

Configuring Multi-Threaded Server

This chapter describes how to configure **multi-threaded server (MTS)**. It includes the following sections:

- Overview
- Configuring MTS with the MTS_DISPATCHERS Parameter
- Enabling Connection Pooling
- Allocating Resources
- Using MTS on Clients
- Overriding MTS on Clients

See Also:

Oracle8i Administrator's Guide for further information about MTS configuration

Oracle8i Designing and Tuning for Performance for further information about tuning MTS parameters

Overview

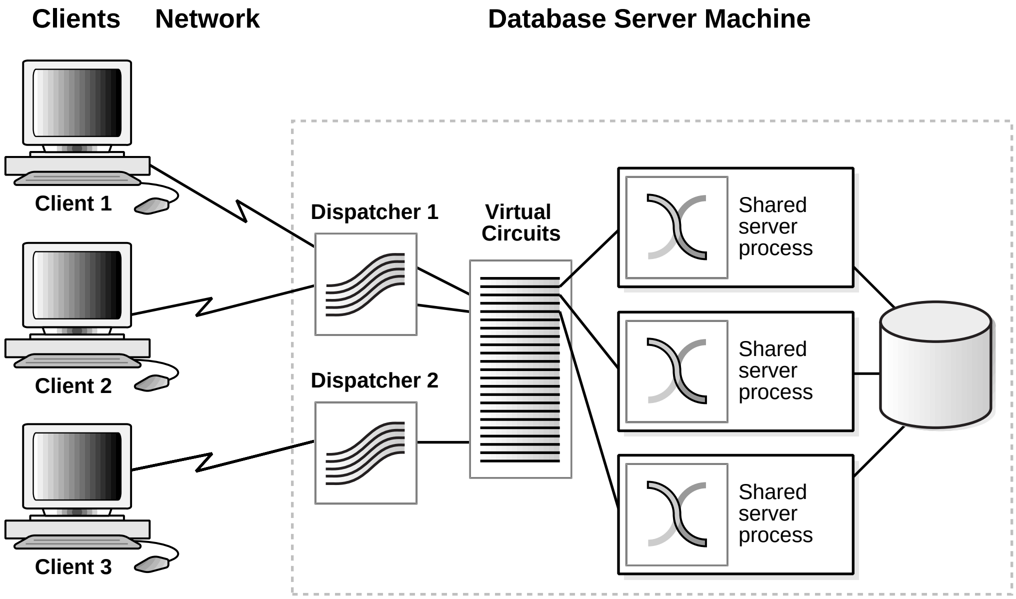
Consider an order entry system with a dedicated server architecture. A customer places an order as a clerk enters the order into the database. For most of the transaction, the clerk is on the telephone talking to the customer and the server process dedicated to the clerk's user process remains idle. The server process is not needed during most of the transaction, and the system is slower for other clerks entering orders because the idle server process is holding system resources.

The MTS architecture eliminates the need for a dedicated server process for each connection (see Figure 9–1). A dispatcher directs multiple incoming network session requests to a pool of shared server processes. An idle shared server process from a shared pool of server processes picks up a request from a common queue. This means a small number of shared servers can perform the same amount of processing as many dedicated servers. Also, since the amount of memory required for each user is relatively small, less memory and process management are required, and more users can be supported.

The advantage of MTS is that system overhead is reduced and less resources are used, allowing the number of users supported to be increased.

See Also: "Multi-Threaded Server Model" on page 2-7

Figure 9–1 MTS Architecture



See Also: "Multi-Threaded Server Model" on page 2-7

Configuring MTS with the MTS_DISPATCHERS Parameter

To enable an MTS configuration, set the MTS_DISPATCHERS parameter in the **initialization parameter file**.

> **Note:** Oracle Database Configuration Assistant enables you to configure this parameter.

After setting this parameter, restart the instance to enable MTS configuration.

MTS_DISPATCHERS should be set in the following manner:

```
mts_dispatchers="(atttribute=value)"
```

One of the following attributes is required to enable MTS: ADDRESS (ADD or ADDR), DESCRIPTION (DES or DESC), or PROTOCOL (PRO or PROT).

Attribute	Description
ADDRESS (ADD or ADDR)	Specifies the network protocol address of the endpoint on which the dispatchers listen.
DESCRIPTION (DES or DESC)	Specifies the network description of the endpoint on which the dispatchers listen, including the network protocol address. The syntax is as follows: `(description=(address=...))`
PROTOCOL (PRO or PROT)	Specifies the network protocol for which the dispatcher generates a listening endpoint. For example: `(protocol=tcp)` **See Also:** Appendix B for further information about protocol address syntax

The attributes CONNECTIONS (CON or CONN), DISPATCHERS (DIS or DISP), LISTENER (LIS or LIST), MULTIPLEX (MUL or MULT), POOL, SERVICE (SER, SERV), SESSIONS (SES or SESS), and TICKS are optional:

Attribute	Description
CONNECTIONS (CON or CONN)	Specifies the maximum number of network connections to allow for each dispatcher The default is operating-system specific. For example, 1024 is the default for Sun Solaris and Windows NT.
DISPATCHERS (DIS or DISP)	Specifies the initial number of dispatchers to start. The default is 1.

LISTENER (LIS or LIST)	Specifies an alias name for the listener(s) with which the **PMON process** registers dispatcher information. The alias should be set to a name which is resolved through a naming method.
	This attribute need only be specified if the:
	▪ Listener is a local listener that uses a non-default address, that is, not TCP/IP, port 1521
	▪ Listener(s) are on other nodes
	Non-default local listeners may also be specified with the LOCAL_LISTENER parameter.
	Important: The listener alias must be resolved through a naming method, such as a `tnsnames.ora` file on the server or an Oracle Names server.
	For example, if the listener alias name is `listener_sales` with two listening endpoints of port 1521, and the chosen naming method is the local naming method, the entry in the `tnsnames.ora` file would look like the following:
	`listener_sales=`
	`(description=`
	`   (address=(protocol=tcp)(host=sales1-server)(port= 1521))`
	`   (address=(protocol=tcp)(host=sales2-server)(port= 1521)))`

Attribute	Description
MULTIPLEX (MUL or MULT)	Enables Oracle Connection Manager's **connection concentration** feature If 1, ON, YES, TRUE, or BOTH is specified, then connection concentration is enabled for both incoming and outgoing network connections. If IN is specified, the connection concentration is enabled for incoming network connections from clients. If OUT is specified, the connection concentration is enabled for outgoing network connections. If 0, NO, OFF, or FALSE is specified, then connection concentration is disabled for both incoming and outgoing network connections. **See Also:** "Enabling Connection Concentration" on page 8-28
POOL	Enables **connection pooling** If a number is specified, enables connection pooling for both incoming and outgoing network connections. The number specified is the timeout in ticks for both incoming and outgoing network connections. If ON, YES, TRUE, or BOTH is specified, enables connection pooling for both incoming and outgoing network connections. A default timeout of 10 ticks is used for both incoming and outgoing network connections. If IN is specified, connection pooling is enabled for incoming network connections and the default timeout of 10 ticks is used for incoming network connections. IN can also be assigned a timeout in ticks value, such as (IN=20). If the numeric value of a specified timeout is 0 or 1, then the default value of 10 ticks is used. If OUT is specified, connection pooling is enabled for outgoing network connections and the default timeout of 10 ticks is used for outgoing network connections. OUT can also be assigned a timeout in ticks value, such as (OUT=20). If the numeric value of a specified timeout is 0 or 1, then the default value of 10 ticks is used. If NO, OFF, or FALSE is specified, then connection pooling is disabled for both incoming and outgoing network connections. **See Also:** "Enabling Connection Pooling" on page 9-9
SERVICE (SER, SERV)	Specifies the service name(s) the dispatchers register with the listeners. If not value are specified, the service names specified with the SERVICE_NAMES parameter are used.

Attribute	Description
SESSIONS (SES or SESS)	The maximum number of network sessions to allow for each dispatcher. The default is operating-system specific. Most operating systems have a default of 16K.
TICKS	The size of a network tick in seconds. The value set is multiplied with the POOL timeout value to get the total connection pool timeout. The default is 15 seconds. For a fast network, Oracle Corporation recommends a tick size of ~1 second. For a slow network, Oracle Corporation recommends a tick size of ~3-4 seconds.

Setting the Initial Number of Dispatchers

The number of dispatchers started at instance startup is controlled by the DISPATCHERS attribute.

Note: Unlike the shared servers, the number of dispatchers does not change dynamically. The number of dispatchers must be explicitly changed with the ALTER SYSTEM command. You can change the number of dispatchers in this manner up to a maximum limit specified by the MTS_MAX_DISPATCHERS parameter. See the *Oracle8i Designing and Tuning for Performance* for further information about this parameter.

The appropriate number of dispatchers for each instance depends upon the performance you want from your database, the host operating system’s limit on the number of connections per process, (which is operating system dependent), and the number of connections required per network protocol.

Calculating the Initial Number of Dispatchers

Once you know the number of possible connections per process for your operating system, calculate the initial number of dispatchers to create during instance startup, per network protocol, using the following formula.

```
number                   maximum number of concurrent sessions
of          = CEIL   (--------------------------------------)
dispatchers                  connections per dispatcher
```

CEIL represents the number roundest to the next highest number.

As an example, assume that a system that has:

- 4000 users concurrently connected via TCP/IP and supports 1,000 connections per process
- 3000 users concurrently connected via SPX and supports 1,000 connections per process

In this case, the DISPATCHERS attribute for TCP/IP should be set to a minimum of four dispatchers and SPX should be set to a minimum of three dispatchers:

```
mts_dispatchers="(protocol=tcp)(dispatchers=4)(connections=1000)"
mts_dispatchers="(protocol=spx)(dispatchers=3)(connections=1000)"
```

Depending on performance, you may need to adjust the number of dispatchers.

Setting Dispatcher Addresses

Example 1 To force the IP address used for the dispatchers, set the following:

```
mts_dispatchers="(address=(protocol=tcp)(host=144.25.16.201))
(dispatchers=2)"
```

This starts two dispatchers that listen on HOST=144.25.16.201. Note that Net8 dynamically selects the TCP/IP port for the dispatcher.

Example 2 To force the exact location of the dispatchers, add the PORT as follows:

```
mts_dispatchers="(address=(protocol=tcp)
(host=144.25.16.201)(port=5000))(dispatchers=1)"
mts_dispatchers="(address=(protocol=tcp)
(host=144.25.16.201)(port=5001))(dispatchers=1)"
```

Note: You can specify multiple MTS_DISPATCHERS in the initialization file, but they must be adjacent to each other.

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 MTS_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

Returning to the example in "Calculating the Initial Number of Dispatchers" on page 9-8, connection pooling can be used to allow each dispatcher 1,000 connections and 4,000 sessions for TCP/IP and 3,000 sessions for SPX. This reduces the configuration to one dispatcher for each protocol, as shown in the following:

```
mts_dispatchers="(protocol=tcp)(dispatchers=1)(pool=on)(tick=1)
(connections=1000)(sessions=4000)"
```

```
mts_dispatchers="(protocol=spx)(dispatchers=1)(pool=on)(tick=1)
(connections=2000)"
```

Allocating Resources

An Oracle8*i* database can be represented by multiple service names. Because of this, a pool of MTS 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 for only clients requesting `adminsales.us.acme.com`.

```
service_names=sales.us.acme.com
instance_name=sales
mts_dispatchers="(protocol=tcp)"
mts_dispatchers="(protocol=tcp)(service=adminsales.us.acme.com)"
```

Using MTS on Clients

If MTS is configured and a client connection request comes when no dispatchers are registered, the requests can be handled by a dedicated server (configured in the `listener.ora` file). If you want a particular client to always use a dispatcher, configure (SERVER=SHARED) in the connect data portion 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 8-8 to set the SERVER parameter

Overriding MTS on Clients

If the database is configured for MTS and a particular client requires a dedicated server, the client can be configured to use a dedicated server in one of the following ways:

- A net service name can be configured 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)))
  ```

- The client profile (`sqlnet.ora` file) can be configured with USE_DEDICATED_SERVER=ON. This adds (SERVER=DEDICATED) to the CONNECT_DATA section of the connect descriptors the client uses. For example:

  ```
  use_dedicated_server=on
  ```

> **Note:** If USE_DEDICATED_SERVER is set to ON, existing (SERVER=*value*) entries are overwritten with (SERVER=DEDICATED).

See Also:

- See "Configuring Advanced Connect Data Parameters" on page 8-8 to set the SERVER parameter.
- See "Routing Connection Requests" on page 8-21 to set the USE_DEDICATED_SERVER parameter.

10

Enabling Net8 Enhancements for Programmers

Net8 includes an application program interface (API) called Net8 OPEN, that allows programmers to develop both database and non-database applications. In addition, Net8 contains several benefits for programmers, including UNIX client programming, signal handler and alarm programming, Bequeath protocol support, and child process termination. This chapter contains the following sections:

- Net8 OPEN
- UNIX Client Programming

Net8 OPEN

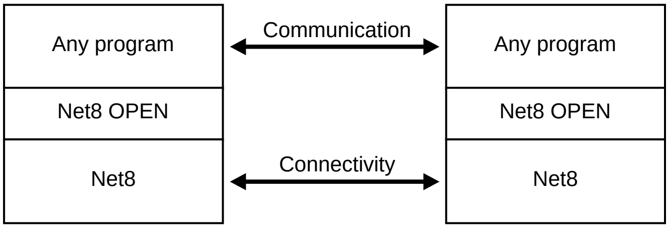
Net8 includes an application program interface (API), called Net8 OPEN, which enables programmers to:

- Develop both database and non-database applications that make use of the Net8 network already deployed in their environment
- Deploy an application developed on one machine to another without having to modify their calls to the network interface

Net8 OPEN provides applications a single common interface to all industry standard network protocols.

The relationship of Net8 OPEN to other products is shown in Figure 10–1.

Figure 10–1 Net8 OPEN



Using Net8 OPEN, you can solve a number of problems, such as:

- Three-tier connectivity (client/agent/server)

  Use any application to communicate with an agent. For example, the agent might be an application server that allows simultaneous connectivity to Oracle and non-Oracle data sources, such as remote information servers.

- Distributed applications

  Build distributed applications that can run over an existing Oracle network without the requirements of a database or additional middleware.

- Enhanced clients

  Integrate non-SQL information with SQL applications. For example, a process control application can communicate with a non-SQL application such as a sensor.

Net8 OPEN API Function Calls

In contrast to a remote procedure call interface, Net8 OPEN provides a byte stream-oriented API that can be used to develop basic applications which send and receive data. Applications developed with Net8 OPEN must ensure that values sent across the network are interpreted correctly at the receiving end.

The Net8 OPEN API consists of five function calls:

- TNSOPEN
- TNSCLOSE
- TNSSEND
- TNSRECV
- TNSCONTROL

TNSOPEN

Description:	Initializes the Net8 OPEN API per-connection handle. This function must be the first Net8 OPEN call that a user makes. Note that `tnsopen()` does not establish a connection; connections are established by the send and receive calls as needed.

Synopsis:	`int tnsopen(handlep, name)`
	`void **handlep;`
	`const char *name;`
	If you are writing a client program that will initiate the connection, "name" contains a net service name in the same format as those in a `tnsnames.ora` file.
	If you are writing a server program, "name" should be NULL. Your server program will pick up the connection automatically when you make the first `tnsrecv()` call to receive data.

Parameters:	`handlep (IN/OUT)`: Address to receive Net8 connection handle `name (IN)`: Net service name
Prerequisites:	The handlep parameter must not be NULL
Returns:	Upon successful completion a zero value is returned. Otherwise, a positive Net8 API error is returned.

TNSCLOSE

Description: Shuts down the connection. The user must call this function to close the connection and release the handle properly.

Synopsis:

```
int tnsclose(handlep)
void **handlep;
```

Parameters: `handlep(IN/OUT)`: Address of a pointer to a Net8 connection handle

Prerequisites: The handlep parameter must not be NULL.

Returns: Upon successful completion a zero value is returned, and *handlep is set to NULL. Otherwise, a positive Net8 API error number is returned.

TNSSEND

Description: Sends data to the Net8 connection handle

In the first call to `tnssend()` on the client side, the connection is established before any data is sent to the handle. The client application must first call `tnssend()` after `tnsopen()` to establish a connection to the server. It is an error if the client application calls `tnsrecv()` first, or if the server program calls `tnssend()` first.

Note that this also means that the `tnssend()` call may return errors related to connection establishment. For example, if you have given the incorrect TNS address, the first error occurs on `tnssend()` call rather than on the `tnsopen()`.

Synopsis:

```
int tnssend(handle, data, length)
void *handle;
const void *data;
size_t *length;
```

Parameters: `handle(IN/OUT)` - Pointer to Net8 connection handle returned by tnsopen()

`data(IN)`: Pointer to data to be sent

`length(IN/OUT)`: Pointer to the length of data to be sent in bytes and the actual number of bytes written on return

Prerequisites: The parameters must not be NULL.

Returns: Upon successful completion a zero value is returned, and the actual number of bytes written is returned as the value pointed to by the length parameter. Otherwise, a positive Net8 API error number is returned.

TNSRECV

Description: Receives data from the Net8 connection handle

In the first call to `tnsrecv()` on the server side, the connection is established before data is received from the handle. The server must first call `tnsrecv()` after `tnsopen()` to accept the connection from the client.

Incoming connections are accepted by the Net8 Listener (tnslsnr), which automatically spawns off a copy of your server program when needed, and hands it the new connection. You must configure the network listener with the location of your server program for this to occurs.

Synopsis:

```
int tnsrecv(handle, data, length)
void *handle;
void *data;
size_t *length;
```

Parameters: `handle(IN/OUT)`: Pointer to Net8 connection handle returned by tnsopen()

`data(IN/OUT)`: pointer to buffer to receive data

`length(IN/OUT)`: Pointer to the length of buffer to receive data and actual number of bytes received on return

Prerequisites: All parameters must not be NULL.

Returns: Upon successful completion a zero value is returned, and the actual number of bytes received is returned as the value pointed to by the length parameter. Otherwise, a positive Net8 API error number is returned.

TNSCONTROL

Description: Sets the connection to blocking or nonblocking mode

Synopsis:

```
int tnscontrol(handle, cmd)
void *handle;
int *cmd;
```

Parameters: `handle(IN)`: Pointer to Net8 connection handle returned by tnsopen()

`cmd(IN)`: Option to apply to the connection. Currently supported values are:

- `TNSAPINONBLOCKING`

  Sets connection into nonblocking mode

- `TNSAPIBLOCKING`

  Sets connection into blocking mode

Prerequisites: The handle must not be NULL, and cmd must be one of the supported commands.

Returns: A zero value is returned if the option is successfully set.

Finding the Net8 OPEN Applications Program Interface

The applications program interface is provided as part of the standard Net8 installation. To use it, you need the following:

- `TNSAPI.H`
- This is the header file which describes the API interfaces and errors. It is provided in `$ORACLE_HOME/network/public` on UNIX and *ORACLE_HOME*`\network\tnsapi\include` on Windows NT.
- The Net8 OPEN library

  This is located with other Oracle networking libraries, and contains the name "`TNSAPI`". Note that the name of the library varies by platform. On UNIX, it is in your `$ORACLE_HOME/network/lib` directory and is named `LIBTNSAPI.A`. On Windows platforms, the *ORACLE_HOME*`/network/tnsapi/lib` contain the files `TNSAPI.DLL` and `TNSAPI.LIB`.
- Sample makefiles

  These are provided for your platform in your network directory. They can be used to determine the appropriate link line to build your application.

Building Your Own Application

Modules which make reference to Net8 OPEN functions should include TNSAPI.H, as follows:

```
#include <tnsapi.h>
```

Your makefile (or other location for your build command) should ensure that the include path is set properly so that it can find TNSAPI.H. Refer to the sample makefiles provided in your installation.

Configuring the System to Use Your Net8 OPEN Application

To configure Net8 to recognize your Net8 OPEN application:

1. Add the location of your server program to the `listener.ora` file, so that the listener knows to start your server if a connection request is received.

   > **Note:** This is not necessary for release 8.1 configurations, as database instance registration registers service information with the listener.

   To do this, choose a system identifier (SID) name for your service similar to that of an Oracle database. Do not pick the same SID as your database.

   For example, if you are configuring a "chat" program, you could call the SID "chatsid". Place the program into the same place as the Oracle server executable, which is normally `$ORACLE_HOME/bin` on UNIX and *`ORACLE_HOME`*`\bin` on Windows NT.

   You would place the following entry in a listener configuration file as follows:

   ```
   sid_list_listener =
     (sid_list =
       (sid_desc =
           (sid_name = chatsid)/*your SID name*/
           (oracle_home = /usr/oracle)/*$ORACLE_HOME bin directory*/
           (program = chatsvr)/*the name of your server program*/)
   ```

   You need to restart the listener, so it will recognize the new service.

2. Add the address of your application server to the `tnsnames.ora` file.

   For example, if your listener is listening on the following address:

   ```
   (description=(address=(protocol=tcp)(host=unixhost)(port=1521)))
   ```

   and, you want people to refer to the service you created above as `chat`, add the following to the `tnsnames.ora` file:

   ```
   chat=
   (description=
       (address=(protocol=tcp)(host=unixhost)(port=1521))
       (connect_data=(service_name=chat)))
   ```

3. Place the executable for your service in the same directory as your Oracle server executable. On UNIX platforms, place the executable in the `$ORACLE_HOME/bin` directory indicated in your `listener.ora` file. In this example, you would place the program "chatsvr" in the location `/usr/oracle/bin/chatsvr.`

   If needed on your operating system, you also must ensure that you have permission to execute your program.

Sample Programs

Two sample applications are provided with Net8 OPEN:

- `finger`

  This is a utility that connects to the server that retrieves information about who is logged in. This utility includes a pair of programs which demonstrate the basic steps involved in building a distributed application. The client program runs on both Solaris and Windows NT; the server is UNIX specific.

- `tftp`

  This sample client and server program is implemented in UNIX to help you with simple file transfers using the tftp protocol.

Net8 OPEN API Errors

This section lists the error numbers that can be returned if one of the above function calls fails. Note that in some cases, connection-related errors may come back from a send or receive call, if the connection has not yet been established at that time.

```
20002 - SDFAIL_TNSAPIE - The underlying "send" command failed in tnssend().
20003 - RECVFAIL_TNSAPIE - The underlying "receive" command failed in tnsrecv().
20004 - INVSVROP_TNSAPIE - Operation is invalid as the server.
20005 - INVCLIOP_TNSAPIE - Operation is invalid as the client.
20006 - HDLUNINI_TNSAPIE - The connection should be initialized by calling
tnsopen().
20007 - INHFAIL_TNSAPIE - Server failed in inheriting the connection from the
listener.
20008 - ACPTFAIL_TNSAPIE - Server failed in accepting the connection request
from the client.
20009 - NULHDL_TNSAPIE - A null handle was passed into the call, which is not
allowed.
20010 - INVOP_TNSAPIE - An invalid operation called was passed into the call.
20011 - MALFAIL_TNSAPIE - A malloc failed in TNS API call.
```

```
20012 - NLINIFAIL_TNSAPIE - Failed in NL initialization.
20013 - NMTOOLONG_TNSAPIE - Service name is too long.
20014 - CONFAIL_TNSAPIE - Client connect request failed.
20015 - LSNFAIL_TNSAPIE - Server failed to listen for connect request.
20016 - ANSFAIL_TNSAPIE - Server failed to answer connect request.
20017 - NMRESFAIL_TNSAPIE - Failed to resolve service name.
20018 - WOULDBLOCK_TNSAPIE - Operation would block.
20019 - CTLFAIL_TNSAPIE - Control call failed.
20020 - TNSAPIE_ERROR - TNS error occurred.
20021 - INVCTL_TNSAPIE - Invalid operation request in control call.
```

UNIX Client Programming

Event programming in UNIX requires the use of a UNIX signal. When an event occurs, a signal flags a process. The process executes code that is relevant to the particular signal generated. UNIX does not allow a single process to set more than one signal handler or alarm for a particular signal call. If a process sets a second signal handler or alarm request on a signal like SIGCHLD (signal on a child process' status change), UNIX nullifies and loses the previous request for the SIGCHLD.

If any part of your application issues one of these requests, signal handling or alarms may cause the system to lose and never respond to that particular request. Depending on the signal requested, the system may not clean up defunct processes properly because of a signal handler problem.

Net8 provides two solutions to allow for the use of signal handling and alarms in tandem with Oracle's usage of those requests:

- Signal Handler and Alarm Programming
- Bequeath

Signal Handler and Alarm Programming

Net8 provides an Operating System Dependent (OSD) call that keeps a table of all signal handler or alarm requests for each signal. Any program that uses the signal handler or alarm is now required to use the Oracle OSD calls. This provides a solution for programmers in UNIX who are not allowed to set more than one signal handler or alarm for a particular call. Any program that uses the signal handler or alarm must use the Oracle OSD calls. This is however, currently available only for internal use.

Until then, if you set all of the client's signal handlers before making any database connections, the OSD call will remember the last signal handler set for the signal and will add it to the signal handler table. Note that by doing this, you cannot disable the signal handler.

Oracle OSD Signal Handling Rules

To use the table-driven shared OSD signal handler for all SIGCHLD calls, you must observe the following rules:

- Know your child process IDs so you can clean up the correct process.
- Use the `waitpid`() call instead of `wait`() on the correct child process ID.
- Set the `waitpid`() call to be non-blocking.

Bequeath

This section is for UNIX application programmers who use both the UNIX signal handler for tracking child process status changes with the SIGCHLD call and Net8 for the networking portion of their application.

When a client application is directed to communicate with an Oracle database on the same machine, it uses the Bequeath protocol to establish the connection. The Bequeath protocol enables the client to retrieve information from the database without using the listener. The Bequeath protocol internally spawns a server process for each client application. In a sense, it performs locally the same operation that a remote listener does for your connection.

Child Process Termination

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. Alternatively, you may configure your client `sqlnet.ora` file to pass this process to the UNIX init process by disabling signal handlers.

Configure the BEQUEATH_DETACH parameter in the `sqlnet.ora` file:

```
bequeath_detach=yes
```

This parameter causes all child processes to be passed over to the UNIX init process (pid = 1). The init process automatically checks for "defunct" child processes and terminates them.

Bequeath automatically chooses to use a signal handler in tracking child process status changes. If your application does not use any signal handling, then this default does not affect you.

Part III

Net8 Testing and Troubleshooting

Part III describes how to establish Net8 connections, and identify and diagnose problems. Part III contains the following chapters:

- Chapter 11, "Establishing a Connection and Testing the Network"
- Chapter 12, "Troubleshooting"

11

Establishing a Connection and Testing the Network

Once you have completed configuring your network, you should make a connection and test each component to ensure that the network is functioning properly. Net8 provides a variety of tools to help you start, test, and control an Oracle Names server, listener, and Oracle Connection Manager.

This chapter outlines procedures to make a connection and test network components using control utilities. This chapter contains the following sections:

- 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/password@connect_identifier
```

This section covers the following topics:

- Using Net8 Control Utilities
- Using Net8 Control Utilities
- Entering a Connect String
- Connect Identifier Syntax Characteristics
- Initiating Connections

Using Net8 Control Utilities

Net8 provides the following tools to help you start, test and control each network component.

- Oracle Names Control Utility (NAMESCTL)
- Listener Control Utility (LSNRCTL)
- Connection Manager Control Utility (CMCTL)

See Also: Appendix A, "Control Utilities"

Using the Oracle Names Control Utility (NAMESCTL)

The Oracle Names Control Utility, NAMESCTL, is a tool that you run from the operating system prompt to start and control the Oracle Names server.

The general form of the Oracle Names Control Utility is:

```
NAMESCTL command
```

You can also issue NAMESCTL commands at the program prompt. When you enter NAMESCTL on the command line, the program is opened. You can then enter the desired commands from the program prompt. For example, the following command starts the Oracle Names server.

```
NAMESCTL> start
```

Using the Listener Control Utility (LSNRCTL)

The Listener Control Utility, LSNRCTL, is a tool that you run from the operating system prompt to start and control the listener. The general form of the Listener Control Utility is:

```
LSNRCTL command [listener_name] [args]
```

You can also issue Listener Control Utility commands at the program prompt. When you enter LSNRCTL on the command line, the program is opened. You can then enter the desired commands from the program prompt. For example, the following command determines the amount of time in seconds the listener will wait for a valid connection request after a connection has been started.

```
LSNRCTL> set connect_timeout 20
```

Using the Connection Manager Control Utility (CMCTL)

The Connection Manager Control Utility, CMCTL, is a tool that you run from the operating system prompt to start and control Oracle Connection Manager. The general form of the Connection Manager Control Utility is:

```
CMCTL command
```

You can also issue CMCTL commands at the program prompt. When you enter CMCTL on the command line, the program is opened. You can then enter the desired commands from the program prompt. For example, the following command starts Oracle Connection Manager.

```
CMCTL> start
```

Net8 Component Startup Overview

Client workstations and other servers connect to a listener with a net service name when logging onto an Oracle 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 steps you should take to start the network components.

Task 1: Start Oracle Names Servers

Task 2: Start Oracle Names Client Cache

Task 3: Start the Listener

Task 4: Start the Database

Task 5: Start Oracle Connection Manager

Note: If the Oracle Names servers use a database to store the network information, you need to start the database first, then start the listener.

Task 1: Start Oracle Names Servers

Start Oracle Names servers using either Net8 Assistant or the NAMESCTL control utility on the machines where Oracle Names server software is installed and configured:

Use the NAMESCTL Control Utility	Use Net8 Assistant

From the command line, enter:
`NAMESCTL`
`NAMESCTL> start`
The START command of NAMESCTL loads the Oracle Names server into memory and tells it to begin executing. At startup, the Oracle Names server loads its configuration and data.
	1. Start Net8 Assistant.
	-On UNIX, run `netasst` at `$ORACLE_HOME/bin`.
	-On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
	2. In the navigator pane, expand the Oracle Names server folder.
	3. Select the Oracle Names server.
	4. From the list in the right pane, select Manage Server from the list box.
	5. Click the Control tab.
	6. Click the Start radio button from the Server Operations field.
	7. Click Apply.

On Windows NT, an Oracle Names server may also be started through the Control Panel:

1. Double-click the Services icon in the Control Panel window.
2. Select the Oracle*HOME_NAME*Names*onames_server*Service service.
3. Click Start to start the service.
4. In the Services window, click Close.

Task 2: Start Oracle Names Client Cache

To avoid clients looking up address information in an Oracle Names server each time, create a client cache of the information. This information is stored in `ckpcch.ora` file located by default in `$ORACLE_HOME/network/names` on UNIX and *`ORACLE_HOME`*`\network\names` on Windows platforms.

To create a client cache:

1. Ensure discovery of Oracle Names servers has been performed, as described in "Task 4: Configure Clients and Database Servers To Use Oracle Names Servers" on page 6-60. The information gathered during discovery is used for a client's cache.
2. Create and start the client cache on the client:

```
NAMESCTL
NAMESCTL> start_client_cache
```

On Windows NT, the client cache may also be started through the Control Panel:

1. Double-click the Services icon in the Control Panel window.
2. Select the Oracle*HOME_NAME*ClientCache service.
3. Click Start to start the service.
4. In the Services window, click Close.

Task 3: Start the Listener

For Net8 to accept connections on the server, a listener must be started with the LSNRCTL control utility on the server:

1. 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, go to Step 2. If the listener is not running, go to Step 3.

   Even if the listener is running, Oracle Corporation advises you to stop the listener, and start it again. To stop the listener, enter:

   ```
   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]
   ```

   LSNRCTL 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 LSNRCTL utility. Enter:

   ```
   LSNRCTL> exit
   ```

On Windows NT, the listener may also be started through the Control Panel:

1. Double-click 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 non-default listener name.
3. Click Start to start the service.
4. In the Services window, click Close.

Task 4: 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 by entering:

   ```
   SQLPLUS /nolog
   ```

2. Connect to Oracle as SYSDBA:

   ```
   SQL> connect username/password as sysdba
   ```

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, Oracle uses the standard initialization file location of `$ORACLE_BASE/admin/`*`db_name`*`/pfile/`*`sid`* on UNIX platforms and `ORACLE_BASE\admin\`*`db_name`*`\pfile\`*`sid`* on Windows NT; if you do not specify a database name Oracle uses the value of the DB_NAME parameter specified in the initialization file.

   > **See Also:** *Oracle8i Administrator's Guide* for further information about starting the database

Task 5: Start Oracle Connection Manager

If Oracle Connection Manager is installed and configured, start it with the CMCTL control utility:

1. From the command line, enter:

   ```
   CMCTL
   CMCTL> start cman
   ```

   CMCTL displays a status message indicating that Oracle Connection Manager has started successfully.

2. Exit from the CMCTL utility. Enter:

   ```
   CMCTL> exit
   ```

On Windows NT, the listener may also be started through the Control Panel:

1. Double-click the Services icon in the Control Panel window.
2. If you are using Oracle Names, select the Oracle*HOME_NAME*CMAdminService to acquire information about available Oracle Names Servers, then click Start. If you are not using Oracle Names, do not start this service.

   The service starts.

3. Select the Oracle*HOME_NAME*CManService to start the Oracle Connection Manager, and click Start.

   The service starts.

4. 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 11-2 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 6, "Configuring Naming Methods" and the tool used for the connection.

The connect strings takes the following basic form:

```
CONNECT username/password@connect_identifier
```

Connect Identifier Syntax Characteristics

A connect identifier cannot contain spaces unless enclosed within single quotes (’ ’) or double quotes (" "). In the following examples, connect identifiers that contain spaces are enclosed in single quotes.

```
CONNECT scott/tiger@’(description=(address=(protocol=tcp)(host=sales-server)
(port=1521))(connect_data=(service_name=sales.us.acme.com)))’
CONNECT scott/tiger@’cn=sales, cn=OracleContext, dc=us, dc=acme, dc=com’
```

Single quotes (’) are required if a double quote (") is used in the connect identifier. For example:

```
CONNECT scott/tiger@'sales@Good"Fast"Food.com'
```

Likewise, double quotes (") are required if a single quote (’) is used in the connect identifier. For example:

```
CONNECT scott/tiger@"cn=sales, cn=OracleContext, ou=Mary's Dept, o=acme"
```

Initiating Connections

There are a number of ways to initiate a connection to an Oracle server. Commonly used methods are described in the following sections:

- 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/password@net_service_name
```

```
SQLPLUS system/password@sales
```

To prevent the password from displaying during a logon, you can leave out the password parameter on the command line. For example:

```
SQLPLUS system@sales
```

You will be prompted to enter your password without it showing on screen.

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 user name and net service name (*username@net_service_name*) in the user name field of the tool logon screen, and typing 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, the *: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 connection, 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/password@net_service_name
```

For example:

```
SQL> CONNECT scott/tiger@serverx
```

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 Oracle Names server (if included in your network layout).
2. Start and test each listener.
3. Start and test each Oracle Connection Manager (if included in your network layout).
4. Test the server with a loopback test.
5. Test client with a connection.

This section covers the following topics:

- Testing an Oracle Names Server
- Testing Network Objects Using Net8 Assistant or NAMESCTL Utility
- Testing a Listener
- Testing Oracle Connection Manager
- Testing Configuration on the Server
- Testing Network Connectivity from the Client

Testing an Oracle Names Server

To test an Oracle Names server, use the NAMESCTL PING command. Following are two ways to PING the Oracle Names server `labrador` in the `us.acme` domain.

From the NAMESCTL prompt, enter:

```
NAMESCTL> ping labrador.us.acme
```

You can test several Oracle Names servers with the same PING command. For example:

```
NAMESCTL> ping huey.uk.acme duey.uk.acme louie.uk.acme
```

PING responds with the time it takes to contact the Oracle Names server and return an acknowledgment. If PING fails, make sure the Oracle Names server is started or double-check the configured address of the Oracle Names server.

Testing Network Objects Using Net8 Assistant or NAMESCTL Utility

The information stored in an Oracle Names server can be queried to verify registration with the QUERY command.

To use the QUERY command, use either Net8 Assistant or the NAMESCTL control utility:

Use Net8 Assistant...	Use NAMESCTL Control Utility...

1. Start Net8 Assistant:
-On UNIX, run `netasst` at `$ORACLE_HOME/bin`.
-On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. Double-click the Oracle Names server folder.
3. Select an Oracle Names server.
4. Choose Manage Data from the drop-down list box.
5. Click the Advanced tab.
6. Select Query.
7. Enter the name of the object to query in the Name field.
8. Optionally, enter the type of the object in the Name field:
-A.SMD: Network addresses, such a database network definitions for net service names.
-CNAME.SMD: Aliases
-DL.RDBMS.OMD: Global database links
-DLCR.RDBMS.OMD: Link qualifiers
-NS.SMD: Oracle Names server address. System data used to communicate between Oracle Names servers
-V1ADD.NPO.OMD: SQL*Net Version 1 connect string
9. Select Execute.
10. Select Save Network Configuration from the File menu.
11. Select Exit from the File menu to exit the Net8 Assistant application.
	For a Oracle Names server in the administrative region, create and register an alias with the Oracle Names server:
	To display all data:
	`NAMESCTL`
	`NAMESCTL> query name *`
	The type of information to retrieve may also be requested. Common object types include:
	-A.SMD: Network addresses, such a database network definitions for net service names.
	-CNAME.SMD: Aliases
	-DL.RDBMS.OMD: Global database links
	-DLCR.RDBMS.OMD: Link qualifiers
	-NS.SMD: Oracle Names server address. System data used to communicate between Oracle Names servers
	-V1ADD.NPO.OMD: SQL*Net Version 1 connect string
	To display the database address for a net service name:
	`NAMESCTL`
	`NAMESCTL> query name a.smd`
	The following example shows a query of the net service name `sales`:
	`NAMESCTL`
	`NAMESCTL> query sales a.smd`
	The QUERY command returns the amount of time the transaction took and information about the network object.

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 Server" on page 11-15.

Testing Oracle Connection Manager

To test Oracle Connection Manager, initiate a connection from a client to any active database for which a source route address has been created.

Testing Configuration on the Server

Once you have configured the network, test the configuration by performing a **loopback test** on the server.

A loopback test uses Net8 to go from the server back to itself, bypassing the Interprocess Communication (IPC). Performing a successful loopback verifies that Net8 is functioning on the server side.

To perform the loopback test, use Net8 Assistant:

1. Start Net8 Assistant:

   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.

   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

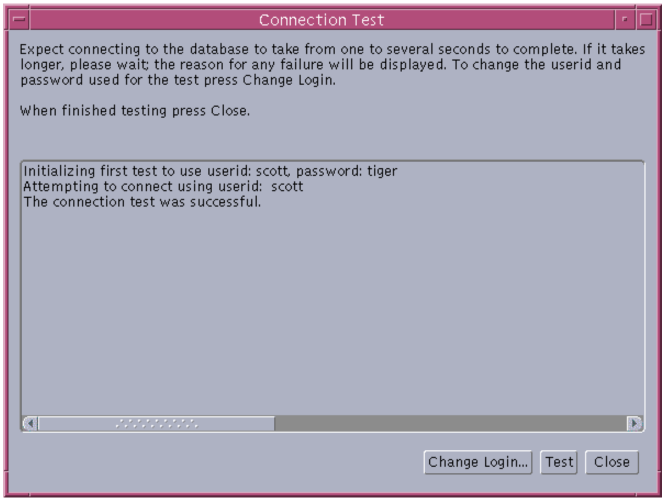
2. In the navigator, expand Directory or Local > Service Naming.

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 "Using Net8 Control Utilities" on page 11-3 to start components.

A successful test results in the following message in the Connect Test dialog box:



Expect connecting to the database to take from one to several seconds to complete. If it takes longer, please wait; the reason for any failure will be displayed. To change the userid and password used for the test press Change Login.

When finished testing press Close.

```
Initializing first test to use userid: scott, password: tiger
Attempting to connect using userid:  scott
The connection test was successful.
```

If the test was not successful:

- Ensure the database and listener are running, then click Test.
- Click Change Login to change the user name and password for the connection, 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 server from each of them by following the instructions in "Entering a Connect String" on page 11-10.

Net8 also provides the following tools to help you evaluate network connectivity:

- TNSPING Utility
- TRCROUTE Utility
- Net8 Assistant

TNSPING Utility

TNSPING is a utility that determines whether or not a service (for example, an Oracle database, an Oracle Names server or any other Oracle service) on a Net8 network can be successfully reached.

If you can connect successfully from a client to a server (or a server to another server) using TNSPING, it displays an estimate of the round trip time (in milliseconds) it takes to reach the Net8 service.

If it fails, it displays a message describing the error that occurred. This allows 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 Server Manager to attempt a connection to the database.

Following are some examples of TNSPING.

Reaching a Database To connect to a database using a net service name of `spotdb`, the following is entered:

```
tnsping spotdb
```

This produces the following message:

```
TNS Ping Utility for SunOS:
Copyright (c) Oracle Corporation 1998.  All rights reserved.
Attempting to contact
(ADDRESS=(PROTOCOL=TCP)(HOST=spot)(PORT=1521))
OK (50msec)
```

To determine whether a connection can be made to the `stprd` database, and to specify that TNSPING try to connect 10 times and then give up, use the following command:

```
tnsping stprd 10
```

This command produces the following message:

```
TNS Ping Utility for SunOS:
Copyright (c) Oracle Corporation 1998. All rights reserved.
Attempting to contact (ADDRESS=(PROTOCOL=TCP)(HOST=spot)(PORT=1521))
OK (290 msec)
OK (100 msec)
OK (70 msec)
OK (70 msec)
OK (60 msec)
OK (70 msec)
OK (70 msec)
OK (80 msec)
OK (180 msec
OK (340 msec)
```

Invalid Net Service Name Below is an example of TNSPING attempting to connect to an invalid net service name:

```
tnsping bad_db
```

This attempt produces the following message:

```
TNS Ping Utility for SunOS:
Copyright (c) Oracle Corporation 1998.  All rights reserved.
TNS-03505: Failed to resolve name
```

Valid Net Service Name Without Listener 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):

```
tnsping testing
```

The following message is returned:

```
TNS Ping Utility for SunOS:
Copyright (c) Oracle Corporation 1998.  All rights reserved.
Attempting to contact (ADDRESS=(PROTOCOL=tcp)(HOST=spot)(PORT=1521))
TNS-12541: TNS:no listener
```

Reaching an Oracle Names Server To check whether a Oracle Names server can be reached, use a command using the Net8 address as in the following:

```
tnsping (address=(protocol=tcp)(host=fido)(port=1575))
```

A message similar to the following will be returned to the user:

```
TNS Ping Utility for SunOS:
Copyright (c) Oracle Corporation 1998. All rights reserved.
Attempting to contact (ADDRESS=(PROTOCOL=TCP)(HOST=fido)(PORT=1575))
OK (70 msec)
```

TRCROUTE Utility

The Trace Route Utility (TRCROUTE) enables administrators to discover what 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 Trace Route Utility displays the information collected on the client screen. You can redirect the TRCROUTE output to a file, and print it if you wish.

Requirements Trace Route works only over Net8 and SQL*Net version 2.3 and later. Every node along the route from client to server must use SQL*Net version 2.3 or later. If a pre-2.3 node is on the path, the following error is displayed:

```
TNS-03603:  Encountered a node with pre-2.3 version of SQL*Net
```

TRCROUTE shows what node along the path is responsible for any errors.

Effect on Performance The Trace Route Utility 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 TCROUTE To invoke TRCROUTE, enter the following from the command line:

```
trcroute net_service_name
```

The following are two examples of trace route output.

Example 11–1 shows a successful Trace Route packet that traveled from a client to a listener.

Example 11–1 Successful Trace Route

```
%trcroute tcp_direct
Trace Route Utility for Solaris:
Copyright (c) Oracle Corporation 1998.  All rights reserved.

Route of TRCROUTE:------------------
Node: Client            Time and address of entry into node:
-------------------------------------------------------------
01-DEC-98 13:26:36 ADDRESS= PROTOCOL=TCP  Host=shining-sun  Port=1581
Node: Server            Time and address of entry into node:
-------------------------------------------------------------
01-DEC-98 13:27:20 ADDRESS= PROTOCOL=TCP  Host=setting-sun  Port=1521
```

Example 11–2 shows an unsuccessful Trace Route packet that could not reach the listener because the listener was not up.

Example 11–2 Trace Route with Error

```
% trcroute tcp_direct
Trace Route Utility for SVR4:
Copyright (c) Oracle Corporation 1998.  All rights reserved.

Route of TRCROUTE:------------------
Node: Client            Time and address of entry into node:
-------------------------------------------------------------
01-DEC-98 11:12:34 ADDRESS= PROTOCOL=TCP  Host=shining-sun  Port=1581
TNS-12224: TNS:no listener
TNS-12541: TNS:no listener
TNS-12560: TNS:protocol adapter error
TNS-03601: Failed in route information collection
```

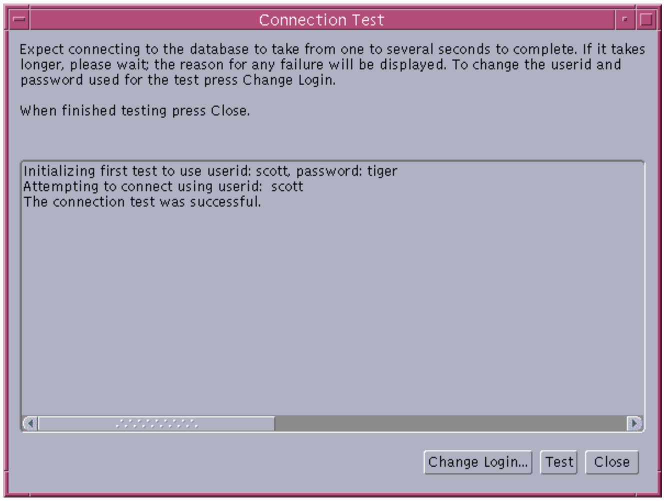
Net8 Assistant

To verify connectivity for a client machine, use Net8 Assistant:

1. Start Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
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 "Using Net8 Control Utilities" on page 11-3 to start components.

   A successful test results in the following message in the Connect Test dialog box:



Expect connecting to the database to take from one to several seconds to complete. If it takes longer, please wait; the reason for any failure will be displayed. To change the userid and password used for the test press Change Login.

When finished testing press Close.

```
Initializing first test to use userid: scott, password: tiger
Attempting to connect using userid:  scott
The connection test was successful.
```

   If the test was not successful:
   - Ensure the database and listener are running, then click Test.
   - Click Change Login to change the user name and password for the connection, then click Test.
5. Click Close to dismiss the Connect Test dialog box.

12

Troubleshooting

Net8 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 includes the following sections:

- Diagnosing Net8
- Resolving the Most Common Error Messages
- Troubleshooting Tips from the Field
- Troubleshooting Network Problems Using Log and Trace Files
- Logging Error Information
- Tracing Error Information
- Contacting Oracle Customer Support

Diagnosing Net8

If you have just completed installing and configuring Net8 and 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 Net8 with an error number or message that is not always indicative of the actual problem. This section helps you determine which parts of Net8 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.

Server Diagnostics

Answer the following questions: (You may need assistance from your server administrator to follow the instructions in this section.)

- 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 above questions / statements, skip this section and continue to "Client Diagnostics" on page 12-4.

If you are unsure, or answered NO to any of the above questions, please continue.

Diagnosing Net8 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 user name and password. For example:

```
SQLPLUS system/manager
```

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 `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows NT.

   The search order for configuration files is as follows:

   a. Current working directory from where an application is executed

   b. TNS_ADMIN environment variable

      If the TNS_ADMIN is not defined as an environment variable on Windows NT, it may be in the registry.

   c. If TNS_ADMIN is not defined, `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows platforms

2. Follow the instructions in "Testing Configuration on the Server" on page 11-15 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. Check the **Problem/Solution Database** website at `http://support.oracle.com` for more specific information on the error received, or contact Oracle Worldwide Support.

Client Diagnostics

At this point, you know the Net8 server side listener works properly, because you could verify at least one of the following statements:

- The server passed a loopback test, showing that the connection worked.
- Other machines connect also using Net8 to this same database.
- Connections from this workstation worked previous to making changes on this machine (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 Oracle protocol(s) as were installed on the server.

   On UNIX, you can run the `adapters` program to verify this. Run `adapters` at `$ORACLE_HOME/bin`.

Output similar to the following appears:

```
Installed Net8 Tranport Protocols are:

    IPC
    TCP/IP
    BEQueath
    SSL
    RAW
...
```

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 file transfer or terminal emulation utilities (FTP, TELNET, and PING) from the workstation to the server where the listener and database reside.
SPX	■ Perform a Netware log in to the machine on which the database is running. ■ Ensure you can map drives or use other Novell services such as Print Servers and File Servers on the Network. ■ Check that the listener service is broadcasting by issuing a DISPLAY SERVERS command from the Novell Server or any Novell File Server on the SPX network.
Named Pipes	■ See other computers or servers on the MSFT network. ■ Ensure you are able to share drives within the MSFT network.

3. Verify that all Net8 software has been installed to ensure that both the Net8 Client and the appropriate protocol are present.

4. Ensure that the client machine has the `tnsnames.ora` and the `sqlnet.ora` files in `$ORACLE_HOME/network/admin` on UNIX and *ORACLE_HOME*`\network\admin` on Windows platforms.

   The search order for `sqlnet.ora` and `tnsnames.ora` follows:

   a. Current working directory from where an application is executed

   b. TNS_ADMIN environment variable

      If the TNS_ADMIN environment variable is not defined as a variable on Windows NT, it may be in the registry.

c. If TNS_ADMIN is not defined, `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows platforms

If you have any other working client machines connecting to your selected Oracle database using Net8, back up your existing files and copy both the working `tnsnames.ora` and `sqlnet.ora` files from the working machine onto the non-working client workstations. This eliminates the possibility of errors in the files.

5. Test the Net8 layer as described in "Testing Network Connectivity from the Client" on page 11-17.

> **Note:** Do *not* to use TNSPING. TNSPING works just like the TCP/IP PING utility and will *not* create and open a socket, nor does it connect with the listener. It just ensures listener is present at the server side.

6. If the connection still fails:

   - Use tracing as described in the following section "Troubleshooting Network Problems Using Log and Trace Files" on page 12-16.
   - Check the **Problem/Solution Database** website at `http://support.oracle.com` for a specific diagnostics bulletin on the error received.
   - Contact Oracle Worldwide Support.

Resolving the Most Common Error Messages

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 and SQL*Forms, which depend on network services from Net8, will normally generate an error message.

A list of the most common network error messages follows:

- `ORA-12154: TNS:could not resolve service name`
- `ORA-12198: TNS:could not find path to destination`
- `ORA-12203: TNS:unable to connect to destination`
- `ORA-12224: TNS:no listener`
- `ORA-12533: TNS:illegal ADDRESS parameters`
- `ORA-12545: TNS:name lookup failure`
- `ORA-12560: TNS:protocol adapter error`
- `ORA-3113: TNS:End of file on communication channel`
- `ORA-3121: No interface driver connection - function not performed`

The following table describes each network error and outlines procedures to troubleshoot them.

Error #: Message	Description/Troubleshooting Procedures
`ORA-12154: TNS:could not resolve service name`	**Cause**: Net8 could not locate the net service name specified in the `tnsnames.ora` configuration file. **Actions**: **1.** Verify that a `tnsnames.ora` file exists. **See Also:** Step 4 on page 12-5 in "Client Diagnostics" on page 12-4 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. If you are not using domain names, and this parameter exists, delete it or disable it by commenting it out. **6.** If you are connecting from a login dialog box, verify that you are not placing an "@" symbol before your connect net service name. **7.** Activate client tracing and re-execute the operation.

Error #: Message	Description/Troubleshooting Procedures

`ORA-12198: TNS:could not find path to destination`
`ORA-12203: TNS:unable to connect to destination`
	Cause: The client is not able to find the desired database.
	Actions:
	1. Verify that you have entered the net service name you wish to reach correctly.
	2. Verify that the net service name ADDRESS parameters in the connect descriptor of your `tnsnames.ora` file are correct.
	3. Verify that your `tnsnames.ora` file is stored in the correct directory.
	See Also: Step 4 on page 12-5 in "Client Diagnostics" on page 12-4 for configuration file location information
	4. Verify that the listener on the remote node has started and is running. Enter:
	`lsnrctl`
	`lsnrctl> status` *`listener_name`*
	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 output indicates the listener is not running, try starting it with the command:
	`lsnrctl> start` *`listener_name`*
	5. If you are connecting from a login box, verify that you are not placing an "@" symbol before your connect net service name.

Error #: Message	Description/Troubleshooting Procedures

ORA-12203:
TNS:unable to connect to destination
	ORA-12203 error is a generic error that often shields secondary errors. For this reason, check the latest sqlnet.log file located in $ORACLE_HOME/network/log on UNIX and *ORACLE_HOME*\network\log on Windows platforms for secondary ORA messages. If after analyzing the log file you determine there are no secondary errors, determine if the problem may be caused by one the following scenarios.
	Cause: The incorrect Oracle protocol for the selected networking protocol is installed. A missing protocol support driver usually produces the following errors in the sqlnet.log or any client trace file:
	■ ORA-12203
	■ ORA-12538
	■ ORA-00508
	Action: Check that you have installed the appropriate Oracle protocol. On UNIX, you can run the adapters program to verify this. Run adapters at $ORACLE_HOME/bin.
	Output similar to the following appears:
	`Installed Net8 Tranport Protocols are:`
	`    IPC`
	`    TCP/IP`
	`    BEQueath`
	`    SSL`
	`    RAW`
	`...`

ORA-12203 continued	**Cause**: An invalid net service name was supplied in the connect string. **Action:** Verify that the net service name supplied in your connect string exists in your tnsnames.ora file and the ADDRESS information for that net service name is valid. Ask yourself the following questions: ■ Is the HOST or SERVICE name correct? ■ Is the PORT specified correct?

Error #: Message	Description/Troubleshooting Procedures

`ORA-12203` continued	**Cause**: Net8 could not find the connect descriptor specified in the `tnsnames.ora` file.
	Action: After verifying that the database is running, check the following:
	1. Verify the listener is running. Enter:
	`lsnrctl`
	`lsnrctl> status listener_name`
	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 output indicates the listener is not running, try starting it with the command:
	`lsnrctl> start listener_name`
	2. Ensure the `tnsnames.ora` file is in the correct location. See Step 4 on page 12-5 in "Client Diagnostics" on page 12-4 for configuration file location information.

`ORA-12203` continued	**Cause**: The destination system's listener is not listening.
	Action: Verify that the remote system's listener is running. Enter:
	`lsnrctl`
	`lsnrctl> status listener_name`
	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 output indicates the listener is not running, try starting it with the command:
	`lsnrctl> start listener_name`

`ORA-12203` continued	**Cause**: There are underlying network transport problems. **Action:** Verify with utilities supplied with the networking protocol being used that the protocol itself is functional. For example, with TCP/IP, try to PING the remote system.
`ORA-12203` continued	**Cause**: The `tnsnames.ora` file is not located in the proper directory. **Action:** Make sure the `tnsnames.ora` file is in the proper location. **See Also:** Step 4 on page 12-5 in "Client Diagnostics" on page 12-4 for configuration file location information

Error #: Message	Description/Troubleshooting Procedures
`ORA-12203` continued	**Cause**: The (HOST=*server_name*) parameter for TCP/IP addresses or (SERVICE=*tns_application*) parameter for SPX addresses is not consistent on the clients and server machines. **Action:** Ensure the values for these parameter are the same on the server and client. For TCP/IP setups, make sure that the HOST parameter in `listener.ora` on the server and in the `tnsnames.ora` file on the client point to the same name, or at least to names that are then translated to the same IP address by each system. This is especially important for servers with multiple IP addresses assigned to the various network interfaces on the server. For SPX setups, the name must be the same on the server and client workstations.
`ORA-12224: TNS:no listener`	**Cause**: The connection request could not be completed because the listener is not running. **Actions**: 1. Ensure that the supplied destination address matches one of the addresses used by the listener. 2. Verify also that this is not a version compatibility problem.
`ORA-12533: TNS:illegal ADDRESS parameters`	**Cause**: The protocol specific parameters in the ADDRESS section of the designated connect descriptor in your `tnsnames.ora` file are incorrect. **Action**: See "Protocol Parameters" on page B-3 for correct protocol syntax.
`ORA-12545: TNS:name lookup failure`	**Cause**: The listener on the remote node cannot be contacted. **Actions**: 1. Verify that the ADDRESS in the `tnsnames.ora` file and the `listener.ora` file is correct. 2. Verify that the listener on the remote node has been started. Enter: `lsnrctl` `lsnrctl> status` *`listener_name`* *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 output indicates the listener is not running, try starting it with the command: `lsnrctl> start` *`listener_name`*

Error #: Message	Description/Troubleshooting Procedures
`ORA-12560: TNS:protocol adapter error`	**Cause**: The listener was unable to start a process connecting the user to the database server. **Actions**: **1.** Turn on tracing and re-execute the operation. **2.** Evaluate the contents of the trace file to diagnose the problem. **See Also:** "Tracing Error Information" on page 12-33
`ORA-3113: TNS:End of file on communication channel`	**Cause**: 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. **Action**: You may need to modify your re-transmission count. For more information about troubleshooting this error, refer to the appropriate operating-system specific documentation.
`ORA-3121: 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: **Cause:** Only the user name and password were specified from a client machine that had no local Oracle database installed. **Action**: Specify a connect string.

Troubleshooting Tips from the Field

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 machine—**This eliminates any internal lookup problems and make the connection slightly faster.

  TCP/IP—Use the internet address rather than the host name in TCP/IP addresses, for example, 198.32.3.5. Change the (HOST =*server_name*) line in the `tnsnames.ora` file with the internet address, for example (HOST=198.32.3.5).

- **Consider possible SPX connection issues—**The workstation requesting a connection be made with a remote listener must first learn the location of that SPX service in the NetWare IPX network.

  The client workstation issues a lookup request for the SPX service. If the service cannot be found, an error is sent back to the workstation.

- **Perform a loopback test—**Perform a loopback test on the server as described in "Testing Configuration on the Server" on page 11-15. 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 machines are fine, the problem may be a timing issue.

- **Verify Whether or Not There Is a Timing Issue—**Timing issues are associated with ORA-12203, ORA-12535, or ORA-12547 errors in the client log files.

  To resolve this, try speeding up the connection by using exact addresses instead of names and increase the 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

Here are some questions to ask yourself when diagnosing a problem:

- **Do all machines have a problem, or is it just one?**

  If one machine works and another does not, and you are confident that the same software (Oracle and third-party products) is installed, 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.

- **Does the third-party application fail, but Oracle applications work?**

Troubleshooting Network Problems Using Log and Trace Files

Net8 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.

For server and listener, log files are by default located in `$ORACLE_HOME/network/log` on UNIX and *`ORACLE_HOME`*`\network\log` on Windows platforms, and trace files are by default located in `$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`\network\trace` on Windows platforms. For client, log and trace files are by default located in the current working directory.

Logging Error Information

All errors encountered in Oracle network products 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 covers the following topics:

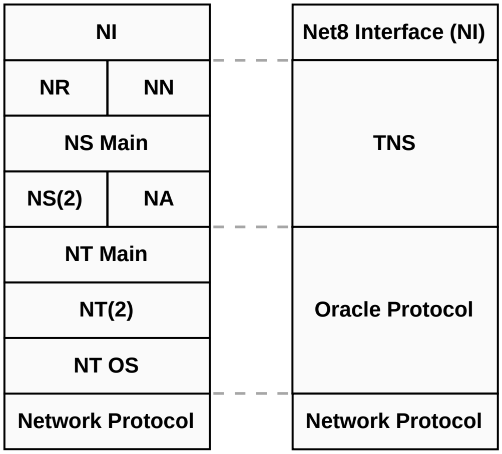
- Error Stacks
- Log File Names
- Understanding and Setting Log Parameters
- Setting Log Settings During Runtime of Control Utilities
- Using Log Files
- Listener’s Log Audit Trail Information
- Understanding Oracle Connection Manager Logs

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.

Figure 12–1 depicts the relationship among Oracle network products as they might appear in an error stack.

Figure 12–1 Network Products and Error Stack Components



The layers in Figure 12–1 are as follows:

NI	Net8 Interface Layer
NR	Network Routing
NN	Network Naming (Oracle Names)
NS	Network Session (main and secondary layers)
NA	Native Services includes Network Authentication (NA) and Network Encryption (NAE)
NT	Network Transport (main, secondary, and operating system layers)

Your network may or may not include all of these components.

Error Example

As an example, suppose that a user of a client application tries to establish a connection with a database server using Net8 and TCP/IP, and the user enters:

```
sqlplus scott/tiger@hrserver.com
```

The SQL*Plus banner is displayed on the screen, and the following error is displayed:

```
ORA-12203: 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, a log file called `sqlnet.log`, contains an error stack corresponding to the `ORA-12203` error as follows:

Example 12–1 Typical Error Stack

```
**************************************************

Fatal OSN connect error 12203, 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:
    nr err code: 12203
    TNS-12203: 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
```

Log File Names

Each Net8 component produces its own log file. The following table provides the default file names and a description of the information they contain:

Log File	Contains Error Information about the...
`sqlnet.log`	Client and / or server
`listener.log`	Listener
`names.log`	Oracle Names Server
`cman_`*`pid`*`.log` on UNIX `cman`*`pid`*`.log` on Windows NT	Oracle Connection Manager CMGW gateway process
`cmadm_`*`pid`*`.log` on UNIX `cmadm`*`pid`*`.log` on Windows NT	Oracle Connection Manager CMADMIN administrative process

Understanding and Setting Log 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 follows:

These log parameters corresponding to the...	...are set in the following configuration files
Client	`sqlnet.ora`
Server	`sqlnet.ora`
Listener	`listener.ora`
Oracle Names Server	`names.ora`
Oracle Connection Manager processes	`cman.ora`

This section cover the following topics:

- sqlnet.ora Parameters
- listener.ora Parameters
- names.ora Parameters
- cman.ora Parameters
- Setting Log Parameters in Configuration Files

See Also: Appendix C, "Configuration Parameters"

sqlnet.ora Parameters

The following parameters settings can be set in the `sqlnet.ora` file:

sqlnet.ora Parameter	Net8 Assistant Field	Description
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_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_DIRECTORY_SERVER	Server Information: Log Directory	Establishes the destination directory for the server log files. By default the server directory is `$ORACLE_HOME/network/log` on UNIX and *`ORACLE_HOME`*`\network\log` on Windows NT.
LOG_FILE_SERVER	n/a	Sets the name of the log file for the server. By default the log name is `sqlnet.log`.

listener.ora Parameters

The following log parameters can be set in the `listener.ora` file:

listener.ora Parameter	Net8 Assistant Field	Description
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 NT, and the file name is defaulted to `listener.log`.

names.ora Parameters

The following log parameters can be set in the `names.ora` file:

names.ora Parameter	Net8 Assistant Option	Description
NAMES.LOG_DIRECTORY	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 NT.
NAMES.LOG_FILE	Log File	Sets the name of the log file for the client. By default the log name is `names.log`.

cman.ora Parameters

The following log parameter can be set in the `cman.ora` file:

cman.ora Parameter	Description
LOG_LEVEL	Establishes the level of logging. Five levels are supported: ■ level 0 - no logging ■ level 1 - basic reporting ■ level 2 - RULE_LIST matching lookup reporting ■ level 3 - relay blocking reporting ■ level 4 - relay I/O counts reporting The CMGW gateway process creates a log file called `cman_`*`pid`*`.log` on UNIX and `cman`*`pid`*`.log` on Windows NT. The CMADMIN administrative process creates a log file called `cmadm_`*`pid`*`.log` on UNIX and `cmadm`*`pid`*`.log` on Windows NT.

Setting Log Parameters in Configuration Files

`sqlnet.ora` file, `listener.ora` files and `names.ora` file logging parameters can be set with the Net8 Assistant. `cman.ora` file logging parameters must be set manually.

> **See Also:** "Oracle Connection Manager Parameters (cman.ora)" on page C-70

To set logging parameters:

1. Start the Net8 Assistant:
   - On UNIX, run `netasst` at `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.
2. Specify the log parameters:

For this log file...	Set logging parameters here...
`sqlnet.log`	1. In the navigator pane, expand Local > Profile. 2. From the list in the right pane, select General. 3. Click the Logging tab. 4. Specify the settings.
`listener.log`	1. In the navigator pane, expand Local > Listeners. 2. Select a listener. 3. From the list in the right pane, select General. 4. Click the Logging and Tracing tab. 5. Specify the settings.
`names.log`	1. In the navigator pane, expand Oracle Names Servers. 2. Select an Oracle Names server. 3. From the list in the right pane, select Configure Server. 4. Click the Adv. tab. 5. Specify the log directory and file name.

3. Choose File > Save Network Configuration.

   The Net8 Assistant application exits.

Setting Log Settings During Runtime of Control Utilities

Logging can be set during a runtime of a control utilities. Note that setting logging with a control utility will 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 LSNRCTL control utility as described in "Listener Control Utility (LSNRCTL)" on page A-3.
- For an Oracle Connection Manager, use the SET LOG_LEVEL from the CMCTL control utility as described in "Oracle Connection Manager Control Utility (CMCTL)" on page A-79.

- For an Oracle Names server, use the SET LOG_FILE_NAME command from the NAMESCTL control utility as described in "Oracle Names Control Utility (NAMESCTL)" on page A-26, or set it through the Net8 Assistant:

  a. Start the Net8 Assistant:

  -On UNIX, run `netasst` at `$ORACLE_HOME/bin`.

  -On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

  b. In the navigator, expand the Oracle Names Servers folder.

  c. Select an Oracle Names server.

  d. From the list in the right pane, select Manage Server.

  e. Click the Logging tab.

  f. Specify the log directory and file name.

  g. Choose File > Save Network Configuration.

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 non-zero 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 re-execute the statement that produced the error message.

Understanding Listener Log Files

This section describes what is recorded in the listener log file, including:

- Listener's Log Audit Trail Information
- Service Registration Event Information
- Direct Hand-Off Information

Listener's Log Audit Trail Information

The listener log file contains audit trail information that allows you to gather and analyze network usage statistics, as well as information indicating the following:

- A client connection request
- A start, stop, status, reload or service command issued by the LSNRCTL Control Utility

Format of the Listener's 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 SID or service name information appear only when a connection is attempted
- A successful connection or command returns a code of 0
- A failure produces a code that maps to an error message

> **See Also:** "Resolving the Most Common Error Messages" on page 12-7 for the most common Net8 error or *Oracle8i Error Messages* for a complete listing of error messages

Typical output to the log file upon a STOP request is shown in Example 12–2:

Example 12–2 Listener Log Event for Successful Reload Request

```
14-SEP-1999 00:29:54 *
(CONNECT_
DATA=(CID=(PROGRAM=)(HOST=dlsun1013)(USER=jdoe))(COMMAND=stop)(ARGUMENTS=64)(SER
VICE=LISTENER)(VERSION=135290880))
* stop * 0
```

Typical output to the log file upon a connection request is shown in Example 12–3:

Example 12–3 Listener Log Event for Successful Connection Request

```
10-AUG-1999 15:28:58 *
(CONNECT_DATA=(service_name=sales.us.acme.com)(CID=(PROGRAM=)(HOST=dlsun1013)
(USER=jdoe)))
* (ADDRESS=(PROTOCOL=tcp)(HOST=144.25.185.246)(PORT=41349)) * establish
* sales.us.acme.com * 0
```

Using Audit Trail Information 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.

Service Registration Event Information

The listener records service registration events. During service registration, the instance background process PMON provides the listener with information about:

- 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

The following service registration-related events are recorded in the `listener.log` file:

Event	Description
SERVICE_REGISTER	The listener received registration information for an instance.
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 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 0, meaning 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" on page 12-7 for the most common Net8 error or *Oracle8i Error Messages* for a complete listing of error messages

Example 12–4 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.

Example 12–4 Listener Log with Service Registration Events

```
-------------------------------
10-AUG-1999 15:28:43 * service_register * sales * 0
10-AUG-1999 15:28:43 * service_register * sales * 0
10-AUG-1999 15:28:58 *
(CONNECT_DATA=(service_name=sales.us.acme.com)(CID=(PROGRAM=)(HOST=dlsun1013)
(USER=jdoe)))
* (ADDRESS=(PROTOCOL=tcp)(HOST=144.25.185.246)(PORT=41349)) * establish
* sales.us.acme.com * 0
10-AUG-1999 15:38:44 * service_update * sales * 0
10-AUG-1999 15:38:44 * service_update * sales * 0
10-AUG-1999 15:48:45 * service_update * sales * 0
10-AUG-1999 15:48:45 * service_update * sales * 0
10-AUG-1999 15:50:57 *
(CONNECT_DATA=(service_name=sales.us.acme.com)(CID=(PROGRAM=)(HOST=dlsun1013)(U
SER=jdoe)))
* (ADDRESS=(PROTOCOL=tcp)(HOST=144.25.185.246)(PORT=41365)) * establish
* sales.us.acme.com * 0
10-AUG-1999 15:51:26 * service_died * sales * 12537
10-AUG-1999 15:51:26 * service_died * sales * 12537
10-AUG-1999 15:52:06 *
```

```
(CONNECT_DATA=(service_name=sales.us.acme.com)(CID=(PROGRAM=)(HOST=dlsun1013)(U
SER=jdoe)))
* (ADDRESS=(PROTOCOL=tcp)(HOST=144.25.185.246)(PORT=41406)) * establish
* sales.us.acme.com * 12514
TNS-12514: TNS:listener could not resolve SERVICE_NAME given in connect
descriptor
--------------------------------
```

Direct Hand-Off Information

The listener records direct hand-off events to MTS dispatchers. 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 0
- A failure produces a code that maps to an error message

> **See Also:** "Resolving the Most Common Error Messages" on page 12-7 for the most common Net8 error or *Oracle8i Error Messages* for a complete listing of error messages

A direct hand-off event in the log file is shown in Example 12–5:

Example 12–5 Listener Log Event for Direct Hand-Off

```
21-MAY-1999 10:54:55 * oracle.aurora.net.SALESHttp2 * handoff * 0
```

Understanding Oracle Connection Manager Logs

The Oracle Connection Manager CMGW gateway process creates a log file called `cman_`*`pid`*`.log` on UNIX and `cman`*`pid`*`.log` on Windows NT. The CMADMIN administrative process creates a log file called `cmadm_`*`pid`*`.log`. The log files are located in `$ORACLE_HOME/network/log` on UNIX and *`ORACLE_HOME`*`\network\log` on Windows NT.

Example 12–6 cman_pid.log

```
(TIMESTAMP=30-OCT-98 18:03:10)(EVENT=10)(VERSION=8.1.6.0.0)
(TIMESTAMP=30-OCT-98 18:03:10)(EVENT=36)(rule_list=
(rule=(src=spcstn)(dst=x)(srv=x)(act=accept)))
(TIMESTAMP=30-OCT-98 18:03:10)(EVENT=32)(PARAMETER_LIST=(MAXIMUM_
RELAYS=1024)(RELAY_STATISTICS=no)(AUTHENTICATION_LEVEL=0)(LOG_LEVEL=1)(SHOW_TNS_
INFO=no)(ANSWER_TIMEOUT=0)(MAXIMUM_CONNECT_DATA=1024)(USE_ASYNC_
CALL=yes)(TRACING=no)(TRACE_DIRECTORY=default)(MAX_FREELIST_BUFFERS=0))
(TIMESTAMP=30-OCT-98 18:03:10)(EVENT=34)(ADDRESS_LIST=
(ADDRESS=(PROTOCOL=tcp)(HOST=)(PORT=1610)(QUEUESIZE=32)))
(TIMESTAMP=30-OCT-98 18:03:12)(EVENT=38)(COMMAND=2)
(TIMESTAMP=30-OCT-98
18:03:27)(EVENT=26)(RLYNO=0)(SRC=(ADDRESS=(PROTOCOL=tcp)(HOST=spcstn.us.oracle.c
om)(PORT=34758)))(DST=(ADDRESS=(PROTOCOL=tcp)(HOST=144.25.187.89)(PORT=1581)))
(TIMESTAMP=30-OCT-98 18:03:43)(EVENT=28)(RLYNO=0)(SINCE=30-OCT-98
18:03:27)(STATISTICS=(IN=(BYTES=0)(PACKETS=0)(DCDS=0)(OUT=(BYTES=0)(PACKETS=0)(D
CDS=0)))
```

Example 12–7 cmadm_pid.log

```
(TIMESTAMP=30-OCT-98 18:03:09)(EVENT=Sent Admin Status to UI)
(TIMESTAMP=30-OCT-98 18:03:10)(EVENT=CMan Registration)
```

cman_*pid*.log Event Codes

The `cman_pid.log` on UNIX and `cmanpid.log` on Windows NT reports events using event code numbers. The event code reported is dependent upon the log level set with the LOG_LEVEL parameter in the `cman.ora` file or with the CMCTL SET LOG_LEVEL command. This section explains what each of these event codes represents.

Log Level 1 Events

Code	Description
10	Gateway is starting up
12	Gateway is shutting down
14	Listening on TNS address(es)
18	Answer failed **See Also:** "Reasons for Event Code 18" on page 12-31
20	Refusing in-coming call **See Also:** "Reasons for Event Code 20" on page 12-31
26	Relay is now open
28	Relay is now closed
30	Statistics report
32	*<parameter_list>*
34	*<address_list>*
36	*<rule_list>*
38	CMCTL command
40	CMCTL command refused because the gateway is busy
42	Dead connection detected
44	Relay has timed out
11	Bad *<address_list>* argument
13	Bad *<parameter_list>* argument
15	Bad *<rule_list>* argument
23	Bad CMCTL record
25	Command line argument is too long

Code	Description
27	Memory allocation failure
29	TNS error
31	TNS error while processing CMCTL requests

Reasons for Event Code 18 The answer can fail due to the following:

Code	Description
1	Timed out
2	Connect data buffer is too small
3	Refused by TNS
4	TNS packet checksum error

Reasons for Event Code 20 The incoming call can be refused if:

Code	Description
1	Gateway is shutting down
1	Gateway is offline
3	No connect data on in-coming call
4	Bad connect data on in-coming call
5	All relays are in use
6	Unable to get relay buffers
7	Fatal TNS error
8	No available ASO service
9	Reject from rule filtering
10	Out-going call failed
11	Refused by Net8/TNS
12	Listener is not running
13	Listener is not reachable
14	Host name lookup failure

Code	Description
15	Protocol adapter (and probably the protocol stack) not loaded
16	No SOURCE_ROUTE set
17	Reject from rule or bad connect string data

Log Level 2 Events

Code	Description
102	Answering in-coming call
104	Making out-going call
105	Accepting in-coming call
106	Rule match report

Log Level 3 Events

Code	Description
202	Call will block (no asynchronous TNS support)
204	Relay blocked **See Also:** "Reasons for Event Code 204"
206	Buffer contains leftover data

Reasons for Event Code 204

Code	Description
1	Waiting for writer to be ready
2	Waiting for writer to clear backlog
3	WOULDBLOCK error on receive
4	WOULDBLOCK or PARTIAL error on send
5	Repeated WOULDBLOCK or PARTIAL send error

Log Level 4 Events

Code	Description
302	Read this many bytes
304	Wrote this many bytes
306	Wrote this many bytes on retry

Tracing Error Information

Tracing produces a detailed sequence of statements that describe network events as they are executed. Tracing an operation allows you to obtain more information on the internal operations of the components of Net8 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.

CAUTION: 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 covers the following topics:

- Trace File Names
- Understanding and Setting Trace Parameters
- Setting Trace Settings During Runtime of Control Utilities
- Evaluating Net8 Traces
- Using the Trace Assistant to Examine Your Trace Files

Trace File Names

Each Net8 component produces its own trace file. The following table provides the default file names and a description of the information they contain:

Trace File	Contains Error Information about the...
`sqlnet.trc`	Client
`svr_`*`pid`*`.trc`	Server
`listener.trc`	Listener
`names.trc`	Oracle Names Server
`cman_`*`pid`*`.trc` on UNIX `cman`*`pid`*`.trc` on Windows NT	Oracle Connection Manager CMGW gateway process
`cmadm_`*`pid`*`.trc` on UNIX `cmadm`*`pid`*`.trc` on Windows NT	Oracle Connection Manager CMADMIN administrative process

Understanding and Setting Trace Parameters

Parameters that control tracing, 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 follows:

These trace parameters corresponding to the...	...are set in the following Configuration Files
Client	`sqlnet.ora`
Server	`sqlnet.ora`
Listener	`listener.ora`
Oracle Names Server	`names.ora`
Oracle Connection Manager processes	`cman.ora`

This sections cover the following topics:

- sqlnet.ora Parameters
- listener.ora Parameters
- names.ora Parameters
- cman.ora Parameters
- Setting Trace Parameters in Configuration Files

> **See Also:** Appendix C for further information about these parameters

sqlnet.ora Parameters

The following parameters settings can be set in the `sqlnet.ora` file:

sqlnet.ora Parameter	Net8 Assistant Field	Description
TRACE_LEVEL_CLIENT	Client Information: Trace Level	Indicates 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 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 support.
TRACE_DIRECTORY_CLIENT	Client Information: Trace Directory	Establishes the destination directory for the client trace file. By default, the client directory is `$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`\network\trace` on Windows NT.
TRACE_FILE_CLIENT	Client Information: Trace File	Sets the name of the log file for the client. By default the log name is `sqlnet.trc`.

sqlnet.ora Parameter	Net8 Assistant Field	Description
TRACE_UNIQUE_CLIENT	Client Information: Unique Trace File Name	Determines whether or not a unique trace file is created for each client trace file. If the value is set to ON, a process identifier is appended to the name of each trace file generated, so that several can coexist. If the value is set to OFF, when a new trace file is created for a client, it overwrites the existing file. This parameter creates unique trace files named `sqlnet`*`pid`*`.trc`
TRACE_LEVEL_SERVER	Server Information: Trace Level	Indicates the level of detail the trace facility records for the server. The trace level value can either be a value within the range of 0 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 support.
TRACE_DIRECTORY_SERVER	Server Information: Trace Directory	Establishes the destination directory for the server trace file. By default, the server directory is `$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`\network\trace` on Windows NT.
TRACE_FILE_SERVER	Server Information: Trace File	Sets the name of the trace file for the client. By default the log name is `svr_`*`pid`*`.trc`.

You can also manually add the following optional tracing parameters for the TNSPING utility to `sqlnet.ora`. TNSPING determines whether or not a service (such as a database, an Oracle Names Server, or other TNS services) on a Net8 network can be successfully reached.

sqlnet.ora Parameter	Description
TNSPING.TRACE_LEVEL	Indicates 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 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 support.
TNSPING.TRACE_DIRECTORY	Establishes the destination directory for TNSPING trace file. By default, the directory is `$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`\network\trace` on Windows NT.

listener.ora Parameters

The following trace parameters can be set in the `listener.ora` file:

listener.ora Parameter	Net8 Assistant Field	Description
TRACE_LEVEL_*listener_name*	Trace Level	Indicates 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 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 support.
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 NT, and the file name is `listener.trc`.

names.ora Parameters

The following trace parameters can be set in the `names.ora` file:

names.ora Parameter	Net8 Assistant Option	Description
NAMES.TRACE_DIRECTORY	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 NT.
NAMES.TRACE_FILE	Trace File	Sets the name of the trace file. By default the trace name is `names.trc`.
NAMES.TRACE_LEVEL	not applicable	Indicates the level of detail the trace facility records for the Oracle Names server. The trace level value can either be a value within the range of 0 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 support.
NAMES.TRACE_UNIQUE	Make Trace File Unique	If set to ON, creates a unique file name for each trace session, allowing multiple trace files to coexist. A process identifier is appended to the name of the trace file generated. This parameter creates unique trace files named `names`*`pid`*`.trc`.

cman.ora Parameters

The following trace parameters can be set in `cman.ora` file:

cman.ora Parameter	Description
TRACING	Determines whether or not tracing is enabled for the Oracle Connection Manager: YES enables tracing for the Oracle Connection Manager. The CMGW gateway process creates a trace file called `cman_`*`pid`*`.trc` on UNIX and `cman`*`pid`*`.trc` on Windows NT. The CMADMIN administrative process creates a trace file called `cmadm_`*`pid`*`.trc` on UNIX and `cmadm`*`pid`*`.trc` on Windows NT.
TRACE_DIRECTORY	Establishes the destination directory for trace file. By default, the directory is `$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`\network\trace` on Windows NT.

Setting Trace Parameters in Configuration Files

`sqlnet.ora`, `listener.ora` and `names.ora` logging parameters can be set with the Net8 Assistant. `cman.ora` logging parameters must be set manually.

> **See Also:** "Oracle Connection Manager Parameters (cman.ora)" on page C-70

To set logging parameters:

1. Start the Net8 Assistant:
   - On UNIX, run `netasst` from `$ORACLE_HOME/bin`.
   - On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

2. Specify the trace parameters:

For this trace file...	Set tracing parameters here...
`sqlnet.trc` (for the client `svr_`*`pid`*`.trc` (for the server)	**1.** In the navigator pane, expand Local > Profile. **2.** From the list in the right pane, select General. **3.** Click the Tracing tab. **4.** Specify the settings.
`listener.trc`	**1.** In the navigator pane, expand Local > Listeners. **2.** Select a listener. **3.** From the list in the right pane, select General. **4.** Click the Logging and Tracing tab. **5.** Specify the settings:
`names.trc`	**1.** In the navigator pane, expand the Oracle Names Servers folder. **2.** Select an Oracle Names server. **3.** From the list in the right pane, select Configure Server. **4.** Click the Adv. tab. **5.** Specify the trace directory and file name.

3. Choose File > Save Network Configuration.

Setting Trace Settings During Runtime of Control Utilities

Tracing can be set during a runtime of a control utilities. Note that setting tracing with a control utility will 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_FILE, SET TRC_DIRECTORY and SET TRC_LEVEL commands from the LSNRCTL control utility as described in "Listener Control Utility (LSNRCTL)" on page A-3.
- For an Oracle Names server, use the SET TRACE_FILE_NAME and SET TRACE_LEVEL commands from the NAMESCTL control utility as described in "Oracle Names Control Utility (NAMESCTL)" on page A-26, or set it through the Net8 Assistant:

  1. Start the Net8 Assistant:

     -On UNIX, run `netasst` at `$ORACLE_HOME/bin`.

     -On Windows NT, choose Start > Programs > Oracle - *HOME_NAME* > Network Administration > Net8 Assistant.

  2. In the navigator pane, expand Oracle Names Servers.
  3. Select an Oracle Names server.
  4. From the list in the right pane, select Manage Server.
  5. Click the Logging tab.
  6. Specify the trace level, directory and file name.
  7. Choose File > Save Network Configuration.

Note: For Oracle Connection Manager, tracing can only be set from the Connection Manager configuration file as described in "Oracle Connection Manager Parameters (cman.ora)" on page C-70.

Evaluating Net8 Traces

Evaluating trace files either manually, or by using the Trace Assistant tool will help you to diagnose and troubleshoot network problems by giving you a better understanding of the following:

- The flow of packets between network nodes
- Which component of Net8 is failing
- Pertinent error codes

Understanding the Flow of Data Packets Between Network Nodes

Net8 performs its functions by sending and receiving data packets.By specifying a trace level of SUPPORT, you can view the actual contents of the Net8 packet in your trace file. The order of the packet types sent and received will help you to determine how your connection was established.

Understanding 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 Net8 packet keywords and describes the types of packets they represent:

Keyword	Packet Type
NSPTCN	Connect
NSPTAC	Accept
NSPTRF	Refuse
NSPTRS	Resend
NSPDA	Data
NSPCNL	Control
NSPTMK	Marker

> **Note:** This data is not viewable if you are using encryption through an Oracle network product or through EBCDIC data.

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 12–8 provides typical packet information:

Example 12–8 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)
```

Understanding Pertinent Error Output

Every time a problem occurs with the connection in Net8, the error code is logged in the trace file with the prefix of <ERROR> or <FATAL>. Example 12–9 depicts typical trace file error output.

Example 12–9 Trace File Error Output

```
nspsend: entry
nspsend: plen=244, type=6
ntpwr: entry
ntpwr: exit
-<ERROR>- nspsend: transport write error
nspsend: error exit
nserror: entry
-<ERROR>- nserror: nsres: id=0, op=65, ns=12541, ns2=12560; nt[0]=511,
nt[1]=61,nt[2]=0
-<ERROR>- nsopen: unable to open transport
nricdt: Call failed...
nricdt: exit
-<ERROR>- osnqper:  error from nricall
-<ERROR>- osnqper:  nr err code: 12203
-<ERROR>- osnqper:  ns main err code: 12541
-<ERROR>- osnqper:  ns (2)  err code: 12560
-<ERROR>- osnqper:  nt main err code: 511
-<ERROR>- osnqper:  nt (2)  err code: 61
-<ERROR>- osnqper:  nt OS   err code: 0
osnqme: entry
osnqme:  reporting nr (1) error: (12203) as rdbms err (12203)
osnqme: exit
-<ERROR>- onstns: Couldn’t connect, returning 12203
nricall: Exiting NRICALL with following termination result -1
nricall: exit
osnqme: entry
osnqme:  reporting nr (1) error: (12203) as rdbms err (12203)
osnqme: exit
-<ERROR>- onstns: Couldn’t connect, returning 12203
-<ERROR>- osnqper:  error from nricall
```

The most efficient way to evaluate error codes is to find the most recent NS error code logged. This is because 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 your connection.

For information about the specific return codes, use the Oracle UNIX error tool `oerr`. Use the `oerr` tool to discover more information about Net8 return codes, by entering the following at any command line prompt:

```
oerr tns error_number
```

Using the Trace Assistant to Examine Your Trace Files

Net8 provides a tool called the Trace Assistant to help you understand the information provided in your 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) SQL*Net or Net8 trace file.

To run the Trace Assistant, enter the following at any command line prompt:

```
trcasst [options] filename
```

The options are described next.

Option	Description
-o	Displays connectivity and Two Task Common (TTC) information. After the -o the following options may be used: ■ c (for summary connectivity information) ■ d (for detailed connectivity information) ■ u (for summary TTC information) ■ t (for detailed TTC information) ■ q (displays SQL commands enhancing summary TTC information)
-p	Oracle Internal Use Only
-s	Displays statistical information
-e	Enables display of 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)

If no options are provided, then the default is -odt -e -s, providing detailed connectivity, detailed Two-Task Common, error level 0, and statistics.

Example 12–10 shows how the Trace Assistant converts trace file information into a more readable format.

Example 12–10 Typical Trace Assistant Conversion

Trace File	Converted by Trace Assistant with option -e0 or -e1

`nsc2addr: normal exit`
`nsopen: entry`
`nsmal: 404 bytes at 0x10d5a48`
`nsopen: opening transport...`
`-<ERROR>- ntus2err: sd=13, op=1, resnt[0]=511, resnt[1]=2, resnt[2]=0`
`-<ERROR>- nserror: nsres: id=0, op=65, ns=12541, ns2=12560; nt[0]=511, nt[1]=2, nt[2]=0`
`-<ERROR>- nsopen: unable to open transport`
	Error found. Error Stack follows:
	id: 00000
	Operation code: 00065
	NS Error 1: 12541
	NS Error 2: 12560
	NT Generic Error: 00511
	Protocol Error: 00146
	OS Error: 00000
	
	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 re-execute 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.
	// Start the listener on the remote machine.

However, other errors may also exist within the trace file that were not logged from the `nserror` function.

Understanding Information Traversing the Network in Net8 Packets

Trace Assistant also allows you to view data packets from both the Net8 and **Two-Task Common (TTC)** communication layers. Trace Assistant offers you two options to view these packets:

- Summary connectivity (using option -oc)
- Detailed connectivity (using option -od)

Net8 Packet Examples

The following examples depict how Trace Assistant presents various packets as they are sent to and from the Net8 layer in a variety of transactions:

- Bequeathed connection
- Redirected connection
- Data packet

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 server. The connection request is always sent by the client, but received by the server (or listener).

Example 12–11 Summary Data Packets Sent in a Bequeathed Connection

Using trcasst -oc <*filename*>

This example shows two packets. The first is the connect packet that is sent from the client to the listener. The second is the accept packet coming back from the server.

```
--->  Send     192 bytes - Connect packet
        Connect data length: 142
(description=(address=(protocol=tcp)(host=dlsun)(port=1521))(connect_
data=(sid=db1)(cid=(program=)(host=dlsun)(user=use1))))
<---  received 24 bytes - accept packet
        Accept data length: 0
```

Example 12–12 Detailed Data Packets Sent in a Bequeathed Connection

Using trcasst -od <*filename*>

This example shows all of the details sent along with the connect data in negotiating a connection.

```
--->  Send     50 bytes - Connect packet
Current NS version number is: 309.
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: 2048
        Maximum TDU size: 5120
        NT protocol characteristics:
                Test for more data
                Spawner is running
                Hang on to Listener connection
                Full duplex I/O
                Urgent data support
                Generate SIGURG signal
                Handoff connection to another
        Line turnaround value: 0
        Connect data length: 234
        Connect data offset: 50
        Connect data maximum size: 2048
                Native Services wanted
                Native Services wanted
Cross facility item 1: 0
        Cross facility item 2: 0
        Connection id: 0x0000000000000000
        Packet data is in the following data packet
--->  Send     244 bytes - Data packet
(DESCRIPTION=(ADDRESS=(PROTOCOL=beq)(PROGRAM=/private/oracle/bin/oracle)(ARGV0=oracle)(A
RGS='(DESCRIPTION=(LOCAL=YES) (ADDRESS=(PROTOCOL=beq))))(DETACH=NO))(CONNECT_
DATA=(CID=(PROGRAM=)(HOST=dlsun)(USER=use1))))
<---  Received 24 bytes - Accept packet
        Accepted NS version number is: 307.
Global options for the connection:
        no attention processing
        Don't care
        Accepted maximum SDU size: 2048
        Accepted maximum TDU size: 4096
        Connect data length: 0
                Native Services wanted
                Native Services wanted
```

Example 12–13 Summary Data Packets Sent in a Redirected Connection

Using trcasst -oc <*filename*>

```
--->  Send     187 bytes - Connect packet
        Connect data length: 153
(DESCRIPTION=(CONNECT_
DATA=(SID=ap347db1)(CID=(PROGRAM=)(HOST=apsun)(USER=use2)))(ADDRESS_
LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=apsun)(PORT=1521))))
<---  Received 8 bytes - Resend packet
--->  Send     187 bytes - Connect packet
        Connect data length: 153
(DESCRIPTION=(CONNECT_DATA=(SID=apdb1)(CID=(PROGRAM=)(HOST=apsun)(USER=use2)))(ADDRESS_
LIST=(ADDRESS=(PROTOCOL=tcp)(HOST=apsun)(PORT=1521))))
<---  Received 24 bytes - Accept packet
        Accept data length: 0
```

Example 12–14 Data Packet

Using trcasst -oc <*filename*> or -od <*filename*>

Once the connection is established, data is given to Net8 from the Two-Task Common layer to be sent to the other node. Both summary and detailed views yield the same summary information.

```
--->  Send     30 bytes - Data packet
<---  Received 201 bytes - Data packet
--->  Send     439 bytes - Data packet
<---  Received 400 bytes - Data packet
```

Two Task Common Packet Examples

TTC handles requests such as open cursor, select rows, and update rows that are directed to the database. All requests are answered by the server. If you request to logon, a response is returned from the database that the request was completed. Example 12–15 through Example 12–17 on page 12-54 show the type of information you can expect.

Summary information for TTC 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 12–15 Two-Task Common Summary Information

Using trcasst -ou <*filename*>

```
(O3LOGA)         1st half of challenge-response logon        80       78
(O3LOGON)        2nd half of challenge-response logon        97       59
(OOPEN)          # 1                                         21       16
(OPARSEX)        # 1                                         245       59
(OCLOSE)         # 1                                         17       11
(OVERSION)                                                   29       16
(OOPEN)          # 2                                         21       16
```

On each line, the first item displayed is the actual request made.The second item is a cursor number, if one is involved with the transaction. The third item is either a listing of the flags or the SQL command that is being answered. The flag indicates that a request has the following characteristics:

!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

The number of bytes sent and received are displayed at the far right.

The OOPEN on line three is a prime example of how the output displays the request/response pairs. The OOPEN appears with a #1 following it indicating that an "Open cursor" request was sent from the client and the server responded with the cursor number 1 that it opened. Because a request/response pairing is placed on one line, you should not combine this option with any of the connectivity options.

Example 12–15 Two-Task Common Summary Information

Using trcasst -ou <*filename*>

```
(OALL7)     # 2 Parse Can Defn=2 Exec Fetch "SELECT A.V     268    100
(OOPEN)     # 3                                            21     16
(OALL7)     # 3 Parse Exec=1 "SELECT USER FROM SYS.DUAL    152    70
(OALL7)     # 3 Defn=1 Fetc                                117    88
(OCLOSE)    # 3                                            17     11
```

On each line, the first item displayed is the actual request made.The second item is a cursor number, if one is involved with the transaction. The third item is either a listing of the flags or the SQL command that is being answered. The flag indicates that a request has the following characteristics:

!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

The number of bytes sent and received are displayed at the far right.

The OOPEN on line three is a prime example of how the output displays the request/response pairs. The OOPEN appears with a #1 following it indicating that an "Open cursor" request was sent from the client and the server responded with the cursor number 1 that it opened. Because a request/response pairing is placed on one line, you should not combine this option with any of the connectivity options.

Example 12–16 Two-Task Common Summary Information

Using trcasst -ot <*filename*>

```
start of user function (TTIFUN)
        1st half of challenge-response logon (O3LOGA)
          Username: applsys
          Terminal: ttyp5
          Machine: ap207sun
          System User: mwarren
          Process: 24459
          Program: aiap45@ap207sun (TNS interface)
return opi parameter (TTIRPA)
        OPI parameter: 3309B1A977A62A3C
start of user function (TTIFUN)
        2nd half of challenge-response logon (O3LOGON)
          Username: applsys
          Terminal: ttyp5
          Machine: ap207sun
          System User: mwarren
          Process: 24459
          Program: aiap45@ap207sun (TNS interface)
ORACLE function complete (TTIOER)
start of user function (TTIFUN)
        Open a cursor
return opi parameter (TTIRPA)
        Cursor #: 1
start of user function (TTIFUN)
        Parse and Execute (OPARSEX) Cursor # 1
alter session set nls_language= 'AMERICAN' nls_territory= 'AMERICA' nls_currency=
'$' nls_iso_currency= 'AMERICA' nls_numeric_characters= '.,' nls_date_format=
'DD-MON-YY' nls_date_language= 'AMERICAN' nls_sort= 'BINARY'
ORACLE function complete (TTIOER)
start of user function (TTIFUN)
        Close cursor (OCLOSE) Cursor # 1
V6 Oracle func complete (TTISTA)
        Succeeded
```

Example 12–17 Detailed SQL Information on Top of Summary Two-Task

Using trcasst -ouq <*filename*>

```
(O3LOGA)        1st half of challenge-response logon                   180      78

(O3LOGON)       2nd half of challenge-response logon                   197      59

(OOPEN)         # 1                                                    21       16

(OPARSEX)       # 1 alter session set nls_language= 'AMERICAN' nls_    245       59
                territory= 'AMERICA' nls_currency= '$' nls_iso_
                currency= 'AMERICA' nls_numeric_characters= '.,' nls_
                date_format= 'DD-MON-YY' nls_date_language=
                'AMERICAN' nls_sort= 'BINARY'

(OCLOSE)        # 1                                                    17       11

(O71SESOPN)     (get session ID)                                       47       18

(OOPEN)         # 1                                                    21       16

(OVERSION)      Oracle7 Server Release 8.1.3.0.0 - Production Release  29       157
                with the distributed and parallel query optionsPL/SQL
                Release 2.1.4.0.0 - Production

(O71SESOPN)     (get session ID)                                       47       18
```

Add q to your summary Two-Task Command to display the detailed SQL information given automatically in the detailed Two-Task option.

Analyze the Data Collected into Appropriate Statistics

The type of statistics gathered is on the order of how many calls (TTC), packets and bytes were sent and received between the network partners. The following example depicts typical trace file statistics:

Example 12–18 Typical Trace File Statistics

Using trcasst -s <*filename*>

```
======================================================================
     Trace File Statistics:
    ----------------------------------------------------------------------
     Net8:
          Total Calls:       466 sent,       491 received,       423 upi
          Total Bytes:    119214 sent,     86614 received
        Average Bytes:       255 sent,       176 received
        Maximum Bytes:      2048 sent,      2048 received
     GRAND TOTAL PACKETS  sent:    466     received:    491
```

Trace File Example

The following example shows a full trace file decoded. This example was created using the Oracle client tool SQL*Plus with the request:

```
connect scott/tiger@june
```

The message `ORA-12154: TNS:could not resolve service name` was displayed on the screen.

Example 12–19 Trace File Example

Description	Trace File Information

Note Trace level and location of the trace file in the Trace Configuration Information section.	`-- TRACE CONFIGURATION INFORMATION FOLLOWS ---`
	`New trace stream is "C:\ORANT\network\trace\net8.trc"`
	`New trace level is 16`
	`--- TRACE CONFIGURATION INFORMATION ENDS ---`

The Oracle Names component cannot find net service name `june.com`. Note client adds the `com` extension to the net service name `june`.	`nnfotran: tnsname.ora entry for name "june.com" not found`
	`nnftqnm: Error querying june.com of attribute A.SMD errcode 406`
	`nnfgrwsp: Query unsuccessful, skipping to next adapter`

Example 12–19 Trace File Example

Description	Trace File Information
Client attempts to access the Oracle Names server `oranamesrvr0` to resolve net service name address.	nnfgrwsp: Switching to ONAMES adapter nnfgrwsp: Original name: june nnfgrwsp: Qualified name: june.com nngsget_get_stream: looking for "(DESCRIPTION=(CONNECT_ DATA=(RPC=ON))(ADDRESS=(PROTOCOL=tcp)(HOST=oranamesrvr0)(PORT =1575)))" nngsget_get_stream: cache miss, opening new stream nngsnad_new_stream_addr: "(DESCRIP TION=(CONNECT_ DATA=(RPC=ON))(ADDRESS=(PROTOCOL =tcp)(HOST=oranamesrvr0)(PORT=1575)))" nngsget_get_stream: no caller address will be sent to callee
Network Routing (NR) performs routing to `oranamesrvr0`.	nricall: entry nric2a: entry nric2a: Getting local community information nriglp: entry nriglp: Looking for local addresses setup by nrigla nriglp: No addresses in the preferred address list nriglp: exit nric2a: TNSNAV.ORA is not present. No local communities entry. nrigla: entry nrigla: Getting local address information nrigla: Simple address... nrigla: No community component so just use straight address nrigla: exit nridst: entry nridst: Resolving address to use to call destination or next hop nridst: Found destination address nridst: Local address nridst: Local destination community found nridst: exit nric2a: This is a local community access nric2a: exit nricall: Got routable address information nricall: Making call with following address information: (DESCRIPTION=(CONNECT_ DATA=(RPC=ON))(ADDRESS=(PROTOCOL=tcp)(HOST=oranamesrvr0)(PORT =1575))) nricdt: entry nricdt: Calling with outgoing connect data (DESCRIPTION=(CONNECT_ DATA=(RPC=ON))(ADDRESS=(PROTOCOL=tcp)(HOST=oranamesrvr0)(PORT =1575)))

Example 12–19 Trace File Example

Description	Trace File Information
Network Session (NS) sets up the session to `oranamesrvr0`.	nscall: entry nscall: connecting... nsc2addr: entry nsc2addr: (DESCRIPTION=(CONNECT_ DATA=(RPC=ON))(ADDRESS=(PROTOCOL=tcp)(HOST=oranamesrvr0)(PORT =1575)))
Network Transport (NT) sets up the transport session.	nttbnd2addr: entry nttbnd2addr: port resolved to 1575 nttbnd2addr: looking up IP addr for host: oranamesrvr0 nttbnd2addr: exitnsopen: entry nsmal: entry nsmal: 330 bytes at 0x30d76e74 nsmal: normal exit nsopen: opening transport... nttcon: entry nttcon: toc = 1 nttcnp: entry nttcnp: creating a socket. nttcnp: exit nttcni: entry nttcni: trying to connect to socket 1. ntt2err: entry
Network Transport (NT) returns the error "no listener" as the Oracle Names Server is not running.	-<ERROR>- ntt2err: soc 1 error - operation=1, ntresnt[0]=511, ntresnt[1]=61 ntresnt[2]=0 ntt2err: exit nttcni: exit nttcon: exit nserror: entry
The error is propagated to the next layer (NS).	-<ERROR>- nserror: nsres: id=0, op=65, ns=12541, ns2=12560; nt[0]=511, nt[1]=61,nt[2]=0 -<ERROR>- nsopen: unable to open transport nsmfr: entry nsmfr: 330 bytes at 0x30d76e74 nsmfr: normal exit nsopen: error exit nscall: error exit nricdt: Call failed... nricfg: entry nricfg: exit nricdt: Call made to destination nricdt: exit

Example 12–19 Trace File Example

Description	Trace File Information
	nricall: Failed to copy originating community name value binding nricall: Exiting NRICALL with following termination result -1 nricall: exit nngsfad_free_stream_addr: "(DESCRIPTION=(CONNECT_DATA=(RPC=ON))(ADDRESS=(PROTOCOL=tcp)(HOST=oranamesrvr0)(PORT=1575)))" -<ERROR>- nngsget_get_stream: open failure, error stack follows
The errors are propagated to the next layer (TNS)	TNS-12224: TNS:no listener TNS-12541: TNS:no listener TNS-12560: TNS:protocol adapter error TNS-00511: No listener nnfgrwsp: Query unsuccessful, skipping to next adapter
The address is not found on any Oracle Names Server as no Oracle Names Server is available.	nnfun2a: address for name "june" not found nngsfad_free_stream_addr: "(DESCRIPTION = (CONNECT_DATA= (RPC=ON)) (ADDRESS=(PROTOCL=tcp) (HOST=oranamesrvr0) (PORT=1575)))" nngtdei_deinit_msg: free message pool block nngtfms_free_msg: message ID -10429 nngtfms_free_msg: message was a request nngtfms_free_msg: message free, type 100 nngtfoa_free_objarr: free message object array nngtfmt_free_msg_type: type-specific message free, type 100 nngtfoa_free_objarr: free message object array nngtfms_free_msg: message ID 0 nngtfms_free_msg: message was a response nngtfms_free_msg: message free, type 0 nngsdei_deinit_streams: deinit nngscls_close_stream: UID 11 not established, ignored nngscls_close_stream: UID 0 not established, ignored osnqrn: Return code from nnfsn2a is 409
Error is returned to the user.	-<ERROR>- onstns: Couldn't connect, returning 12154 onstns: exit osnqtg: Count in the OSN global area is now 0 rigbd: entry nrigbd: exit osnqtg: Count in the NL global area is now 0

Trace File Example Summary

This trace file provides a summary of what occurs with Net8 when you encounter the error "`ORA-12154: Could not resolve service name`". In this example, a client is unsuccessful in making a connection to net service name `june`. This is because a NAMES.DEFAULT_DOMAIN = COM parameter setting exists in the `sqlnet.ora` file. This parameter adds the `com` extension to all net service names requested, including the net service name `june`. Unfortunately, this net service name is defined in neither the client's `tnsnames.ora` file, nor an Oracle Names server. To troubleshoot this problem, the user should perform one of the following:

- Edit the `sqlnet.ora` file to remove the NAMES.DEFAULT_DOMAIN configuration parameter
- Request a connection to `june.com` instead of `june`

Contacting Oracle Customer Support

If you are still unable to resolve your problems or if you are requested to contact Oracle Customer Support to report the error, please have the following information at hand:

- The hardware and operating system release number on which your application(s) 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
- A Net8 trace, if possible; if not, the log file is sufficient

Part IV

Reference

Part IV provides reference material for this guide. It contains the following appendices:

- Appendix A, "Control Utilities"
- Appendix B, "Protocol Addresses"
- Appendix C, "Configuration Parameters"
- Appendix D, "LDAP Schema for Net8"

A

Control Utilities

Net8 provides control utilities to control the listener, Oracle Names and Oracle Connection Manager. This appendix lists the commands that are available with each utility, including any applicable prerequisites, passwords, syntax or argument rules, and usage notes or examples to help you use them.

The three control utilities described in this appendix are:

- Listener Control Utility (LSNRCTL)
- Oracle Names Control Utility (NAMESCTL)
- Oracle Connection Manager Control Utility (CMCTL)

Operating Modes

The control utilities operate for the duration of a runtime session. You can run utilities in one of three modes:

- Interpreter mode—The control utility is loaded from the control utility command line. When loaded, the program displays a prompt, such as:

  ```
  LSNRCTL>
  ```

- Command line mode—You can also execute most commands from the operating system command line by running the control utility with a complete command as a parameter to the program. In this case, the control utility will load and execute the command, then return to the operating system prompt. Sample commands are:

  ```
  NAMESCTL start
  NAMESCTL status sales.com
  ```

- Batch command mode—You can combine commands in a standard text file, then run them as a sequence of commands. To execute in batch mode, use the format:

  ```
  control_utility @file_name
  ```

- You can use either REM or # to identify comments in the batch script; all other lines are considered commands. Any commands that would typically require confirmation do not require confirmation during batch execution.

Listener Control Utility (LSNRCTL)

You can manage the listener with the Listener Control Utility (LSNRCTL). The general syntax of the LSNRCTL is as follows:

```
LSNRCTL command [listener_name]
```

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.

LSNRCTL looks for a `listener.ora` file in the directory defined by the TNS_ADMIN environment variable or in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows NT. It looks for a listener named LISTENER. If LSNRCTL cannot find an entry for a listener named LISTENER, unless specified otherwise, it assumes that the command is to be applied to the default listener—that is, a listener named LISTENER listening on TCP/IP, port 1521 and IPC, key PNPKEY.

LSNRCTL contains several types of commands:

- Operational commands such as START, STOP, and so forth.
- Modifier commands, such as SET *command*
- Informational commands, such as STATUS, SHOW *command*
- Command utility operational commands such as EXIT, QUIT, and HELP

You can use any of these commands to perform basic management functions on one or more listeners. Additionally, you can view and change listener parameter settings.

SET and SHOW Commands

You can use the SET command to change some parameter values for a listener or the LSNRCTL environment during the LSNRCTL control utility session. You can then save those values or settings with the SAVE_CONFIG command. You can use the SHOW command to display the current value of a configuration setting.

Security

If the PASSWORDS_*listener_name* parameter is set in the `listener.ora` file or the CHANGE_PASSWORD command has been used to create a new, encrypted password, the LSNRCTL control utility will require a SET PASSWORD command for any sensitive operations, such as stopping the listener.

LSNRCTL Commands

The following commands are available through the LSNRCTL:

CHANGE_PASSWORD

Purpose:

Enables you to dynamically change the encrypted password of a listener set with the PASSWORDS_*listener_name* parameter in the `listener.ora` file

This command does not change unencrypted passwords already established in with the PASSWORDS_*listener_name* parameter in the `listener.ora` file. It only establishes a new password, or changes a password that has been encrypted in the `listener.ora` file.

Prerequisites:

None

Password required if one has been set:

Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:

From the operating system:

```
LSNRCTL change_password [listener_name]
```

From the LSNRCTL utility:

```
LSNRCTL> change_password [listener_name]
```

Arguments:

Listener name, if the default name of LISTENER is not used

Usage Notes:

The control utility prompts you for your old password, then for the new one. It asks you to re-enter the new one, then changes it. Neither the old nor the new password displays during this procedure.

Example:

```
LSNRCTL> change_password
Old password:
New password:
Reenter new password:
Connecting to (ADDRESS=(PROTOCOL=ipc)(KEY=iris))
Password changed for LISTENER
The command completed successfully
```

DBSNMP_START

Purpose: Starts the SNMP subagent for an Oracle database running on the same node

Prerequisites: The command must be run locally.

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL dbsnmp_start
```

From the LSNRCTL utility:

```
LSNRCTL> dbsnmp_start
```

Arguments: None

Usage Notes: None

Example:

```
LSNRCTL> dbsnmp_start
```

DBSNMP_STATUS

Purpose: Verifies whether the SNMP subagent for an Oracle database is running

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL dbsnmp_status
```

From the LSNRCTL utility:

```
LSNRCTL> dbsnmp_status
```

Arguments: None

Usage Notes: DBSNMP STATUS must be run on the same node the Oracle database is on.

Example:

```
LSNRCTL> dbsnmp_status
```

DBSNMP_STOP

Purpose: Stops the SNMP subagent for an Oracle database running on the same node

Prerequisites: Must be run locally

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL dbsnmp_stop
```

From the LSNRCTL utility:

```
LSNRCTL> dbsnmp_stop
```

Arguments: None

Usage Notes: None

Example:

```
LSNRCTL> dbsnmp_stop
```

EXIT

Purpose: Quits LSNRCTL, and returns you to the operating system prompt

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL exit
```

From the LSNRCTL utility:

```
LSNRCTL> exit
```

Arguments: None

Usage Notes: This command is identical to the QUIT command.

Example:

```
LSNRCTL> exit
```

HELP

Purpose: Provides a list of all the LSNRCTL commands available. In response to one of the HELP commands, LSNRCTL displays help on how to use the command.

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL help
LSNRCTL help [command]
```

From the LSNRCTL utility:

```
LSNRCTL help
LSNRCTL> help [command]
```

Arguments: HELP command. Commands are shown in the following example output.

Example:

```
LSNRCTL> help
The following operations are available
An asterisk (*) denotes a modifier or extended command:
change_password
dbsnmp_start
dbsnmp_stop
dbsnmp_status
exit
quit
reload
services
set*
show*
spawn
start
status
stop
trace
version
```

QUIT

Purpose: Exits LSNRCTL, and returns you to the operating system prompt

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL quit
```

From the LSNRCTL utility:

```
LSNRCTL> quit
```

Arguments: None

Usage Notes: This command is identical to the EXIT command.

Example:

```
LSNRCTL> quit
```

RELOAD

Purpose: Shuts down everything except listener addresses, and re-reads the `listener.ora` file. This command enables you to add or change services without actually stopping the listener.

Prerequisites: This will not work on valid nodes. In this case, the listener must be stopped and restarted.

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
LSNRCTL reload [listener_name]
```

From the LSNRCTL utility:

```
LSNRCTL> reload [listener_name]
```

Arguments: Listener name, if the default name of LISTENER is not used

Example:

```
LSNRCTL> reload
```

SAVE_CONFIG

Purpose: Compares the current configuration state of the listener, including trace level, trace file, trace directory, logging, and connect timeout, to the `listener.ora` file. Any changes are stored in the `listener.ora`, preserving formatting, comments, and case as much as possible. Prior to modification of the `listener.ora` file, a back up of the file, called `listener.bak`, is created.

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
LSNRCTL save_config [listener_name]
```

From the LSNRCTL utility:

```
LSNRCTL> save_config [listener_name]
```

Arguments: Listener name, if the default name of LISTENER is not used

Usage Notes: This is used by an administrator to save all on-line configuration changes to the listener configuration file. This is especially useful for saving changed encrypted passwords.

Example:

```
LSNRCTL> save_config listener
```

SERVICES

Purpose: Provides detailed information about the service handlers (dispatchers, dedicated servers, and prespawned dedicated servers) the listener forwards client connection requests to, including how many connections have been established and refused

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
LSNRCTL services [listener_name]
```

From the LSNRCTL utility:

```
LSNRCTL> services [listener_name]
```

Arguments: Listener name, if the default name of LISTENER is not used

SERVICES

Usage Notes:

The SET DISPLAYMODE command changes the format and the detail level of the output.

Example:

The example below shows SERVICES output in NORMAL mode. The output shows:

- An instance named `ORCL` belongs to two services, `s1` and `s2`**,** and has a total of three service handlers.
- Service `s1` is handled by one dispatcher only (and this dispatcher handles only clients requesting service `s1`), as specified by `Relevant handlers: 1` in the output below.
- Service `s2` is handled by one dispatcher and one dedicated server, as specified by `Relevant handlers: 2` in the output below.

```
LSNRCTL> set displaymode normal
LSNRCTL> services
Connecting to
(DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=net)(QUEUESIZE=32)))
Services Summary...
Service "s1"            has 1 instances.
    Instance "ORCL"
      Status: READY  Total handlers: 3  Relevant handlers: 1
        D000 established:0 refused:0 current:0 max:1022
state:ready

(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1013.us.oracle.com)(PORT=15
82))
            Session: NS
Service "s2"            has 1 instances.
    Instance "ORCL"
      Status: READY  Total handlers: 3  Relevant handlers: 2
        D0001 established:0 refused:0 current:0 max:1022
state:ready

(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1013.us.oracle.com)(PORT=53
101))
            Session: NS
        DEDICATED established:0 refused:0 current:0 max:1
state:ready
            Session: NS
The command completed successfully
```

SET

Purpose: Lists the available configuration commands that can be set for the current LSNRCTL session.

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL set
LSNRCTL set [command]
```

From the LSNRCTL utility:

```
LSNRCTL> set
LSNRCTL set [command]
```

Arguments: SET commands. Commands are shown in the following example output.

Example:

```
LSNRCTL> set
The following operations are available with set.
An asterick (*) denotes a modifier or extended command.
connect_timeout
current_listener
displaymode
log_file
log_directory
log_status
password
raw_mode
save_config_on_stop
startup_waittime
trc_file
trc_directory
trc_level
use_plugandplay
```

SET CONNECT_TIMEOUT

Purpose: Specifies the amount of time in seconds the listener will wait for a valid connection request after a connection has been started

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL set connect_timeout {time}
```

From the LSNRCTL utility:

```
LSNRCTL> set connect_timeout {time}
```

Arguments: Time in seconds

Usage Notes: None

Example:

```
LSNRCTL> set connect_timeout 20
Connecting to (ADDRESS=(PROTOCOL=ipc)(KEY=iris))
LISTENER parameter "connect_timeout" set to 20
The command completed successfully
```

SET CURRENT_LISTENER

Purpose: Enables you to set the name of a listener that you want to set or show parameters

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL set current_listener [listener_name]
```

From the LSNRCTL utility:

```
LSNRCTL> set current_listener [listener_name]
```

Arguments: Listener name, if the default name of LISTENER is not used

Usage Notes: When SET CURRENT_LISTENER is set, LSNRCTL commands act on the listener you set. You do not have to specify the name of the listener.

Example:

```
LSNRCTL> set current_listener [listener_name]
```

SET DISPLAYMODE

Purpose: Changes the format and level of detail for the LSNRCTL SERVICES and LSNRCTL STATUS commands

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL set displaymode [argument]
```

From the LSNRCTL utility:

```
LSNRCTL> set displaymode [argument]
```

Arguments: COMPAT (default) - Displays output that is compatible with older versions of the listener

NORMAL - Displays output in a formatted and descriptive output. Oracle recommends this mode

VERBOSE - Displays all data received from the listener in a formatted and descriptive output

RAW - Displays all data received from the listener without any formatting. This output should only be used if recommended by Oracle WorldWide Support.

Example:

```
LSNRCTL> set displaymode normal
```

SET LOG_DIRECTORY

Purpose: Sets the directory where the listener log file is written. By default, the log files are written to `$ORACLE_HOME/network/log` on UNIX and `ORACLE_HOME/network/log` on Windows NT.

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
LSNRCTL set log_directory {directory}
```

From the LSNRCTL utility:

```
LSNRCTL> set log_directory {directory}
```

Arguments: Directory path

SET LOG_DIRECTORY

Example:

```
LSNRCTL> set log_directory /usr/oracle/admin
Connecting to (ADDRESS=(PROTOCOL=ipc)(KEY=iris))LISTENER
parameter "log_directory" set to /usr/oracle/admin
The command completed successfully
```

SET LOG_FILE

Purpose: Sets the name for the listener log file. By default, the log file name is `listener.log`.

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
LSNRCTL set log_file {file_name}
```

From the LSNRCTL utility:

```
LSNRCTL> set log_file {file_name}
```

Arguments: File name

Example:

```
LSNRCTL> set log_file list.log
Connecting to (ADDRESS=(PROTOCOL=ipc)(KEY=iris))LISTENER
parameter "log_file" set to list.log
The command completed successfully
```

SET LOG_STATUS

Purpose: Turns listener logging on or off

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
LSNRCTL set set log_status [argument]
```

From the LSNRCTL utility:

```
LSNRCTL> set set log_status [argument]
```

Arguments: ON | OFF (default)

Example:

```
LSNRCTL> set log_status on
```

SET PASSWORD

Purpose:

Sets the password for privileged LSNRCTL utility commands, such as SAVE_CONFIG and STOP

The password entered should match one set for the PASSWORDS_listener_name parameter in the `listener.ora` file or the LSNRCTL utility's CHANGE_PASSWORD command.

Syntax:

From the operating system:

```
LSNRCTL set password
```

From the LSNRCTL utility:

```
LSNRCTL> set password
```

The syntax for unencrypted passwords is:

`set password` *`password`*

Arguments:

Password

Usage Notes:

You can enter this command when you start up the shell or any time during your session. (You must enter the SET PASSWORD command before you can stop the listener.)

The preferred, secure way to enter your password is in interactive mode.

Example:

```
LSNRCTL> set password
enter listener password: password
```

SET SAVE_CONFIG_ON_STOP

Purpose:

Specifies whether or not changes made by the LSNRCTL SET commands are saved to the `listener.ora` file after the LSNRCTL session has ended

The saving of all values occurs right before the LSNRCTL session exits, taking as much care as possible to preserve the formatting, comments, and letter case. To have all parameters saved right away, use the SAVE_CONFIG command.

Any changes are stored in the `listener.ora` file, preserving formatting, comments, and case as much as possible. Prior to modification of the `listener.ora` file, a back up of the file, called `listener.bak`, is created.

Password required if one has been set:

No

SET SAVE_CONFIG_ON_STOP

Syntax:

From the operating system:

```
LSNRCTL set save_config_on_stop  [argument]
```

From the LSNRCTL utility:

```
LSNRCTL> set save_config_on_stop  [argument]
```

Arguments:

ON | OFF (default)

Example:

```
LSNRCTL> set save_config_on_stop on
```

SET STARTUP_WAITTIME

Purpose:

Sets the amount of time the listener sleeps before responding to a START command

Prerequisites:

None

Password required if one has been set:

Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:

From the operating system:

```
LSNRCTL set startup_waittime [time]
```

From the LSNRCTL utility:

```
LSNRCTL> set startup_waittime [time]
```

Arguments:

Time in seconds

Example:

```
LSNRCTL> set startup_waittime 10
Connecting to (ADDRESS=(PROTOCOL=ipc)(KEY=iris))LISTENER
parameter "startup_waittime" set to 10
The command completed successfully
```

SET TRC_DIRECTORY

Purpose: Sets the directory where the listener trace file is written. By default, the trace file(s) are written to `$ORACLE_HOME/network/trace` on UNIX and `ORACLE_HOME/network/trace` on Windows NT.

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL set trc_directory {directory}
```

From the LSNRCTL utility:

```
LSNRCTL> set trc_directory {directory}
```

Arguments: Directory path

Example:

```
LSNRCTL> set trc_directory /usr/oracle/admin
Connecting to (ADDRESS=(PROTOCOL=ipc)(KEY=iris))
LISTENER parameter "trc_directory" set to /usr/oracle/admin
The command completed successfully
```

SET TRC_FILE

Purpose: Sets the name for the listener trace file. By default, the trace file name is `listener.trc`.

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL set trc_file {file_name}
```

From the LSNRCTL utility:

```
LSNRCTL> set trc_file {file_name}
```

Arguments: File name

Example:

```
LSNRCTL> set trc_file list.trc
Connecting to (ADDRESS=(PROTOCOL=ipc)(KEY=iris))
LISTENER parameter "trc_file" set to list.trc
The command completed successfully
```

SET TRC_LEVEL

Purpose: Sets tracing for the listener at a specific level

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
LSNRCTL set trc_level {level}
```

From the LSNRCTL utility:

```
LSNRCTL> set trc_level {level}
```

Arguments: Trace level:

- OFF - No trace output
- USER - User trace information
- ADMIN - Administration trace information
- SUPPORT - WorldWide Customer Support trace information

Usage Notes: This command has the same functionality as TRACE.

Example:

```
LSNRCTL> set trc_level admin
```

SET USE_PLUGANDPLAY

Purpose: Instructs the listener to register its database information with an Oracle Names server

Password required if one has been set: No

Syntax: From the operating system:

```
LSNRCTL set use_plugandplay [argument]
```

From the LSNRCTL utility:

```
LSNRCTL> set use_plugandplay [argument]
```

Arguments: ON | OFF (default)

Example:

```
LSNRCTL> set use_plugandplay on
```

SHOW

Purpose:

Lists the available commands that can be shown for the current LSNRCTL session. In response to one of the SHOW commands, LSNRCTL displays the current setting of the listener for that parameter.

All of the SET commands listed except SET PASSWORD have equivalent SHOW commands.

Prerequisites:

None

Password required if one has been set:

Yes for SHOW SAVE_CONFIG_ON_STOP and SHOW USE_PLUGANDPLAY

If a password is set, the SET PASSWORD command must be issued prior to these commands.

Syntax:

From the operating system:

```
LSNRCTL show
LSNRCTL show [listener_name] [command]
```

From the LSNRCTL utility:

```
LSNRCTL> show
```

```
LSNRCTL> show [listener_name] [command]
```

Arguments:

- Listener name, if the default name of LISTENER is not used
- SHOW command. Commands are shown in the following example output.

Example:

```
LSNRCTL> show
The following properties are available with SHOW:
An asterisk (*) denotes a modifier or extended command:
connect_timeout
current_listener
displaymode
log_file
log_directory
log_status
password
raw_mode
save_config_on_stop
startup_waittime
trc_file
trc_directory
trc_level
use_plugandplay
```

SPAWN

Purpose:	Starts a program stored on the machine on which the listener is running, and which is listed with an alias in the `listener.ora` file
Prerequisites:	None
Password required if one has been set:	Yes If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:	From the operating system:
	`LSNRCTL spawn [listener_name] alias (arguments=arg1,arg2,)`
	From the LSNRCTL utility:
	`LSNRCTL> spawn [listener_name] alias (arguments=arg1,arg2,)`

Arguments:	■ Listener name, if the default name of LISTENER is not used ■ Alias name of the program as listed in the `listener.ora` file ■ Arguments sent to the program that is to be spawned
Example:	`LSNRCTL> spawn nstest_alias (ARGUMENTS='')`

START

Purpose:	Starts the named listener
Prerequisites:	Listener must not already be running.
Password required if one has been set:	No

Syntax:	From the operating system:
	`LSNRCTL start [listener_name]`
	From the LSNRCTL utility:
	`LSNRCTL> start [listener_name]`

Arguments:	Listener name, if the default name of LISTENER is not used

START

Usage Notes:

To start a listener configured in the `listener.ora` file with a name other than LISTENER, include that name.

For example, if the listener name is TCP_LSNR, enter:

```
LSNRCTL start tcp_lsnr
```

Or, from the LSNRCTL utility prompt, enter:

```
LSNRCTL> start tcp_lsnr
```

Example:

```
LSNRCTL> start
Starting /private/dsteiner/sales/bin/tnslsnr: please wait...

TNSLSNR for Solaris: Version 8.1.6.0.0
System parameter file is
/private/dsteiner/sales/network/admin/listener.ora
Log messages written to
/private/dsteiner/sales/network/log/listener.log
Trace information written to
/private/dsteiner/sales/network/trace/listener.trc
Listening on:
(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=dlsun1770)(PORT=152
1))(PROTOCOL_STACK=(PRESENTATION=TTC)(SESSION=NS)))

Connecting to
(ADDRESS=(PROTOCOL=TCP)(HOST=dlsun1770)(PORT=1521))
STATUS of the LISTENER
------------------------
Alias                    LISTENER
Version                  TNSLSNR for Solaris: Version
8.1.6.0.0
Start Date               08-JUL-99 15:50:10
Uptime                   0 days 0 hr. 0 min. 0 sec
Trace Level              user
Security                 OFF
SNMP                     OFF
Listener Parameter File
/private/dsteiner/sales/network/admin/listener.ora
Listener Log File
/private/dsteiner/sales/network/log/listener.log
Listener Trace File
/private/dsteiner/sales/network/trace/listener.trc
Services Summary...
  PLSExtProc            has 1 service handler(s)
The command completed successfully
```

STATUS

Purpose:

Displays basic status information about a listener, including:

- Version of listener
- Start time and up time
- Tracing level
- Current setting of the logging and tracing options
- `listener.ora` being used
- Whether a password is encrypted in `listener.ora` file
- Whether the network listener can respond to queries from an SNMP-based network management system
- The list of database services registered with this listener
- The address(es) the listener is listening on

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system:

```
LSNRCTL status [listener_name]
```

From the LSNRCTL utility:

```
LSNRCTL> status [listener_name]
```

Arguments:

Listener name, if the default name of LISTENER is not used

Usage Notes:

The SET DISPLAYMODE command changes the format and level of the detail of the output.

STATUS

Example:

The example below shows STATUS output in NORMAL mode. The output contains:

- Listening endpoints summary
- Services summary, which is an abbreviated version of LSNRCTL SERVICES output

```
LSNRCTL> set displaymode normal
LSNRCTL> status
Connecting to
(DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=net)(QUEUESIZE=32)))
STATUS of the LISTENER
------------------------
Alias                     LISTENER
Version                   TNSLSNR for Solaris: Version
8.1.6.0.0 -
Development
Start Date                08-SEP-1999 10:52:32
Uptime                    0 days 0 hr. 0 min. 54 sec
Trace Level               support
Security                  OFF
SNMP                      OFF
Listener Parameter File   /listener.ora
Listener Log File         /vobs/oracle/network/log/listener.log
Listener Trace File
/vobs/oracle/network/trace/listener.trc
Listening Endpoints Summary...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=net)))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1013.us.oracle.c
om)(PORT=1521)))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1013.us.oracle.c
om)(PORT=5522
))(PRESENTATION=GIOP))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1013.us.oracle.c
om)(PORT=5523
))(PRESENTATION=oracle.aurora.net.Echo))

(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1013.us.oracle.c
om)(PORT=8080
))(PRESENTATION=oracle.aurora.net.BabyHttp2))
Services Summary...
Service "s1"            has 1 instances.
    Instance "ORCL"
      Status: READY  Total handlers: 3  Relevant handlers: 1
Service "s2"            has 1 instances.
    Instance "ORCL"
      Status: READY  Total handlers: 3  Relevant handlers: 2
The command completed successfully
```

STOP

Purpose: Stops the named listener

Prerequisites: The listener must be running.

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
LSNRCTL stop [listener_name]
```

From the LSNRCTL utility:

```
LSNRCTL> stop [listener_name]
```

Arguments: Listener name, if the default name of LISTENER is not used

Usage Notes: Be careful when stopping a listener. On some platforms and with some protocols, when a listener is stopped any Net8 connections currently running are shut down. In some situations the connections continue, but it is then not possible to start the listener again until the running processes have been closed. It is good practice to send a warning message to all network users before stopping a listener.

Example:

```
LSNRCTL> stop
```

TRACE

Purpose: Turns on tracing for the listener

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
LSNRCTL trace level [listener_name]
```

From the LSNRCTL utility:

```
LSNRCTL> trace level [listener_name]
```

TRACE

Arguments:

Trace level:

- OFF - No trace output
- USER - User trace information
- ADMIN - Administration trace information
- SUPPORT - WorldWide Customer Support trace information

Listener name, if the default name of LISTENER is not used

Usage Notes:

This command has the same functionality as SET TRC_LEVEL.

Example:

```
LSNRCTL> trace admin listener
```

VERSION

Purpose:

Displays the current LSNRCTL version

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system:

```
LSNRCTL version [listener_name]
```

From the LSNRCTL utility:

```
LSNRCTL> version [listener_name]
```

Arguments:

Listener name, if the default name of LISTENER is not used.

Example:

```
LSNRCTL> version listener1
```

Oracle Names Control Utility (NAMESCTL)

The Oracle Names Control Utility (NAMESCTL) is a tool that you run from the operating system prompt to start and control Oracle Names servers. The general syntax of the NAMESCTL is as follows:

```
NAMESCTL command
```

It contains several types of commands:

- Operational commands such as START, STOP, RESTART, and so forth.
- Modifier commands, such as SET *property*
- Informational commands, such as STATUS, SHOW *property*, and PING
- Command utility operational commands, such as EXIT, QUIT, and HELP

You can use any of these commands to perform basic management functions on one or more Oracle Names server. Additionally, you can view and change parameter settings.

SET and SHOW Commands

You can use the SET command to change some parameter values for an Oracle Names server or the NAMESCTL environment during the NAMESCTL control utility session. You can then save those values or settings with the SET SAVE_CONFIG_ON_STOP command or the SET SAVE_CONFIG_INTERVAL command. You can use the SHOW command to display the current value of a configuration setting.

Security

If the NAMES.PASSWORD parameter is set in the `names.ora`, the NAMESCTL control utility will require a SET PASSWORD command for any sensitive operations, such as stopping the Oracle Names server.

If NAMESCTL.SERVER_PASSWORD parameter is set in the `sqlnet.ora` file on the node running NAMESCTL, you will *not* be prompted to use the SET PASSWORD command each time a secure operation is performed.

If you are concerned with the security implications of explicitly putting an Oracle Names server password in the client `sqlnet.ora` file, you can omit the NAMESCTL.SERVER_PASSWORD parameter and always use the command:

```
NAMESCTL> set password
```

When passed over the network, the password is encrypted regardless of how it set in the names.ora file unless the NAMESCTL.INTERNAL_ENCRYPT_PASSWORD is set to FALSE in the sqlnet.ora file. If this parameter is set to FALSE, the password is not encrypted.

NAMESCTL's Distributed Operation

The Oracle Names Control Utility operates on an Oracle Names server on the same machine as any other Oracle Names servers in the network. This is very useful when a single administrator is managing all of the Oracle Names servers in a region, or wants to check the availability of a specific Oracle Names server.

Most commands accept the name of an Oracle Names server as the last argument indicating which Oracle Names server to perform the command against. If omitted, the current SET Oracle Names server is used. For example:

```
show system_queries dolphin.acme
```

will display the system queries on the Oracle Names server dolphin.acme and when they will next occur. Prior to performing a series of commands against an individual Oracle Names server, enter

```
NAMESCTL> set server onames_server
```

Confirmation Mode in NAMESCTL

Some of the NAMESCTL commands require your confirmation before they are executed. When you issue the command, you are prompted:

```
confirm:[yes or no]
```

Enter "yes" to execute the command; enter "no" to cancel the command.

You can turn confirmation mode off by using by setting the parameter NAMESCTL.NOCONFIRM to TRUE in the sqlnet.ora file.

NAMESCTL Commands

The following commands are available through the Oracle Names Control Utility (NAMESCTL):

DELEGATE_DOMAIN

Purpose:

Defines a domain as the start of a subregion of the current administrative region

Prerequisites:

none

Password required if one has been set:

No

Syntax:

From the operating system prompt:

```
NAMESCTL delegate_domain {domain}{names_server}
{(address=...)}
```

From the NAMESCTL utility:

```
delegate_domain {domain}{names_server}{(address=...)}
```

Arguments:

{*domain*} - Specifies the domain name.

{*onames_server*} - Specifies the Oracle Names server name.

{(address=...)} - Specifies the Oracle Names server protocol address.

See Also: Appendix B for further information about defining protocol addresses

Usage Notes:

This command provides a dynamic way to subdivide the namespace.

Unless a domain is delegated from a region, the Oracle Names servers in that region will assume authority over all sub-domains. In order to delegate a domain, you must first create a new region.

Once a domain is delegated, the Oracle Names servers in the current region will forward subsequent operations to the sub-region where the domain is administered by Oracle Names servers.

Examples:

```
NAMESCTL> delegate_domain webwidgets.acme.com
ns1.webwidgets.acme.com (address=(protocol=tcp)
(host=fred.webwidgets.acme.com)(port=1575))
```

DOMAIN_HINT

Purpose:

Enables you to specify domain hints for requests for data from remote regions. A domain hint contains the name of a remote domain and at least one address of an Oracle Names server in that domain. This enables the Oracle Names server to forward the request to a specific address, reducing network traffic is reduced.

Without a domain hint, an Oracle Names server forwards a request on to whatever remote Oracle Names servers it knows, who then forwards the request to the root Oracle Names server in its region. The root Oracle Names server will forward the request to the Oracle Names servers which has information on the domain that the request refers to.

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system:

```
NAMESCTL domain_hint{domain}{names_server}{(address=...)}
```

From the NAMESCTL utility:

```
domain_hint {domain}{names_server}{(address=...)}
```

Arguments:

{*domain*} - Specifies the domain name.

{*onames_server*} - Specifies the Oracle Names server name.

{(address=...)} - Specifies the Oracle Names server protocol address.

See Also: Appendix B for further information about defining protocol addresses

Usage Notes:

Any region that is not the root region will need at least the root region defined using this command in order to find objects in any other region. You can provide additional hints as optimizations to provide local Oracle Names servers with direct access to certain other regions.

Examples:

```
NAMESCTL> domain_hint acme.com ns0.acme.com
(address=(protocol=tcp) (host=top.acme.com)(port=1575))
```

DUMP_LDAP

Purpose:

Queries all the authoritative data in a domain or region and exports the data to a LDAP-compliant directory service or into **LDAP Data Interchange Format (LDIF)** file, which can later be loaded into a directory

Prerequisites:

The directory must already have an Oracle Context and Oracle schema.

Password required if one has been set:

A password for the directory may be required.

Syntax:

Exporting Data to an LDIF File

```
NAMESCTL> dump_ldap [source] [destination]
[options] {-f filename}
```

Exporting Data To a Directory

```
NAMESCTL> dump_ldap [source] [destination]
[options] {-h host} {-p port} {-D user_dn}
{-w password}
```

Arguments:

[*source*]= {*domain*} [-R]

{*domain*} - Specifies the domain's objects to be migrated. The default domain is the root. The operation is forwarded to a server who is authoritative for that domain if the server which the client contacts is not authoritative.

[-R] - Specifies that all authoritative subdomains of the given domain should also be migrated.

DUMP_LDAP

[*destination*] - Specifies the DN in the directory where objects should be placed in the DIT

A DN can be specified in one of following ways:

- A complete DN that includes all the tree levels. For example, `(dn:cn=OracleContext,dc=com,dc=acme)` specifies that object be exported to `cn=OracleContext` under `acme.com`.
- A template to accommodate the full-intended depth of the tree by using wildcards for RDNs. For example, `(dn:o,ou,dc)` maps to a DIT structure that contains an Organization (`o`), and OrganizationUnit (`ou`), and a domain component (`dc`).

  Note: If the directory's DIT contains an `cn=OracleContext` RDN, use the -x option when using a RDN template. This way, a database service and net service names will be mapped under `cn=OracleContext`.

The -R (recursive) argument in the source specification determines how the destination DN is mapped to the directory's DIT.

- If -R is *not* specified in the source, the source domain specified maps directly into the DIT specified by the destination DN.
- If the -R argument is used, then data will be exported from all subdomains starting at the given domain. Any subdomains map to the destination RDNs specified. If RDNs in the destination DN are wildcarded—that is, specified without a value—subdomains are mapped to the RDN type with their own value.

  Note: If the DIT contains an `cn=OracleContext` RDN for each sub-domain, use the -x option. This way, database services and net service names will be mapped under `cn=OracleContext` for each subdomain.

Note: If no DN is specified, domains will be mapped to domain components (`dc`) in the DIT.

DUMP_LDAP

[*options*] - Arguments that specify how the export of data is to occur:

-a - Specifies that all objects are to be added to the DIT. If an object already exists in the DIT, an error occurs. If this option is not specified, existing objects are modified, but no additions are made.

-c - Specifies that the export should continue on error

-n - Does not perform an actual export. This argument enables you to perform a test run

-q - Runs the operation in quiet mode

-x - Adds `cn=OracleContext` to the destination DN.

{-f *filename*} - Specifies that the migrated objects be dumped into an LDIF file, which can later be loaded into a directory. The default file names is `onames.ldif`.

These arguments specify the directory server location:

[-h *host*] - Specifies the host name of the directory server

[-p *port*] - Specifies the port number the directory is configured to listen on. The default TCP/IP port number is 389

These arguments specify the authentication credentials:

[-D *user_dn*] - Specifies a directory administrator that has been given add and modify privilege in the form of a DN. For example, `c=us,o=acme,ou=admin,cn=scott` is the DN for an administrator named Scott.

[-w *password*] -Specifies the password for the directory administrator

Usage Notes:

See Also: "Exporting Network Objects from an Oracle Names Server" on page 6-35

Examples:

Exporting Data to a LDIF File

```
NAMESCTL> dump_ldap sj.us.sczi.com - R
(dn:cn=OracleContext,ou=sj,dc=us,dc=sczi,dc=com) -f
test.ldif
```

Exporting Data Directly Into a Directory

```
NAMESCTL> dump_ldap sj.us.sczi.com - R
(dn:cn=OracleContext,ou=sj,dc=us,dc=sczi,dc=com) -a -h
dlsun1598 -p 389 -D cn=orcladmin -w welcome
```

DUMP_TNSNAMES

Purpose:	Writes the names and addresses of all address defined in the local region into a `tnsnames.ora` file
Prerequisites:	None
Password required if one has been set:	No

Syntax:	From the operating system:
	`NAMESCTL dump_tnsnames`
	From the NAMESCTL utility:
	`dump_tnsnames`

Arguments:	None
Usage Notes:	Writes the name and address of all addresses defined in the local region into `tnsnames.ora` (everything with an address-type record, A.SMD). The command creates or partially overwrites the `tnsnames.ora` file. Any entries in the existing `tnsnames.ora` which are not defined in the Oracle Names servers will remain. Any definitions in `tnsnames.ora` which are also defined in the Oracle Names servers will be overwritten. Entries which are defined in the Oracle Names servers but not in the `tnsnames.ora` will be added.
Examples:	`NAMESCTL> dump_tnsnames`

EXIT

Purpose: Exits from the NAMESCTL utility

Prerequisites: The NAMESCTL utility must be loaded.

Password required if one has been set: No

Syntax: From the NAMESCTL utility: `EXIT`

Arguments: None

Usage Notes: EXIT has no effect on any Oracle Names servers.

It affects only the NAMESCTL utility.

This command is identical to the QUIT command.

Example:

```
namsctl> exit
NAMESCTL finished.
```

FLUSH

Purpose: Instructs the Oracle Names server to clear all remote region information from its local cache checkpoint file, which has a default of `ckpcch.ora`

Prerequisites: Only relevant with an environment with multiple regions. (In central administration there is no non-authoritative data.)

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system prompt:

```
NAMESCTL flush [onames_server] [onames_server]*
```

From the NAMESCTL utility:

```
flush [onames_server] [onames_server]*
```

Arguments: Zero or more Oracle Names servers separated by a space. When no arguments are supplied, only the current Oracle Names server's cache is flushed of the foreign names

FLUSH

Usage Notes:

FLUSH erases *all* remote data that has been cached. Typically, you should flush the foreign data cache when:

- A large volume of data changes in the network and the normal TTL aging mechanism will take too long.
- When unidentifiable errors in name resolution of cached foreign data are occurring. Flushing all foreign data from the cache forces it to be looked up again when it is requested the next time.

Names are flushed from the current Oracle Names server. The current Oracle Names server is either the default preferred Oracle Names server or the one set by using the SET SERVER command.

Examples:

```
NAMESCTL> flush
Confirm [yes or no]: yes
```

FLUSH_NAME

Purpose:

Instructs the Oracle Names server to clear a specific region's information from its local cache checkpoint file, which has a default of `ckpcch.ora`

Prerequisites:

This parameter is only meaningful for an environment with multiple regions. (In central administration, there is no non-authoritative data.)

Password required if one has been set:

Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:

From the operating system prompt:

```
NAMESCTL flush_name {domain}
```

From the NAMESCTL utility:

```
flush_name {domain}
```

Arguments:

A single domain name

Usage Notes:

FLUSH_NAME erases only data cached from outside the Oracle Names server's region (that is, non-authoritative data). It is typically flushed when a name is behaving unusually, suggesting the source copy may have changed.

Names are flushed from the current Oracle Names server. The current Oracle Names server is either the default preferred Oracle Names server or the one set by using the SET SERVER command.

Examples:

```
NAMESCTL> flush_name mountain.acme.com
```

HELP

Purpose: Provides a list of all the NAMESCTL commands available. In response to one of the HELP commands, NAMESCTL displays help on how to use the command.

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL help
NAMESCTL help [command]
```

From the NAMESCTL utility:

```
NAMESCTL help
NAMESCTL> help [command]
```

Arguments: Help command. Commands are shown in the following example output.

Example:

```
NAMESCTL> help
The following operations are available:
An asterisk (*) denotes a modifier or extended command:
exit
flush
flush_name
log_stats
ping
query
quitreload
repeat*
reset_stats
restart
set*
show
shutdown
start
startup
status
stop
version
```

LIST_DELEGATED

Purpose: Lists all the delegated domains for the root region or a specified domain

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL list_delegated [domain]
```

From the NAMESCTL utility:

```
NAMESCTL> list_delegated [domain]
```

Arguments: Domain name

Usage Notes: This command is intended to help you understand the current domain structure before exporting data from Oracle Names to an LDAP-compliant directory.

See Also: "Exporting Network Objects from an Oracle Names Server" on page 6-35

Examples:

```
NAMESCTL> list_delegated
europe.acme.com
asia.acme.com
africa.acme.com
```

LIST_DOMAINS

Purpose: Lists all the domains in the root region or subdomains for a specified domain

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL _list_domain [domain]
```

From the NAMESCTL utility:

```
NAMESCTL _list_domain [domain]
```

Arguments: domain name

LIST_DOMAINS

Usage Notes:

This command is intended to help you understand the current domain structure before exporting data from Oracle Names to an LDAP-compliant directory.

See Also: "Exporting Network Objects from an Oracle Names Server" on page 6-35

Examples:

```
NAMESCTL> list_domains
com
sczi.com
us.sczi.com
sj.us.sczi.com
```

LIST_OBJECTS

Purpose:

Lists all the network objects for the root region or a specified domain

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system:

```
NAMESCTL list_domain [-R] [domain]
```

From the NAMESCTL utility:

```
NAMESCTL> list_domain [-R] [domain]
```

Arguments:

[-R] - Specifies that all authoritative subdomains of the given domain be listed.

[*domain*] - Specifies the domain name.

Usage Notes:

This command is intended to help you understand the current domain structure before exporting data from Oracle Names to an LDAP-compliant directory.

See Also: "Exporting Network Objects from an Oracle Names Server" on page 6-35

Examples:

```
NAMESCTL> list_obmects
partsdb.widgets.acme.com
toolsdb.widgets.acme.com
partsdb.components.widgets.acme.com
sparepartsdb.gadgets.widgets.acme.com
```

LOAD_TNSNAMES

Purpose: Loads all net service names and addresses defined in one or more `tnsnames.ora` files into an Oracle Names server.

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system prompt:

```
NAMESCTL load_tnsnames {tnsnames.ora} [tnsnames.ora]
```

From the NAMESCTL utility:

```
load_tnsnames {tnsnames.ora}[tnsnames.ora]
```

Arguments: One or more `tnsnames.ora` files

Usage Notes: This command would typically be used once when a site begins using Oracle Names after having used `tnsnames.ora` files. Run this command once per region. The names defined during this operation will be defined permanently and will be propagated from one Oracle Names server to all the rest in the region.

Example:

```
NAMESCTL>load_tnsnames network/admin/tnsnames.ora
Name:                 koala.lab.npd.us.oracle.com
Response status:      normal, successful completion
Name:                 devdd.rdbms.us.oracle.com
Response status:      normal, successful completion
Name:                 envyd.lab.npd.us.oracle.com
Response status:      normal, successful completion
Name:                 stealth.npd.us.oracle.com
Response status:      normal, successful completion
Name:                 null.us.oracle.com
Response status:      normal, successful completion
Name:                 slime.lab.npd.us.oracle.com
Response status:      normal, successful completion
Name:                 felix.hp.us.oracle.com
Response status:      normal, successful completion
Name:                 dtnet1.dec.oracle.com
Response status:      normal, successful completion
Name:                 devds.rdbms.us.oracle.com
Response status:      normal, successful completion
```

LOG_STATS

Purpose:	Logs the current set of statistics to the configured log file for that Oracle Names server. The log file has a default of `names.log`.
Prerequisites:	None
Password required if one has been set:	Yes If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:	From the operating system prompt:
	`NAMESCTL log_stats [`*`onames_server`*`] [`*`onames_server`*`]*`
	From the NAMESCTL utility:
	`log_stats [`*`onames_server`*`] [`*`onames_server`*`]*`

Arguments:	Zero or more Oracle Names server names separated by a space. When no arguments are supplied, only the statistics for the current Oracle Names server are reset.
Usage Notes:	Statistics may be logged if the STATUS command or other behavior indicates some data that you would like to capture in the log. LOG_STATS does not affect the current log statistics interval.

Example:	`NAMESCTL> log_stats`
	`Statistics counters logged.`

PASSWORD

Purpose:	Sets an encrypted password for privileged NAMESCTL utility commands, such as STOP, RESTART and RELOAD.
Prerequisites:	The NAMESCTL utility must be loaded.
Password required if one has been set:	N/A
Syntax:	From the NAMESCTL utility: `password [`*`password`*`]`
Arguments:	Text string matching the value encrypted in the NAMES.PASSWORD parameter in the `names.ora` file.

PASSWORD

Usage Notes:

This command does not change a password already established with the NAMES.PASSWORD parameter in the `names.ora` file. It simply sets a NAMESCTL variable. Then, the value stored is sent from NAMESCTL with any command request to the Oracle Names server, and the value is compared to the value configured on the Oracle Names server. If they match, operations requiring passwords are allowed.

Only "privileged" operations are affected, that is, operations that alter the functioning of the Oracle Names server. Operations such as SHOW or STATUS are not considered privileged, and do not require a password.

The password can either be passed as an argument of the PASSWORD command, or if no argument is given, it will be prompted for. Note that the input is not displayed on the screen as it is typed.

When passed over the network the password is always encrypted, regardless of how it is set.

Example:

```
NAMESCTL> password open_sesame
NAMESCTL> password
Enter name server password:
```

PING

Purpose:

Contacts the current Oracle Names server, or named Oracle Names servers, and displays the request/response time

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system prompt:

```
NAMESCTL ping [onames_server] [onames_server]*
```

From the NAMESCTL utility:

```
ping [onames_server] [onames_server]*
```

Arguments:

Zero or more Oracle Names server names separated by a space. When no arguments are supplied, only the current Oracle Names server is pinged.

Usage Notes:

Ping ensures that an Oracle Names server is functioning and shows typical response times from the location of the NAMESCTL user to an Oracle Names server.

Example:

```
NAMESCTL> ping nserver.com
Round trip time is 0.04 seconds
```

QUERY

Purpose:	Retrieves the contents of a network object stored in the Oracle Names server
Prerequisites:	None
Password required if one has been set:	No

Syntax

From the operating system prompt:

`NAMESCTL query` *`object_name`* `[`*`object_type`*`] [`*`modifiers`*`]`

From the NAMESCTL utility:

`query` *`object_name`* `[`*`object_type`*`] [`*`modifiers`*`]`

COMMON OBJECT TYPES:

- A.SMD—Network addresses, as with database service definitions or net service names
- CNAME.SMD—Aliases (sometimes referred to as "canonical name")
- DL.RDBMS.OMD—Global database links
- DLCR.RDBMS.OMD—Global database link qualifiers
- NS.SMD—Oracle Names server addresses. System data used to communicate between Oracle Names servers
- V1ADD.NPO.OMD—SQL*Net Version 1 connect string

VALID MODIFIERS:

- AUTHORITY—Forces the query to be resolved at the source of the data (in the administrative region where the data is considered local) even if the data is in the local cache. This could be used if the administrator suspects that the data has changed at the source.
- NOFORWARD—Query for the data, but don’t forward the request. When the data is not local, and noforward is specified, the query will not be resolved.
- TRACE—Enables a trace of the path to the answer. This is useful whenever you want to find out which Oracle Names servers the request went to

Arguments: Mandatory network object name and network object type

QUERY

Usage Notes:

QUERY can be used to test that a defined piece of data can be found, and that the contents are correct.

The QUERY command always operates on the current Oracle Names server, either the default, or as specified using the SET SERVER command.

If the QUERY command is used with just a name as a parameter, the Oracle Names server responds with the number of pieces of data with that name, and the time required to complete the operation.

If the QUERY command is used with the name and type supplied as arguments; the specific name is looked up and returned to the user.

The QUERY command can take multiple arguments if appropriate. For example:

```
query sales.com a.smd authority trace
```

Example:

```
NAMESCTL> query bones.dem.medicine a.smd
Total response time:0.04 seconds
Response status:normal, successful completion
Authoritative answer:yes
Number of answers:1
Canonical name:bones.dem.medicine
TTL: 1 day
Alias translations:
    from:bones.dem.medicine
    to: bones.dem.medicine
Answers:
    data type is "a.smd"
        Syntax is ADDR:...(DESCRIPTION=(ADDRESS=
(PROTOCOL=TCP)(Host=bones-pc)
(Port=1521))(CONNECT_DATA=(SERVICE_NAME=bones.dem.medicine)))
```

QUIT

Purpose: Quits the NAMESCTL utility

Prerequisites: The NAMESCTL utility must be loaded.

Password required if one has been set: No

Syntax: From the NAMESCTL utility:

```
QUIT
```

Arguments: None

Usage Notes: QUIT has no effect on any Oracle Names servers. It affects only the NAMESCTL utility.

The QUIT command is functionally equivalent to the EXIT command.

Example:

```
NAMESCTL> quit
nl-00851: NAMESCTL finished
```

REGISTER

Purpose: Registers a network object to an Oracle Names server

Password required if one has been set: No

Syntax: From the NAMESCTL utility:

```
register object_name [-t service_type]
[-d [(description=](address=(protocol_address_
information))[)]] [-h onames_server]
[-l listener_name]
```

TYPE OF SERVICE:

- ORACLE_DATABASE
- ORACLE_LISTENER
- ORACLE_NAMESERVER

See Also: Appendix B for further information about protocol addresses and parameters

Arguments: Mandatory object name. The service, address data, and host are not necessary to make the registration process appear to work. However, they are necessary to make the registration useful. In other words, an object name registered without an address cannot be used.

REGISTER

Usage Notes:

Provides a manual mechanism for registering a service, its type, its hostname, and its address. Both the type of service and the data can be any valid string, but the typical registration has either "database" or "listener" as type of service, and the TNS address as the data.

The object registration is propagated to all other well known Oracle Names servers in the region.

Example:

```
NAMESCTL> register parts -t oracle_database -d (DESCRIPTION=
(ADDRESS= (PROTOCOL=TCP)(HOST=nineva)(PORT=1575)) (CONNECT_
DATA=(SERVICE_NAME=db3)))
```

RELOAD

Purpose:

Forces the Oracle Names server to check immediately for data changes in its administrative region. If there are any changes, the Oracle Names server reloads all database service names, net service names, global database links, and aliases.

Prerequisites:

None

Password required if one has been set:

Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:

From the operating system prompt:

```
NAMESCTL reload [onames_server] [onames_server]*
```

From the NAMESCTL utility:

```
reload [onames_server] [onames_server]*
```

Arguments:

Zero or more Oracle Names server names separated by a space. When no arguments are supplied, only the current Oracle Names server is reloaded.

Usage Notes:

All Oracle Names servers load their data directly from the database specified by the NAMES.ADMIN_REGION configuration parameter in the `names.ora` file.

In an environment with multiple regions, RELOAD affects only the data for the current administrative region. All foreign data in the cache is unchanged.

Example:

```
NAMESCTL> reload
Server reloaded.
```

REORDER_NS

Purpose: Creates the file which lists local Oracle Names servers and their listening addresses

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system prompt:

```
NAMESCTL reorder_ns [(description=](address=(protocol_
address_information))[)]
```

From the NAMESCTL utility:

```
reorder_ns [(description=](address=(protocol_address_
information))[)]
```

See Also: Appendix B for further information about protocol addresses and parameters

Arguments: An optional Oracle Names server address will be used as the initial server to contact.

Usage Notes: This command generates the file which defines Oracle Names server names and addresses to enable clients to contact Oracle Names servers for name lookup.

The REORDER_NS command performs the following tasks;

1. Finds an Oracle Names server: 1. It looks in the `sqlnet.ora` for a preferred Oracle Names server parameter, 2. tries calling a well-known servers, 3. tries locating a local Oracle Names server using TCP/IP on port 1575.
2. Sends a query for all the Oracle Names servers in the local region
3. Sends a PING to each of these servers
4. Sorts the list of servers by increasing order of response time
5. Writes an Oracle Names server List with the sorted list of names and addresses

Example:

```
NAMESCTL> reorder_ns (address=
(protocol=tcp)(host=nineva)(port=1575))
```

REPEAT

Purpose:

Performs QUERY, REGISTER, TIMED_QUERY, or UNREGISTER multiple times to compute average return rates

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system prompt:

```
NAMESCTL repeat number query type
```

From the NAMESCTL utility:

```
repeat number query type
```

Arguments:

number - Integer

type - Object types, as specified in the QUERY command

Usage Notes:

Repeat is useful for understanding the average response time over a number of requests.

Do not specify too large a number here; while the number of iterations are occurring, the NAMESCTL utility cannot do anything else.

Example:

```
NAMESCTL> repeat 10 query manatee a.smd
Number of requests: 10
Average response time: 0.01 seconds
Minimum response time: 0.01 seconds
Maximum response time:0.04 seconds
Total response time:0.14 seconds
Response status:normal, successful completion
Authoritative answer:yes
Number of answers: 1
TTL: 1 day
Answers:
    data type is "a.smd"
        Syntax is ADDR:(DESCRIPTION=(ADDRESS=
(PROTOCOL=TCP)(Host=salmon)
(Port=1522))(CONNECT_DATA=(SID=otter)))
```

RESET_STATS

Purpose: Resets the Oracle Names server statistics to the original values of the Oracle Names server at startup

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system prompt:

```
NAMESCTL reset_stats [onames_server] [onames_server]*
```

From the NAMESCTL utility:

```
reset_stats [onames_server] [onames_server]*
```

Arguments: Zero or more Oracle Names server names separated by a space. When no arguments are supplied, only the current Oracle Names server's statistics are reset.

Usage Notes: RESET_STATS has the same effect as waiting for the RESET_STATS_INTERVAL to conclude, except that it happens immediately.

Example:

```
NAMESCTL> reset_stats
Confirm [yes or no]: yes
Server statistics reset.
```

RESTART

Purpose: Initiates a reset of an Oracle Names server to its original state at startup

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system prompt:

```
NAMESCTL restart [onames_server] [onames_server]*
```

From the NAMESCTL utility:

```
restart [onames_server] [onames_server]*
```

Arguments: Zero or more Oracle Names server names separated by a space. When no arguments are supplied, only the current Oracle Names server is restarted.

RESTART

Usage Notes:

RESTART is the same as STARTUP except that the Oracle Names server is already running.

Data is reloaded, statistics are reset, and all foreign data is flushed. Valid foreign cache data (that is, data with a TTL greater than 0) is retrieved from the checkpoint files. (The TTL value must be set to more than 0.)

Example:

```
NAMESCTL> restart
Confirm [yes or no]: yes
Server restarted.
```

SET

Purpose:

Lists the available configuration commands that can be set for the current NAMESCTL session

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system:

```
NAMESCTL set
NAMESCTL set [command]
```

From the LSNRCTL utility:

```
NAMESCTL> set
NAMESCTL set [command]
```

Arguments:

SET commands. Commands are shown in the following example output.

SET

Example:

```
NAMESCTL> set
The following operations are available after set
An asterisk (*) denotes a modifier or extended command:
cache_checkpoint_interval
default_domain
forwarding_available
log_file_name
log_stats_interval
NAMESCTL_trace_level
password
requests_enabled
reset_stats_interval
save_config_interval
save_config_on_stop
server
trace_file_name
trace_level
```

SET CACHE_CHECKPOINT_INTERVAL

Purpose: Sets how often all collected information about remote regions is saved in the local cache file, which as a default of `ckpcch.ora`

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL set cache_checkpoint_interval time
```

From NAMESCTL utility:

```
set cache_checkpoint_interval time
```

Arguments: Time is in seconds

For example, to increase the CACHE_CHECKPOINT_INTERVAL to 36 hours, the following can be set:

```
set cache_checkpoint_interval 129600
```

Usage Notes: Minimum: 10 seconds

Maximum: 259200 (3 days)

Default: 0 (disabled)

Example:

```
NAMESCTL> set cache_checkpoint_interval 10
```

SET DEFAULT_DOMAIN

Purpose: Sets the domain from which the NAMESCTL client most often looks up names resolution requests, or changes the domain specified by the NAMES.DEFAULT_DOMAIN in the `sqlnet.ora` file

Prerequisites: The NAMESCTL utility must be loaded.

Password required if one has been set: No

Syntax: From the NAMESCTL utility:

```
set default_domain domain_name
```

Arguments: Domain name

Usage Notes: When a default domain is set, it is automatically appended to any unqualified net service name or service name. For example, if the default domain is set to `us.acme`, the global name `sales.us.acme` could be queried using:

```
NAMESCTL> query sales
```

Example:

```
NAMESCTL> set default_domain us.acme.com
Default domain is now "US.ACME.COM"
```

SET FORWARDING_AVAILABLE

Purpose: Turns on or off forwarding to remote Oracle Names servers for client name requests

Prerequisites: Oracle Names server must be running.

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
NAMESCTL set forwarding_available on|off|yes|no
```

From the NAMESCTL utility:

```
set forwarding_available off
```

Arguments: ON | OFF | YES | NO

Default Value: OFF

SET FORWARDING_AVAILABLE

Usage Notes:

This setting is intended only for Oracle Names servers that have no local clients and are exclusively handling requests from remote Oracle Names servers. This usually would only apply to Oracle Names servers in the root region when the root is configured without clients or services. If such an Oracle Names server is a performance bottleneck in cross-region request processing, then disabling forwarding in that Oracle Names server will cut its workload in half. Rather than forward the request and return the answer, the Oracle Names server simply tells the requestor the address of the Oracle Names server that can answer the request. Note that there is no overall reduction in work; the work is simply displaced from the non-forwarding Oracle Names server to the requesting Oracle Names server.

WARNING: If SET FORWARDING_AVAILABLE is set to off, any clients who rely directly on that Oracle Names server will be unable to resolve remote names. Clients are not capable of redirecting their requests as Oracle Names servers are. Their requests will fail at that point, even if other Oracle Names servers are listed in the NAMES.PREFERRED_SERVERS configuration parameter.

Example:

```
NAMESCTL> set forwarding_available off
Request processing is now disabled.
```

SET LOG_FILE_NAME

Purpose:

Sets the name for the Oracle Names server's log file. By default, the log file name is `names.log`.

Prerequisites:

None

Password required if one has been set:

Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:

From the operating system:

```
NAMESCTL set log_file_name file_name
```

From the NAMESCTL utility:

```
set log_file_name file_name
```

Arguments:

file name of the log file.

Usage Notes:

The LOG_FILE_NAME changes the destination of all logging messages.

Example:

```
NAMESCTL> set log_file_name namesvr1
```

SET LOG_STATS_INTERVAL

Purpose:

Changes the frequency with which the statistics are logged to the log file, which has a default of `names.log`

Prerequisites:

None

Password required if one has been set:

Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:

From the operating system:

```
NAMESCTL set log_stats_interval time
```

From the NAMESCTL utility:

```
NAMESCTL> set log_stats_interval time
```

Arguments:

Time is in seconds or [<*n*> DAY[S]] [<*hh*>:<*mi*>:<*ss*>]

For example, to increase the LOG_STATS_INTERVAL to 36 hours, both of the following can be set:

```
set log_stats_interval 129600
set log_stats_interval 1 day 12:00:00
```

You can specify any valid combination, such as the number of days combined with number of hours, minutes, and seconds; or just the number in hours.

Restrictions:

Minimum Value: 10 seconds

Maximum Value: no maximum

Special Value: 0 (which means never reset)

Default value: 0 (no logging)

Usage Notes:

The LOG_STATS_INTERVAL value is initially set based on the value configured in NAMES.LOG_STATS_INTERVAL parameter in the `sqlnet.ora` file when the Oracle Names server is loaded. By default, the value is 0 (no logging). This command is intended to override that value during server operation.

Example:

```
NAMESCTL> set log_stats_interval 7200
Statistic counter logging interval is now 2 hours
```

SET NAMESCTL_TRACE_LEVEL

Purpose: Sets the level at which the NAMESCTL utility can be traced

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
NAMESCTL set NAMESCTL_trace_level level
```

From the NAMESCTL utility:

```
NAMESCTL> set NAMESCTL_trace_level level
```

Arguments: OFF - Tracing is not set

USER - Tracing is set to identify user-induced error conditions

ADMIN - Tracing is set to identify installation-specific problems

SUPPORT - Tracing is set to a level appropriate for support

Usage Notes: Tracing assists in diagnosing unexpected or unidentifiable failures in processing the NAMESCTL utility. Tracing writes a series of events from normal NAMESCTL processing to an operating system file for review by the administrator.

Tracing output is at three levels OFF (none), USER (basic information), or ADMIN.

When no arguments are supplied, the setting is reset to the value in the client's `sqlnet.ora` file. The default setting is OFF.

Example:

```
NAMESCTL> set NAMESCTL_trace_level admin
Controller's local trace level changed from 0 to 4
```

SET PASSWORD

Purpose:

Sets the password for privileged NAMESCTL utility commands, such as STOP, RESTART and RELOAD.

The password entered should match the one set for the NAMES.PASSWORD parameter in the `names.ora` file.

Prerequisites:

The NAMESCTL utility must be loaded.

Password required if one has been set:

Not applicable

Syntax:

From the NAMESCTL utility:

```
NAMESCTL> set password [password]
```

Arguments:

Text string matching the value stored in the current Oracle Names server parameter NAMES.PASSWORD

Usage Notes:

SET PASSWORD does not change the Oracle Names server's password. It simply sets a NAMESCTL variable that is sent over to the Oracle Names server with any NAMESCTL command and is compared to the value configured on the Oracle Names server. If they match, operations requiring passwords are allowed.

Only "privileged" operations are affected, that is, operations that alter the functioning of the Oracle Names server. Operations such as SHOW or STATUS are not considered privileged, and do not require a password.

The password can either be passed as an argument of the SET PASSWORD command, or if no argument is given, it will be prompted for. Note that the input is not displayed on the screen as it is typed.

When passed over the network the password is always encrypted, regardless of how it is set.

Example:

```
NAMESCTL> set password open_sesame
NAMESCTL> set password
enter name server password:
```

SET REQUESTS_ENABLED

Purpose: Determines whether the current Oracle Names server will respond to requests

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
NAMESCTL requests_enabled on|off
```

From the NAMESCTL utility:

```
NAMESCTL> set requests_enabled on|off
```

Arguments: ON | OFF

Usage Notes: Setting this property to OFF will send refusals to all clients that approach with names resolution requests. This is primarily useful for diagnostics when an Oracle Names server is functioning unexpectedly.

Example:

```
NAMESCTL> set requests_enabled off
Confirm [yes or no]: yes
General request processing is now disabled
```

SET RESET_STATS_INTERVAL

Purpose: Changes the time between the statistics being reset to zero or initial values in the current server.

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
NAMESCTL set reset_stats_interval time
```

From the NAMESCTL utility:

```
NAMESCTL> set reset_stats_interval time
```

SET RESET_STATS_INTERVAL

Arguments:

Time is in seconds or [*n DAY*[S]] [*hh:mi:ss*]

For example, to increase the RESET_STATS_INTERVAL to 72 hours, the following can be set:

```
set reset_stats_interval 259200
set reset_stats_interval 3 days
```

Restrictions:

Minimum Value: 10 seconds

Maximum Value: no maximum

Default value: 0 (never reset)

Usage Notes:

The RESET_STATS_INTERVAL value is initially set based on the NAMES.RESET_STATS_INTERVAL parameter when the Oracle Names server is loaded. This command is intended to override that value during Oracle Names server operation.

Example:

```
NAMESCTL> set reset_stats_interval 1 day
Statistic counter reset interval is now 24 hours
```

SET SAVE_CONFIG_INTERVAL

Purpose:

Saves any changes made by NAMESCTL SET commands to the `names.ora` at an interval

Password required if one has been set:

Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:

From the operating system:

```
NAMESCTL set save_config_interval time
```

From NAMESCTL utility:

```
set save_config_interval time
```

Arguments:

Time is in seconds

Example:

```
NAMESCTL> set save_config_interval 10
```

SET SAVE_CONFIG_ON_STOP

Purpose:

Specifies whether or not changes made by the NAMESCTL SET commands are saved to the `names.ora` file after the NAMESCTL session has ended.

The saving of all values occurs right before the NAMESCTL session exits, taking as much care as possible to preserve the formatting, comments, and letter case.

Password required if one has been set:

Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax:

From the operating system:

```
NAMESCTL set save_config_on_stop on|off
```

From the NAMESCTL utility:

```
NAMESCTL> set save_config_on_stop on|off
```

Arguments:

ON | OFF

Example:

```
NAMESCTL> set save_config_on_stop on
```

SET SERVER

Purpose:	Sets the name of the Oracle Names server
Prerequisites:	The NAMESCTL utility must be loaded
Password required if one has been set:	No
Syntax:	From the NAMESCTL utility: `NAMESCTL> set server [`*`onames_server`*`] [(description=](address=(`*`protocol_address_information`*`))[)]` **See Also:** Appendix B for further information about protocol addresses and parameters
Arguments:	Valid Oracle Names server or valid Oracle Names server address If there are no arguments, use the values set by the NAMES.PREFERRED_SERVERS parameter in the `sqlnet.ora` file.
Usage Notes:	SET SERVER enables switching between multiple Oracle Names servers while running the NAMESCTL utility. The qualifier can be a name where the name is defined in the memory of the current Oracle Names server, or it can be the TNS address of any Oracle Names server. The Oracle Names server name specified is resolved through normal name lookup. Another Oracle Names server can only be set if the current Oracle Names server knows or can retrieve its address. If no current Oracle Names server is set, you must enter an address to complete this command.
Example:	`NAMESCTL> set server server1.us.acme`

SET TRACE_FILE_NAME

Purpose: Sets the name for the Oracle Names server's trace file. By default, the trace file name is `names.trc`.

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL set trace_file_name file_name
```

From NAMESCTL utility:

```
NAMESCTL> set trace_file_name file_name
```

Arguments: file name

Example:

```
NAMESCTL> set trace_file_name namesvr1
```

SET TRACE_LEVEL

Purpose: Sets tracing for the Oracle Names server at a specific level.

Prerequisites: None

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
NAMESCTL set trace_level level
```

From the NAMESCTL utility:

```
NAMESCTL> set trace_level level
```

Arguments: Trace level:

- OFF- No trace output
- USER - User trace information
- ADMIN - Administration trace information
- SUPPORT - WorldWide Customer Support trace information

SET TRACE_LEVEL

Usage Notes:

Tracing assists in diagnosing unexpected or unidentifiable failures in processing the current Oracle Names server. Tracing writes a series of events from normal Oracle Names server processing to an operating system file for review by the administrator.

After the TRACE_LEVEL is set, tracing begins immediately. All operations are traced until it is reset to trace level OFF.

Trace files can grow very large. Remember to turn trace level off after diagnosing the problem.

Example:

```
NAMESCTL> set trace_level admin
Trace level is now 6.
```

SHOW

Purpose:

Lists the available commands that can be shown for the current NAMESCTL session. In response to one of the SHOW commands, NAMESCTL displays the current setting for that parameter.

All of the SET commands listed except SET PASSWORD have equivalent SHOW commands.

Prerequisites: None

Syntax:

From the operating system:

```
NAMESCTL show
NAMESCTL show [command]
```

From the NAMESCTL utility:

```
NAMESCTL> show

NAMESCTL> show [command]
```

SHOW

Arguments:

```
NAMESCTL> show
The following operations are available after show
An asterisk (*) denotes a modifier or extended command:
cache_checkpoint_interval
default_domain
forwarding_available
log_file_name
log_stats_interval
NAMESCTL_trace_level
requests_enabled
reset_stats_interval
save_config_interval
save_config_on_stop
server
status
system_queries
trace_file_name
trace_level
version
```

Example:

```
CMCTL> show trace_level
```

SHOW CACHE_CHECKPOINT INTERVAL

Purpose: Shows the frequency with which the Oracle Names server's cache is written to the cache checkpoint file, which as a default of `ckpcch.ora`

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show cache_checkpoint_interval
```

From the NAMESCTL utility:

```
NAMESCTL> show cache_checkpoint_interval
```

Arguments: None

SHOW CACHE_CHECKPOINT INTERVAL

Usage Notes:

The CACHE_CHECKPOINT_INTERVAL is initially set with the value in NAMES.CACHE_CHECKPOINT_INTERVAL in the names.ora file. By default, the value is 0, which disables cache checkpointing. Data written to the cache checkpoint file includes net service names and addresses, and Oracle Names server addresses which were learned by the Oracle Names server as a result of forwarding a query to a foreign region on behalf of the client.

Example:

```
NAMESCTL> show cache_checkpoint_interval
Cache checkpoint interval is currently 8 minutes 20 seconds
```

SHOW DEFAULT_DOMAIN

Purpose:

Shows the domain set by the SET DEFAULT_DOMAIN or the NAMES.DEFAULT_DOMAIN in the sqlnet.ora file

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system:

```
NAMESCTL show default_domain
```

From the NAMESCTL utility:

```
NAMESCTL> show default_domain
```

Arguments:

None

Usage Notes:

When a default domain is set, it is automatically appended to any unqualified net service name or service name. For example, if the default domain is set to us.acme, the global name sales.us.acme could be queried using:

```
NAMESCTL> query sales
```

Example:

```
NAMESCTL> show default_domain
Current default domain is "com"
```

SHOW FORWARDING_AVAILABLE

Purpose: Shows whether the Oracle Names server is forwarding client requests to remote Oracle Names servers

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show forwarding_available
```

From the NAMESCTL utility:

```
NAMESCTL> show forwarding_available
```

Arguments: Zero or more Oracle Names servers separated by a space. If no names are given, then the setting is displayed for the current server.

Usage Notes: By default, all Oracle Names servers forward requests to remote Oracle Names servers. If forwarding is disabled, then requests to remote Oracle Names server will be redirected to an Oracle Names server in the region which is authoritative to the requested name.

Disabling forwarding can reduce the load on a particular Oracle Names server, but makes it impossible to direct clients to remote Oracle Names servers.

Use the SET FORWARDING_AVAILABLE command to turn forwarding on or off.

Example:

```
NAMESCTL> show forwarding_available
Request forwarding is currently enabled
```

SHOW LOG_FILE_NAME

Purpose: Shows the name of the file where the Oracle Names server writes logging information

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show log_file_name
```

From the NAMESCTL utility:

```
NAMESCTL> show log_file_name
```

Arguments: None

Usage Notes: The log file name is initially set with the value in NAMES.LOG_FILE in the `names.ora` file. The default file name is `names.log`.

Example:

```
NAMESCTL> show log_file_name
Log file name is currently
/private/ora23/network/names.log
```

SHOW LOG_STATS_INTERVAL

Purpose: Displays the frequency with which the statistics are logged to the log file

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show [onames_server] [onames_server]*
log_stats_interval
```

From NAMESCTL utility:

```
NAMESCTL> show [onames_server] [onames_server]*
log_stats_interval
```

Arguments: Zero or more Oracle Names servers separated by a space. If no names are given, then the setting is displayed for the current server.

SHOW LOG_STATS_INTERVAL

Usage Notes:

The LOG_STATS_INTERVAL is initially set with the value in NAMES.LOG_STATS_INTERVAL in the `names.ora` file. By default, the value is 0, or no logging.

Example:

```
NAMESCTL> show log_stats_interval
Statistic counter logging is currently disabled
```

SHOW NAMESCTL_TRACE_LEVEL

Purpose:

Displays the level at which the NAMESCTL utility is being traced

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system:

```
NAMESCTL show NAMESCTL_trace_level
```

From the NAMESCTL utility:

```
NAMESCTL> show NAMESCTL_trace_level
```

Arguments:

None

Usage Notes:

Tracing assists in diagnosing unexpected or unidentifiable failures in processing the NAMESCTL utility. Tracing writes a series of events from normal NAMESCTL processing to an operating system file for review by the administrator.

Tracing output is at three levels OFF (none), USER (basic information), or ADMIN (maximum amount of information).

Example:

```
NAMESCTL> show NAMESCTL_trace_level
Controller’s trace level is currently 0
```

SHOW REQUESTS_ENABLED

Purpose: Shows whether or not the Oracle Names server is responding to requests

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show [onames_server] [onames_server]*
requests_enabled
```

From the NAMESCTL utility:

```
NAMESCTL> show [onames_server] [onames_server]*
requests_enabled
```

Arguments: Zero or more Oracle Names servers separated by a space. If no names are given, then the setting is displayed for the current server.

Usage Notes: If requests are disables, all requests to the Oracle Names server will be refused.

Example:

```
NAMESCTL> show requests_enabled
General request processing is currently enabled
```

SHOW RESETS_STATS_INTERVAL

Purpose: Shows the interval set on how often the statistics are dumped to the log file

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show reset_stats_interval
```

From the NAMESCTL utility:

```
NAMESCTL> show reset_stats_interval
```

Usage Notes: If RESET_STATS_INTERVAL is initially set with the value in NAMES.RESET_STATS_INTERVAL. By default the value is set to 0, or no reset. This results in the Oracle Names server accumulating statistics the entire time it runs. For example, if statistics are reset every day, then the statistics will represent totals for the day rather than the entire time the server has been running.

Example:

```
NAMESCTL> show reset_stats_interval
Statistic counter reset interval is currently 5 minutes
```

SHOW SAVE_CONFIG_INTERVAL

Purpose: Displays the interval in seconds that the SET command is scheduled to save to the `names.ora` file

Prerequisites: The command only displays an interval if the SET SAVE_CONFIG_INTERVAL command was set.

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show save_config_interval
```

From NAMESCTL utility:

```
show save_config_interval
```

Example:

```
NAMESCTL> set save_config_interval 10
```

SHOW SAVE_CONFIG_ON_STOP

Purpose: Indicates whether NAMESCTL SET command is scheduled to save its changes to the `names.ora` file

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show save_config_on_stop
```

From the NAMESCTL utility:

```
NAMESCTL> show save_config_on_stop
```

Arguments: None

Example:

```
NAMESCTL> show save_config_on_stop
```

SHOW SERVER

Purpose:

Displays the current Oracle Names server name

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system:

```
NAMESCTL show server
```

From the NAMESCTL utility:

```
NAMESCTL> show server
```

Arguments:

None

Usage Notes:

SHOW SERVER displays the current Oracle Names server that commands will operate on

Example:

```
NAMESCTL> show server
currently managing name server "NameServer.us.oracle.com
Version banner is "Oracle Names for SunOS: Version 8.1.6.0.0"
```

SHOW STATUS

Purpose:

Displays the general status information about the Oracle Names server

Prerequisites:

None

Password required if one has been set:

No

Syntax:

From the operating system:

```
NAMESCTL show [onames_server] [onames_server]*
status
```

From the NAMESCTL utility:

```
NAMESCTL> show [onames_server] [onames_server]*
status
```

Arguments:

Zero or more Oracle Names servers separated by a space. If no names are given, then the setting is displayed for the current server.

Usage Notes:

Shows the current state of an Oracle Names server.

This command is identical to the STATUS command.

Example:

```
NAMESCTL> show status
Version Banner is "Oracle Names for SunOS: Version 8.1.6.0.0"
Server has been running for:1 day 2 hours 3 minutes 35.16
seconds....
```

SHOW SYSTEM_QUERIES

Purpose: Displays the next occurrence of all system queries

Prerequisites: This is only relevant for distributed configurations. There are no system queries with only one administrative region.

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show system_queries
```

From the NAMESCTL utility:

```
NAMESCTL> show system_queries
```

Arguments: None

Usage Notes: System queries are performed at intervals to keep information among Oracle Names servers current.

There is no specific action that can change the activities listed as system queries. Being able to show them gives the administrator an understanding of when a system change will occur, and can assist in a decision to RESTART.

Example:

```
NAMESCTL> show system_queries
System query index number:1
Query ID:49824
Query next issued in:2 hours 55 min 3.84 seconds
Query state:2
Name:""
Desired data type:ns.smd
```

SHOW TRACE_FILE_NAME

Purpose: Displays the trace file name and path for the current Oracle Names server

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show trace_file_name
```

From the NAMESCTL utility:

```
NAMESCTL> show trace_file_name
```

SHOW TRACE_FILE_NAME

Arguments: None

Usage Notes: The trace file name is initially set with the value in the NAMES.TRACE_FILE in the `names.ora` file.The default value is `names.trc`. This file must be a valid file name, and the file must be writable to the Oracle Names server.

Example:

```
NAMESCTL> show trace_file_name
Trace file name is currently
/private/ora23/network/names.trc
```

SHOW TRACE_LEVEL

Purpose: Displays the trace level for tracing the current Oracle Names server

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show trace_level
```

From the NAMESCTL utility:

```
NAMESCTL> show trace_level
```

Arguments: None

Usage Notes: Tracing assists in diagnosing unexpected or unidentifiable failures in processing the current Oracle Names server. Tracing writes a series of events from normal Oracle Names server processing to an operating system file for review by the administrator.

Tracing output is at three levels OFF (none), USER (basic information), or ADMIN (maximum amount of information).

Example:

```
NAMESCTL> show trace_level
Trace level is currently 0
```

SHOW VERSION

Purpose: Displays the current version and name of the Oracle Names server

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL show [onames_server] [onames_server]*
version
```

From the NAMESCTL utility:

```
NAMESCTL> show [onames_server] [onames_server]*
version
```

Arguments: Zero or more Oracle Names servers separated by a space. If no names are given, then the setting is displayed for the current server.

Usage Notes: This banner identifies the server by name and version. This can be useful when clearing up minor difficulties. This command is enabled every time you connect NAMESCTL to a server.

Example:

```
NAMESCTL> show version
Currently managing Oracle Names server "NameServer.com"
Version banner is "Oracle Names for SunOS: Version 8.1.6.0.0"
```

SHUTDOWN

Purpose: Stops one or more Oracle Names servers.

Prerequisites: The Oracle Names server(s) must be started.

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
NAMESCTL shutdown [onames_server] [onames_server]*
```

From the NAMESCTL utility:

```
NAMESCTL> shutdown [onames_server] [onames_server]*
```

Arguments: Zero or more Oracle Names server names separated by a space. When no arguments are supplied, only the current Oracle Names server is shut down.

SHUTDOWN

Usage Notes:

SHUTDOWN stops the current Oracle Names server and unloads the program from memory. an Oracle Names server should only be shut down for operational reasons like upgrades or machine maintenance. The preferred way to stop and start an Oracle Names server is using the RESTART command because you can perform it from anywhere in the network. If SHUTDOWN and START are processed individually, they must occur on the Oracle Names server machine.

This command is identical to the STOP command.

Example:

```
NAMESCTL> shutdown
Confirm [yes or no] yes
Server shut down.
```

START

Purpose:

Loads the Oracle Names service program and starts loading system and local administrative region data

Prerequisites:

Oracle Names server must be stopped.

Password required if one has been set:

No

Syntax:

From the operating system:

```
NAMESCTL start
```

From the NAMESCTL utility:

```
NAMESCTL> start
```

Arguments:

None

Usage Notes:

START is the command used to initially load an Oracle Names server into memory. At startup, the Oracle Names server reads its configuration files to set up its operating parameters, then loads all data for the administrative region.

Security on Oracle Names server startup is supplied through the operating system Oracle Names is installed on. Because an Oracle Names server must be started from a local session, network security is not an issue.

This command is identical to the STARTUP command.

START

Example:

```
NAMESCTL> start
Starting "/private/dsteiner/sales/bin/names"...server
successfully started

Currently managing name server "namesrv1.us.oracle.com"
Version banner is "Oracle Names for Solaris: Version
8.1.6.0.0"

Server name:
namesrv1.us.oracle.com
Server has been running for:             0.16 seconds
Request processing enabled:              yes
Request forwarding enabled:              yes
Requests received:                       0
Requests forwarded:                      0
Foreign data items cached:               0
Region data next checked for reload in:  not set
Region data reload check failures:       0
Cache next checkpointed in:              not set
Cache checkpoint interval:               not set
Cache checkpoint file name:
/private/dsteiner/sales/network/names/ckpcch.ora
Statistic counters next reset in:        not set
Statistic counter reset interval:        not set
Statistic counters next logged in:       not set
Statistic counter logging interval:      not set
Trace level:                             0
Trace file name:
/private/dsteiner/sales/network/trace/names_10841.trc
Log file name:
/private/dsteiner/sales/network/log/names.log
System parameter file name:
/private/dsteiner/sales/network/admin/names.ora
Command-line parameter file name:        ""
Administrative region name:              ""
Administrative region description:       ""
ApplTable Index:                         0
Contact                                  ""
Operational Status                       0
Save Config on Stop                      yes
```

START_CLIENT_CACHE

Purpose: Starts the client cache daemon process

Prerequisites: The client cache daemon process must be stopped.

An Oracle Names server List must exist before you run the client cache daemon process.

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL start_client_cache
```

From the NAMESCTL utility:

```
NAMESCTL> start_client_cache
```

Arguments: None

Usage Notes: Once started, the client cache daemon process will store all information received from an Oracle Names server, making lookup faster.

Example:

```
NAMESCTL> start_client_cache
```

STARTUP

Purpose: Loads the Oracle Names service program and starts loading system and local administrative region data

Prerequisites: Oracle Names server must be stopped.

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL startup
```

From the NAMESCTL utility:

```
NAMESCTL> startup
```

Arguments: None

Usage Notes: This command is identical to the START command.

Example: See example for START.

STATUS

Purpose: Displays statistics for one or more Oracle Names servers as well as many of its internal settings

Prerequisites: Oracle Names server must be started.

Password required if one has been set: No

Syntax: From the operating system:

```
NAMESCTL status [onames_server] [onames_server]*
```

From NAMESCTL utility:

```
NAMESCTL> status [onames_server] [onames_server]*
```

Arguments: Zero or more Oracle Names server names separated by a space. When no arguments are supplied, status is given only for the current Oracle Names server.

Usage Notes: STATUS shows the activity of the Oracle Names server over time and its state at a point in time.

Example:

```
NAMESCTL> status
Version banner is "Oracle Names for SunOS:
8.1.6.0.0"
Server name:NSERVER.com
Server has been running for:1 day 20 hours ........
```

STOP

Purpose: Stops one or more Oracle Names servers

Prerequisites: Oracle Names server must be started.

Password required if one has been set: Yes

If a password is set, the SET PASSWORD command must be issued prior to this command.

Syntax: From the operating system:

```
NAMESCTL stop  [onames_server] [onames_server]*
```

From the NAMESCTL utility:

```
NAMESCTL> stop  [onames_server] [onames_server]*
```

Arguments: Zero or more Oracle Names server names separated by a space. When no arguments are supplied, only the current Oracle Names server is stopped.

STOP

Usage Notes:

STOP stops the current Oracle Names server and unloads the program from memory. an Oracle Names server should only be shut down for operational reasons like upgrades or machine maintenance. The preferred way to stop and start an Oracle Names server is using the RESTART command because you can issue it from anywhere in the network. If STOP and START are processed individually, they must occur on the Oracle Names server machine.

This command is identical to the SHUTDOWN command.

Example:

```
NAMESCTL> stop
Confirm [yes or no]: yes
Server shut down
```

TIMED_QUERY

Purpose:

Show all data in the Oracle Names server cache

Password required if one has been set:

No

Syntax:

From the operating system:

```
NAMESCTL timed_query
```

From the NAMESCTL utility:

```
NAMESCTL> timed_query [time]
```

Arguments:

Time in seconds

Usage Notes:

TIMED_QUERY returns all data since *time*. The *time* argument returns all objects registered after a given time. To use the *time* argument, the first TIMED_QUERY dumps out all information available since startup. At the end of the first dump is a "last timestamp" number which gives a bookmark as to where the last dump of information ended. To see all logged data since that point, provide the "last timestamp" number in the *time* argument.

UNREGISTER

Purpose: Removes a network object from an Oracle Names server

Prerequisites: None

Password required if one has been set: No

Syntax: From the operating system

```
NAMESCTL unregister object_name  [-d
[(description=](address=(protocol_address_information))[)]]
[-h hostname] [-l listener_name]
```

From the NAMESCTL utility:

```
NAMESCTL> unregister object_name [-d
[(description=](address=(protocol_address_information))[)]]
[-l listener_name]
```

See Also: Appendix B for further information about protocol addresses and parameters

Arguments: Mandatory object name and the address, listener, or hostname that it was registered with.

Usage Notes: Provides a manual mechanism for unregistering a service. The definition for that object is removed from the Oracle Names servers in the region. If the object was registered with an address, listener name, or a hostname, the address, listener name, or hostname must be provided on the command line in order to unregister the object.

Example:

```
NAMESCTL> unregister parts -t oracle_database -d
(description= (address=
(protocol=tcp)(host=nineva)(port=1575)) (connect_
data=(service_name=db3)))
```

VERSION

Purpose:	Displays the current version and name of the Oracle Names server
Prerequisites:	None
Password required if one has been set:	No

Syntax:	From the operating system:
	`NAMESCTL version`
	From the NAMESCTL utility:
	`NAMESCTL> version`

Arguments:	Zero or more Oracle Names servers separated by a space. If no names are given, then the setting is displayed for the current server
Usage Notes:	This banner identifies the server by name and version. This can be useful when clearing up minor difficulties. This command is enabled every time you connect NAMESCTL to a server.

Example:	`NAMESCTL> version`
	`Currently managing Oracle Names server "NameServer.com"`
	`Version banner is "Oracle Names for SunOS: Version 8.1.6.0.0"`

Oracle Connection Manager Control Utility (CMCTL)

The Oracle Connection Manager Control Utility (CMCTL) is a tool that you run from the operating system prompt to start and control Oracle Connection Manager. The general syntax of CMCTL is as follows:

```
CMCTL command [process_type]
```

process_type is the names of process that the command is being executed on. The choices are:

- `cman` for both the CMGW (Oracle Connection Manager Gateway Process) gateway and CMADMIN (Oracle Connection Manager Administrative Process) administrative processes (recommended)
- `cm` for the CMGW process

Note: `cm` should only be used to reserve resources. The CMGW process performs all Oracle Connection Manager basic functions and can run without the CMADMIN process.

CMCTL also supports an `adm` process type for the CMADMIN process. This process type should not be used.

For example, to start both the gateway and administration processes, you would execute the following:

From the operating system:

```
CMCTL start cman
```

From the CMCTL program:

```
CMCTL> start cman
```

The CMCTL utility contains several types of commands:

- Operational commands such as START, STOP, SHUTDOWN, and so forth.
- Modifier commands, such as SET *command*
- Informational commands, such as STATUS, STATS and SHOW *command*
- Command utility operational commands, such as EXIT, QUIT, and HELP

SET and SHOW Commands

You can use the SET command to change some parameter values for an Oracle Connection Manager or the CMCTL environment during a CMCTL control utility session. These settings are only valid for the current CMCTL session. You cannot save parameter settings to the `cman.ora` file.

You can use the SHOW command to display the current value of a configuration setting.

Shutdown Commands

When shutting down Oracle Connection Manager, Oracle recommends using SHUTDOWN rather than STOP and STOPNOW. SHUTDOWN offers all the functionality of STOP and STOPNOW.

CMCTL Commands

The following commands are available through CMCTL:

ACCEPT_CONNECTIONS

Purpose: Enables or disables Oracle Connection Manager to accept new connections

Prerequisites: Oracle Connection Manager must be running.

Syntax: From the operating system:

```
CMCTL accept_connections [argument]
```

From the CMCTL utility:

```
CMCTL> accept_connections [argument]
```

Arguments: ON (default) - Enables Oracle Connection Manager to accept new connections.

OFF - Rejects new connections to Oracle Connection Manager

Usage Notes: If set to OFF, existing connections are not affected.

Example:

```
CMCTL> accept_connections off
Profile of the CMAN
----------------------------------------
Current state             offline
```

CLOSE_RELAY

Purpose: Enables a connection identified by relay number to be shut down

Note: Use the SHOW RELAY command to obtain the relay numbers for connections.

Prerequisites: Oracle Connection Manager must be running.

Syntax: From the operating system:

```
CMCTL close_relay [argument]
```

From the CMCTL utility:

```
CMCTL> close_relay [argument]
```

Arguments: *relay_num* - Specifies the relay number of the connection to terminate immediately.

ALL - Immediately terminates all connections

CLOSE_RELAY

Usage Notes: Use this command with caution. From the time the SHOW RELAY command is executed to the time this command is executed, the same relay may have been reused by another connection.

Example:

```
CMCTL> close_relay 0
Relay is not active

CMCTL> close_relay 0
The command completed successfully
```

EXIT

Purpose: Exits the CMCTL utility program.

Prerequisites: None

Syntax: From the operating system:

```
CMCTL exit
```

From the CMCTL utility:

```
CMCTL> exit
```

Usage Notes: This command is identical to QUIT command.

HELP

Purpose: Provides a list of all the CMCTL commands available. In response to one of the HELP commands, CMCTL displays help on how to use the command.

Prerequisites: None

Syntax: From the operating system:

```
CMCTL help
CMCTL help [command]
```

From the CMCTL utility:

```
CMCTL> help
CMCTL> help [command]
```

Arguments: HELP command. Commands are shown in the following example output.

HELP

Example:

```
CMCTL> help
The following operations are available
An asterisk (*) denotes a modifier or extended command:
accept_connections
close_relay
exit
quit
set*
show*
shutdown
start
stats
status
stop
stopnow
version
```

QUIT

Purpose: Quits CMCTL, and returns you to the operating system prompt.

Prerequisites: None

Syntax: From the operating system:

```
CMCTL quit
```

From the CMCTL utility:

```
CMCTL> quit
```

Usage Notes: This command is identical to the EXIT command.

SET

Purpose: Lists the available configuration commands that can be set for the current CMCTL session.

Prerequisites: None

Syntax: From the operating system:

```
CMCTL set
CMCTL set [command]
```

From the CMCTL utility:

```
CMCTL set
CMCTL> set [command]
```

SET

Arguments: SET commands. Commands are shown in the following example output.

Example:

```
CMCTL> set
The following operations are available
An asterisk (*) denotes a modifier or extended command:
authentication_level
displaymode
log_level
relay_statistics
```

SET AUTHENTICATION_LEVEL

Purpose: Sets the level of security for the Oracle Connection Manager

Prerequisites: Oracle Connection Manager must be running.

Syntax: From the operating system:

```
CMCTL set authentication_level level
```

From the CMCTL utility:

```
CMCTL> set authentication_level level
```

Arguments: Authentication level:

0 (default) - No authenticating is required for client connections.

1 - Rejects connections that are not employing Secure Network Service (SNS) to perform client authentication

Note: SNS is included with Oracle Advanced Security

Example:

```
CMCTL> authentication_level 0
Profile of the CMAN
-----------------------------------------
AUTHENTICATION_LEVEL    = 0
```

SET DISPLAYMODE

Purpose: Changes the output of CMCTL START, CMCTL STATS, CMCTL STATUS, CMCTL STOP, CMCTL VERSION commands

Prerequisites: None

Syntax: From the operating system:

```
CMCTL set displaymode [argument]
```

From the CMCTL utility:

```
CMCTL> set displaymode [argument]
```

Arguments: COMPAT (default) - Displays output that is compatible with older versions of Oracle Connection Manager.

VERBOSE (recommended) - Displays a formatted and descriptive output.

Example:

```
CMCTL> set displaymode compat
Current display mode is COMPATible
```

SET LOG_LEVEL

Purpose: Sets the log level for the Oracle Connection Manager

Prerequisites: Oracle Connection Manager must be running.

Syntax: From the operating system:

```
CMCTL set log_level [level]
```

From the CMCTL utility:

```
CMCTL> set log_level [level]
```

Arguments: 0 through 4:

- level 0 (default) - no logging
- level 1 - basic reporting
- level 2 - RULE_LIST matching lookup reporting
- level 3 - relay blocking reporting
- level 4 - relay I/O counts reporting

Example:

```
CMCTL> set log_level 0

Profile of the CMAN
----------------------------------------
LOG_LEVEL              = 0
```

SET RELAY_STATISTICS

Purpose:

Turns statistic collection pertaining to the I/O of connections of the Oracle Connection Manager on or off

When a relay closes and this command is set to ON, the following statistics are recorded in the appropriate LOG_RECORD of the `cman_`*`pid`*`.log` file on UNIX and `cman`*`pid`*`.log` file on Windows NT:

- Number of IN bytes
- Number of OUT bytes
- Number of IN packets
- Number of OUT packets

Prerequisites:

Oracle Connection Manager must be running.

Syntax:

From the operating system:

```
CMCTL set relay_statistics [argument]
```

From the CMCTL utility:

```
CMCTL> set relay_statistics [argument]
```

Arguments:

ON

OFF (default)

Example:

```
CMCTL> set relay_statistics off

Profile of the CMAN
-----------------------------------------
RELAY_STATISTICS       = no
```

SHOW

Purpose:

Lists the available commands that can be shown for the current CMCTL session. In response to one of the SHOW commands, CMCTL displays the current setting for that parameter.

Prerequisites:

None

Syntax:

From the operating system:

```
CMCTL show
CMCTL show [command]
```

From the LSNRCTL utility:

```
CMCTL> show
```

```
CMCTL> show [command]
```

SHOW

Arguments: SHOW command. Commands are shown in the following example output.

Example:

```
CMCTL> show
The following operations are available
An asterisk (*) denotes a modifier or extended command:
address
ALL
displaymode
profile
relay
rules
```

SHOW ADDRESS

Purpose: Lists the listening protocol address of the Oracle Connection Manager

Prerequisites: None

Syntax: From the operating system:

```
CMCTL show address
```

From the CMCTL utility:

```
CMCTL> show address
```

Arguments: None

Usage Notes: The address is set with the CMAN parameter in the `cman.ora` file.

Example:

```
CMCTL> show address

Address List
-----------------------------------------
(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1594)(PORT=1630)(QUEUESIZE=3
2))
(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1594)(PORT=1631)(QUEUESIZE=3
2))
```

SHOW ALL

Purpose: Displays output for SHOW ADDRESS, SHOW PROFILE and SHOW RULES commands

Prerequisites: None

Syntax: From the operating system:

```
CMCTL show all
```

From the CMCTL utility:

```
CMCTL> show all
```

Arguments: None

Example:

```
CMCTL> show all
Address List
-----------------------------------------
(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1594)(PORT=1630)(QUEUESIZE=3
2))
(ADDRESS=(PROTOCOL=tcp)(HOST=dlsun1594)(PORT=1631)(QUEUESIZE=3
2))

Profile of the CMAN
-----------------------------------------
MAXIMUM_RELAYS          = 2048
RELAY_STATISTICS        = yes
AUTHENTICATION_LEVEL    = 0
LOG_LEVEL               = 4
ANSWER_TIMEOUT          = 0
MAXIMUM_CONNECT_DATA    = 1024
USE_ASYNC_CALL          = yes
TRACING                 = yes
TRACE_DIRECTORY         = default
MAX_FREELIST_BUFFERS    = 0
REMOTE_ADMIN            = no

Rule List
-----------------------------------------
(rule=(src=144.25.185.60)(dst=x)(srv=x)(act=accept))
(rule=(src=sguan-pc)(dst=x)(srv=x)(act=accept))
```

SHOW DISPLAYMODE

Purpose:

Shows the current display mode used for the CMCTL START, CMCTL STATS, CMCTL STATUS, CMCTL STOP, CMCTL VERSION commands

COMPAT (default) - Displays output that is compatible with older versions of Oracle Connection Manager.

VERBOSE - Displays a formatted and descriptive output.

Syntax:

From the operating system:

```
CMCTL show displaymode
```

From the CMCTL utility:

```
CMCTL> show displaymode
```

Example:

```
CMCTL> show displaymode
Current display mode is VERBose
```

SHOW PROFILE

Purpose:

Lists the current parameter settings for the Oracle Connection Manager. Information is obtained from the CMAN_PROFILE parameters in the `cman.ora` file and any changes made with the CMCTL SET command.

Syntax:

From the operating system:

```
CMCTL show profile
```

From the CMCTL utility:

```
CMCTL> show profile
```

Example:

```
CMCTL> show profile
Profile of the CMAN
-----------------------------------------
MAXIMUM_RELAYS         = 2048
RELAY_STATISTICS       = yes
AUTHENTICATION_LEVEL   = 0
LOG_LEVEL              = 4
SHOW_TNS_INFO          = yes
ANSWER_TIMEOUT         = 0
MAXIMUM_CONNECT_DATA   = 1024
USE_ASYNC_CALL         = yes
TRACING                = yes
TRACE_DIRECTORY        = default
MAX_FREELIST_BUFFERS   = 0
REMOTE_ADMIN           = no
```

SHOW RELAY

Purpose: Displays the current status of a selected relay (connection) or all active relays for the Oracle Connection Manager

Prerequisites: Oracle Connection Manager must be running.

Syntax: From the operating system:

```
CMCTL show relay {argument}
```

From the CMCTL utility:

```
CMCTL> show relay {argument}
```

Arguments: *relay_num* - Shows the status information for a specific relay.

ACTIVE - Shows the list of active relays.

Usage Notes: For active relays, only relay numbers are shown. For a given relay number, the following information is displayed:

- Relay number
- Source address (client-side endpoint)
- Destination address (server-side endpoint)
- Number of IN bytes
- Number of IN packets
- Number of IN DCD probes
- Number of OUT bytes
- Number of OUT packets
- Number of OUT DCD probes

SHOW RELAY

Example:

```
CMCTL> show relay active

Active Relays
-----------------------------------------
0000 0001
CMCTL> show relay 0

Relay Information
-----------------------------------------
Relay number              0
Start-up time             22-JUL-1999 19:47:17
Src
(ADDRESS=(PROTOCOL=tcp)(HOST=144.25.185.60)(PORT=35279))
Dest
(ADDRESS=(PROTOCOL=tcp)(HOST=144.25.185.60)(PORT=1521))
Number of IN bytes        438
Number of IN packets      7
Number of IN DCD probes   0
Number of OUT bytes       364
Number of OUT packets     7
Number of OUT DCD probes  0
```

SHOW RULES

Purpose: Lists the current Net8 access rules used by the Oracle Connection Manager.

Prerequisites: Oracle Connection Manager must be running.

Syntax: From the operating system:

```
CMCTL show rules
```

From the CMCTL utility:

```
CMCTL> show rules
```

Arguments: None

Usage Notes: Rules are set with the CMAN_RULES parameter in the `cman.ora` file.

Example:

```
CMCTL> show rules

Rule List
-----------------------------------------
(rule=(src=144.25.185.60)(dst=x)(srv=x)(act=accept))
(rule=(src=sguan-pc)(dst=x)(srv=x)(act=accept))
```

SHUTDOWN

Purpose: Shuts down the Oracle Connection Manager processes

Prerequisites: None

Syntax: From the operating system:

```
CMCTL shutdown [argument] [cman]
```

From the CMCTL utility:

```
CMCTL> shutdown [argument] [cman]
```

Arguments: NORMAL (default) - No new connections will be accepted and the Oracle Connection Manager terminates after all existing connections close.

ABORT - The Oracle Connection Manager shuts down immediately, closing down all open connections.

`cman` - Starts both CMGW and CMADMIN processes. NORMAN and ABORT use this argument. Therefore, it is not necessary to explicitly specify it.

Example:

```
CMCTL> shutdown

The command completed successfully

CMCTL> shutdown abort

The command completed successfully
```

START

Purpose: Starts Oracle Connection Manager

Prerequisites: Oracle Connection Manager using the same listening address must not be running.

Syntax: From the operating system:

```
CMCTL start [process_type]
```

From the CMCTL utility:

```
CMCTL start [process_type]
```

Arguments: `cman` (default) - Starts both CMGW and CMADMIN processes.

`cm` - Starts the CMGW process.

Usage Notes: The SET DISPLAYMODE command changes the format and the level of detail of the output.

START

Example: COMPAT Mode:

```
CMCTL> start cman
ADMIN Status:
(STATUS=(VERSION=8.1.6.0.0)(STARTED=22-JUL-1999
19:34:15)(STATE=RUNNING))
CMAN Status:
(STATUS=(VERSION=8.1.6.0.0)(STARTED=22-JUL-1999
19:34:17)(STATE=running))
```

VERBOSE Mode:

```
CMCTL> start cman

Starting /vobs/oracle/bin/cmadmin: please wait...
STATUS of the cman_admin
-----------------------------------------
ADMIN Version             8.1.6.0.0
Start-up time             22-JUL-1999 19:40:00
Current state             RUNNING


Starting /vobs/oracle/bin/cmgw: please wait...
STATUS of the cman
-----------------------------------------
CMAN  Version             8.1.6.0.0
Start-up time             22-JUL-1999 19:40:02
Current state             running
```

STATS

Purpose: Displays statistical information for the Oracle Connection Manager

Prerequisites: Oracle Connection Manager must be running.

Syntax: From the operating system:

`CMCTL stats [`*`process_type`*`]`

From the CMCTL utility:

`CMCTL stats [`*`process_type`*`]`

Arguments: `cman` (default) - Displays statistics for the CMGW process.

`cm` - Displays statistics for the CMGW process.

STATS

Usage Notes:

The following statistics are displayed:

- TOTAL_RELAYS (COMPAT) | Total number of connections handled (VERBOSE)

  The total number of connections the Oracle Connection Manager has established since it started.
- ACTIVE_DELAYS (COMPAT) | Number of currently active relays (VERBOSE)

  The number of currently active connections.
- MOST_RELAYS (COMPAT) | Peak active relays (VERBOSE)

  The maximum number of concurrent connections the Oracle Connection Manager has ever held since it started.
- OUT_OF_RELAY (COMPAT) | Total refusals due to max_relays exceeded (VERBOSE)

  The total number of connect request refusals due to out-of-relay since the Oracle Connection Manager started.
- TOTAL_REFUSED (COMPAT) | Total number of connections refused (VERBOSE)

  The total number of connect request refusals since the Oracle Connection Manager started.

Usage Notes:

The SET DISPLAYMODE command changes the format and the level of detail of the output.

Example:

COMPAT Mode:

```
CMCTL> stats
CMAN Status:
(STATISTICS=(TOTAL_RELAYS=0)(ACTIVE_RELAYS=0)(MOST_
RELAYS=0)(OUT_OF_RELAY=0)(TOTAL_REFUSED=0))
```

VERBOSE Mode:

```
CMCTL> stats
STATISTICS of CMAN
-----------------------------------------
Total number of connections handled          0
Number of currently active relays            0
Peak active relays                           0
Total refusals due to max_relays exceeded    0
Total number of connections refused          0
```

STATUS

Purpose: Displays basic status information, including version, start time and current statistics

Prerequisites: None

Syntax: From the operating system:

```
CMCTL status [process_type]
```

From the CMCTL utility:

```
CMCTL status [process_type]
```

Arguments: `cman` (default) - Display status information for both the CMGW and CMADMIN processes.

`cm` - Displays status information for the CMGW process.

Usage Notes: The SET DISPLAYMODE command changes the format and the level of detail of the output.

Example: COMPAT Mode:

```
CMCTL> status
CMAN Status:
(STATUS=(VERSION=8.1.6.0.0)(STARTED=22-JUL-1999
19:34:17)(STATE=running))
ADMIN Status:
(STATUS=(VERSION=8.1.6.0.0)(STARTED=22-JUL-1999
19:34:15)(STATE=RUNNING))
```

VERBOSE Mode:

```
CMCTL> status
STATUS of the cman
----------------------------------------
CMAN  Version            8.1.6.0.0
Start-up time            22-JUL-1999 19:40:02
Current state            running

STATUS of the cman_admin
----------------------------------------
ADMIN Version            8.1.6.0.0
Start-up time            22-JUL-1999 19:40:00
Current state            RUNNING
```

STOP

Purpose:

Shuts down Oracle Connection Manager immediately, prompting you if there are open connections

Prerequisites:

Oracle Connection Manager must be running.

Syntax:

From the operating system:

`CMCTL stop [`*`process_type`*`]`

From the CMCTL utility:

`CMCTL stop [`*`process_type`*`]`

Arguments:

`cman` (default) - Stops both CMGW and CMADMIN processes.

`cm` - Stops both CMGW and CMADMIN processes.

Usage Notes:

Oracle recommends the SHUTDOWN command to this command.

If you issue a stop command while connections remain active, you will be prompted to confirm the stop.

The SET DISPLAYMODE command changes the format and the level of detail of the output.

Example:

COMPAT Mode:

```
CMCTL> stop
The command completed successfully.
```

VERBOSE Mode:

```
CMCTL> stop
```

STOPNOW

Purpose: Shuts down Oracle Connection Manager immediately, closing down all open connections without warning

Prerequisites: Oracle Connection Manager must be running.

Syntax: From the operating system:

```
CMCTL stopnow
```

From the CMCTL utility:

```
CMCTL stopnow
```

Arguments: `cman` (default) - Stops both CMGW and CMADMIN processes.

`cm` - Stops both CMGW and CMADMIN processes.

Usage Notes: Oracle recommends the SHUTDOWN command to this command.

Example:

```
CMCTL> stopnow
```

VERSION

Purpose: Displays the current version and name of the CMCTL utility.

Prerequisites: None

Syntax: From the operating system:

```
CMCTL version
```

From the CMCTL utility:

```
CMCTL> version
```

Usage Notes: The SET DISPLAYMODE command changes the format of the output.

Example: COMPAT Mode:

```
CMCTL> version
CMCTL Version            8.1.6.0.0
```

VERBOSE Mode:

```
CMCTL> version
CMCTL Version            8.1.6.0.0
CMAN  Version            8.1.6.0.0
```

B

Protocol Addresses

A network object is identified by a protocol address. When a connection is made, the client and the receiver of the request (listener, Oracle Names server, 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, and grants a connection based on its address information matching the client information.

This appendix includes the following sections:

- ADDRESSes and ADDRESS_LISTs
- Protocol Parameters
- Recommended Port Numbers
- Port Number Limitations

ADDRESSes and ADDRESS_LISTs

Protocol address are comprised of ADDRESS and ADDRESS_LIST elements:

ADDRESS

Purpose: Defines a protocol address. This parameter may be embedded under ADDRESS_LIST or DESCRIPTION. A DESCRIPTION is typically only used in a `tnsnames.ora` or `listener.ora` file.

See Also: Each protocol has its own required parameters needed in the address, as explained in "Protocol Parameters" on page B-3.

Example:

```
(address=
 (protocol=tcp)
 (host=sales-pc)
 (port=1521))
```

ADDRESS_LIST

Purpose: Defines a list of addresses that share common characteristics.

Example:

```
(address_list=
 (load_balance=on)
 (address=
  (protocol=tcp)
  (host=sales-pc)
  (port=1521))
 (address=
  (protocol=tcp)
  (host=hr-pc)
  (port=1521)))
(address_list=
 (address=
  (protocol=tcp)
  (host=finance-pc)
  (port=1521)))
```

Protocol Parameters

The listener, Oracle Names server and Oracle Connection Manager are identified by protocol addresses. The table below describes the parameters used by the supported Oracle protocols:

Protocol	Parameter	Description
All	PROTOCOL	Indicates the type of network on which the TNS-based application resides
Bequeath	PROGRAM	Identifies the Oracle8*i* executable
Bequeath	ARGV0	Identifies the service name
Bequeath	ARGS	Identifies the source of the connection (local client)
IPC	KEY	Indicates a way of identifying the server. Oracle Corporation recommends using the name of the service
Named Pipes	SERVER	Indicates the name of your Oracle8*i* server computer
Named Pipes	PIPE	Indicates the pipe name you use to connect to your server (the same PIPE keyword you specified on server with Named Pipes). This name can be any arbitrary name.
LU6.2	LU_NAME	Identifies the Oracle8*i* server; must be a fully-qualified name
LU6.2	LLU or LOCAL_LU	Identifies the local LU alias. This parameter cannot be used with LLU_NAME
LU6.2	LLU_NAME or LOCAL_LU_NAME	Specifies the local LU name; must be a fully qualified name. This parameter cannot be used with LLU.
LU6.2	MODE or MDN	Identifies the log mode entry of the LU6.2 network; the value is typically ORAPLU62
LU6.2	PLU or PARTNER_LU_NAME	Identifies the Oracle8*i* server; must be a fully qualified name.This parameter cannot be used with PLU_LA.
LU6.2	PLU_LA or PARTNER_LU_LOCAL_ALIAS	Identifies the partner LU alias of the Oracle8*i* server. This parameter cannot be used with PLU.
LU6.2	TP_NAME or TPN	Identifies the transaction program name of the host machine. This parameter is not required for a connection to an MVS host.

Protocol	Parameter	Description
SPX	SERVICE	Defines the name of the TNS-based application on the network
TCP/IP and TCP/IP with SSL	HOST	Identifies the host name or IP address of the machine
TCP/IP and TCP/IP with SSL	PORT	Identifies the host name of IP address of the machine

Recommended Port Numbers

Oracle Corporations recommends the following port numbers:

Port	Description
1521	Default listening port for client connections to the listener. In future releases, this port number may change to the officially registered port number of 2483 for TCP/IP and 2484 for TCP/IP with SSL.
2481	Recommended and officially registered listening port for client connections to the Oracle8*i* JServer option using TCP/IP
2482	Recommended and officially registered listening port for client connections to the Oracle8*i* JServer using TCP/IP with SSL
2484	Recommended and officially registered listening port for client connections to the listener using TCP/IP with SSL
1575	Default and officially registered listening port for client connections to an Oracle Names server using TCP/IP or TCP/IP with SSL
1630	Default and officially registered listening port for client connections to Oracle Connection Manager
1830	Default and officially registered listening port for administrative commands to Oracle Connection Manager

Port Number Limitations

Oracle Corporation allows port numbers from 1 to 65535. Port numbers below 1024 are reserved for use by privileged processes on many operating systems.

If your listener is configured to use a port number below 1024, you will not be able to start it using LSNRCTL utility. LSNRCTL, even if itself running as root, cannot start the listener as `root` (on UNIX systems), however, you can always run listener executable (`tnslsnr`) as a root. This can be done by entering the command `tnslsnr` *`listener_name`* at the operating system prompt. Note that after the listener has been started it can be further administered (including stopping it) by LSNRCTL utility.

C

Configuration Parameters

This appendix provides complete listing of all Net8 configuration parameters. It contains the following sections:

- Syntax Rules for Configuration Files
- Profile Parameters (sqlnet.ora)
- Local Naming Parameters (tnsnames.ora)
- Listener Parameters (listener.ora)
- Oracle Names Parameters (names.ora)
- Oracle Connection Manager Parameters (cman.ora)
- Protocol-Specific Parameters (protocol.ora)
- Directory Server Access Parameters (ldap.ora)
- Configuration File Changes in Release 8.1

Syntax Rules for Configuration Files

The configuration files in a Net8 network consist of parameters which include keyword-value pairs. Keyword-value pairs are surrounded by parentheses:

```
parameter=(keyword=value)
```

Some keywords have other keyword-value pairs as their values:

```
(keyword=
    (keyword=value)
    (keyword=value))
```

For example, the address portion of a local naming configuration file (`tnsnames.ora`) might include the following lines:

```
(address=
    (protocol=tcp)
    (host=max)
    (port=1521))
```

Set up configuration files so that indentation reflects what keyword is the parent or owner of other keyword-value pairs. This format is not required, but it does make the files much easier to read and understand.

Even if you do not choose to indent your files in this way, you must indent a wrapped line by at least one space, or it will be misread as a new parameter. The following layout is acceptable:

```
(address=(protocol=tcp)
  (host=max)(port=1521))
```

The following layout is *not* acceptable:

```
(address=(protocol=tcp)
(host=max)(port=1521))
```

Further Syntax Rules for Configuration Files

The following rules apply to the syntax of configuration files:

- Any keyword in a configuration file that begins a parameter that includes one or more keyword-value pairs must be in the far left column of a line. If it is indented by one or more spaces, it is interpreted as a continuation of the previous line.
- All characters must belong to the *network character set* (see the next section).
- Keywords are not case sensitive. Values may be case sensitive, depending on the operating system and protocol.
- Spaces around the "=" sign are optional in keyword-value pairs.
- There is a hierarchy of keywords in that some keywords are always followed by others. At any level of the hierarchy, keywords can be listed in any order. For example, the following entries are equally valid:

```
(address=
   (protocol=tcp)
   (host=martha)
   (port=1521))

(address=
   (protocol=tcp)
   (port=1521)
   (host=martha))
```

- Keywords cannot contain spaces. Values must not contain spaces unless enclosed within double quotes (") or single quotes (’).
- The maximum length of a connect descriptor is 4 kilobytes.
- Comments can be included using the pound sign # at the beginning of a line. Anything following the sign to the end of the line is considered a comment.
- If the keyword-value pair consists of a single word or a concatenation of words on either side of the equal sign, no parentheses are needed.

Network Character Set

The network character set for keyword values consists of the following characters. Connect descriptors must be made up of single-byte characters.

A-Z, a-z

0-9

( ) < > / \

, . : ; ’ "=- _

$ + * # & ! % ? @

Within this character set, the following symbols are reserved:

( )= \ " ’ #

Reserved symbols are used as delimiters, not as part of a keyword or a value unless the keyword or value is quoted. Either single or double quotes can be used to enclose a value containing reserved symbols. To include a quote within a value that is surrounded by quotes, use different quote types. The backslash (\) is used as an escape character.

A specific example of the use of reserved symbols is a numeric DECnet object within an address. An OBJECT can be a name such as ABC or #123. These would be entered in the form:

```
(OBJECT=ABC)
```

or

```
(OBJECT=\#123)
```

Because the "#" sign is a reserved symbol, the character must be preceded by a backslash.

The following characters may be used within a connect descriptor, but not in a keyword or value:

<space> <tab> <CR> <newline>

Character Set

The listener name, net service name, and Oracle Names server are limited to the following character set:

[a...z] [A...Z] [0...9] _

The first character must be an alphabetical character. In general, up to 64 characters is acceptable. A database service name must match the global database name defined by the database administrator, which consists of a database name (originally limited to eight characters), and the database domain. Net service names and global database names are not case sensitive.

Profile Parameters (sqlnet.ora)

The following parameters are available in a profile.

The `sqlnet.ora` file is located in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows platforms, or in the directory specified by the TNS_ADMIN environment variable or registry value.

BEQUEATH_DETACH	
Purpose:	Turns signal handling on or off for UNIX systems
Default:	NO which leaves signal handling on.
Values:	■ YES - Turns off signal handling ■ NO - Leaves signal handling on
Example:	`bequeath_detach=yes`

DAEMON.TRACE_DIRECTORY	
Purpose:	Specifies the destination directory of the Oracle Enterprise Manager daemon trace file
Default:	`$ORACLE_HOME/network/trace` on UNIX platforms and *`ORACLE_HOME`*`/network/trace` on Windows platforms
Example:	`daemon.trace_directory=/oracle/traces`

DAEMON.TRACE_LEVEL

Purpose:	Turns tracing on/off to a certain specified level for the Oracle Enterprise Manager daemon
Default:	OFF
Values	■ OFF - No trace output ■ USER - User trace information ■ ADMIN - Administration trace information ■ SUPPORT - WorldWide Customer Support trace information
Example:	`daemon.trace_level=user`

DAEMON.TRACE_MASK

Purpose:	Specifies that only the Oracle Enterprise Manager daemon trace entries are logged into the trace file
Default:	`$ORACLE_HOME/network/trace` on UNIX platforms and *`ORACLE_HOME`*`/network/trace` on Windows platforms
Example:	`daemon.trace_mask=(106)`

DISABLE_OOB

Purpose:	If turned OFF, enables Net8 to send and receive "break" messages using urgent data provided by the underlying protocol. If turned 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 documentation to determine if the protocols you are using support urgent data requests. TCP/IP is an example of a protocol that supports this feature.
Default:	OFF
Example:	`disable_oob=ON`

LOG_DIRECTORY_CLIENT

Purpose:	Specifies the destination directory for the client log file
Default:	Current directory from which the executable is started
Example:	`log_directory_client=/oracle/network/log`

LOG_DIRECTORY_SERVER

Purpose:	Specifies the destination directory for the server log file
Default:	Current directory from which the executable is started
Example:	`log_directory_server=/oracle/network/log`

LOG_FILE_CLIENT

Purpose:	Specifies the name of the log file for the client
Default:	`sqlnet.log`
Example:	`log_file_client=client`

LOG_FILE_SERVER

Purpose:	Specifies the name of the log file for the server
Default:	`sqlnet.log`
Example:	`log_file_server=svr.log`

NAMES.DCE.PREFIX

Purpose:	Specifies the DCE cell name (prefix) to use for name lookupS
Default:	`/.:/subsys/oracle/names`
Example:	`names.dce.prefix=/.:/subsys/oracle/names`

NAMES.DEFAULT_DOMAIN

Purpose:	Sets the domain from which the client most often looks up names resolution requests. When this parameter is set, the default domain name is automatically appended to any unqualified net service name or service name. For example, if the default domain is set to `us.acme.com`, the connect string `CONNECT scott/tiger@sales` gets searched as `sales.us.acme.com`. If the connect string includes the domain extension, (such as `CONNECT scott/tiger@sales.acme.com`), the domain is not appended.
Default:	NULL
Example:	`names.default_domain=com`

NAMES.DIRECTORY_PATH

Purpose:	Specifies the order of the naming methods that used for client name resolution lookups
Default:	TNSNAMES, ONAMES, HOSTNAME
Values:	■ TNSNAMES ■ ONAMES ■ HOSTNAME ■ DCE ■ LDAP ■ NIS ■ NOVELL
Example:	`names.directory_path=(tnsnames, onames)`

NAMES.INITIAL_RETRY_TIMEOUT

Purpose:	Determines how long a client waits for a response from an Oracle Names server before reiterating the request to the next server in the preferred servers list
Default:	15
Minimum Value:	1
Maximum Value:	600
Example:	`names.initial_retry_timeout=20`

NAMES.MAX_OPEN_CONNECTIONS

Purpose:	Determines how many connections an Oracle Names client can have open at one time
Default:	10
Minimum Value:	3
Maximum Value:	64
Example:	`names.max_open_connections=3`

NAMES.MESSAGE_POOL_START_SIZE

Purpose:	Determines the initial number of messages allocated in the client's message pool which are used for forwarded message requests
Default:	10
Minimum Value:	3
Maximum Value:	256
Example:	`names.message_pool_start_size=10`

NAMES.NIS.META_MAP

Purpose:	Specifies the **map** file to be used to map **NIS** attributes to an NIS mapname
Default:	`sqlnet.maps`
Example:	`names.nis.meta_map=sqlnet.maps`

NAMES.NDS.NAME.CONTEXT

Purpose:	Specifies the naming context in the **NDS** tree where the database object resides
Default:	None
Example:	`cn=Payroll.o=Oracle`

NAMES.PREFERRED_SERVERS

Purpose: Indicates the name, addresses, and order of Oracle Names servers that are used for a client’s name requests

Default: None

Example:

```
names.preferred_servers= (address_list=
  (address=(protocol=ipc)(key=n23))
  (address=(protocol=tcp)(host-nineva)(key=1575))
  (address=(protocol=tcp)(host=cicada)(key=1575)))
```

NAMES.REQUEST_RETRIES

Purpose: Specifies the number of times the client should try each Oracle Names server in the list of preferred Oracle Names servers before allowing the operation to fail

Default: 1

Minimum Value: 1

Maximum Value: 5

Example:

```
names.request_retries=5
```

NAMESCTL.INTERNAL_ENCRYPT_PASSWORD

Purpose: If set to TRUE, NAMESCTL encrypts the password when it is sent to the Oracle Names server.

If set to FALSE, NAMESCTL does not encrypt the password when it is sent to the Oracle Names server. This enables unencrypted passwords to be set in the `names.ora` file with the NAMES.PASSWORD parameter.

Default: TRUE

Values: TRUE | FALSE

Example:

```
namesctl.internal_encrypt_password=true
```

NAMESCTL.INTERNAL_USE

Purpose	If set to TRUE, NAMESCTL enables a set of internal undocumented commands. All internal commands are preceded by an underscore in order to distinguish them as internal.

NAMESCTL.NO_INITIAL_SERVER

Purpose:	If set to TRUE, NAMESCTL suppresses any error messages when unable to connect to a default Oracle Names server.
Default:	FALSE
Values	TRUE | FALSE
Example:	`namesctl.no_initial_server=true`

NAMESCTL.NOCONFIRM

Purpose:	Indicates whether sensitive commands (STOP, RELOAD, RESTART) should be prompted with a confirmation when running the NAMESCTL utility
Default:	OFF
Values	ON | OFF
Example:	`namesctl.noconfirm=on`

NAMESCTL.SERVER_PASSWORD

Purpose:	Indicates the value that matches the configured password set in the `names.ora` file with the NAMES.PASSWORD parameter. This eliminates the need to enter the password with the SET PASSWORD command each time you use the NAMESCTL utility to perform secure options, such as STOP, RESTART and RELOAD.
Example:	`namesctl.server_password=secret`

NAMESCTL.TRACE_LEVEL

Purpose:	Indicates the level at which the NAMESCTL program should be traced
Default:	OFF
Values:	OFF, USER, ADMIN, SUPPORT
Example:	`namesctl.trace_level=admin`

NAMESCTL.TRACE_FILE

Purpose:	Indicates the file in which the NAMESCTL trace output is placed
Default:	`namesctl_`*`pid`*`.trc`
Example:	`namesctl.trace_file=nmsctl`

NAMESCTL.TRACE_DIRECTORY

Purpose:	Indicates the directory where trace output from the NAMESCTL utility is placed
Default:	`$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`/network/trace` on Windows NT
Example:	`namesctl.trace_directory=/oracle/trace`

NAMESCTL.TRACE_UNIQUE

Purpose:	Indicates whether a process identifier is appended to the name of each trace file generated, so that several can co-exist
Default:	ON
Values:	ON | OFF
Example:	`namesctl.trace_unique=on`

OSS.SOURCE.MY_WALLET

Purpose:	Specifies the location of wallets. Wallets are certificates, keys and trustpoints processed by SSL **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	None

Example:	`oss.source.my_wallet=`
	`(source=`
	`(method=file)`
	`(method_data=`
	`(directory=/home/smalladi/oss)))`

SQLNET.AUTHENTICATION_SERVICES

Purpose:	Enables one or more authentication services. If authentication has been installed, it is recommended that this parameter be set to either NONE or to one of the authentication methods.
Default:	None
Values:	**Authentication Available with Net8:** NONE - No authentication methods are used. A valid user name and password can be used to access the database. ALL - Enables all authentication methods to be used BEQ - Uses the Bequeath protocol NDS - Uses NDS authentication NTS - Uses Windows NT native authentication **Authentication Available with Oracle Advanced Security:** KERBEROS5 - Uses Kerberos authentication SECURID - Uses SecurID authentication CYBERSAFE - Uses Cybersafe authentication IDENTIX - Uses Identix authentication RADIUS - Uses RADIUS authentication DCEGSSAPI - Uses DCE GSSAPI authentication **See Also:** *Oracle Advanced Security Administrator's Guide*
Example:	`sqlnet.authentication_services=(beq, kerberos5, cybersafe)`

SQLNET.AUTHENTICATION_KERBEROS5_SERVICE

Purpose: Defines the name of the service used to obtain a Kerberos service ticket

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

Default: None

Example: `sqlnet.authentication_kerberos5_service= oracle`

SQLNET.AUTHENTICATION_GSSAPI_SERVICE

Purpose: Defines the CyberSAFE service principal

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

SQLNET.CLIENT_REGISTRATION

Purpose: Sets a unique identifier for this client machine. This identifier is passed to the listener with any connection request and is included in the Audit Trail. The identifier can be any alphanumeric string up to 128 characters long.

Default: None

Example: `sqlnet.client_registration 1432`

SQLNET.CRYPTO_CHECKSUM_CLIENT

Purpose: Specifies the checksum behavior for the client

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

Default: ACCEPTED

Values:

- ACCEPTED - Turn on the security service if the other side wants it.
- REJECTED - Do not turn on the security service even if the other side wants it.
- REQUESTED - Turn on the security service if the other side allows it.
- REQUIRED - Turn on the security service or do not make the connection.

Example: `sqlnet.crypto_checksum_client=accepted`

SQLNET.CRYPTO_CHECKSUM_SERVER

Purpose: Specifies the checksum behavior for the server

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

Default: ACCEPTED

Values:

- ACCEPTED - Turn on the security service if the other side wants it.
- REJECTED - Do not turn on the security service even if the other side wants it.
- REQUESTED - Turn on the security service if the other side allows it.
- REQUIRED - Turn on the security service or do not make the connection.

Example: `sqlnet.crypto_checksum_server=accepted`

SQLNET_CRYPTO_CHECKSUM_TYPE_CLIENT

Purpose: Specifies a list of crypto-checksum algorithms this client is allowed to use

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

Default: MD5

Values: MD5 - RSA Data Security's MD5 algorithm

Example: `sqlnet.crypto_checksum_types_client=(md5)`

SQLNET_CRYPTO_CHECKSUM_TYPE_SERVER

Purpose: Specifies a list of crypto-checksum algorithms this server is allowed to use

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

Default: MD5

Values: MD5 - RSA Data Security's MD5 algorithm

Example: `sqlnet.crypto_checksum_types_server=(md5)`

SQLNET.CRYPTO_SEED

Purpose:	Specifies the characters used when generating cryptographic keys. The more random the characters are, the stronger the keys are. The string should be 10-70 random characters. This is required for when encryption or checksumming are turned on. Encryption is turned on if the SQLNET.ENCRYPTION_CLIENT parameter is specified for the client and the SQLNET.ENCRYPTION_SERVER parameter is specified for the server; checksumming is turned on if the SQLNET.CRYPTO_CHECKSUM_CLIENT parameter is specified for the client and the SQLNET.CRYPTO_CHECKSUM_SERVER parameter is specified for the server. **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	qwertyuiopasdfghjkl;zxcvbnm,.s1
Example:	`sqlnet.crypto_seed="qwertyuiopasdfghjkl;zxcvbnm,.s1"`

SQLNET.ENCRYPTION_CLIENT

Purpose:	Turns encryption on for the client **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	ACCEPTED
Values:	■ ACCEPTED - Turn on the security service if the other side wants it. ■ REJECTED - Do not turn on the security service even if the other side wants it. ■ REQUESTED - Turn on the security service if the other side allows it. ■ REQUIRED - Turn on the security service or do not make the connection.
Example:	`sqlnet.encryption_client=accepted`

SQLNET.ENCRYPTION_SERVER

Purpose:	Turns encryption on for the client **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	ACCEPTED
Values:	■ ACCEPTED - Turn on the security service if the other side wants it. ■ REJECTED - Do not turn on the security service even if the other side wants it. ■ REQUESTED - Turn on the security service if the other side allows it. ■ REQUIRED - Turn on the security service or do not make the connection.
Example:	`sqlnet.encryption_server=accepted`

SQLNET.ENCRYPTION_TYPES_CLIENT

Purpose:	Specifies a list of encryption algorithms this client is allowed to use **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	All algorithms are used if none are specified.
Values:	One or more of the following: RC4_40 - This is RSA RC4 (40-bit key size) RC4_56 - This is RSA RC4 (56-bit key size) RC4_128 - This is RSA RC4 (128-bit key size) DES - This is Standard DES (56-bit key size) DES40 - This is DES40 (40-bit key size)
Example:	`sqlnet.encryption_types_client=(rc4_40)`

SQLNET.ENCRYPTION_TYPES_SERVER

Purpose:	Specifies a list of encryption algorithms this server is allowed to use when acting as a server **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	All algorithms are used if none are specified.
Values:	RC4_40 - This is RSA RC4 (40-bit key size) RC4_56 - This is RSA RC4 (56-bit key size) RC4_128 - This is RSA RC4 (128-bit key size) DES - This is Standard DES (56-bit key size) DES40 - This is DES40 (40-bit key size)
Example:	`sqlnet.encryption_types_server=(rc4_40, des, ...)`

SQLNET.EXPIRE_TIME

Purpose:	Determines time interval to send a probe to verify the session is alive **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	None
Minimum Value:	0 minutes
Recommended Value:	10 minutes
Example:	`sqlnet.expire_time=10`

SQLNET.IDENTIX_FINGERPRINT_DATABASE

Purpose:	Specifies the service name or alias for the authentication fingerprint database **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	None
Example:	`sqlnet.identix_fingerprint_database=fingrdb`

SQLNET.IDENTIX_FINGERPRINT_DATABASE_USER

Purpose:	Specifies the well-known user name for the fingerprint database **See Also:** *Oracle Advanced Security Administrator’s Guide*
Default:	None
Example:	`sqlnet.identix_fingerprint_database_user=manager`

SQLNET.IDENTIX_FINGERPRINT_DATABASE_PASSWORD

Purpose:	Specifies the well-known password for the fingerprint database **See Also:** *Oracle Advanced Security Administrator’s Guide*
Default:	None
Example:	`sqlnet.identix_fingerprint_database_password=`*password*

SQLNET.IDENTIX_FINGERPRINT_METHOD

Purpose:	Specifies the method name for the fingerprint database. The method name must be ORACLE. **See Also:** *Oracle Advanced Security Administrator’s Guide*
Default:	None
Example:	`sqlnet.identix_fingerprint_method=oracle`

SQLNET.KERBEROS5_CC_NAME

Purpose:	Specifies the complete path name to the Kerberos credentials cache file **See Also:** *Oracle Advanced Security Administrator’s Guide*
Default:	`/usr/tmp/krbcache` on UNIX and `c:\tmp\krbcache` on Windows platforms
Example:	`sqlnet.kerberos5_cc_name= /usr/tmp/krbcache`

SQLNET.KERBEROS5_CLOCKSKEW

Purpose:	Specifies how many seconds can pass before a Kerberos credential is considered out of date **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	300
Example:	`sqlnet.kerberos5_clockskew=1200`

SQLNET.KERBEROS5_CONF

Purpose:	Specifies the complete path name to the Kerberos configuration file, which contains the realm for the default KDC and maps realms to KDC hosts **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	`/krb5/krb.conf` on UNIX and `c:\krb5\krb.conf` on Windows platforms
Example:	`sqlnet.kerberos5_conf=/krb5/krb.conf`

SQLNET.KERBEROS5_KEYTAB

Purpose:	Specifies the complete path name to the Kerberos principal/secret key mapping file, which is used to extract keys and decrypt incoming authentication information **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	`/etc/v5srvtab` on UNIX and `c:\krb5\v5srvtab` on Windows platforms
Example:	`sqlnet.kerberos5_keytab=/etc/v5srvtab`

SQLNET.KERBEROS5_REALMS

Purpose:	Specifies the complete path name to the Kerberos realm translation file, which provides a mapping from a host name or domain name to a realm **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	`/krb5/krb.realms` on UNIX and `c:\krb5\krb.realms` on Windows platforms
Example:	`sqlnet.kerberos5_realms= /krb5/krb.realms`

SQLNET.RADIUS_ALTERNATE

Purpose: Specifies an alternate RADIUS server used in case the primary server is unavailable. The value can be either the IP address or host name of the server

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

Default: None

Example:

```
sqlnet.radius_alternate=radius2
```

SQLNET.RADIUS_ALTERNATE_PORT

Purpose: Listening port of the alternate RADIUS server

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

Default: 1645

Example:

```
sqlnet.radius_alternate_port= 1667
```

SQLNET.RADIUS_ALTERNATE_RETRIES

Purpose: Specifies the number of times the Oracle Server should resend messages to the alternate RADIUS server

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

Default: 3

Example:

```
sqlnet.radius_alternate_retries=4
```

SQLNET.RADIUS_AUTHENTICATION

Purpose: Specifies the location of the primary RADIUS server, either by its host name or IP address

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

Default: local host

Example:

```
sqlnet.radius_authenetication= officeacct
```

SQLNET.RADIUS_AUTHENTICATION_INTERFACE

Purpose:	Specifies the class containing the user interface used to interact with the user **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	DefaultRadiusInterface
Example:	`sqlnet.radius_authenetication_interface=`

SQLNET.RADIUS_AUTHENTICATION_PORT

Purpose:	Listening port of the primary RADIUS server **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	1645
Example:	`sqlnet.radius_authenetication_port= 1667`

SQLNET.RADIUS_AUTHENTICATION_RETRIES

Purpose:	Specifies the number of times the Oracle Server should resend messages to the primary RADIUS server **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	3
Example:	`sqlnet.radius_authenetication_retries=4`

SQLNET.RADIUS_AUTHENTICATION_TIMEOUT

Purpose:	Specifies the number of seconds the Oracle Server should wait for a response from the primary RADIUS server **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	5 seconds
Example:	`sqlnet.radius_authenetication_timeout= 10`

SQLNET.RADIUS_CHALLENGE_RESPONSE

Purpose: Turns challenge response on or off

Default: OFF

Values: ON | OFF

Example: `sqlnet.radius_challenge_response=ON`

SQLNET.RADIUS_SECRET

Purpose: Specifies the location of the RADIUS secret key

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

Default: `$ORACLE_HOME/network/security/radius.key` on UNIX and *`ORACLE_HOME`*`/network/security/radius.key` on Windows NT

Example: `sqlnet.radius_secret= oracle/bin/admin/radiuskey`

SQLNET.RADIUS_SEND_ACCOUNTING

Purpose: Turns accounting ON and OFF. If enabled, packets are sent to the active RADIUS server at listening port plus one. The default port is 1646

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

Default: OFF

Values: ON | OFF

Example: `sqlnet.radius_send_accounting=ON`

SSL_CLIENT_AUTHENTICATION

Purpose: Specifies whether or not a client—in addition to the server—is authenticated using SSL

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

Default: TRUE

Values: TRUE | FALSE

Example: `ssl_cipher_suite=(ssl_dh_dss_with_des_cdc_sha)`

SSL_CIPHER_SUITES

Purpose:	Controls what combination of encryption and data integrity is used by SSL **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	None
Values:	For further information about Cipher Suites, see the *Oracle Advanced Security Administrator's Guide.*
Example:	`ssl_cipher_suite=(ssl_rsa_with_rc4_138_md5)`

SSL_VERSION

Purpose:	Forces the version of the SSL connection Clients and servers must use a compatible version. **See Also:** *Oracle Advanced Security Administrator's Guide*
Default:	UNDETERMINED
Values:	UNDETERMINED | 2.0 | 3.0
Example:	`ssl_version=2.0`

TNSPING.TRACE_DIRECTORY

Purpose:	Specifies the destination directory for the TNSPING utility trace file
Default:	`$ORACLE_HOME/network/trace`
Example:	`tnsping.trace_directory=/oracle/traces`

TNSPING.TRACE_LEVEL

Purpose:	Turns tracing on / off at specified level for the TNSPING utility
Default:	OFF
Values	■ OFF - No trace output ■ USER - User-induced error conditions trace information ■ ADMIN - Administration trace information ■ SUPPORT - WorldWide Customer Support trace information
Example:	`tnsping.trace_level=admin`

TRACE_DIRECTORY_CLIENT

Purpose:	Specifies the destination directory for the client trace file
Default:	`$ORACLE_HOME/network/trace`
Example:	`trace_directory_client=/oracle/traces`

TRACE_DIRECTORY_SERVER

Purpose:	Specifies the destination directory for the client trace file
Default:	`$ORACLE_HOME/network/trace`
Example:	`trace_directory_server=/oracle/traces`

TRACE_FILE_CLIENT

Purpose:	Specifies the name of the client trace file
Default:	`sqlnet.trc`
Example:	`trace_file_client=clientsqlnet.trc`

TRACE_FILE_SERVER

Purpose:	Specifies the name of the server trace file
Default:	svr_*pid*.trc
Example:	`trace_file_server=svrsqlnet.trc`

TRACE_FILELEN_CLIENT

Purpose: 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_listener_name parameter.

Example: `trace_filelen_client=100`

TRACE_FILELEN_SERVER

Purpose: Specifies the size of the 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_listener_name parameter.

Example: `trace_filelen_server=100`

TRACE_FILENO_CLIENT

Purpose: Specifies the number of trace files for client 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 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.

Default: None

Example: `trace_fileno_server=3`

TRACE_FILENO_SERVER

Purpose:	Specifies the number of trace files for server 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 `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.
Default:	None
Example:	`trace_fileno_server=3`

TRACE_LEVEL_CLIENT

Purpose:	Turns tracing on or off on the client at a specified level
Default:	OFF
Values	■ OFF - No trace output ■ USER - User trace information ■ ADMIN - Administration trace information ■ SUPPORT - WorldWide Customer Support trace information
Example:	`trace_level_client=user`

TRACE_LEVEL_SERVER

Purpose:	Turns tracing on or off on the server at a specified level
Default:	OFF
Values	■ OFF - No trace output ■ USER - User trace information ■ ADMIN - Administration trace information ■ SUPPORT - WorldWide Customer Support trace information
Example:	`trace_level_server=admin`

TRACE_TIMESTAMP_CLIENT

Purpose:	Adds a timestamp in form of *dd-month-yyyy hh*:*mm*:*ss* to a trace event in the client trace file, which has a default name of `sqlnet.trc`
Default:	OFF
Values	ON or TRUE | OFF or FALSE
Example:	`trace_timestamp_server=true`

TRACE_TIMESTAMP_SERVER

Purpose:	Adds a timestamp in form of *dd-month-yy hh*:*mm*:*ss* to a trace event in the sever trace file, which has a default name of `svr_`*`pid`*`.trc`
Default:	OFF
Values	ON or TRUE | OFF or FALSE
Example:	`trace_timestamp_server=true`

TRACE_UNIQUE_CLIENT

Purpose:	Determines whether or not a unique trace file is created for each client trace file. If the value is set to ON, a process identifier is appended to the name of each trace file, so that several files named `sqlnet`*`pid.`*`trc`. can coexist. If the value is set to OFF, when a new trace file is created for a client, it overwrites the existing file.
Default:	ON
Example:	`trace_unique_client=on`

USE_CMAN

Purpose:

If set to TRUE, instructs the client to use a connect descriptor that contains an address list with a first address for Oracle Connection Manager location and a second address for the listener location. This way, the client is routed to an Oracle Connection Manager.

The following example shows two address lists. While the first address list routes the client to an Oracle Connection Manager, the second address list routes the client directly to a listener.

```
sales=
 (description=
   (load_balance=on)
   (failover=on)
   (address_list=
     (source_route=yes)
     (address=(protocol=tcp)(host=host1)(port=1630))
     (address=(protocol=tcp)(host=host2)(port=1521)))
   (address_list=
     (address=(protocol=tcp)(host=host3)(port=1521)))
   (connect_data=(service_name=sales.us.acme.com)))
```

Without USE_CMAN=TRUE, the client picks one of the address lists at random and fails over to the other address list if the chosen ADDRESS_LIST fails. With USE_CMAN=TRUE, the client always uses the first address list.

If no Oracle Connection Manager addresses are available, connections are routed through any available listener address.

Note: If you are using Oracle Connection Manager with Oracle Names, this option must be set on clients and Oracle Names servers.

Default: FALSE

Values: TRUE | FALSE

Example: `use_cman=true`

USE_DEDICATED_SERVER

Purpose:

If set to ON, (SERVER=DEDICATED) is automatically appended to a connect descriptor's connect data.This way, connections from this client use dedicated servers, even if MTS is configured.

Adds (SERVER=DEDICATED) to the CONNECT_DATA section of the connect descriptors the client uses. This parameter overrides any current value SERVER parameter.

Default:

OFF

Values:

- ON - Spawns dedicated server processes
- OFF - Hands off request to existing server processes

Example:

```
use_dedicated_server=on
```

Local Naming Parameters (tnsnames.ora)

With the local naming method, net service names are added to the `tnsnames.ora` file. The `tnsnames.ora` file is located in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows platforms, or in the directory specified by the TNS_ADMIN environment variable or registry value.

A net service name is an alias mapped to a database network address contained in a connect descriptor. A connect descriptor contains the location of the listener through a protocol address and the service name of the database to which to connect. Clients and servers that are clients of other servers use this net service name when making a connection with an application.

General Syntax

The basic syntax for a `tnsnames.ora` file is shown in Example C–1. DESCRIPTION contains the connect descriptor, ADDRESS contains the listener address, and CONNECT_DATA contains the database service identification information.

Example C–1 General Syntax of tnsnames.ora

```
net_service_name=
 (description=
   (address= (protocol_address_information))
   (connect_data=
     (service_name=service_name)))
```

Multiple Descriptions

A tnsnames.ora file can contain net service names with one or more connect descriptors. Each connect descriptor can contain one or more listener addresses. Example C–2 shows two connect descriptors with multiple addresses. DESCRIPTION_LIST defines a list of connect descriptors.

Note: The Net8 Assistant does not support multiple connect descriptors in a net service name.

Example C–2 Multiple Descriptions in tnsnames.ora

```
net_service_name=
 (description_list=
  (description=
   (address= (protocol_address_information))
   (address= (protocol_address_information))
   (address= (protocol_address_information))
   (connect_data=
     (service_name=service_name)))
  (description=
   (address= (protocol_address_information))
   (address= (protocol_address_information))
   (address= (protocol_address_information))
   (connect_data=
     (service_name=service_name))))
```

Multiple Address Lists

The tnsnames.ora file also supports connect descriptors with multiple lists of addresses, each with its own characteristics. In Example C–3, two address lists are presented. The first address list features client load balancing and no connect-time failover, affecting only those listener ADDRESSes within the ADDRESS_LIST. The second address list features connect-time failover and no client load loading balance, affecting only those listener ADDRESSes within the ADDRESS_LIST. The client first tries either the first or second address at random, then tries addresses three and four sequentially.

Note: The Net8 Assistant only supports the creation of one address list in a connect descriptor.

Example C–3 Multiple Address Lists in tnsnames.ora

```
net_service_name=
 (description=
  (address_list=
   (load_balance=on)
   (failover=off)
   (address= (protocol_address_information))
   (address= (protocol_address_information)))
  (address_list=
   (load_balance=off)
   (failover=on)
   (address= (protocol_address_information))
   (address= (protocol_address_information)))
  (connect_data=
   (service_name=service_name)))
```

> **Note:** Address lists do not have to be embedded in an ADDRESS_LIST if there is only one list, as was the case prior to release 8.1.

Changed Functionality of Client Load Balancing and DESCRIPTION_LISTs

Prior to release 8.1, DESCRIPTION_LISTs were used for client load balancing of multiple listeners. While client load balancing is still on by default for DESCRIPTION_LISTs, client load balancing can also be explicitly specified for an ADDRESS_LIST or associated with a set of ADDRESSes or set of DESCRIPTIONs. Shown in Example C–4 is a comparison of a `tnsnames.ora` file prior to release 8.1 and for release 8.1, using client load balancing in an Oracle Parallel Sever environment:

Example C–4 Client Load Balancing for Release 8.1 and Prior to Release 8.1

Client Load Balancing Prior to Release 8.1

The following example shows a release 8.0 `tnsnames.ora` file with net service names of `op`, `op1`, and `op2`.

`op` allows a client to randomly choose one of the two instances. `op1` and `op2` allow a client to connect to a specific Oracle Parallel Server instance.

```
op=
 (description_list=
  (description=
     (address=
       (protocol=tcp)
       (host=opsnt1)
       (port=1521))
      (connect_data=(sid=op1)))

   (description=
      (address=
        (protocol=tcp)
        (host=opsnt2)
        (port=1521))
       (connect_data=(sid=op2))))
op1=
  (description=
     (address=
       (protocol=tcp)
       (host=opsnt1)
       (port=1521))
     (connect_data=(sid=op1)))
op2=
  (description=
     (address=
       (protocol=tcp)
       (host=opsnt1)
       (port=1521))
     (connect_data=(sid=op2)))
```

Client Load Balancing in Release 8.1

The following example shows a release 8.1 `tnsnames.ora` file with net service names of `op`, `op1` and `op2`.

`op` allows a client to randomly to connect to database service `op.com` and choose one of the two instances using multiple ADDRESSes rather than multiple DESCRIPTIONs. `op1` and `op2` allow a client to connect to a specific Oracle Parallel Server instance.

```
op=
 (description=
     (load_balance=on)
     (address=
       (protocol=tcp)
       (host=opsnt1)
       (port=1521))
     (address=
       (protocol=tcp)
       (host=opsnt2)
       (port=1521))
     (connect_data=
       (service_name=op.com)))
op1=
  (description=
     (address=
       (protocol=tcp)
       (host=opsnt1)
       (port=1521))
     (connect_data=
       (service_name=op)
       (instance_name=op1)))
op2=
  (description=
     (address=
       (protocol=tcp)
       (host=opsnt2)
       (port=1521))
     (connect_data=
       (service_name=op)
       (instance_name=op2)))
```

Examples

When a client goes through an Oracle Connection Manager for a connection to a service, the first address (or set of addresses) is for a client connection to an Oracle Connection Manager and the second address (or set of addresses) is for an Oracle Connection Manager connection to a service using listener.

When the Oracle Connection Manager parameter SOURCE_ROUTE is set to YES, it instructs the client to go through all the addresses listed before connecting.

Shown in Example C–5 is an example with multiple Oracle Connection Manager addresses:

Example C–5 Multiple Oracle Connection Manager Addresses tnsnames.ora

```
sample1=
 (description=
   (source_route=yes)
   (address=(protocol=tcp)(host=host1)(port=1630))    # hop 1
   (address_list=
     (failover=on)
     (load_balance=off)                                # hop 2
     (address=(protocol=tcp)(host=host2a)(port=1630))
     (address=(protocol=tcp)(host=host2b)(port=1630)))
   (address=(protocol=tcp)(host=host3)(port=1521))    # hop 3
   (connect_data=(service_name=sales.us.acme.com)))
```

In Example C–5:

1. The client is instructed to connect to an address of the first Oracle Connection Manager, as indicated by:

   ```
   (address=(protocol=tcp)(host=host1)(port=1630))
   ```

2. The first Oracle Connection Manager then connects to another Oracle Connection Manager by first trying the address of one Oracle Connection Manager. If the first address fails, it tries the second address. This is indicated by:

   ```
   (address_list=
     (failover=on)
     (load_balance=off)
     (address=(protocol=tcp)(host=host2a)(port=1630))
     (address=(protocol=tcp)(host=host2b)(port=1630))
   ```

3. The Oracle Connection Manager then connects to the database service using the following listener address:

   ```
   (address=(protocol=tcp)(host=host3)(port=1521))
   ```

Shown in Example C–6 is an example of client load balancing among two Oracle Connection Managers and two listener addresses:

Example C–6 Client Load Balancing in tnsnames.ora

```
sample2=
 (description=
   (load_balance=on)
   (failover=on)
   (address_list=
     (source_route=yes)
     (address=(protocol=tcp)(host=host1)(port=1630))
     (address=(protocol=tcp)(host=host2)(port=1521)))
  (address_list=
     (source_route=yes)
     (address=(protocol=tcp)(host=host3)(port=1630))
     (address=(protocol=tcp)(host=host4)(port=1521)))
   (connect_data=(service_name=sales.us.acme.com)))
```

In the example above:

1. The client is instructed to pick an ADDRESS_LIST at random and to failover to the other if the chosen ADDRESS_LIST fails. This is indicated by the LOAD_BALANCE and FAILOVER parameters being set to ON.
2. When an ADDRESS_LIST is chosen, the client first connects to the Oracle Connection Manager, using the Oracle Connection Manager address that uses port 1630 indicated for the ADDRESS_LIST.
3. The Oracle Connection Manager than connects to the database service, using the listener address indicated for the ADDRESS_LIST.

DESCRIPTIONS and DESCRIPTION_LISTs

DESCRIPTION

Purpose: Beginning of a connect descriptor, containing the definition of a database listening address and the service name to which to connect. This parameter can be embedded under a DESCRIPTION_LIST.

Example:

```
net_service_name=
(description=
  (address=...)
  (connect_data=(service_name=sales.us.acme.com)))
```

DESCRIPTION_LIST

Purpose: Defines a list of connect descriptors for a particular net service name

Example:

```
net_service_name=
(description_list=
 (description=
  (address=...)
  (connect_data=(service_name=sales.acme.com)))
 (description=
  (address=...)
  (connect_data=(service_name=sales2.us.acme.com))))
```

ADDRESSes and ADDRESS_LISTs

ADDRESS

Purpose:

Defines a listener protocol address. This parameter can be embedded under a ADDRESS_LIST or DESCRIPTION.

See Also: Appendix B for descriptions of the correct parameters to use for each protocol

Example:

```
net_service_name=
(description=
 (address=(protocol=tcp)(host=sales-svr)(port=1521))
  (connect_data=(service_name=sales.us.acme.com))
```

ADDRESS_LIST

Purpose:

Defines a list of addresses. If there is only address list, ADDRESS_LIST is not necessary. This parameter can be embedded under a DESCRIPTION or DESCRIPTION_LIST.

Example:

```
net_service_name=
 (description=
  (address_list=
   (address=(protocol=tcp)(host=sales1-svr)(port=1521))
   (address=(protocol=tcp)(host=sales2-svr)(port=1521)))
  (address_list=
   (failover=on)
   (address=(protocol=tcp)(host=sales3-svr)(port=1521))
   (address=(protocol=tcp)(host=sales4-svr)(port=1521))
  (connect_data=(service_name=sales.us.acme.com)))
```

Optional Parameters for Lists

FAILOVER

Purpose:

When set to ON, instructs Net8, at connect time, to fail over to a different address if the first address fails. When set to OFF, instructs Net8 to try one address.

This parameter can be embedded under a DESCRIPTION_LIST, DESCRIPTION or ADDRESS_LIST.

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 connect-time failover

Default:

ON for DESCRIPTION_LISTs, DESCRIPTIONs, and ADDRESS_LISTs

Values:

ON | OFF | YES | NO | TRUE | FALSE

Example:

```
net_service_name=
 (description=
  (failover=on)
  (address=(protocol=tcp)(host=sales1-svr)(port=1521))
  (address=(protocol=tcp)(host=sales2-svr)(port=1521)))
  (connect_data=(service_name=sales.us.acme.com)))
```

LOAD_BALANCE

Purpose:

When set to ON, instructs Net8 to progress through the list of addresses in a random sequence, balancing the load on the various listener or Oracle Connection Manager addresses. When set to OFF, this parameter instructs Net8 to try the addresses sequentially until one succeeds.

This parameter can be embedded under a DESCRIPTION_LIST, DESCRIPTION or ADDRESS_LIST.

Default:

ON for DESCRIPTION_LISTs

Values:

ON | OFF | YES | NO | TRUE | FALSE

Example:

```
net_service_name=
 (description=
  (load_balance=on)
  (address=(protocol=tcp)(host=sales1-svr)(port=1521))
  (address=(protocol=tcp)(host=sales2-svr)(port=1521)))
  (connect_data=(service_name=sales.us.acme.com))
```

SDU

Purpose:

Instructs Net8 to optimize the transfer rate of data packets being sent across the network with the session data unit (SDU) size you specify. This parameter must be embedded under DESCRIPTION.

Default:

2048

Values:

The SDU size should be set as a multiple of the normal transport frame size. Since the normal size for Ethernet frame size is 1500, the most efficient SDU size over an Ethernet protocol should be a multiple of 1500.

Example:

```
net_service_name=
 (description=
  (sdu=3000)
  (address=...)
  (address=...)
  (connect_data=
    (server_name=sales.com))
```

SOURCE_ROUTE

Purpose:

When set to ON or YES, instructs Net8 to use each address in order until the destination reached.

This parameter is required for Oracle Connection Manager. For Oracle Connection Manager, an initial connection from the client to the Oracle Connection Manager is required, and a second connection from the Oracle Connection Manager to the listener is required.

This parameter can be embedded under a DESCRIPTION_LIST, DESCRIPTION or ADDRESS_LIST.

Default:

OFF

Values:

YES | NO | ON | OFF

Example:

```
net_service_name=
 (description=
  (source_route=on)
  (address=(protocol=tcp)(host=cman-pc)(port=1630))
  (address=(protocol=tcp)(host=sales1-svr)(port=1521)))
  (connect_data=(service_name=sales.us.acme.com))
```

TYPE_OF_SERVICE

Purpose:

Specifies 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 an Oracle Rdb and Oracle database, and you want the application to randomly choose (load balance). This parameter must be embedded under a DESCRIPTION.

Example:

```
net_service_name=
 (description_list=
  (description=
   (address=...)
   (connect_data=
    (service_name=generic)
    (rbd_database=[.mf]mf_personal.rdb)
    (global_name=alpha5))
    (type_of_service=rdb_database))
  (description=
   (address=(protocol=tcp)(host=sales1-svr)(port=1521))
   (connect_data=
    (service_name=sales.us.acme.com))
    (type_of_service=oracle8_database)))
```

CONNECT_DATA Section

CONNECT_DATA

Purpose: Defines the database to which to connect. This parameter must be embedded under a DESCRIPTION

Example:

```
net_service_name=
 (description=
  (address=(protocol=tcp)(host=sales1-svr)(port=1521))
  (address=(protocol=tcp)(host=sales2-svr)(port=1521))
  (connect_data=(service_name=sales.us.acme.com)))
```

CONNECT_DATA permits the following parameters:

- FAILOVER_MODE
- GLOBAL_NAME
- HS
- INSTANCE_NAME
- RDB_DATABASE
- SERVER
- SERVICE_NAME
- SID

FAILOVER_MODE

Purpose:	Instructs Net8 to fail over to a different listener if the first listener fails during runtime. Depending upon the configuration, session or any SELECT statements which were in progress are automatically failed over. This parameter must be embedded under CONNECT_DATA. This type of failover is called Transparent Application Failover (TAF) and should not be confused with the connect-time failover FAILOVER parameter. **See Also:** ▪ "Transparent Application Failover for High Availability" on page 2-21 for conceptual information ▪ "Configuring Transparent Application Failover" on page 8-12 for further complete configuration information

FAILOVER_MODE

Sub-Parameters

FAILOVER_MODE supports the following sub-parameters:

- BACKUP - Specifies the failover node by its net service name. A separate net service name must be created for the failover node.
- TYPE - **(Required)** Specifies the type of failover. Three types of Net8 failover functionality are available by default to Oracle Call Interface (OCI) applications:

  SESSION: Fails over the session; that is, 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: Allows 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, in which no failover functionality is used. This can also be explicitly specified to prevent failover from happening.
- METHOD: Determines how fast failover occurs from the primary node to the backup node:

  BASIC: Establishes connections at failover time. This option requires almost no work on the backup server until failover time.

  PRECONNECT: Pre-establishes connections. This provides faster failover but requires that the backup instance be able to support all connections from every supported instance.
- RETRIES: Specifies the number of times to attempts to connect. If DELAY is specified, RETRIES defaults to 5 retry attempts.
- DELAY: Specifies the amount of time in seconds to wait between connect attempts. If RETRIES is specified, DELAY defaults to 1 second.

Example:

For implementation examples, see "TAF Implementation" on page 8-14.

GLOBAL_NAME

Purpose: Identifies the Oracle Rdb database. This parameter must be embedded under CONNECT_DATA.

Example:

```
net_service_name=
 (description=
   (address=...)
   (address=...)
   (connect_data=
    (service_name=generic)
    (rdb_database= [.mf]mf_personal.rdb)
    (global_name= alpha5)))
```

HS

Purpose: Instructs Net8 to connect to a non-Oracle system. This parameter must be embedded under CONNECT_DATA.

Default: None

Values: OK

Example:

```
net_service_name=
 (description=
   (address=...)
   (address=...)
   (connect_data=
    (sid=sales6)
    (hs=ok)))
```

INSTANCE_NAME

Purpose:

Identifies the database instance to access. The instance name can be obtained from the INSTANCE_NAME parameter in the initialization parameter file.

This parameter must be embedded under CONNECT_DATA.

See Also: "Understanding Connect Descriptors" on page 6-2 for information about setting the instance name string

Example:

```
net_service_name=
 (description=
   (address=...)
   (address=...)
   (connect_data=
    (service_name=sales.com)
    (instance_name=sales1)))
```

RDB_DATABASE

Purpose:

Specifies the file name of an Oracle RDB database. This parameter must be embedded under CONNECT_DATA.

Example:

```
net_service_name=
 (description=
   (address=...)
   (address=...)
   (connect_data=
    (service_name=sales.com)
    (rdb_database= [.mf]mf_personal.rdb)))
```

SERVER

Purpose:

Instructs the listener to connect the client to a specific type of service handle. This parameter must be embedded under CONNECT_DATA.

Values:

DEDICATED | SHARED

DEDICATED connects this client to a dedicated server process.

SHARED connects the client to a shared server.

Note: MTS must be configured in order to connect the client to a shared server.

Note: The USE_DEDICATED_SERVER in the `sqlnet.ora` file overrides this parameter.

SERVER

Example:

```
net_service_name=
 (description=
   (address=...)
   (address=...)
   (connect_data=
    (server_name=sales.com)
    (server=dedicated)))
```

SERVICE_NAME

Purpose:

Identifies the release 8.1 service to access. This parameter must be embedded under CONNECT_DATA.

The SERVICE_NAME is typically set to the global database name, a name comprised of the database name (DB_NAME) and domain name (DB_DOMAIN), entered during installation or database creation.

See Also:

- "Database Identification by Service Name Rather than SID" on page 2-32 for information about the use of SERVICE_NAME
- "Understanding Connect Descriptors" on page 6-2 for information on setting the instance name

Example:

```
net_service_name=
 (description=
   (address=...)
   (address=...)
   (connect_data=
    (service_name=sales.com)))
```

SID

Purpose:	Identifies the database instance for an Oracle8 or Oracle7 database. If the database is Oracle8*i*, use SERVICE_NAME rather than SID. **See Also:** "Database Identification by Service Name Rather than SID" on page 2-32 This parameter must be embedded under CONNECT_DATA.
Example:	*net_service_name*= (description= (address=...) (address=...) (connect_data= **(sid=sales)**))

Listener Parameters (listener.ora)

The following items are available in the network listener configuration file (`listener.ora`).

- Listener Address Section
- SID_LIST_listener_name Static Service Section
- Control Parameters

The `listener.ora` file is located in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows NT, or in the directory specified by the TNS_ADMIN environment variable or registry value.

Listener Address Section

The listener address section of the `listener.ora` file defines the protocol address(es) of the listener.

DESCRIPTION

Purpose: Defines listener protocol address(es)

See Also: "Configuring Connections to Non-Oracle Database Services" on page 8-45 for configuration information

Example:

```
listener_name=
 (description=
  (address=(protocol=ipc)(key=extproc0))
  (address=(protocol=tcp)(host=sales-pc)(port=1521)))
```

ADDRESS

Purpose: Defines a listener protocol address. This parameter can be embedded under a ADDRESS_LIST or DESCRIPTION.

See Also: Appendix B for descriptions of the correct parameters to use for each protocol

Example:

```
listener_name=
 (description=
  (address=(protocol=ipc)(key=extproc0))
  (address=(protocol=tcp)(host=sales-pc)(port=1521)))
```

ADDRESS_LIST

Purpose: Defines lists of listener protocol addresses. If there is only one list of addresses, it is not necessary to use this parameter. This parameter can be embedded under a DESCRIPTION.

Example:

```
listener_name=
 (description=
  (address_list=
   (address=(protocol=ipc)(key=extproc0))
   (address=(protocol=tcp)(host=sales-pc)(port=1521))))
```

SID_LIST_*listener_name* Static Service Section

You can use the SID_LIST section of the `listener.ora` to statically configure service information with the listener.

The SID_LIST is required for backward compatibility with Oracle8 or Oracle7 databases, as well as external procedures and heterogeneous services, and some management tools, including Oracle Enterprise Manager.

Oracle8*i* database information is dynamically registered with the listener during instance startup. Therefore, this information is not required, unless Oracle Enterprise Manager is used to monitor an Oracle8*i* database.

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=
    (global_dbname=global_database_name)
    (sid_name=sid)
    (oracle_home=oracle_home)
    (prespawn_max=99)
    (prespawn_list=
      (prespawn_desc=
        (protocol=tcp)
        (pool_size=10)
        (timeout=2))
      (prespawn_desc=...)))
   (sid_desc=...))
```

SID_LIST contains the following parameters:

GLOBAL_DBNAME

Purpose:

Identifies the global database name of the database, a name comprised of the database name and database domain. You can obtain the GLOBAL_DBNAME value from the SERVICE_NAMES parameter in the initialization parameter file.

This parameter must be embedded under SID_DESC and should match the value of the SERVICE_NAMES parameter.

Example:

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=
    (global_dbname=oracle.com)
    (sid_name=orcl)
    (oracle_home=/usr/oracle)))
```

ORACLE_HOME

Purpose:

Identifies the Oracle home location of the service. This parameter must be embedded under SID_DESC.

Example:

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=
     (sid_name=extproc)
     (oracle_home=/usr/oracle)
     (program=extproc)))
```

PROGRAM

Purpose:

Identifies the service’s executable program name

Example:

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=
     (sid_name=extproc)
     (oracle_home=oracle)
     (program=extproc)))
```

SID_NAME

Purpose:

Identifies the Oracle System Identifier (SID) of the instance. You can obtain the SID value from the INSTANCE_NAME parameter in the initialization parameter file.

This parameter must be embedded under SID_DESC.

Example:

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=
    (global_dbname=oracle.com)
    (sid_name=orcl)
    (oracle_home=/usr/oracle)))
```

SID_LIST

Purpose:

Identifies a list of SID descriptions

Example:

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=...)
   (sid_desc=...))
```

SID_DESC

Purpose:

Provides service information for a specific database instance

This parameter can be embedded under SID_LIST.

Example:

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=...)
   (sid_desc=...))
```

PRESPAWN_MAX

Purpose:

Specifies the maximum number of prespawned dedicated server processes the listener creates. This number must be at least as many as the sum of the pool size for each protocol. Set this value to a large number so that prespawned dedicated server processes are always available for new connections.

This parameter must be embedded under SID_LIST.

Note: Prespawned dedicated servers cannot be configured on Windows NT.

Example:

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=
    (global_dbname=oracle.com)
    (sid_name=sid)
    (oracle_home=/usr/oracle)
    (prespawn_max=99)))
```

PRESPAWN_LIST

Purpose:

Specifies a list of prespawnd dedicated server process protocol descriptions. This parameter must be embedded under SID_ DESC.

Example:

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=
    (global_dbname=oracle.com)
    (sid_name=sid)
    (oracle_home=/usr/oracle)
    (prespawn_max=99)
    (prespawn_list=
      (prespawn_desc=
        (protocol=tcp)
        (pool_size=10)
        (timeout=2))
        (prespawn_desc=...)))
```

PRESPAWN_DESC

Purpose: Defines the protocol on which the listener creates prespawned dedicated server processes, as well as characteristics of the prespawned dedicated server. Each protocol must have its own PRESPAWN_DESC description. This parameter must be embedded under PRESPAWN_LIST.

See Also: "Configuring Prespawned Dedicated Servers" on page 7-16 for descriptions of PROTOCOL, POOL_SIZE, and TIMEOUT.

Example:

```
sid_list_listener_name=
  (sid_list=
   (sid_desc=
    (global_dbname=oracle.com)
    (sid_name=sid)
    (oracle_home=/usr/oracle)
    (prespawn_max=99)
    (prespawn_list=
      (prespawn_desc=
        (protocol=tcp)
        (pool_size=10)
        (timeout=2))
        (prespawn_desc=...))))
```

Control Parameters

The following parameters control the behavior of the listener:

CONNECT_TIMEOUT_*listener_name*

Purpose: Determines the amount of time in seconds the listener waits for a valid connection request after a connection has been started

Default: 10 seconds

Example: `connect_timeout_listener=12`

LOG_DIRECTORY_*listener_name*

Purpose: Specifies the destination directory for the listener log file

Default: `$ORACLE_HOME/network/log` on UNIX and `*ORACLE_HOME*/network/log` on Windows NT

Example: `log_directory_listener=/oracle/network/admin/log`

LOG_FILE_*listener_name*

Purpose: Specifies the name of the log file for the listener

Default: `listener.log`

Example: `log_file_listener=list.log`

LOGGING_*listener_name*

Purpose: Turns logging on or off

Default: ON

Values: ON | OFF

Example: `logging_listener=on`

OSS.SOURCE.MY_WALLET

Purpose: Specifies the location of wallets. Wallets are certificates, keys and trustpoints processed by SSL that allow for secure connections.

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

Default: None

Example:

```
oss.source.my_wallet=
   (source=
      (method=file)
      (method_data=
        (directory=/home/smalladi/oss)))
```

PASSWORDS_*listener_name*

Purpose: Sets an unencrypted password for a listener, so that certain privileges operations, such as SAVE_CONFIG and STOP, used from the LSNRCTL utility are secure. Allows one or more passwords. An encrypted password can set with the LSNRCTL utility's CHANGE_PASSWORD command.

Default: oracle

Example: `passwords_listener=(oracle8)`

SAVE_CONFIG_ON_STOP_*listener_name*

Purpose:	If set to TRUE, any parameters which were modified through the LSNCRCTL SET command replace prior `listener.ora` file settings
Default:	FALSE
Example:	`save_config_on_stop_listener=true`

SSL_CLIENT_AUTHENTICATION

Purpose:	Specifies whether or not a client is authenticated using SSL
Default:	TRUE
Values:	TRUE | FALSE
Usage Notes:	The database authenticates the client. Therefore, this value should be set to FALSE. If this parameter is set to TRUE, the listener attempts to authenticate the client, which can result in a failure. **See Also:** *Oracle Advanced Security Administrator's Guide*
Example:	`ssl_cipher_suite=(ssl_dh_dss_with_des_cdc_sha)`

STARTUP_WAIT_TIME_*listener_name*

Purpose:	Sets the number of seconds that the network listener sleeps before responding to the first LSNRCTL STATUS command
Default:	0 seconds
Example:	`startup_wait_time_listener=5`

TRACE_DIRECTORY_*listener_name*

Purpose:	Specifies the destination directory for the listener trace files
Default:	`$ORACLE_HOME/network/trace` on UNIX and *ORACLE_HOME*`/network/trace` on Windows NT
Example:	`trace_directory_listener=/oracle/network/admin/trace`

TRACE_FILE_*listener_name*

Purpose:	Specifies the name of the trace file for the listener
Default:	`listener.trc`
Example:	`trace_file_listener=list.trc`

TRACE_FILELEN_*listener_name*

Purpose:	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.
Default:	Unlimited
Example:	`trace_filelen_listener=100`

TRACE_FILENO_*listener_name*

Purpose:	Specifies the number of trace files for listener tracing. When this parameter is set along with the TRACE_FILEN_*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.
Default:	1
Example:	`trace_fileno_listener=3`

TRACE_LEVEL_*listener_name*

Purpose:	Turns tracing on / off to a certain specified level
Default:	OFF
Values	- OFF - No trace output - USER - User trace information - ADMIN - Administration trace information - SUPPORT - WorldWide Customer Support trace information
Example:	`trace_level_listener=admin`

TRACE_TIMESTAMP_*listener_name*

Purpose:	Adds a timestamp in form of *dd-month-yyyy hh*:*mm*:*ss* to a trace event in the listener trace file
Default:	OFF
Values	ON or TRUE | OFF or FALSE
Example:	`trace_timestamp_listener=true`

USE_PLUG_AND_PLAY_*listener_name*

Purpose:	Instructs the listener to register database information with an Oracle Names server
Default:	OFF
Values:	ON | OFF
Example:	`use_plug_and_play=on`

Oracle Names Parameters (names.ora)

The following parameters are available in an Oracle Names configuration file (`names.ora`). `names.ora` is located in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows NT.

NAMES.ADDRESSES

Purpose:

Lists the protocol address(es) on which the Oracle Names server listens. Any valid ADDRESS or ADDRESS_LIST is allowed.

See Also:

- "ADDRESSes and ADDRESS_LISTs" on page B-2 for a description of address syntax
- "Protocol Parameters" on page B-3 for descriptions of the correct parameters to use for each protocol

Default:

```
names.addresses=
  (address=(protocol=tcp)(host=oranamesrvr0)(port=1575))
```

NAMES.ADMIN_REGION

Purpose:

Describes the data source for an administrative region. If set, this parameter defines a database as a repository for information. If this parameter is not set, replication of data between Oracle Names server caches.

Default:

NULL

Example:

```
names.admin_region=
 (region=
  (description=
    (address=(protocol=tcp)
    (host=nineva)(port=1575)))
   (connect_data=
    (service_name=sales.us.acme.com)))
   (userid=system)
   (password=manager)
   (refresh=172800)
   (retry=2700)
   (expire-8700)
   (version=34619392))
add descriptions for various attributes
```

NAMES.AUTHORITY_REQUIRED

Purpose:	Determines whether system queries require authoritative answers
Default:	FALSE
Example:	`names.authority_required=true`

NAMES.AUTO_REFRESH_EXPIRE

Purpose:	Specifies the amount of time in seconds the Oracle Names server caches other region's database server addresses which have been obtained through the NAMES.DOMAIN.HINTS parameter. At the end of this interval, the Oracle Names server issues a query to the other region database servers to refresh the address.
Default:	600 seconds
Acceptable Values:	60-1209600 seconds
Example:	`names.auto_refresh_expire=1200000`

NAMES.AUTO_REFRESH_RETRY

Purpose:	Specifies the interval in seconds that the Oracle Names server retries the Oracle Names servers on its domain hint list
Default:	180
Minimum Value:	60
Maximum Value	3600
Example:	`names.auto_refresh_retry=180`

NAMES.CACHE_CHECKPOINT_FILE

Purpose:	Specifies the name and path of the file to which the Oracle Names server writes its checkpoint file. explain
Default:	`$ORACLE_HOME/network/names/ckpcch.ora` on UNIX and `*ORACLE_HOME*\network\names\ckpcch.ora` on Windows NT

Example:	`names.cache_checkpoint_`
	`file=c:\oracle\network\names\cacheck.ora`

NAMES.CACHE_CHECKPOINT_INTERVAL

Purpose:	Indicates the interval in seconds in which an Oracle Names server writes a checkpoint of its stored data to a checkpoint file. Each Oracle Names server can periodically write its cached data to a file to protect against startup failures.
Default:	0 (disabled)
Minimum Value:	10 seconds
Maximum Value:	259200 seconds (3 days)
Example:	`names.cache_checkpoint_interval=24`

NAMES.CONFIG_CHECKPOINT_FILE

Purpose:	Specifies the name and path of the file used to checkpoint Oracle Names server configuration settings
Default:	`$ORACLE_HOME/network/names/ckpcfg.ora` on UNIX and `*ORACLE_HOME*\network\names\ckpcfg.ora` on Windows NT

Example:	`names.config_checkpoint_`
	`file=c:\oracle\network\names\configck.ora`

NAMES.CONNECT_TIMEOUT

Purpose:	Limits the amounts of time in seconds the Oracle Names server waits for the connection from a client to complete
Default:	3 seconds
Minimum Value:	1 second
Maximum Value:	600 seconds
Example:	`names.connect_timeout=8`

NAMES.DEFAULT_FORWARDERS

Purpose: Address list of other Oracle Names servers which are used to forward queries

Example:

```
names.default_forwarders= (forwarder_list=
  (forwarder=(name=rootserv1.com)(address=(protocol=tcp)
      (port=4200)(host=roothost))))
```

NAMES.DEFAULT_FORWARDERS_ONLY

Purpose:	When set to TRUE, the Oracle Names server forwards queries only to those Oracle Names servers listed as default forwarders with the NAMES.DEFAULT_FORWARDERS parameter
Default:	FALSE
Usage Notes:	If set to FALSE, Oracle Names servers listed as default forwarders are called before Oracle Names servers found in the cache.

NAMES.DOMAIN_HINTS

Purpose: Lists the names, addresses and domains of all servers in one or more remote regions. Enables the Oracle Names server to know about other regions' Oracle Names servers. This includes at least the root region for all Oracle Names servers who are not in the root region. Other regions can be provided as optimization requires.

Example:

```
names.domain_hints=
 (hint_desc=(hint_list=
   (hint=(name=rootserv1.com)
     (address=(protocol=tcp)(host=nineva)(port=4200))))
```

NAMES.DOMAINS

Purpose: List of domains in the server’s local region, as well as the default time to live (TTL) for data in those domains.

Example:

```
names.domains=
 (domain_list=
    (domain=(name=)(min_ttl=86400))
    (domain=(name=com)(min_ttl=8640)))
```

NAMES.FORWARDING_AVAILABLE

Purpose: If set to ON, the Oracle Names server forwards client request to remote Oracle Names server. If set to OFF, clients without access to the network outside the local domain are unable to resolve names.

Default ON

Values: ON | OFF

Example: `names.forwarding_available=off`

NAMES.FORWARDING_DESIRED

Purpose: If set to TRUE, the Oracle Names server provides remote Oracle Names server address location information to clients. This way, clients are redirected to the appropriate Oracle Names server. If set to FALSE, the Oracle Names server connects to the remote Oracle Names server on behalf of clients.

Default: TRUE

Values: TRUE | FALSE

Example: `names.forwarding_desired=true`

NAMES.KEEP_DB_OPEN

Purpose:	Specifies whether or not to attempt to keep the TNS connection to the region database open between operations. If set to FALSE, the connection is closed after each load, reload or reload-check.
Default:	TRUE
Values:	TRUE | FALSE
Example:	`names.keep_db_open=FALSE`

NAMES.LOG_DIRECTORY

Purpose:	Specifies the destination directory where the log file for Oracle Names server operational events are written
Default:	`$ORACLE_HOME/network/log` on UNIX and *ORACLE_HOME*`\network\log` on Windows NT
Example:	`names.log_directory=c:\oracle\network\names`

NAMES.LOG_FILE

Purpose:	Indicates the name of the output file to which Oracle Names server operational events are written. The file name extension is always `.log`. Do not enter an extension for this parameter.
Default:	NAMES
Example:	`names.log_file=onames`

NAMES.LOG_STATS_INTERVAL

Purpose:	Specifies the number of seconds between full statistical dumps in the log file
Default:	0 (0=OFF)
Minimum Value:	10 seconds
Maximum Value:	none
Example:	`names.log_stats_interval=12`

NAMES.LOG_UNIQUE

Purpose:	If set to TRUE, the log file name are unique and do not overwrite existing log files
Default:	FALSE
Values:	TRUE | FALSE
Example:	`names.log_unique=true`

NAMES.MAX_OPEN_CONNECTIONS

Purpose:	Specifies the number of connections that the Oracle Names server can have open at any given time. The value is generated as the value 10 or the sum of one connection for listening, five for clients, plus one for each remote domain defined in the local administrative region, whichever is greater. The calculated value is acceptable for most installations.
Default:	Calculated based on entered data.
Minimum Value:	2
Maximum Value:	64
Example:	`names.max_open_connections=52`

NAMES.MAX_REFORWARDS

Purpose:	Specifies the maximum number of times the server attempts to forward an operation
Default:	2
Minimum Value:	1
Maximum Value:	15
Example:	`names.max_reforwards=2`

NAMES.MESSAGE_POOL_START_SIZE

Purpose:	Determines the initial number of messages allocated in the server's message pool which are used for incoming or outgoing forwarded messages
Default:	10
Minimum Value:	3
Maximum Value:	256
Example:	`names.message_pool_start_size=10`

NAMES.NO_MODIFY_REQUESTS

Purpose:	If set to TRUE, the server refuses any operations which modify the data in its region
Default:	FALSE
Values:	TRUE | FALSE
Example:	`names.no_modify_requests=true`

NAMES.NO_REGION_DATABASE

Purpose:	If set to TRUE, the server does not look for a region database
Default:	FALSE
Example:	`names.no_region_database=true`

NAMES.PASSWORD

Purpose:	Sets an encrypted password for an Oracle Names server, so that certain privileged operations, such as STOP, RESTART and RELOAD, used from the NAMESCTL utility are secure If this parameter is set with the Net8 Assistant, the password is encrypted. A clear-text password can be made manually. If the password is clear-text, ensure the NAMESCTL.INTERNAL_ENCRYPT_PASSWORD parameter is set in the `sqlnet.ora` file is set to FALSE.
Default:	None
Example:	`names.password=625926683431aa55`

NAMES.REGION_CHECKPOINT_FILE

Purpose: Specifies the name and path of the file used to checkpoint region data (for example, domain addresses, database addresses of Oracle Names servers in the local region)

Default: `$ORACLE_HOME/network/names/ckpreg.ora` on UNIX and `*ORACLE_HOME*\network\names\ckpreg.ora` on Windows NT

Example: `names.region_checkpoint_file=c:\oracle\network\names\regionck.ora`

NAMES.RESET_STATS_INTERVAL

Purpose: Specifies the number of seconds during which the statistics collected by the Oracle Names servers should accumulate. At the frequency specified, they are reset to zero. The default value of 0 means never reset statistics.

Default: 0 (never reset)

Minimum Value: 10 seconds

Maximum Value: none

Example: `names.reset_stats_interval=15`

NAMES.SAVE_CONFIG_ON_STOP

Purpose: If set to TRUE, the Oracle Names server saves its runtime configuration settings back into the `names.ora` file. Any parameters which were modified through NAMESCTL SET operations replace prior `names.ora` settings.

Default: FALSE

Example: `names.save_config_on_stop=FALSE`

NAMES.SERVER_NAME

Purpose:	Each Oracle Names server is uniquely identified by a name. All configuration references to a particular Oracle Names server use this name.
Default:	ONAMES_*onames_server*
Example:	`names.server_name=namesrv1.us.oracle.com`

NAMES.TRACE_DIRECTORY

Purpose:	Indicates the name of the directory to which trace files from a Oracle Names server trace session are written.
Default:	`$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`/network/trace` on Windows NT
Example:	`names.trace_directory=/oracle/network/admin/trace`

NAMES.TRACE_FILE

Purpose:	Indicates the name of the output file from a Oracle Names server trace session. The file name extension is always `.trc`.
Default:	NAMES
Example:	`names.trace_file=onames`

NAMES.TRACE_FUNC

Purpose:	Enables internal mechanism to control tracing by function name
Default:	FALSE
Example:	`names.trace_func=false`

NAMES.TRACE_LEVEL

Purpose: Indicates the level at which the Oracle Names server is to be traced

Default: OFF

Values

- OFF - No trace output
- USER - User trace information
- ADMIN - Administration trace information
- SUPPORT - WorldWide Customer Support trace information

Example: `names.trace_level=admin`

NAMES.TRACE_UNIQUE

Purpose: Indicates whether each trace file has a unique name, allowing multiple trace files to coexist. If the value is set to ON, a process identifier is appended to the name of each trace file generated.

For example, trace files names `names`*`pid`*`.trc` are created if the default trace file name, `names.trc`, is used.

Default: ON

Values ON | OFF

Example: `names.trace_unique=on`

Oracle Connection Manager Parameters (cman.ora)

The following parameters are available in an Oracle Connection Manager configuration file `cman.ora`. The `cman.ora` file is located in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows NT, or in the directory specified by the TNS_ADMIN environment variable or registry value.

CMAN

Purpose:	Specifies listening addresses for the Oracle Connection Manager gateway process, CMGW.
Default:	`cman=(address=(protocol=tcp)(host=`*`local_host`*`)(port=1630))`

Syntax:	`cman=`
	`  ([address_list= ]`
	`    (address= ...)`
	`   [(address= ...)])`

CMAN_ADMIN

Purpose:	Specifies listening addresses for the Oracle Connection Manager administrative process, CMADMIN (Oracle Connection Manager Administrative Process)
Default:	`cman_admin=(address=(protocol=tcp)(host=`*`anyhost`*`)(port=1830))`

Syntax:	`cman_admin=`
	` ([address_list= ]`
	`    (address= ...)`
	`   [(address= ...)])`

CMAN_PROFILE

Purpose: Sets parameters related to the Oracle Connection Manager

Default:

- ANSWER_TIMEOUT=0
- AUTHENTICATION_LEVEL=0
- LOG_LEVEL=0
- MAX_FREELIST_BUFFERS=0
- MAXIMUM_CONNECT_DATA=1024
- MAXIMUM_RELAYS=128
- RELAY_STATISTICS=NO
- REMOTE_ADMIN=NO
- SHOW_TNS_INFO=NO
- TRACE_DIRECTORY=`$ORACLE_HOME/network/trace` on UNIX and *`ORACLE_HOME`*`\network\trace` on Windows NT
- TRACE_FILELEN=unlimited
- TRACE_FILENO=1
- TRACE_TIMESTAMP=NO
- TRACING=NO
- USE_ASYNC_CALL=YES

CMAN_PROFILE

Values:

- ANSWER_TIMEOUT=[0 to *n*]
- AUTHENTICATION_LEVEL=[0 | 1]
- LOG_LEVEL=[0-4]
- MAXIMUM_CONNECT_DATA=[257-4096]
- MAX_FREELIST_BUFFERS=[0 to 10240]
- MAXIMUM_RELAYS=[1 to 2048]
- RELAY_STATISTICS=[YES | TRUE | ON | 1 | NO | FALSE | OFF | 0]
- REMOTE_ADMIN=[YES | TRUE | ON | 1 | NO | FALSE | OFF | 0]
- SHOW_TNS_INFO=[YES | TRUE | ON | 1 | NO | FALSE | OFF | 0]
- TRACE_DIRECTORY=*directory*
- TRACE_FILELEN=size of file in KBs
- TRACE_FILENO=*number*
- TRACE_TIMESTAMP=[YES | TRUE | ON | 1 | NO | FALSE | OFF | 0]
- TRACING=[YES | TRUE | ON | 1 | NO | FALSE | OFF | 0]
- USE_ASYNC_CALL=[YES | TRUE | ON | 1 | NO | FALSE | OFF | 0]

Example:

```
cman_profile=
   (parameter_list=
      (maximum_relays=512)
      (log_level=1)
      (relay_statistics=yes)
      (remote_admin=yes)
      (show_tns_info=yes)
      (use_async_call=yes)
      (authentication_level=0)
      (tracing=yes)
      (trace_timestamp=yes)
      (trace_filelen=100)
      (trace_fileno=2))
```

CMAN_PROFILE Attributes	Description
ANSWER_TIMEOUT	Determines the number of seconds that Oracle Connection Manager uses to time out the protocol handshake associated with an incoming connection request The range is 0 to *n*.
AUTHENTICATION_LEVEL	Determines the level of security the Oracle Connection Manager can enforce: ■ 1 instructs the Oracle Connection Manager to reject connect requests that are not using Secure Network Services (SNS). SNS is part of the Oracle Advanced Security option. ■ 0 (default) instructs the Oracle Connection Manager not to check for SNS between the client and server
LOG_LEVEL	Determines the level of logging performed by the Oracle Connection Manager: ■ Default is 0, which means no logging is performed ■ Range is 0 to 4: Level 1 - basic reporting Level 2 - RULE_LIST matching lookup reporting Level 3 - relay blocking reporting Level 4 - relay I/O counts reporting The CMGW gateway process creates a log file called `cman_`*`pid`*`.log`, and the CMADMIN administrative process creates a log file called `cmadm_`*`pid`*`.log`. The log files are located in `$ORACLE_HOME/network/log` on UNIX and *`ORACLE_HOME`*`\network\log` on Windows NT.
MAX_FREELIST_BUFFERS	Determines the maximum number of buffers that TNS keeps in its freelist for later re-use instead of returning them to the operating system after a relay gets closed. The range is 0 to 10240.
MAXIMUM_CONNECT_DATA	Limits the connect data string length of the incoming connection requests The range is 257 to 4096.
MAXIMUM_RELAYS	Determines the maximum number of concurrent connections Oracle Connection Manager supports The range is 1 to 2048.

CMAN_PROFILE Attributes	Description
RELAY_STATISTICS	Determines if I/O statistics are recorded YES instructs Oracle Connection Manager to maintain statistics pertaining to relay I/O activities, such as: ▪ Number of IN bytes ▪ Number of OUT bytes ▪ Number of IN packets ▪ Number of OUT packets This information is stored in the `cman_pid.log` file. NO records no I/O statistics.
REMOTE_ADMIN	Determines if remote access to an Oracle Connection Manager is allowed YES allows access from a remote CMCTL session to Oracle Connection Manager. NO allows only access to the local Oracle Connection Manager. This prevents a user running a remote CMCTL session access to Oracle Connection Manager.
SHOW_TNS_INFO	Determines if TNS information are to be recorded YES instructs Oracle Connection Manager to include TNS information in the `cman_pid.log` file. NO does not include TNS events in the log file.
TRACING	Determines whether or not tracing is enabled for the Oracle Connection Manager YES enables tracing for the Oracle Connection Manager. The CMGW gateway process creates a trace file called `cman_pid.trc`, and the CMADMIN administrative process creates a trace file called `cmadm_pid.trc`.
TRACE_DIRECTORY	Specifies the trace directory. You must set this parameter to a non-default location if you plan to use the TRACE_FILELEN or TRACE_TIMESTAMP parameters.
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. **Note:** This parameter requires a non-default trace directory be specified with the TRACE_DIRECTORY parameter.

CMAN_PROFILE Attributes	Description
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 re-used, 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 CMGW gateway trace files would be named `cman1_`*`pid`*`.trc`, `cman2_`*`pid`*`.trc` and `cman3_`*`pid`*`.trc` and the CMADMIN administrative trace files would be named `cmadm1_`*`pid`*`.trc`, `cmadm2_`*`pid`*`.trc` and `cmadm3_`*`pid`*`.trc`. In addition, trace events in the trace files are preceded by the sequence number of the file.
TRACE_TIMESTAMP	Adds a timestamp in form of *dd-month-yyyy hh*:*mm*:*ss* to a trace event in the trace file **Note:** This parameter requires a non-default trace directory be specified with the TRACE_DIRECTORY parameter.
USE_ASYNC_CALL	Determines whether or not Oracle Connection Manager uses asynchronous functions while in the answering or calling phase of establishing a Net8 connection YES instructs the Oracle Connection to use all asynchronous functions. NO instructs the Oracle Connection Manager not to use asynchronous functions. **Note:** Oracle Connection Manager supports out-of-band breaks; it forwards it on to the server.

CMAN_RULES

Purpose:

Sets the filtering rules for the network access control portion of Oracle Connection Manager

If no rules are specified, all connections are accepted. Use of this feature depends on whether you want to use connection concentration and multi-protocol support features with or without filtering.

Syntax:

```
cman_rules=
 (rule_list=
   (rule=
    (src=host)
    (dst=host)
    (srv= service_name|sid)
    (act=accept|reject))
    [(rule= ...)])
```

Values:

- SRC - Source host name or IP address (in dot notation) of session request
- DST - Destination server host name or IP address (in dot notation)
- SRV - Database server SID
- ACT - Accept or reject incoming requests with the previous characteristics.

Usage Notes

The wildcard for host name is the single character ’x’. In the case of an IP address (d.d.d.d), you can wildcard the individual d’s with an ’x’.

If client’s connect string contain both SID and SERVICE_NAME, then both names requested need to be permitted by the rules respectively in order for the client’s access to be allowed.

Example:

```
cman_rules=
 (rule_list=
   (rule=
    (src=client1-pc)
    (dst=sales-pc)
    (srv= sales.acme.com)
    (act=reject))
   (rule=
    (src=src=144.25.23.45)
    (dst=144.25.187.200)
    (srv= db1)
    (act=accept)))
```

Protocol-Specific Parameters (protocol.ora)

Some protocols have platform-specific parameters. These are stored in the `protocol.ora` file. The `protocol.ora` file is located in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows platforms, or in the directory specified by the TNS_ADMIN environment variable or registry value.

Listed next are some of the common parameters. For a complete listing of parameters for your operating system and platform, see your operating-system-specific documentation.

***protocol*.EXCLUDED_NODES**

Purpose:	Specifies which clients are restricted access to the database.
Syntax:	`*protocol*.excluded_nodes= (*hostname* | ip_address, *hostname* | ip_address,...)`
Example:	`tcp.excluded_nodes= (hr.com, 144.25.5.25)`

***protocol*.INVITED_NODES**

Purpose:	Specifies which clients are allowed access to the database. This list takes precedence over the *protocol*.EXCLUDED_NODES parameter if both lists are present.
Syntax:	`*protocol*.invited_nodes= (*hostname* | ip_address, *hostname* | ip_address,...)`
Example:	`tcp.invited_nodes= (sales.com, 144.185.5.73)`

***protocol*.VALIDNODE_CHECKING**

Purpose:	Checks for the *protocol*.INVITED_NODES and *protocol*.EXCLUDED_NODES to determine which clients to allow or deny access.
Default:	NO
Values:	YES | NO
Example:	`tcp.validnode_checking=yes`

TCP.NODELAY	
Purpose:	Specifies no delays in buffer flushing within the TCP/IP protocol stack.
Default:	NO
Values:	YES | NO
Example:	`tcp.nodelay=yes`

Directory Server Access Parameters (ldap.ora)

The following parameters are configured in the `ldap.ora` file by the Net8 Configuration Assistant on the client and server.

Do not modify these parameters or their settings. The `ldap.ora` file is located in `$ORACLE_HOME/network/admin` on UNIX and *`ORACLE_HOME`*`\network\admin` on Windows platforms, or in the directory specified by the TNS_ADMIN environment variable or registry value.

DIRECTORY_SERVERS	
Purpose:	Contains a list of primary and alternate LDAP directory services with their host and port numbers
Values:	host:port[:sslport]
Example:	`directory_servers=dlsun1778:389, raffles:400:636`

DIRECTORY_SERVER_TYPE	
Purpose:	Contains the type of LDAP directory server that is being used
Values:	OID | AD | NDS OID - Oracle Internet Directory AD - Microsoft's Active Directory NDS - Novell Directory Services
Example:	`directory_type=oid`

DEFAULT_ADMIN_CONTEXT	
Purpose:	Specifies the administrative context
Values:	Valid DN
Example:	`networkobj_context="o=OracleSoftware,c=US"`

Configuration File Changes in Release 8.1

This section describes the following:

- Obsolete Parameters for Release 8.1
- Configuration File Syntax Changes
- tnsnames.ora Changes
- cman.ora Changes

Obsolete Parameters for Release 8.1

The following table describes the obsolete parameters for release 8.1:

Parameter	File	Description
AUTOMATIC_IPC	`sqlnet.ora`	This parameter used to force sessions through IPC addresses. Due to performance issues, this parameter has been removed. You should configure an IPC address instead.
NAMES.USE_PLUG_AND_PLAY	`names.ora`	This was used to enable / disable the Dynamic Discovery Option. There are other mechanisms available to discover other Oracle Names server. **See Also:** "Configuring the Oracle Names Method" on page 6-47
NAMES.DOMAIN_CHECKPOINT FILE	`names.ora`	Name of the file used to checkpoint domain data (all the database addresses and other data in the region). You should use the NAMES.REGION_ CHECKPOINT_FILE parameter to create a checkpoint file with this information.
NAMES.TOPOLOGY_ CHECKPOINT_FILE	`names.ora`	Name of the file used to checkpoint topology data (domain addresses of servers in the local region). You should use the NAMES.REGION_ CHECKPOINT_FILE parameter to create a checkpoint file with this information.

Configuration File Syntax Changes

The following table describes the syntax differences between Net8 release 8.0 and Net8 release 8.1:

Affected element	Net8 release 8.0	Net8 release 8.1

Connecting with service name and instance name rather than system identifier (SID) in the `tnsnames.ora` file	The SID of the database had to be specified in the CONNECT_DATA section of the `tnsnames.ora` file, as shown in the following:
	service_name=
	 (description=
	  (address_list=
	   (address=...)
	   (address=...))
	  (connect_data=**(sid=sales)**)
		Because a database can include services that span multiple instances, SID has been replaced by service name and, optionally, instance name.
		The following entry allows a client to connect to the `sales.com` service:
		net_service_name=
		 (description=
		  (address=...)
		  (address=...)
		  (connect_data=
		    **(service_name=sales.com)**))
		The following entry allows a client to connect to the `op1` instance, which is a part of the `op.com` service:
		net_service_name=
		 (description=
		  (address=...)
		  (address=...)
		  (connect_data=
		    (service_name=op.com)
		    **(instance_name=op1)**))
		SERVICE_NAME has a value of the global database name, a name comprised of the database name and domain name, and INSTANCE_NAME has a value of the SID, entered during installation or database creation.
		See Also:
		■ "Database Identification by Service Name Rather than SID" on page 2-32
		■ "Understanding Connect Descriptors" on page 6-2

Affected element	Net8 release 8.0	Net8 release 8.1

Client load balancing with DESCRIPTION_ LIST and ADDRESS_ LIST in the `tnsnames.ora` file	DESCRIPTION_LISTs were used for mapping multiple connect descriptors to net service names and for client load balancing of multiple listeners.
	In order to client load balance, the user had to define separate connect descriptors for each listener.
		While client load balancing is still ON by default for DESCRIPTION_LISTs, load balancing can also be explicitly specified with the new LOAD_BALANCE parameter for an ADDRESS_LIST or associated with a set of ADDRESSes or set DESCRIPTIONs. The following entry associates client load balancing with a list of addresses, affecting all listener ADDRESSes:
		`net_service_name=`
		` (description=`
		`  (load_balance=on)`
		`  (address=...)`
		`  (address=...)`
		`  (connect_data=`
		`    (service_name=sales.com)))`
		Client load balancing works with any version of the client, but requires an Oracle8*i* release 8.1 server.
		See Also:
		- "Configuring Address List Parameters" on page 8-4
		- "Changed Functionality of Client Load Balancing and DESCRIPTION_LISTs" on page C-33.

Affected element	Net8 release 8.0	Net8 release 8.1

Connect-Time Failover with ADDRESS_LIST in the `tnsnames.ora` file	An ADDRESS_LIST implied connect-time failover, where a client connect fails over to a different listener if the first listener fails. Net8 and SQL*Net proceeded through all the listener addresses until one succeeded.	Failover is now supported by default for ADDRESS_LISTs, DESCRIPTION_LISTs, and a set of DESCRIPTIONs. Failover can also be explicitly specified with the new FAILOVER parameter for a set of ADDRESSes. The following entry associates connect-time failover with a list of addresses, affecting all listener ADDRESSes:
		`net_service_name=`
		`(description=`
		`(failover=on)`
		`(address=...)`
		`(address=...)`
		`(connect_data=`
		`(service_name=sales.com)))`
		Connect-time failover works with any version of the client, but requires an Oracle8*i* release 8.1 server.
		See Also:
		■ "Configuring Address List Parameters" on page 8-4
		■ "Local Naming Parameters (tnsnames.ora)" on page C-31

Routing connections through an Oracle Connection Manager with SOURCE_ROUTE=ON in the `tnsnames.ora` file	The SOURCE_ROUTE parameter, which creates a source route of addresses through an Oracle Connection Manager to the destination service, had to be outside of an ADDRESS_LIST:
	`service_name=`
	`(description=`
	`(source_route=on)`
	`(address_list=`
	`(address=...)`
	`(address=...))`
	`(connect_data=`
	`(sid=sales)))`
		SOURCE_ROUTE is no longer outside of an ADDRESS_LIST. Instead, it is now associated with a list of ADDRESSes, as shown in the following:
		`net_service_name=`
		`(description=`
		`(source_route=on)`
		`(address=...)`
		`(address=...)`
		`(connect_data=`
		`(service_name=sales.com)))`
		See Also:
		■ "Configuring Clients for Oracle Connection Manager" on page 8-38
		■ "Local Naming Parameters (tnsnames.ora)" on page C-31

Affected element	Net8 release 8.0	Net8 release 8.1

Nested addresses within ADDRESS_LIST in the `tnsnames.ora` file and `listener.ora` file	If there was only one list of addresses, it was required to use an ADDRESS_LIST, as shown in the following:
	service_name=
	`(description=`
	`  `**`(address_list=`**
	`   (address=...)`
	`   (address=...))`
	`  (connect_data=(sid=sales))`
		Address lists in Net8 do not have to be embedded in an ADDRESS_LIST if there is only one list, as shown in the following:
		net_service_name=
		` (description=`
		`   (address=...)`
		`   (address=...)`
		`    (connect_data=`
		`      (service_name=sales.com)))`
		Nested addresses work with any version of the client, but require an Oracle8*i* database.
		See Also:
		■ "Configuring the Local Naming Method" on page 6-5
		■ "Local Naming Parameters (tnsnames.ora)" on page C-31

SID_LIST_*listener_name* information in the `listener.ora` file	The `listener.ora` file required a definition of the SID of the database served by the listener. The SID of the database had to be specified in the SID_LIST section of the `listener.ora` file. For example:
	`sid_list_listener=(sid_list=`
	`(sid_desc=`
	` (global_dbname=sales.com)`
	` (sid_name=db1)`
	` (oracle_home=/usr/bin/oracle)))`
		Because database instances and multi-threaded server dispatchers are now automatically registered with the listener, it is no longer necessary to explicitly define database information, unless you are:
		■ Connecting to an Oracle8 or Oracle7 database
		■ Using a management tool, such as Oracle Enterprise Manager, that requires this information
		See Also: "Configuring Static Service Information" on page 7-13

Affected element	Net8 release 8.0	Net8 release 8.1

Address for CMADMIN process in `cman.ora` file	An address was only required for the Oracle Connection Manager gateway process, CMGW. The administrative process, CMADMIN used an internal IPC address.
	`cman=(address=(protocol=tcp)(host=cman-pc)(port=1630))`
		A protocol address is now required for both CMADMIN and CMGW.
		This new address is depicted by the new CMAN_ADMIN parameter, as shown in the following:
		`cman=(address=(protocol=tcp)(host=cman-pc)(port=1630))`
		`cman_admin=(address=(protocol=tcp)(host=cman-pc)(port=1830))`
		See Also:
		■ "Oracle Connection Manager Processes" on page 4-5 for conceptual information
		■ "Enabling Connection Concentration" on page 8-28 for configuration information

tnsnames.ora Changes

The following table describes the new parameters in the `tnsnames.ora` file for Net8 release 8.1:

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>FAILOVER</td><td>

When set to ON, instructs Net8 at connect time to fail over to a different listener if the first listener fails. It determines how many addresses are tried, as shown in the following:

```
net_service_name=
 (description=
   (failover=on)
   (address=...)
   (address=...)
   (connect_data=
    (service_name=service_name)))
```

When set to OFF, instructs Net8 to try one address.

By default, this parameter is set to ON for ADDRESS_LISTs, DESCRIPTION_LISTs and a set of DESCRIPTIONs.

Note: Failover during an active session can be configured using Transparent Application Failover parameters, as described in "Configuring Transparent Application Failover" on page 8-12.

See Also:

- "Configuring Address List Parameters" on page 8-4
- "Local Naming Parameters (tnsnames.ora)" on page C-31

</td></tr>
</table>

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>INSTANCE_NAME</td><td>

In addition to using the SERVICE_NAME parameter, the INSTANCE_NAME parameter can also be used to identify the database instance to access. INSTANCE_NAME is only necessary for an Oracle Parallel server database, where the specific instance to which to connect is important to identify. The following example shows descriptions for a service, `op.us.acme.com`, that spans multiple instances, `op1` and `op2`. The end user can connect to the `op.us.acme.com` service through either instance.

```
net_service_name=
  (description=
     (address=...)
     (connect_data=
        (service_name=op.us.acme.com)
        (instance_name=op1)))
net_service_name=
  (description=
     (address=...)
     (connect_data=
        (service_name=op.us.acme.com)
        (instance_name=op2)))
```

The INSTANCE_NAME is typically the SID entered during installation or database creation.

See Also: "Understanding Connect Descriptors" on page 6-2 for information about the instance name string

</td></tr>
<tr><td>LOAD_BALANCE</td><td>

When set to ON, instructs Net8 to progress through the list of listener addresses in a random sequence, balancing the load on the various listeners, as shown in the following:

```
net_service_name=
 (description=
  (load_balance=on)
  (address=...)
  (address=...)
  (connect_data=
    (service_name=sales.us.acme.com)))
```

When set to OFF, instructs Net8 to try the addresses sequentially until one succeeds.

By default, this parameter is set to ON for DESCRIPTION_LISTs.

See Also:

- "Configuring Address List Parameters" on page 8-4
- "Local Naming Parameters (tnsnames.ora)" on page C-31

</td></tr>
</table>

Parameter	Description
SERVICE_NAME	Identifies the Oracle8*i* database service to which to connect, as shown in the following: *net_service_name*= (description= (address=...) (address=...) (connect_data= **(service_name=sales.com)**)) The SERVICE_NAME is typically the global database name, a name comprised of the database name and domain name, entered during installation or database creation. **See Also:** "Understanding Connect Descriptors" on page 6-2 for information about the service name string

cman.ora Changes

The following table describes the new parameters in the `cman.ora` file for Net8 release 8.1:

Parameter	Description
CMAN_ADMIN	Identifies the address for the administrative process CMADMIN process, as shown in the following: **cman_admin=** **(address=** **(protocol=tcp)** **(host= cman-pc)** **(port=1650))** When the Oracle Connection Manager Control Utility, CMCTL, is run, it locates the CMADMIN protocol address to execute its commands. It is not necessary to explicitly specify CMAN_ADMIN if you are using the default address of TCP/IP on port 1830. **See Also:** ■ "Enabling Connection Concentration" on page 8-28 ■ "Oracle Connection Manager Control Utility (CMCTL)" on page A-79

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>REMOTE_ADMIN</td><td>Determines if remote access to an Oracle Connection Manager is allowed.
<pre>
cman_profile=
   (parameter_list=
      (remote_admin=yes))
</pre>
<b>See Also:</b> "Oracle Connection Manager Control Utility (CMCTL)" on page A-79</td></tr>
</table>

D

LDAP Schema for Net8

This appendix describes the **Oracle schema** object classes and attributes defined in the directory for Net8 objects. This appendix does not describe object classes and attributes reserved for future functionality or used by other Oracle products.

This appendix includes the following sections:

- Structural Object Classes
- Attributes

Structural Object Classes

The Oracle schema supports the following structural object classes for Net8 lookups:

- orclDBServer
- orclNetService
- orclNetDescription
- orclNetDescriptionList
- orclNetAddress
- orclNetAddressList

orclDBServer

Description

Defines the attributes for database service entries

Attributes

orclNetDescName

orclVersion

orclNetService

Description

Defines the attributes for net service name entries

Attributes

orclNetDescName

orclVersion

orclNetDescription

Description

Specifies a connect descriptor, containing the listener protocol address and the connect information to the service

Attributes

- orclNetAddrList
- orclNetInstanceName
- orclNetConnParamList
- orclNetFailover
- orclNetLoadBalance
- orclNetProtocolStack
- orclNetSdu
- orclNetServiceName
- orclNetSourceRoute
- orclSid
- orclVersion

orclNetDescriptionList

Description

Specifies a list of connect descriptors

Attributes

- orclNetDescList
- orclVersion

orclNetAddress

Description

Specifies a listener protocol address

Attributes

- orclNetAddressString
- orclNetProtocol
- orclVersion

orclNetAddressList

Description

Specifies a list of addresses

Attributes

- orclNetAddrList
- orclNetFailover
- orclNetLoadBalance
- orclNetSourceRoute
- orclVersion

Attributes

The following table lists the attributes used for the object classes. This list is subject to change.

Attribute	Description
orclNetAddrList	Identifies one or more listener protocol addresses
orclNetAddressString	Defines a listener protocol address
orclNetConnParamList	Placeholder for future connect data parameters
orclNetDescList	Identifies one or more connect descriptors
orclNetDescName	Identifies a connect descriptor or a list of connect descriptors
orclNetFailover	Turns connect-time failover on for an address list
orclNetInstanceName	Specifies the instance name to access
orclNetLoadBalance	Turns client load balancing on for an address list
orclNetProtocol	Identifies the protocol used in the orclAddressString attribute
orclNetProtocolStack	Identifies the presentation and session layer information for connections to Oracle8*i* JServer
orclNetSdu	Specifies the session data unit (SDU) size
orclNetServiceName	Specifies Oracle8*i* database service name in the CONNECT_DATA portion
orclNetSourceRoute	Instructs Net8 to use each address in order until the destination is reached
orclSid	Specifies the Oracle8 or Oracle7 Oracle System Identifier (SID) in the CONNECT_DATA portion of a connection descriptor
orclVersion	Specifies the version of software used to create the entry

Glossary

A

Access Control List (ACL)

The group of access directives that you define. The directives grant levels of access to specific data for specific clients and / or groups of clients.

ACL

See **Access Control List (ACL)**.

access control

See **Net8 access control**.

address

A unique network location used to identify a network object, such as a listener address, Oracle Connection Manager, or Oracle Names server. Addresses have a specific format and must be unique.

administrative context

A directory entry under which an **Oracle Context** resides. An administrative context can be a **directory naming context**. During directory access configuration, clients are configured with an administrative context in the directory configuration file (`ldap.ora`). The administrative context specifies the location of the Oracle Context in the directory whose entries a client expects to access

administrative region

An organizational entity for administering Net8 network components. Each administrative region includes:

- One or more domains
- One or more Oracle Names servers
- One or more databases and listeners

alias

An alternative name for an existing network object. Once an alias is created, it is resolved to the same name as the initial network object. An Oracle Names server stores aliases for any defined net service name, database server or database link.

API

See **Net8 Open**.

ASCII character set

Stands for 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 an 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 may be supported, depending on whether or not **Oracle Advanced Security** is installed:

- CyberSafe
- SecurID
- RADIUS
- Identix

- Kerberos
- **SSL**
- **Windows NT native authentication**

B

Bequeath protocol

If the listener and server exist on the same node, the listener may create or spawn dedicated servers as connect requests are received. Dedicated servers are committed to one network session only and exist for the duration of that network session. The sequence of events that occur when the listener creates a dedicated server process and passes or "bequeaths" control of a network session to it.

C

cache

Memory that stores recently-accessed data to so that subsequent requests to access the same data can be processed quickly.

CDS

Cell Directory Services. See **Cell Directory Services (CDS)**.

Cell Directory Services (CDS)

An external naming method that enables users to transparently use Oracle tools and applications to access Oracle8*i* databases in a Distributed Computing Environment (DCE) environment.

central administration

An Oracle Names network where network management consists of one **administrative region** for the entire network. With central administration, all Oracle Names servers know about one another and about all the services in the network. Contrast with **delegated administration**.

client

A user, software application, or computer that requests the services, data, or processing of another application or computer. In a two-task environment, 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 a client can randomly choose between the listeners for its connect requests if more than one listener services a single database. 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 methods**, client and server **logging** and **tracing**, the domain from which to request names, and other client options for **Oracle Names** and **Oracle Advanced Security**.

client-server architecture

Software architecture based on a separation of processing between two CPUs, one acting as the client in the transaction, requesting and receiving services, and the other as the server that provides services in a transaction.

cman.ora file

A configuration file that specifies protocol addresses for incoming requests and administrative commands, as well as Oracle Connection Manager parameters and **Net8 access control** rules.

configuration files

Files that are used to identify and characterize the components of a network. Configuration is largely a process of naming network components and identifying relationships among those components.

connect data

A portion of the **connect descriptor** that defines the destination database **service name** or **Oracle System Identifier (SID)**. In the example below, SERVICE_NAME defines a database service called `sales.us.acme.com`:

```
(description=
 (address= (protocol=tcp)(host=sales-pc)(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 Oracle release 8.1 database or its **Oracle System Identifier (SID)** for Oracle release 8.0 or version 7 databases. The network route provides, at a minimum, the location of the listener through use of a network address.

connect identifier

A **net service name** or **service name**, that resolves to a **connect descriptor**. Users initiate a connect request by passing a user name and password along with a connect identifier in a connect string for the service to which they wish to connect, for example:

```
CONNECT username/password@connect_identifier
```

connect-time failover

A client connect request is forwarded to a another listener if first listener is not responding. Connect-time failover is enabled by **service registration**, because the listener knows if an instance is up prior to attempting a connection.

connection

An interaction between two processes on a network. Connections are originated by an initiator (client), who requests a connection with a destination (server).

connection concentration

A feature of Oracle Connection Manager that consolidates multiple connection requests from clients to establish a single connection to a server in order to conserve server resources.

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 physical network connections to a multi-threaded server. A feature of Oracle Connection Manager that consolidates multiple connection requests from clients to establish a single connection to a server in order to conserve server resources.

connection request

A notification sent by an initiator and received by a listener that indicates that the initiator wants to start a connection.

connect string

Information the user passes to a service to connect, such as user name, password and **net service name**. For example:

```
CONNECT username/password@net_service_name
```

D

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 link

A network object stored in the local database or in the network definition that identifies a remote database, a communication path to that database, and optionally, a username and password. Once defined, the database link is used to access the remote database.

A public or private database link from one database to another is created on the local database by a DBA or user.

A global database link is created automatically from each database to every other database in a network with Oracle Names. Global database links are stored in the network definition.

See also **global database link**, **private database link**, and **public database link**.

decentralized administration

See **delegated administration**.

dedicated server

A server that requires a dedicated server process for each user process. There is one server process for each client. Net8 sends the address of an existing server process back to the client. The client then resends its connect request to the server address provided. Contrast with **multi-threaded server (MTS)**.

default domain

The domain **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 "." character within it.

delegated administration

A Net8 network where network management is delegated to one or more administrative regions below the **root administrative region**. Also referred to as distributed or decentralized administration. Contrast with **central administration**.

delegated administrative region

A region hierarchically below the **root administrative region**. Any region other than the root administrative region.

destination

The client that is the endpoint of a connection. The initiator of the connection requires some data or service of the destination.

Directory Information Tree (DIT)

A hierarchical tree-like structure in a **directory sever** of the **Distinguished Names (DNs)** of the entries.

directory naming

A **naming method** that resolves a database service or net service name to a connect descriptor, stored in a central directory server.

A directory provides central administration of database services and net service names, reducing the work effort associated with adding or relocating services. Although net service names can be configured to alias a service, the directory can refer to a database service directly without using a net service name. To further aid with configuration ease, the database service is automatically added as an entry to the directory during installation.

directory naming context

A subtree which is of significance within a directory server. It is usually the top of some organizational subtree. Some directories only allow one such context which is fixed; others allow none to many to be configured by the directory administrator.

directory sever

A LDAP-compliant directory server that is accessed with the Lightweight Directory Access Protocol (LDAP). A directory can provide centralized storage and retrieval of database network components, user and corporate policies preferences, user authentication and security information, replacing client-side and server-side localized 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 **multi-threaded server (MTS)**.

Distinguished Name (DN)

Name of entry in a **directory sever**. 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 administration

See **delegated administration**.

distributed processing

Division of front-end and back-end processing to different computers. Net8 supports distributed processing by transparently connecting applications to remote databases.

domain

Any tree or subtree within the **Domain Name System (DNS)** namespace. Domain is most commonly used to refer to a group of computers whose host names share a common suffix, the domain name.

Domain Name System (DNS)

Domain Name System (DNS) is a system for naming computers and network services that is organized into a hierarchy of **domains**. 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 Net8, DNS translate the host name in a TCP/IP address into an IP address.

domestic domains

The set of domains that are managed within a given administrative region. Domains are only domestic relative to a region; they are never domestic in any absolute sense. Also referred to as local domains.

DNS

Domain Name System. See **Domain Name System (DNS)**.

E

error message

A message from a computer program informing you of a potential problem or condition preventing program or command execution.

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. A enterprise role is created by a Database Administrator of a particular database. An enterprise role can be granted or revoked to one or more enterprise users. The information for granting and revoking these roles is also stored in the directory. In addition to creating global roles, a Database Administrator can also grant roles and privileges to a database schema. Everyone sharing a schema gets these local roles and privileges in addition to the enterprise roles.

Each enterprise user has a unique identity across an enterprise. Enterprise users connect to individual databases via a schema. Enterprise users are assigned enterprise roles which determine their access privileges on databases.

enterprise user

Each enterprise user has a unique identity across an enterprise. Enterprise users connect to individual databases via a schema. Enterprise users are assigned enterprise roles which determine their access privileges on databases.

entry

The building block of a directory, it contains information about an object of interest to directory users.

external naming

A **net service name** resolution that uses a supported third-party naming service, such as **NIS** or **NDS**.

external procedures

Functions or procedures written in a third-generation language (3GL) that can be called from PL/SQL code. Only C is supported for external procedures.

F

failover

See **connect-time failover**.

firewall support

See **Net8 access control**.

flat naming model

An Oracle Names infrastructure in which there is only one domain. All names must be unique within that domain.

foreign domains

The set of domains not managed within a given administrative region. Domains are only foreign relative 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 Names Server caching performance.

G

General Inter-ORB Protocol (GIOP)

A **presentation layer** type that is used in Java option connections to provide messaging.

global database link

A **database link** that links each database in a network to all other databases. This enables any user of any database in the network to specify a global object name in a SQL statement or object definition. An **Oracle Names server** can store global database links.

A database link that links each database in a network to all other databases. This enables any user of any database in the network to specify a global object name in a SQL statement or object definition.

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 recommends that your database domain mirror the network domain.

The global database name is the default service name of database, as specified by the SERVICE_NAMES parameter in the **initialization parameter file**.

H

Heterogeneous Services

An integrated component that provides the generic technology for accessing non-Oracle systems from the Oracle 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 **net service name** resolution that enables users in a TCP/IP environment to resolve net service names via 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 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.

I

initialization parameter file

File that contains information to initialize the database and instances.

instance

An instance of the running Oracle8*i* software referencing the database. When a database is started on a database server (regardless of the type of computer), Oracle allocates a memory area called the **System Global Area (SGA)** and starts one or more Oracle processes. This combination of the SGA and the Oracle processes is called an instance. 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 parallel server database.

Inter-ORB Protocol (IIOP)

An implementation of **General Inter-ORB Protocol (GIOP)** over TCP/IP or TCP/IP with SSL for connections to **Oracle8i JServer**.

Interprocess Communication

A protocol used by client applications the reside on the same node as the listener to communicate with the database.

IIOP Clients

Clients that use the **General Inter-ORB Protocol (GIOP)** presentation to access the Java option. These clients include:

- Enterprise JavaBeans (EJBs)
- CORBA Servers
- Java Stored Procedures

IP address

Used to identify a node on a network. Each machine on the network is assigned a unique IP address, which is made up of the network ID, plus 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**.

J

Java Database Connectivity (JDBC) Drivers

Drivers that provide Java programmers access to an Oracle database.

K

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.

L

LDAP

Lightweight Directory Access Protocol. The framework of design conventions supporting industry-standard **directory severs**.

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 the Net8 Configuration Assistant that contains the following directory access information:

- Type of directory
- Location of the directory
- Default administrative context the client or server will use to look up or configure connect identifiers for connections to database services

The `ldap.ora` file resides in `$ORACLE_HOME/network/admin` on UNIX platforms and *`ORACLE_HOME`*`\network\admin` on Windows platforms.

link qualifier

A qualifier appended to a global database link to provide alternate settings for the database user name and password credentials. For example, a link qualifier of `emp` can be appended to a global database link of `sales.us.acme.com`.

listener

A process that resides on the server whose responsibility is to listen for incoming client connection requests and manage the traffic to the server.

Every time a client (or server acting as a client) requests a network session with a server, a listener receives the actual request. If the client's information matches the listener's information, the listener grants a connection to the server.

listener.ora file

A configuration file for the listener that identifies the:

- Listener name
- Protocol addresses that it is accepting connection requests on
- Services it is listening for

The `listener.ora` file typically resides in `$ORACLE_HOME/network/admin` on UNIX platforms and *`ORACLE_HOME`*`\network\admin` on Windows NT.

An Oracle release 8.1 databases does not require identification of the database service because of **service registration**. However, static service configuration is required for an Oracle release 8.1 databases if you plan to use Oracle Enterprise Manager.

Listener Control Utility (LSNRCTL)

A utility included with Net8 to control various listener functions, such as to start, stop, and get 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.

Net8 supports **client load balancing** and **connection load balancing**.

local naming

A **net service name** resolution 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 Net8 is functioning on the server side.

LU6.2 protocol

Logical Unit Type 6.2. A protocol enables an Oracle application on a PC to communicate with an Oracle database. This communication occurs over an System Network Architecture (SNA) network with the Oracle database on a host system that supports Advanced Program-to-Program Communication (APPC) architecture.

M

map

Files used by the **Network Information Service (NIS)** `ypserv` program to handle name requests.

Microsoft's Active Directory

A LDAP-compliant directory service 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 users 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.

MTS

See **multi-threaded server (MTS)**.

multi-protocol support

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.

multi-threaded server (MTS)

A 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 MTS 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 a small pool of server processes can server a large amount of clients. Contrast with **dedicated server**.

multiplexing

Combining multiple sessions for transmission over a single transport connection in order to conserve the operating system's resources. See also **connection concentration**

N

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 (distributed applications). Named Pipes enables client/server conversation over a network using Named Pipes.

naming method

The method used by a client application to resolve a **connect identifier** to a **connect descriptor** when attempting to connect to a database service. Net8 provides five naming methods:

- **local naming**
- **directory naming**
- **host naming**
- **Oracle Names**
- **external naming**

naming model

The set and structure of domains within which names can be allocated.

In a flat naming model, there is a single domain.

In a hierarchical naming model, the highest level is the root domain, and all other domains are hierarchically related.

NDS authentication

An **authentication method** that enables a client single login access to a multi-server and multi-database network under a single NDS directory tree.

NDS

See **Novell Directory Services (NDS).**

net service name

A simple name for a service that resolves to a **connect descriptor**. Users initiate a connect request by passing a user name and password along with a net service name in a connect string for the service to which they wish to connect:

```
CONNECT username/password@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
- Oracle Names server
- External naming service, such as **NDS**, **NIS** or **CDS**

Net8

Oracle's remote data access software that enables both client-server and server-server communications across any network. Net8 supports distributed processing and distributed database capability. Net8 runs over and interconnects many communications protocols. Net8 is backward compatible with SQL*Net version 2.

Net8 access control

A feature of Oracle Connection Manager that sets rules for denying or allowing certain clients to access designated servers. Also known as firewall support.

Net8 Assistant

A graphical user interface tool that combines configuration abilities with component control to provide an integrated environment for configuring and managing Net8. It can be used on either the client or server.

You can use Net8 Assistant to configure the following network components:

- Naming: Define **connect identifiers** and map them to **connect descriptors** to identify the network location and identification of a service. Net8 Assistant supports configuration of connect descriptors in local `tnsnames.ora` files, centralized LDAP-compliant directory service, or an Oracle Names 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.

Net8 Configuration Assistant

A post-installation tool that configure basic network components after installation, including:

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

Net8 Open

The application program interface (API) to Net8 that enables programmers to develop both database and non-database applications that make use of the Net8 network already deployed in their environment. Net8 Open provides applications a single common interface to all industry standard network protocols.

network

A group of two or more computers linked together through hardware and software to allow the sharing of data and/or 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 Net8 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 or a Oracle Names server.

network protocol

See **Oracle Protocol**.

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 service

In an Oracle application network, a service performs tasks for its service consumers; for example, a Names Server provides name resolution services for clients.

Network Session (NS)

NS is a **session layer** types that is used in typical Net8 connection to establish and maintaining the connection between a client application and server.

NI

Network Interface

NIS

Network Information Service. See **Network Information Service (NIS)**.

NN

Network Naming (Oracle Names)

node

A computer or terminal that is part of a network.

Novell Directory Services (NDS)

A distributed computing infrastructure that stores information about network resources on your network. NDS provides easy network access regardless of the user's physical location or the location of needed resources.

Using NDS, you can simplify network administration by using objects to represent any network resource in the network file system.

NPI

See **Network Program Interface (NPI)**.

NR

Network Routing

NS

Network Session. See **Network Session (NS)**.

NT

Network Transport. See **transport**.

O

object class

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 in the same object class share the same attributes.

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 below it and provides a service to the layer above.

Oracle Advanced Security

A product that provides a comprehensive suite of security features to protect enterprise networks and securely extend 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 datatypes, 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 a Connection Manager can then take advantage of the connection concentration, Net8 access control, or multi-protocol support features configured on that Connection Manager.

Oracle Connection Manager Control Utility (CMCTL)

A utility included with Net8 to control various functions, such as to start, stop, and get the status of the Oracle Connection Manager.

Oracle Context

A **RDN** of `cn=OracleContext`, under which all Oracle software relevant information is kept, including entries for **directory naming** and **enterprise user** security.

There may be one or more than one Oracle Context in a directory. An Oracle Context can be associated with a **directory naming context**.

Oracle Database Configuration Assistant

A tool that enables you to create, delete, and modify a database.

Oracle Internet Directory

A directory service implemented as an application on the Oracle release 8.1 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 access protocol, with the high performance, scalability, robustness, and availability of the Oracle8*i* Server.

Oracle Names

An Oracle directory service made up of a system of Oracle Names servers that provide name-to-address resolution for each Net8 service on the network.

Oracle Names Control Utility (NAMESCTL)

A utility included with Oracle Names to control various functions for Oracle Names servers, such as to start, stop, and get the status of an Oracle Names server.

Oracle Names server

A server that uses Oracle Names to store a service's network address along with its simple name so that client applications can request connections with simple names, rather than lengthy addresses.

Oracle Program Interface (OPI)

A networking layer that is 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

A set of rules that defines how data is transported across networks. There are several industry standard transport protocols, such as TCP/IP and SPX.

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 (as it relates to LDAP)

A set of rules that determine what can be stored in **directory sever**. Oracle has its own schema that is applied to many types of Oracle entries, including Net8 entries. The Oracle schema for Net8 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 was used to identify the database. The SID was included in the part of the connect descriptor in a **tnsnames.ora file**, and in the definition of the listener in the **listener.ora file**.

ORACLE_HOME

An alternate name for the top directory in the Oracle directory hierarchy on some directory-based operating systems.

OSI

Open Systems Interconnection. See **Open Systems Interconnection (OSI)**.

Oracle8*i* JServer

Provides support for Java stored procedures, JDBC, SQLJ, Common Object Request Broker Architecture (CORBA), and Enterprise JavaBeans (EJBs) in the Oracle8*i* database.

P

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, etc. Packet information can be useful in troubleshooting.

parameter

Information passed to a program, command, or function, such as a file specification, a keyword, or a constant value.

password

A string (word or phrase) used for data security and known only to its owner. Passwords are entered in conjunction with an operating system login ID, Oracle username, or account name, in order to connect to an operating system or software application (such as the Oracle database). Whereas the username or ID is public, the secret password ensures that only the owner of the username can use that name, or access that data.

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.

preferred Oracle Names server

The Oracle Names server(s) preferred by a client for names resolution; usually the Oracle Names Server that is physically closest to the client, or available over the least expensive network link.

presentation layer

The presentation layer manages the representation of information that application layer entities either communicate or reference in their communication. Example of session layers are **Two-Task Common (TTC)** and **General Inter-ORB Protocol (GIOP)**.

prespawned dedicated server

Prespawned dedicated server processes are prestarted by the listener before any incoming connection requests. They improve the time it takes to establish a connection on servers where **multi-threaded server (MTS)** is not used or not supported on a given machine. They also use allocated memory and system resources better by recycling server processes for use by other connections with shutting down and recreating a server.

private database link

A database link created by one user for his or her exclusive use.

See also **database link**, **global database link**, and **public database link**.

profile

A collection of parameters that specifies preferences for enabling and configuring Net8 features on the client or server. A profile is stored and implemented through the `sqlnet.ora` file.

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**, **Oracle Names server** 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 configure the same addresses.

protocol stack

Designates a particular **presentation layer** and **session layer** combination.

public database link

A database link created by a DBA on a local database which is accessible to all users on that database.

See also **database link**, **global database link**, and **private database link**.

R

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 uniquely address the entry. In the example, `cn=sales,dc=us,dc=acme,dc=com`, `cn=sales` is the RDN.

region

See **administrative region**.

region database

Tables in an Oracle database that store Oracle Names information.

root administrative region

The highest level administrative region in a distributed installation. The root administrative region contains the root domain.

root domain

The highest level domain in a hierarchical naming model.

RPC

Remote Procedure Call

S

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).

Sequenced Packet Exchange (SPX)

service handler

A service handler can be a multi-threaded server **dispatcher**, **dedicated server**, or **prespawned dedicated server**.

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 the following 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 (dispatchers and dedicated servers) available for each instance

  This enables the listener to direct a client's request appropriately.
- Dispatcher, instance, and node load information

  This load information enables the listener to determine which dispatcher can best handle a client connection's request. If all dispatchers are blocked, the listener can spawn a dedicated server for the connection.

service replication

A process that fully replicates a directory system on the network. New services need to register with only one Names Server. The service replication process automatically distributes the new registration to all other active Names Servers on the network.

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**, a name comprised of 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 combined values of the SERVICE_NAMES parameter in the initialization parameter file.

The service name is included in the **connect data** part of the **connect descriptor**.

session layer

The session layer provides the services needed by the **presentation layer** entities that enable them to organize and synchronize their dialogue and manage their data exchange. This layer establishes, manages, and terminates network sessions between the client and server. An example of a session layer is **Network Session (NS)**.

SID

Oracle System Identifier. 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 only valid for version 7.*x* and version 8.0 Oracle databases, as release 8.1 instance information is now automatically registered with the listener. Static configuration is also required for other services, such as external procedures and heterogeneous services, and some management tools, including Oracle Enterprise Manager.

SPX protocol

Sequenced Packet Exchange protocol. A protocol known for high performance and acceptance among many major network management systems, in particular, Novell Advanced NetWare.

SQL*Net

Net8's precursor. An Oracle product that works with the Oracle Server and enables two or more computers that run the Oracle RDBMS or Oracle tools such as SQL*Forms to exchange data through a network. SQL*Net supports distributed processing and distributed database capability. SQL*Net runs over and interconnects many communications protocols.

sqlnet.ora file

A configuration file for the client or server that specifies:

- Client domain to append to unqualified service names or net service names
- Order of naming methods the client should use when resolving a name
- Logging and tracing features to use
- Route of connections
- Preferred Oracle Names servers
- External naming parameters
- Oracle Advanced Security parameters

The `sqlnet.ora` file typically resides in `$ORACLE_HOME/network/admin` on UNIX platforms and *`ORACLE_HOME`*`\network\admin` on Windows platforms.

SSL

Secure Sockets Layer. 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**.

system or topology data

Data used by the Oracle Names server to control regular functioning or communicate with other Oracle Names servers. Includes interchanges, root region’s Oracle Names servers, and any delegated regions’ Oracle Names servers.

T

TCP/IP protocol

Transmission Control Protocol/Internet Protocol. The de facto standard Ethernet protocol used for client/server conversation over a network.

TCP/IP with SSL protocol

TCP/IP with Secure Sockets Layer. A protocol that enables an Oracle application on a client to communicate with remote Oracle databases through TCP/IP and SSL (if the Oracle database is running on a host system that supports network communication using TCP/IP and SSL).

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 Net8 and **Two-Task Common (TTC)**.

TNS

See **Transparent Network Substrate (TNS)**.

tnsnames.ora file

A configuration file that contains **net service name** mapped to **connect descriptors**. 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 Oracle Parallel Server 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 Net8, Oracle Connection Manager and Oracle Names, 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. Net8 uses **Oracle Protocols** for the transport layer.

TTC

Two-Task Common. See **Two-Task Common (TTC)**.

Two-Task Common (TTC)

TTC is a **presentation layer** type that is used in typical Net8 connection to provide character set and data type conversion between different character sets or formats on the client and server.

U

user name

The name by which a user is known to the Oracle Server and to other users. Every username is associated with a password, and both must be entered to connect to an Oracle database.

UPI

User Program Interface

V

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.

W

well-known Oracle Names server

Addresses for one or more Oracle Names servers hardcoded into both the Oracle Names server and its clients. Oracle Names servers then become available at these well known addresses, so that clients do not need to be told, by way of configuration files, where to find the server.

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

Symbols

" symbol
  reserved, in configuration filets, C-4
# symbol
  reserved, in configuration files, C-4
(, 4-23
() symbol
  reserved, in configuration files, C-4
= symbol
  reserved, in configuration files, C-4
| (slash) symbol
  reserved, in configuration files, C-4
’ symbol
  reserved, in configuration files, C-4

Numerics

1024 port, B-5
1521 port, 6-52, 7-7, B-4
  if not using, 7-10
1575 port, 4-24, 6-55, B-4, C-59
1630 port, 4-5, 8-28, 8-32, 8-35, B-4
1646 port, C-23
1830 port, 8-28, 8-32, B-4, C-85
2481 port, B-4
2482 port, B-4
2483 port, B-4

A

ACCEPT_CONNECTIONS command, A-81
Access Control Lists (ACLs)
  described, 2-51
access control to database
  with Oracle Connection Manager, 8-35
  with protocol.ora, C-77
ACT parameter, 8-36, C-76
ACTIVE_DELAYS, A-94
ADDRESS (ADD or ADDR) attribute, 9-4
ADDRESS parameter, B-2, C-38, C-49
ADDRESS_LIST parameter, B-2, C-38, C-49
addresses, configuring, 7-6
administrative context
  described, 2-41
  selecting, 6-19
administrative regions
  defined, 6-47
  delegated, 4-19
  delegated, below root, 4-21
  delegating domains, 6-73
  described, 4-12
  in Oracle Names, 4-12
  root, 4-19
Advanced Program-to-Program Communication (APPC), 4-4
Always Use Dedicated Server option in Net8 Assistant, 8-21
ANSWER_TIMEOUT parameter, C-73
APPC architecture, 4-4
application program interface (API), for non-Oracle data sources, 10-2

applications
building with Net8 OPEN, 10-7
configuring system to use custom, 10-8
samples provided with Net8 OPEN, 10-9
architecture
connection pooling, 2-12
dedicated server, 2-13
multi-threaded server, 2-7
Net8, 3-1
Oracle Connection Manager, 4-5
Oracle Names, 4-10
ARGS parameter, B-3
ARGV0 parameter, B-3
A.SMD record, 7-11, 11-14, A-42
Attempts Per Names Server field in Net8 Assistant, 6-63
attributes
c, 2-40
cn, 2-40
commonName, 2-40
country, 2-40
defined, 2-40
domain component, 2-40
o, 2-41
orclDescList, D-5
orclDescName, D-5
orclLoadBalance, D-5
orclNetAddrList, D-5
orclNetAddrString, D-5
orclNetConnParamList, D-5
orclNetFailover, D-5
orclNetInstanceName, D-5
orclNetProtocol, D-5
orclNetProtocolStack, D-5
orclNetSdu, D-5
orclNetServiceName, D-5
orclNetSourceRoute, D-5
orclSid, D-5
orclVersion, D-5
organization, 2-41
organizationUnitName, 2-41
ou, 2-41
audit trail
described, 12-25
script for using information, 12-26
authentication methods
CyberSAFE, 4-27
Identix, 4-27
Kerberos, 4-27
NDS, 4-25
Novell Directory Services, 4-25
RADIUS, 4-27
SecurID, 4-27
SSL, 4-27
Windows NT native authentication, 4-25
AUTHENTICATION_LEVEL parameter, C-73
AUTOMATIC_IPC parameter, 8-21, C-80

B

BACKUP parameter, 8-13, C-44
bequeath, 2-19, 10-12
described, 4-4
event programming, 10-11
Bequeath protocol
ARGS parameter, B-3
ARGV0 parameter, B-3
PROGRAM parameter, B-3
BEQUEATH_DETACH parameter, 8-23, 10-12, C-5
bequeathed sessions, 2-19
boldface, defined, xxii
brackets, defined, xxii
buffer flushing, described, 5-9
building custom applications with Net8 OPEN, 10-7

C

c attribute, 2-40
CDS external naming, 2-37, 4-25, 6-79, 8-20
Cell Directory Services external naming, 2-37, 4-25, 8-20
configuring, 6-79
centralized naming, 6-4
advantages and disadvantages, 5-4
described, 2-36
establishing a connection with, 6-4
recommended for, 5-4
CHANGE_PASSWORD command, A-4

character set
  for net service name, C-5
  network, for keyword values, C-4
child process termination, 10-12
ckcfg.ora file, 6-49
ckpcch.ora file, 6-49, 11-6
ckpreg.ora file, 6-49
client application layer, described, 3-6
client configuration
  access control with Oracle Connection Manager, 8-37
  connection concentration, 8-30
  connect-time failover, 2-21
  default domains, 8-17
  discovering Oracle Names servers, 6-60
  load balancing requests among listeners, 2-20, 8-4, C-40
  local naming, 6-15
  log files, 12-20
  multi-protocol support, 4-9, 8-33
  multi-threaded server, 9-10
  Oracle Connection Manager address, 8-39
  Oracle Connection Manager features, 8-38
  Oracle Rdb connections, 8-53
  overriding multi-threaded server configuration, 9-10
  preferred Oracle Names servers, 8-26
  shared servers, 9-10
  sqlnet.log file, 12-20
  trace files, 12-35
  Transparent Application Failover (TAF), 2-21
client connections
  methods of, 11-11
  syntax, 11-10
client load balancing, xx, 2-20, C-40
  configuring, 8-4
  using to improve network performance, 5-8
Client Registration ID field in Net8 Assistant, 8-23
client testing
  connectivity, 11-17
  using application, 11-11
  using OS, 11-11
  using special commands, 11-12
  with 3GL, 11-12
  with Net8 Assistant, 11-22
client troubleshooting, 12-4
client32net.cfg file, 6-87
CLOSE_RELAY command, A-81
cmadm_*pid*.log file, 12-19, 12-29
cmadm_*pid*.trc file, 12-34
CMADMIN process, 4-6
CMAN parameter, C-70
CMAN_ADMIN parameter, C-70, C-85, C-88
cman_*pid*.log file, 12-19, 12-29
cman_*pid*.trc file, 12-34
CMAN_PROFILE parameter, C-71
CMAN_RULES parameter, 8-36, C-76
cman.ora file
  configuration parameter reference, C-70 to C-76
  creating, 8-28, 8-32, 8-35
  parameters
    ACT, C-76
    ACT parameter, 8-36
    ANSWER_TIMEOUT, C-73
    AUTHENTICATION_LEVEL, C-73
    CMAN, C-70
    CMAN_ADMIN, C-70, C-85, C-88
    CMAN_PROFILE, C-71
    CMAN_RULES, C-76
    DST, C-76
    DST parameter, 8-36
    LOG_LEVEL, C-73
    MAX_FREELIST_BUFFERS, C-73
    MAXIMUM_CONNECT_DATA, C-73
    MAXIMUM_RELAYS, C-73
    RELAY_STATISTICS, C-74
    REMOTE_ADMIN, C-74, C-89
    SHOW_TNS_INFO, C-74
    SRC, 8-36, C-76
    SRV, 8-36, C-76
    TRACE_DIRECTORY, C-74
    TRACE_FILELEN, C-74
    TRACE_TIMESTAMP, C-75
    TRACING, C-74
    USE_ASYNC_CALL, C-75
  release 8.1, changes in, C-88 to C-89

CMCTL STATS command
  ACTIVE_DELAYS, A-94
  MOST_RELAYS, A-94
  OUT_OF_RELAY, A-94
  TOTAL_REFUSED, A-94
  TOTAL_RELAYS, A-94
CMCTL utility
  command reference, A-81 to A-97
  commands
    ACCEPT_CONNECTIONS, A-81
    CLOSE_RELAY, A-81
    EXIT, A-82
    HELP, A-82
    QUIT, A-83
    SET, A-83
    SET AUTHENTICATION_LEVEL, A-84
    SET DISPLAYMODE, A-85
    SET LOG_LEVEL, A-85
    SET RELAY_STATISTICS, A-86
    SHOW, A-80, A-86
    SHOW ADDRESS, A-87
    SHOW ALL, A-88
    SHOW DISPLAYMODE, A-89
    SHOW PROFILE, A-89
    SHOW RELAY, A-90
    SHOW RULES, A-91
    SHUTDOWN, A-80, A-92
    START, 11-9, A-92
    STATS, A-93
    STATUS, A-95
    STOP, A-96
    STOPNOW, A-97
    VERSION, A-97
  described, 4-30
  function of and syntax format
  SET commands, A-80
  shutdown, preferred mechanism, A-80
  testing the network, 11-4
  using, 11-4
CMGW process, 4-5
cn attribute, 2-40
CNAME.SMD record, 11-14, A-42
comments in configuration files, C-3
communications, stack, 3-2
configuration files
  cman.ora, C-70 to C-76
  listener.ora, C-48 to C-58
  names.ora, C-59 to C-69
  protocol addresses, B-1
  syntax rules, C-2
  tnsnames.ora, C-31 to C-48
Configure Prespawned Servers dialog box, 7-19
configuring
  clients
    access control with Oracle Connection Manager, 8-37
    connection concentration, 8-30
    connect-time failover, 2-21, 8-4
    default domains, 8-17
    local naming, 6-15
    multi-protocol support, 8-33
    naming methods in profile, 8-19
    Oracle Connection Manager, 8-38
    Oracle Names, 6-60
    Transparent Application Failover, 2-21
  connection concentration, 8-28
  connection pooling, 9-9
  connection requests in a profile, 8-21
  directory naming, 6-16 to 6-46
  domains, default, 8-26
  external naming, 6-79 to 6-88
  external procedures, 8-45
  Heterogeneous Services connections, 7-13, 8-49
  host naming, 6-74 to 6-78
  listeners, 7-1 to 7-20
    directory naming, 6-28
    host naming, 6-75
    local naming, 6-15
    Oracle Names, 6-64
  local naming, 6-5 to 6-15
  localized management, 6-4
  multiple addresses, 8-2
  multiple domains in Oracle Names, 6-53, 6-58
  multi-protocol support, 8-32
  naming methods, 6-1 to 6-88

net service names
directory naming, 6-23
external naming, 6-79
local naming, 6-6
Oracle Names, 6-65
Net8 access control, 8-35
Net8 OPEN, 10-7
Oracle Names, 6-47 to 6-74
Oracle Names servers, 6-51
Oracle Rdb database connections, 8-52
overview, 6-2
prespawned dedicated servers, 7-16
protocol addresses, 7-6, B-1
region database for Oracle Names, 6-50, 6-54
replicated data among Oracle Names servers, 6-55
servers
connection concentration, 8-29
local naming, 6-6
Oracle Names, 6-60
system to use custom applications, 10-8
connect descriptors
described, 1-5, C-31
connect identifiers
syntax, 11-10
connect operations, 2-3
connect strings, 11-2
CONNECT_DATA parameter, C-42
CONNECT_TIMEOUT_*listener_name* parameter, C-54
connection concentration, 4-6, 8-28
advantages relative to connection pooling, 5-7
using to improve network performance, 5-7
connection load balancing, xx, 2-26
service registration, 2-24
connection pooling
advantages relative to connection concentration, 5-7
configuring, 9-9
using to improve network performance, 2-12, 5-6
using with multi-threaded server, 9-9
connections, 2-9
adjusting listener queue size to avoid errors, 7-12
adjusting listener queuesize to avoid errors, 5-6
bequeath, 2-19
concurrent, increasing number of, 7-12
connect strings, entering, 11-2, 11-10
dedicated servers, 2-13, 8-21
directory naming, 6-29
external procedures, 8-45
Heterogeneous Services, 8-49
host naming, 6-78
local naming, 6-15
methods of, 11-11
multi-threaded server, 2-7, 9-2
NDS external naming, 6-83
Network Information Service external naming, 6-79
NIS external naming, 6-79
Novell Directory Services external naming, 6-83
Oracle Connection Manager, 1-3, 8-21
Oracle Names, 6-74
Oracle Rdb databases, 8-52
Oracle8*i* JServer, 3-16
routing, 1-3
startup of components, 11-4
to services, 1-5
CONNECTIONS (CON or CONN) attribute, 9-5
connect-time failover, xx, C-39
configuring, 8-4
GLOBAL_DBNAME parameter in listener.ora file, 7-13, 8-4, C-39
service registration, 2-23
control utilities
CMCTL, 4-30, A-81 to A-97
described, A-1
LSNRCTL, 4-30, A-4 to A-25
NAMESCTL, 4-31, A-28 to A-79
curly brackets, defined, xxii
CyberSAFE authentication method
See Oracle Advanced Security

D

DAEMON.TRACE_DIRECTORY parameter, C-5
DAEMON.TRACE_LEVEL parameter, C-6
DAEMON.TRACE_MASK parameter, C-6
data transfer, maximizing, 5-8

database links, 4-15, 6-68
- global database link, 4-15
- private database, 4-15
- public database, 4-15

Database Services tab in Net8 Assistant, 7-14
databases
- distributed, 2-2
- repository for Oracle Names, 4-13
- starting, 11-8

DBAccess Interface, 3-13
DBSNMP_START command, A-5
DBSNMP_STATUS command, A-5
DBSNMP_STOP command, A-6
dc attribute, 2-40
DCE Integration, described, 4-28
dead connection detection
- described, 2-6
- limitations, 8-23
- SQLNET.EXPIRE_TIME parameter, C-18

dedicated servers
- bequeathed sessions, 2-19
- described, 2-13
- multi-threaded servers contrasted with, 9-2
- prespawned dedicated servers, 2-16
- routing connections, 8-21

Default Domain field in Net8 Assistant, 6-63
default network domains, 8-17
- in Oracle Names, 4-17

DEFAULT_ADMIN_CONTEXT parameter, C-79
DELAY parameter, 8-13, C-44
DELEGATE_DOMAIN command, 6-73, A-28
delegated administrative regions
- below root, 4-21
- diagram, 4-20
- domains to, 6-73
- in Oracle Names, 4-19

DESCRIPTION parameter, 6-5, C-37, C-49
DESCRIPTION_LIST parameter, C-37
diagnosing
- client problems, 12-4
- errors, *See* troubleshooting

direct hand-off, xx
- described, 2-9
- events in listener.log, 12-28

directory access configuration, 6-17
Directory Information Tree (DIT), defined, 2-40
directory naming
- administrative context, 2-41
- advantages and disadvantages, 5-3
- architecture, 2-40
- authentication methods
  - native, 2-51
  - simple, 2-51
  - SSL, 2-51
  - strong, 2-51
- configuring, 6-16 to 6-46
- connecting to database services, 2-44
- connecting with, 6-29
- connections
  - using an entry's absolute name, 2-46
  - using an entry's relative name, 2-45
- default configuration, 6-17
- described, 2-35
- directory access configuration, 6-17
- establishing a connection with, 6-29
- exporting
  - net service names from a tnsnames.ora file, 6-32
  - objects from Oracle Names, 6-35 to 6-46, A-30
- JDBC clients, 3-15
- ldapmodify tool, 6-31
- ldap.ora file
  - creating, 6-18
  - described, 2-44
  - parameters, C-78
- listener configuration for, 6-28
- object classes, 2-54
- Oracle Context, 2-41
- Oracle Database Configuration Assistant, 6-22
- Oracle schema, 2-54
- OracleDBCreators group, 2-43, 2-44, 2-53, 6-22
- OracleNetAdmins group, 2-44, 2-52, 6-22, 6-23, 6-31
- OracleSecurityAdmins group, 2-53, 6-22
- overview, 2-38
- recommended for, 5-3
- security, 2-51

Directory Server Migration Wizard, 6-32

directory servers
- Directory Information Tree (DIT), 2-40
- Distinguished Name (DN), 2-40
- ldapmodify tool, 6-31
- Microsoft's Active Directory, 2-39
- Novell Directory Services external naming, 2-39
- Oracle Internet Directory, 2-39
- performance, 2-50
- security, 2-51

DIRECTORY_SERVER_TYPE parameter, C-78
DIRECTORY_SERVERS parameter, C-78
Disable Out-of-Band Break option in Net8 Assistant, 8-24
DISABLE_OOB parameter, 8-24, C-6
disconnecting
- abnormal termination, 2-6
- additional connect request, 2-6
- from servers, 2-6
- user initiated, 2-6

discovering Oracle Names servers
- described, 4-24
- procedure, 6-59

dispatchers, 2-7
- described, 2-7
- number to start, 9-7
- setting the number of, 9-7

DISPATCHERS (DIS or DISP) attribute, 9-5
Distinguished Name (DN), defined, 2-40
distributed databases, 2-2
distributed processing, 2-2
DLCR.RDBMS.OMD record, 11-14, A-42
DL.RDBMS.OMD record, 11-14, A-42
DNS, described, Glossary-9
Domain Name Services, described, Glossary-9
DOMAIN_HINT command, A-29
domains
- default network, 4-17
- delegating, 6-73
- described, 4-12, 6-47
- multiple hierarchically, 6-53, 6-58
- multiple hierarchically related, 4-18
- required by root administrative regions, 4-19
- with a single domain naming structure, diagram, 4-16

DST parameter, 8-36, C-76
DUMP_LDAP command, A-30
DUMP_TNSNAMES command, A-33
Dynamic Discovery Option, 4-22

E

ellipsis, defined, xxii
enterprise user security, 2-41
entry, defined, 2-40
errors
- 20002-20021 for Net8 OPEN, 10-9
- contacting Oracle Customer Support, 12-59
- example of trace data, 12-45
- in trace file, 12-45
- NNL-00018, 6-54
- NNL-00024, 6-54
- ORA-00508, 12-10
- ORA-1017, 12-3
- ORA-1034, 12-3
- ORA-12154, 12-8
- ORA-12198, 12-9
- ORA-12203, 12-9, 12-10, 12-14
  - sample error stack, 12-18
- ORA-12224, 12-12
- ORA-12533, 12-12
- ORA-12535, 12-14
- ORA-12538, 12-10
- ORA-12545, 12-12
- ORA-12547, 12-14
- ORA-12560, 12-13
- ORA-3113, 12-13
- ORA-3121, 12-13
- resolving, 12-1 to 12-59
- using log file to track, 12-24

error stack
- described, 12-17
- entries in log files, 12-24
- sample, 12-18
- typical layers in, 12-17

establishing a session
- external naming, 2-38
- through shared servers, 2-8, 2-14
- using centralized naming, 2-36
- using directory naming, 2-35
- using host naming, 2-37
- using local naming, 2-35
- using Oracle Names, 2-35

exception operations, described, 2-7

EXIT command
- of CMCTL, A-82
- of LSNRCTL, A-6
- of NAMESCTL, A-34

external naming
- advantages and disadvantages, 5-4
- CDS, 2-37, 4-25, 6-79, 8-20
- Cell Directory Services, 2-37, 4-25, 6-79, 8-20
- configuring, 6-79 to 6-88
- described, 2-37
- establishing a connection with, 2-38
- features, 4-24
- NDS, 2-37, 4-24, 6-79, 8-20
- Network Information Service, 2-37, 4-24, 6-79, 8-20
- NIS, 2-37, 4-24, 6-79, 8-20
- Novell Directory Services, 2-37, 4-24, 6-79, 8-20
- recommended for, 5-4

external procedures
- configuring, 8-45
- described, 8-45
- extproc program, 8-45

extproc program, 8-45

F

failover
- connect-time, 2-21, 8-4, C-39
- Transparent Application Failover, 2-21, 8-12, C-43
- with Oracle Enterprise Manager, 2-21, 5-8

FAILOVER parameter, 2-21, 2-30, 8-4, C-39, C-86

FAILOVER parameters, 8-4

FAILOVER_MODE parameter, 8-12, 8-13, C-43

features, new
- client load balancing, xx
- cman.ora file, C-88 to C-89
- connection load balancing, xx
- connect-time failover, xx
- direct hand-off, xx
- directory naming, xx
- listener.ora file, C-84
- Net8 Configuration Assistant, xxi
- Oracle Connection Manager, xxi
- service naming, xx
- service registration, xx
- SERVICE_NAME parameter, xxi
- TCP/IP with SSL, xxi
- tnsnames.ora file, C-86 to C-88

finger utility, 10-9

firewall support, 4-8
- Net8 Firewall Proxy, 4-8
- Oracle Connection Manager, 4-8

FLUSH command, A-34

FLUSH_NAME command, A-35

G

General Inter-Orb Protocol
- described, 3-16
- presentation, 3-16

GIOP
- described, 3-16
- presentation, 3-16

global database link qualifiers, 6-68

global database links, 4-15, 6-68
- adding link qualifiers, 6-70
- registering with Oracle Names, 6-69

global database name
- configuring on the listener, 7-13, C-51
- defined, 6-3
- described, 7-13, C-51

Global Database Name field in Net8 Assistant, 7-13, 8-9, 8-52

GLOBAL_DBNAME parameter, 7-13, C-51

GLOBAL_NAME parameter, 8-9, 8-52, C-45

H

HELP command
  of CMCTL, A-82
  of LSNRCTL, A-7
  of NAMESCTL, A-36
Heterogeneous Services
  configuring connections to, 8-49
  described, 8-49
hierarchical naming model
  described, 4-17
  diagram, 4-17
host naming
  advantages and disadvantages, 5-4
  configuring, 6-74 to 6-78
  connecting with, 6-78
  described, 2-36
  establishing a connection with, 2-37, 6-78
  listener configuration for, 6-75
  recommended for, 5-4
  requirements, 6-74
HOST parameter, B-4
hosts file, 4-22
HS parameter, 8-9, C-45

I

Identix authentication method
  See Oracle Advanced Security
Initial Preallocated Requests field in Net8 Assistant, 6-63
initialization parameter file
  INSTANCE_NAME parameter, 2-32
  LOCAL_LISTENER parameter, 7-7, 7-10
  MTS_DISPATCHERS parameter, 9-4
  SERVICE_NAMES parameter, 2-32, 9-6
installation
  default configuration
    directory naming, 6-17
    listeners, 7-3
    local naming, 6-6
  Oracle Names, 6-48
Instance Name field in Net8 Assistant, 8-9
INSTANCE_NAME parameter, 2-32, 6-3, 8-9, C-46, C-87
IPC protocol
  KEY parameter, B-3
IPC, parameters for addresses, B-3
italic, defined, xxii

J

Java DataBase Connectivity (JDBC) Drivers
  See JDBC
Java DBAccess Interface layer, described, 3-13
Java NS, described, 3-14
Java Sockets, described, 3-14
JavaNet, described, 3-14
JavaTTC, described, 3-13
JDBC
  directory naming, 3-15
  drivers, 3-11
  Java DBAccess Interface, 3-13
  Java NS, 3-14
  Java Sockets, 3-14
  JavaNet, 3-14
  OCI, 3-11
  TCP/IP Protocol, 3-14
  Thin, 3-11

K

Kerberos authentication method
  See Oracle Advanced Security
KEY parameter, B-3
keyword syntax rules, for configuration files, C-3
keyword values, network character set for, C-4

L

LDAP schema
  attributes, D-1 to D-5
  object classes, D-1 to D-5
ldapmodify tool, 6-31
ldap.ora file
  creating, 2-44, 6-18
  DEFAULT_ADMIN_CONTEXT parameter, C-78
  DIRECTORY_SERVERS parameter, C-78
  DIRECTORY_TYPE parameter, C-78

link layer, described, 3-4
LIST_DELEGATED command, A-37
LIST_DOMAINS command, A-37
LIST_OBJECTS command, A-38
LISTENER (LIS or LIST) attribute, 9-5
listener addresses, B-1
Listener Control Utility (LSNRCTL)
  See LSNRCTL utility
listener.log file, 12-19
listener.ora file
  configuration parameter reference, C-48 to C-58
  described, 1-5
  parameters
    ADDRESS, C-49
    ADDRESS_LIST, C-49
    CONNECT_TIMEOUT_*listener_name*, C-54
    DESCRIPTION, C-49
    GLOBAL_DBNAME, 7-13, C-51
    LOG_DIRECTORY_*listener_name*, C-54
    LOG_FILE_*listener_name*, C-55
    LOGGING_*listener_name*, C-55
    ORACLE_HOME, 7-13, 8-47, 8-49, C-51
    OSS.MY.WALLET, C-55
    PASSWORDS_*listener_name*, C-55
    POOL_SIZE, 7-17
    PRESPAWN_DESC, C-54
    PRESPAWN_LIST, C-53
    PRESPAWN_MAX, 7-16, C-53
    PROGRAM, 8-47, 8-49, C-51
    PROTOCOL, 7-16
    SAVE_CONFIG_ON_STOP_*listener_name*, C-56
    SID_DESC, C-52
    SID_LIST_*listener_name*, C-50, C-52
    SID_NAME, 7-13, 8-47, 8-49, C-52
    SQLNET.CLIENT_AUTHENTICATION, C-56
    STARTUP_WAITTIME_*listener_name*, C-56
    TIMEOUT, 7-17
    TRACE_DIRECTORY_*listener_name*, C-56
    TRACE_FILE_*listener_name*, C-57
    TRACE_FILEN_*listener_name*, C-57
    TRACE_FILENO_*listener_name*, C-57
    TRACE_LEVEL_*listener_name*, C-58
    TRACE_TIMESTAMP_*listener_name*, C-58
    USE_PLUG_AND_PLAY_*listener_name*, C-58
  release 8.1, changes in, C-84
    SID_LIST_*listener_name* not required, C-84
listeners, 7-13, 8-49
  adjusting queuesize for, 7-12
  client load balancing, xx, 2-20
  configuration file, 1-5, 6-2
  configuring, 7-1 to 7-20
    address for Oracle Connection Manager, 8-42
    address list, 8-2
    directory naming method, 6-28
    external procedures connections, 7-13, 8-45
    global database name, 7-13
    Heterogeneous Services connections, 8-49
    host naming method, 6-75
    local naming method, 6-15
    multihomed hosts, 7-7
    non-default address, 7-10
    Oracle Enterprise Manager, 7-13
    Oracle home directory, 7-13
    Oracle Names, 6-64
    Oracle System Identifier, 7-13
    Oracle8*i* JServer, 7-8
    prespawned dedicated servers, 7-16
    protocol addresses, 7-6, B-1
    SID, 7-13
  connection load balancing, xx, 2-26
  connect-time failover, 2-21
  default address, 2-24
  default configuration, 7-3
  direct hand-off, xx, 2-9
  handling concurrent connections, 7-12
  increasing queuesize, 7-12
  log files, 12-21, 12-25
    audit trail, 12-25
    configuring, 12-22
    direct hand-off event information, 12-28
    service registration event information, 12-26
  multihomed hosts, 7-7
  multiple addresses, 8-2
  overview, 2-5
  prespawned dedicated servers, 2-17
  queuesize, 5-6

redirect connections, 2-9
service naming, xx
service registration, xx, 2-22
starting
from the command line, 11-7
from the Control Panel, 11-7
testing configuration, 11-15
trace files, 12-38
Transparent Application Failover, 2-21
listener.trc file, 12-34
LLU parameter, B-3
LLU_NAME parameter, B-3
load balancing
client, 2-20, C-40
connection, 2-26
LOAD_BALANCE parameter, 8-4, C-40, C-82, C-87
LOAD_TNSNAMES command, 6-67, A-39
local naming
advantages and disadvantages, 5-3
client configuration for, 6-15
configuring, 6-5 to 6-15
connecting with, 6-15
default configuration, 6-6
described, 2-35
establishing a connection with, 2-35, 6-15
listener configuration for, 6-15
recommended for, 5-3
server configuration for, 6-6
LOCAL_LISTENER parameter, 7-7, 7-10, 9-5
LOCAL_LU parameter, B-3
LOCAL_LU_NAME parameter, B-3
log files
cmadm_*pid*log, 12-22
cmadm*pid*.log, 12-22
cman_*pid*.log, 12-22
cman*pid*.log, 12-22
default names for, 12-19
for listener, 12-25
listener.log, 12-21
names.log, 12-21
sqlnet.log for clients and servers, 12-20
using to track errors, 12-24
LOG_DIRECTORY_CLIENT parameter, 12-20, C-7
LOG_DIRECTORY_*listener_name* parameter, 12-21, C-54
LOG_DIRECTORY_SERVER parameter, 12-20, C-7
LOG_FILE_CLIENT parameter, 12-20, C-7
LOG_FILE_*listener_name* parameter, 12-21, C-55
LOG_FILE_SERVER parameter, 12-20, C-7
LOG_LEVEL parameter, 12-22, C-73
LOG_STATS command, A-40
LOGGING_*listener_name* parameter, C-55
loopback test, 12-4
lowercase courier, defined, xxii
LSNRCTL utility, A-3
command reference, A-4 to A-25
commands
CHANGE_PASSWORD, A-4
DBSNMP_START, A-5
DBSNMP_STATUS, A-5
DBSNMP_STOP, A-6
EXIT, A-6
HELP, A-7
QUIT, A-8
RELOAD, A-8
SAVE_CONFIG, A-9
SERVICES, A-9
SET, A-11
SET CONNECT_TIMEOUT, A-12
SET CURRENT_LISTENER, A-12
SET DISPLAYMODE, A-13
SET LOG_DIRECTORY, A-13
SET LOG_FILE, A-14
SET LOG_STATUS, A-14
SET PASSWORD, 11-7, A-15
SET STARTUP_WAITTIME, A-16
SET TRC_DIRECTORY, A-17
SET TRC_FILE, A-17
SET TRC_LEVEL, A-18
SET USE_PLUGANDPLAY, A-18
SHOW, A-19
SHOW CONNECT_TIMEOUT, A-19
SHOW CURRENT_LISTENER, A-19
SHOW DISPLAYMODE, A-19
SHOW LOG_DIRECTORY, A-19
SHOW LOG_FILE, A-19
SHOW LOG_STATUS, A-19
SHOW PASSWORD, A-19

SHOW RAW_MODE, A-19
SHOW SAVE_CONFIG_ON_STOP, A-19
SHOW STARTUP_WAITTIME, A-19
SHOW TRC_DIRECTORY, A-19
SHOW TRC_FILE, A-19
SHOW TRC_LEVEL, A-19
SHOW USE_PLUGANDPLAY, A-19
SPAWN, A-20
START, 11-7, A-20
STATUS, 11-7, A-22
STOP, 11-7, A-24
TRACE, A-24
VERSION, A-25
described, 4-30
function of and syntax format
SET commands, A-3
SHOW command, A-3
starting a listener, 7-5
stopping a listener, 7-5
testing the network, 11-3
LU_NAME parameter, B-3
LU6.2 parameter
LU_NAME, B-3
MDN parameter, B-3
LU6.2 protocol, B-3
described, 4-4
LLU parameter, B-3
LLU_NAME, B-3
LOCAL_LU parameter, B-3
LOCAL_LU_NAME, B-3
MODE parameter, B-3
parameters for addresses, B-3
PARTNER_LU_LOCAL_ALIAS parameter, B-3
PARTNER_LU_NAME parameter, B-3
PLU parameter, B-3
PLU_LA parameter, B-3
supporting networks, 4-4
supporting topologies, 4-4
TP_NAME protocol, B-3
TPN parameter, B-3

M

MAX_FREELIST_BUFFERS parameter, C-73
maximizing data transfer, by adjusting SDU size, 5-8
Maximum Open Connections field in Net8 Assistant, 6-63
Maximum Prespawned Servers field in Net8 Assistant, 7-16
Maximum Wait Each Attempt field in Net8 Assistant, 6-63
MAXIMUM_CONNECT_DATA parameter, C-73
MAXIMUM_RELAYS parameter, C-73
MDN parameter, B-3
METHOD parameter, 8-13, C-44
Microsoft's Active Directory, 2-39
MODE parameter, B-3
MOST_RELAYS, A-94
MTS
See multi-threaded server
MTS_DISPATCHERS parameter, 8-29, 9-4, 9-9
ADDRESS attribute, 9-4
CONNECTIONS attribute, 9-5
DISPATCHERS attribute, 9-5
LISTENER attribute, 9-5
MULTIPLEX attribute, 8-29, 9-6
POOL attribute, 9-6
PROTOCOL attribute, 8-29, 9-4
SERVICE attribute, 9-6
SESSIONS attribute, 9-7
setting initially, 9-7
TICKS attribute, 9-7
multihomed hosts, 7-7
multiple addresses, 8-2, 8-4
configuring client load balancing, 8-7
configuring connect-time failover, 8-7
configuring source routing to Oracle Connection Manager, 8-38
MULTIPLEX (MUL or MULT) attribute, 8-29, 9-6
multiplexing
used in connection concentration, 5-7
multi-protocol support, 4-8
configuring, 8-32

multi-threaded server
- allocating resources, 9-9
- configuring dispatchers, 9-7
- connection load balancing, 2-26
- connection pooling, 2-12
- dedicated server contrasted with, 9-2
- described, 2-7
- dispatchers, 2-7
- initialization parameters, 9-4
- routing session requests to, 2-8, 2-14
- starting, 9-4
- tnsnames.ora file parameters, 2-9
- using with connection pooling, 9-9
- using with Oracle Connection Manager, 4-6, 8-28
- virtual circuits, 2-7

N

Named Pipes protocol
- described, 4-3
- parameters for addresses, B-3
- PIPE parameter, B-3
- SERVER parameter, B-3

Names Wizard, 6-51, 6-55

NAMES.ADDRESSES parameter, 6-52, 6-56, C-59

NAMES.ADMIN_REGION parameter, 6-52, C-59

NAMES.AUTHORITY_REQUIRED parameter, C-60

NAMES.AUTO_REFRESH_EXPIRE parameter, C-60

NAMES.AUTO_REFRESH_RETRY parameter, C-60

NAMES.CACHE_CHECKPOINT_FILE parameter, C-61

NAMES.CACHE_CHECKPOINT_INTERVAL parameter, C-61

NAMES.CONFIG_CHECKPOINT_FILE parameter, C-61

NAMES.CONNECT_TIMEOUT parameter, C-62

NAMESCTL utility, A-26
- command reference, A-28 to A-79
- commands
  - DELEGATE_DOMAIN, A-28
  - DOMAIN_HINT, A-29
  - DUMP_LDAP, A-30
  - DUMP_TNSNAMES, A-33
  - EXIT, A-34
  - FLUSH, A-34
  - FLUSH_NAME, A-35
  - HELP, A-36
  - LIST_DELEGATED, A-37
  - LIST_DOMAINS, A-37
  - LIST_OBJECTS, A-38
  - LOAD_TNSNAMES, A-39
  - LOG_STATS, A-40
  - PASSWORD, A-40
  - PING, A-41
  - QUERY, 11-14, A-42
  - QUIT, A-44
  - REGISTER, A-44
  - RELOAD, A-45
  - REORDER_NS, A-46
  - REPEAT, A-47
  - RESET_STATS, A-48
  - RESTART, A-48
  - SET, A-26, A-49
  - SET CACHE_CHECKPOINT_INTERVAL, A-50
  - SET DEFAULT_DOMAIN, A-51
  - SET FORWARDING_AVAILABLE, A-51
  - SET LOG_FILE_NAME, A-52
  - SET LOG_STATS_INTERVAL, A-53
  - SET NAMESCTL_TRACE_LEVEL, A-54
  - SET PASSWORD, A-55
  - SET REQUESTS_ENABLED, A-56
  - SET RESET_STATS_INTERVAL, A-56
  - SET SAVE_CONFIG_INTERVAL, A-57
  - SET SAVE_CONFIG_ON_STOP, A-58
  - SET SERVER, A-59
  - SET TRACE_FILE_NAME, A-60
  - SET TRACE_LEVEL, A-60
  - SHOW, A-26, A-61
  - SHOW CACHE CHECKPOINT_INTERVAL, A-62

SHOW DEFAULT_DOMAIN, A-63
SHOW FORWARDING_AVAILABLE, A-64
SHOW LOG_FILE_NAME, A-65
SHOW LOG_STATS_INTERVALE, A-65
SHOW NAMESCTL_TRACE_LEVEL, A-66
SHOW REQUEST_ENABLED, A-67
SHOW RESETS_STATS_INTERVAL, A-67
SHOW SAVE_CONFIG_INTERVAL, A-68
SHOW SAVE_CONFIG_ON_STOP, A-68
SHOW SERVER, A-69
SHOW STATS, A-69
SHOW SYSTEM_QUERIES, A-70
SHOW TRACE_FILE_NAME, A-70
SHOW TRACE_LEVEL, A-71
SHOW VERSION, A-72
SHUTDOWN, A-72
START, A-73
START_CLIENT_CACHE, 11-6, A-75
STARTUP, A-75
STATUS, A-76
STOP, A-76
TIMED_QUERY, A-77
UNREGISTER, A-78
VERSION, A-79
confirmation mode, A-27
delegating domains, 6-73
described, 4-31
discovering Oracle Names servers, 6-59, 6-61
distributed operation, A-27
function of and syntax format
global database links, 6-69
loading tnsnames.ora file into Oracle Names, 6-67
modes of operation, A-2
net service names, 6-66
querying objects, 11-14
starting an Oracle Names server, 11-5
testing network objects with, 11-14
testing the network, 11-3
NAMESCTL.INTERNAL_ENCRYPT_PASSWORD parameter, C-10
NAMESCTL.INTERNAL_USE parameter, C-11
NAMESCTL.NO_INITIAL_SERVER parameter, C-11
NAMESCTL.NOCONFIRM parameter, C-11
NAMESCTL.SERVER_PASSWORD parameter, C-11
NAMESCTL.TRACE_DIRECTORY parameter, C-12
NAMESCTL.TRACE_FILE parameter, C-12
NAMESCTL.TRACE_LEVEL parameter, C-12
NAMESCTL.TRACE_UNIQUE parameter, C-12
NAMES.DCE.PREFIX parameter, C-7
NAMES.DEFAULT_DOMAIN parameter, 6-7, 6-63, 8-18
NAMES.DEFAULT_FORWARDERS parameter, C-62
NAMES.DEFAULT_FORWARDERS_ONLY parameter, C-62
NAMES.DEFAULT.DOMAIN parameter, C-8
NAMES.DIRECTORY_PATH parameter, 8-20, C-8
CDS, 8-20
HOSTNAME, 8-20
LDAP, 8-20
NDS, 8-20
NIS, 8-20
ONAMES, 8-20
TNSNAMES, 8-20
NAMES.DOMAIN_CHECKPOINT FILE parameter, C-80
NAMES.DOMAIN_HINTS parameter, C-62
NAMES.DOMAINS parameter, 6-53, 6-54, 6-58, C-63
NAMES.FORWARDING_AVAILABLE parameter, C-63
NAMES.FORWARDING_DESIRED parameter, C-63
namesini.sql script, 6-51
NAMES.INITIAL_RETRY_TIMEOUT parameter, 6-63, C-8
NAMES.KEEP_DB_OPEN parameter, C-64
names.log file
, 12-19
NAMES.LOG_DIRECTORY parameter, 12-21, C-64
NAMES.LOG_FILE parameter, 12-21, C-64
NAMES.LOG_STATS_INTERVAL parameter, C-64
NAMES.LOG_UNIQUE parameter, C-65

NAMES.MAX_OPEN_ CONNECTIONS
    parameter, 6-63
NAMES.MAX_OPEN_CONNECTIONS
    parameter, C-9, C-65
NAMES.MAX_REFORWARDS parameter, C-65
NAMES.MESSAGE_POOL_ START_SIZE
    parameter, 6-63
NAMES.MESSAGE_POOL_START_SIZE
    parameter, C-9, C-66
NAMES.NDS.NAME_CONTEXT parameter, 6-86
NAMES.NDS.NAME.CONTEXT parameter, C-9
NAMES.NIS.META_MAP parameter, C-9
NAMES.NO_MODIFY_RESPONSE
    parameter, C-66
NAMES.NO_REGION_DATABASE
    parameter, C-66
names.ora file
  configuration parameter reference, C-59 to C-69
  obsolete parameters, C-80
  parameters
    NAMES.ADDRESSES, C-59
    NAMES.ADMIN_REGION, C-59
    NAMES.AUTHORITY_REQUIRED, C-60
    NAMES.AUTO_REFRESH_EXPIRE, C-60
    NAMES.AUTO_REFRESH_RETRY, C-60
    NAMES.CACHE_CHECKPOINT_
      FILE, C-61
    NAMES.CACHE_CHECKPOINT_
      INTERVAL, C-61
    NAMES.CONFIG_CHECKPOINT_
      FILE, C-61
    NAMES.CONNECT_TIMEOUT, C-62
    NAMES.DEFAULT_FORWARDERS, C-62
    NAMES.DEFAULT_FORWARDERS_
      ONLY, C-62
    NAMES.DOMAIN_CHECKPOINT
      FILE, C-80
    NAMES.DOMAIN_HINTS, C-62
    NAMES.DOMAINS, C-63
    NAMES.FORWARDING_
      AVAILABLE, C-63
    NAMES.FORWARDING_DESIRED, C-63
    NAMES.KEEP_DB_OPEN, C-64
    NAMES.LOG_DIRECTORY, C-64
    NAMES.LOG_FILE, C-64
    NAMES.LOG_STATS_INTERVAL, C-64
    NAMES.LOG_UNIQUE, C-65
    NAMES.MAX_OPEN_
      CONNECTIONS, C-65
    NAMES.MAX_REFORWARDS, C-65
    NAMES.MESSAGE_POOL_START_
      SIZE, C-66
    NAMES.NO_MODIFY_REQUESTS, C-66
    NAMES.NO_REGION_DATABASE, C-66
    NAMES.PASSWORD, C-66
    NAMES.REGION_CHECKPOINT_
      FILE, C-67
    NAMES.RESET_STATS_INTERVAL, C-67
    NAMES.SAVE_CONFIG_ON_STOP, C-67
    NAMES.SERVER_NAME, C-68
    NAMES.TOPOLOGY_CHECKPOINT_
      FILE, C-80
    NAMES.TRACE_DIRECTORY, C-68
    NAMES.TRACE_FILE, C-68
    NAMES.TRACE_FUNC, C-68
    NAMES.TRACE_LEVEL, C-69
    NAMES.TRACE_UNIQUE, C-69
    NAMES.USE_PLUG_AND_PLAY, C-80
NAMES.PASSWORDS parameter, C-66
names*pid*.trc file, 12-39
NAMES.PREFERRED_SERVERS parameter, 8-26,
    8-27, C-10
NAMES.REGION_CHECKPOINT_FILE
    parameter, C-67
NAMES.REQUEST_RETRIES parameter, 6-63,
    C-10
NAMES.RESET_STATS_INTERVAL
    parameter, C-67
NAMES.SAVE_CONFIG_ON_STOP
    parameter, C-67
NAMES.SERVER_NAME parameter, 6-52, 6-56
NAMES.SERVER_NAMES parameter, C-68
NAMES.TOPOLOGY_CHECKPOINT_FILE
    parameter, C-80
NAMES.TRACE_DIRECTORY parameter, 12-39,
    C-68
NAMES.TRACE_FILE parameter, 12-39, C-68
NAMES.TRACE_FUNC parameter, C-68
NAMES.TRACE_LEVEL parameter, 12-39, C-69
NAMES.TRACE_UNIQUE parameter, 12-39, C-69

names.trc file, 12-34
namesupg.sql script, 6-51
NAMES.USE_PLUG_AND_PLAY parameter, C-80
naming methods
  CDS, 2-37, 6-79
  Cell Directory Services, 2-37, 6-79
  centralized, 6-4
  choosing, 5-3
  described, 2-34
  directory naming, 2-35, 6-16 to 6-46
  external naming, 2-37, 4-24, 6-79 to 6-88
  host naming, 2-36, 6-74 to 6-78
  local naming, 2-35, 6-5 to 6-15
  localized, 6-4
  NDS, 2-37, 6-79
  Network Information Service, 2-37, 6-79
  NIS, 2-37, 6-79
  Novell Directory Services, 2-37, 6-79
  Oracle Names, 2-36, 6-47 to 6-74
  prioritizing order, 8-19
naming model
  hierarchical, 4-17
  single domain, 4-16
native authentication for directory naming, 2-51
NDS external naming, 2-37, 6-79, 8-20
  connecting with, 6-83
  establishing a connection with, 6-83
Net Service Name Wizard, 6-7, 6-23
net service names
  adding an address, 8-2
  configure an Oracle Connection Manager address, 8-39
  creating in Oracle Names, 6-66
  directory naming, 6-23
  external naming, 2-37, 6-79
  external procedures, 8-48
  local naming, 2-35, 6-6
  multiple addresses, 8-2, 8-4
  Oracle Names, 2-36, 6-65
  prioritizing naming methods, 8-19
  testing with TNSPING, 11-19
NET_CONFIG_METHOD parameter, C-79
Net8
  concept described, 1-2
  connect operations, 2-3
  control utilities, 11-3
  exception operations, 2-7
  JDBC drivers, 3-11
  operations, 2-7
  overview, 1-2
  primary functions, 2-7
Net8 access control, 4-8, 8-35
Net8 Assistant, 8-52
  adding addresses, 8-2
  clients
    client load balancing, 8-4
    connect-time failover, 8-4
    default network domains, 8-17
    local naming method, 6-7
    Oracle Connection Manager, 8-39
  Default Domain field, 6-63
  delegating domains to administrative regions, 6-73
  described, 4-29
  directory naming, 2-43
  Directory Server Migration Wizard, 6-32
  discovering Oracle Names servers, 6-61
  external procedure connections, 8-45
  Heterogeneous Services connections, 8-49
  Instance Name field, 8-9
  listeners
    Configure Prespawned Servers dialog box, 7-19
    configuring, 7-5
    Database Services tab, 7-14
    Global Database Name field, 7-13, 8-9, 8-52
    Maximum Prespawned Servers field, 7-16
    non-default through Oracle Names, 7-11
    Oracle Home Directory field, 7-13, 8-47, 8-49
    prespawned dedicated servers, 7-18
    Program Name field, 8-47, 8-49
    protocol addresses, 7-6
    Register Service with Oracle Names option, 6-64
    registering database information with Oracle Names, 6-64
    SID field, 7-13
    static service information, 7-13

local naming method, 6-7
log files, 12-22
multiple address options, 8-4
Names Wizard, 6-51, 6-55
Net Service Name Wizard, 6-7, 6-23
net service names, 6-7
Oracle Names
  aliases, 6-72
  Attempts Per Names Server field, 6-63
  Default Domain field, 6-63
  discovering, 6-61
  global database link qualifiers, 6-70
  global database links, 6-69
  Initial Preallocated Requests field, 6-63
  Maximum Open Connections field, 6-63
  Maximum Wait Each Attempt field, 6-63
  net service names, 6-66
  non-default listener, 7-11
  querying network objects, 11-14
  preferred Oracle Names servers, 8-26
  region database, 6-51
  registering database information from listener, 6-64
  starting Oracle Names servers, 11-14
Oracle Rdb Database, 8-53
Oracle RDB Database field, 8-9
profiles
  advanced options, 8-23
  Always Use Dedicated Server option, 8-21
  Client Registration ID field, 8-23
  Disable Out-of-Band Break option, 8-24
  naming method specification, 8-19
  Prefer Connection Manager Routing option, 8-21
  preferred Oracle Names servers, 8-26
  routing connections, 8-21
  TNS Time Out Value option, 8-23
  Turn Off UNIX Signal Handling option, 8-23
RDB Database field, 8-52
routing connection requests, 8-21
Session Data Unit (SDU) field in Net8 Assistant, 8-9
SID field for listeners, 8-47, 8-49
specifying naming methods, 8-19
starting, 4-29
starting an Oracle Names server, 6-59, 11-5
testing
  client configuration, 11-22
  server configuration, 11-15
Timeout field, 7-17
trace files, 12-40
Type of Service field, 8-9
Use for Heterogeneous Services option, 8-9, 8-51
Net8 Client, described, 4-2
Net8 Configuration Assistant
  clients
    administrative context selection, 6-19
    directory access, 6-21
    local naming method, 6-11
  described, 4-28
  listener configuration, 7-4
  local naming method, 6-11
  net service names, 6-11
  OracleDBCreators group, 2-44, 2-53, 6-22
  OracleNetAdmins group, 2-52, 6-22
  OracleSecurityAdmins group, 2-53, 6-22
  servers
    administrative context selection, 6-19
    directory access, 6-22
    Oracle Context creation, 2-43, 6-22
    Oracle schema creation, 6-22
  starting, 4-28
Net8 Firewall Proxy, 4-8
Net8 OPEN
  API function calls, 10-3
  compatibility with C language, 10-3
  configuration requirements, 10-7
  configuring system to use applications, 10-8
  error messages, 10-9
  finding API, 10-7
  for distributed applications, 10-2
  integrating with non-SQL information, 10-2
  library, 10-7
  sample applications, 10-9
Net8 Server, described, 4-2
Network Authentication (NA)
  layer in error stacks, 12-17
network character set, keyword values, C-4
network configuration

centralized management, 6-4
localized management, 6-4
Network Encryption (NAE), layer in error stacks, 12-17
Network Information Service external naming, 2-37, 4-24, 8-20
configuring, 6-79
connecting with, 6-79
establishing a connection with, 6-79
Network Interface (NI)
described, 3-8
layer in error stacks, 12-17
Network Naming (NN)
layer in error stacks, 12-17
network performance, improving
by adjusting SDU size, 5-8
by client load balancing, 5-8
client load balancing, 2-20, C-40
connection concentration, 5-7
connection pooling, 2-12, 5-6
listener queuesize, 5-6
Network Program Interface (NPI), 3-9
Network Routing (NR)
layer in error stacks, 12-17
Network Security
DES_40 encryption, 4-27
described, 4-27
MD5 checksumming, 4-27
RC4_128 encryption, 4-27
RC4_40 encryption, 4-27
RC4_56 encryption, 4-27
Network Services (NA), layer in error stacks, 12-17
Network Session (NS)
described, 3-8
Network Session (NS), layer in error stacks, 12-17
Network Transport (NT), layer in error stacks, 12-17
new features
client load balancing, xx
connection load balancing, xx
connect-time failover, xx
direct hand-off, xx
directory naming, xx
Net8 Configuration Assistant, xxi
Oracle Connection Manager, xxi
service naming, xx
service registration, xx
SERVICE_NAMES, xxi
TCP/IP with SSL, xxi
NI layer, 3-8
NIS external naming, 2-37, 4-24, 6-79, 8-20
connecting with, 6-79
establishing a connection with, 6-79
NIS Maps, 6-80
NNL-00018 error message, 6-54
NNL-00024 error message, 6-54
nodes, described, 3-2
Novell Directory Services external naming, 2-37, 2-39, 8-20
configuring, 6-79
connecting with, 6-83
described, 4-24
establishing a connection with, 6-83
NPI, 3-9
NS layer, 3-8
NS.SMD record, 11-14, A-42

O

o attribute, 2-41
object classes
described, 2-54
orclDBServer, 2-54, D-2
orclNetAddress, 2-54, D-4
orclNetAddressList, 2-54, D-4
orclNetDescription, 2-54, D-3
orclNetDescriptionList, 2-54, D-3
orclNetService, 2-54, D-2
obsolete parameters in release 8.1, C-80
OCI, described, 3-7
ONRS_CONFIG table, 6-49
Open System Interconnection (OSI) model
client application layer, 3-4
described, 3-2
link layer, 3-4
physical, 3-4
presentation layer, 3-4
session layer, 3-4
transport layer, 3-4
operations

connect, 2-3
exception, 2-7
Net8, 2-7
OPI, described, 3-9
ORA-00508 error message, 12-10
ORA-1017 error messages, 12-3
ORA-1034 error messages, 12-3
ORA-12154 error message, 12-8
ORA-12198 error message, 12-9
ORA-12203 error message, 12-9, 12-10, 12-14
sample error stack, 12-18
ORA-12224 error message, 12-12
ORA-12533 error message, 12-12
ORA-12535 error message, 12-14
ORA-12538 error message, 12-10
ORA-12545 error message, 12-12
ORA-12547 error message, 12-14
ORA-12560 error message, 12-13
ORA-3113 error message, 12-13
ORA-3121 error message, 12-13
Oracle Advanced Security
CyberSAFE authentication method, 4-27
DCE Integration, 4-28
DES_40 encryption, 4-27
described, 4-25
features of, 4-25
Identix authentication method, 4-27
Kerberos authentication method, 4-27
MD5 checksumming, 4-27
Network Security, 4-27
RADIUS authentication method, 4-27
RC4_128 encryption, 4-27
RC4_40 encryption, 4-27
RC4_56 encryption, 4-27
SecurID authentication method, 4-27
Single Sign-On, 4-27
SSL authentication method, 4-27
Oracle Call Interface (OCI) layer, described, 3-7
Oracle Connection Manager
access control, 4-8
CMADMIN process, 4-6
CMGW process, 4-5
configuration parameter reference, C-70 to C-76
configuring
access control, 8-35
connection concentration, 8-28
multi-protocol support, 8-32
Net8 access control, 8-35
protocol address for listener, 8-42
protocol address for Oracle Connection Manager, 8-39
protocol addresses, B-1
connection concentration, 4-6
log files, 12-22
configuring, 12-22
understanding, 12-29
multiple protocol support, 4-8
multi-threaded server, 4-6, 8-28
Net8 access control, 4-8
Net8 Firewall Proxy, 4-8
processes, 4-5
role in multiple protocol environments, 4-9
routing connections, 1-3, 8-21
SOURCE_ROUTE parameter, C-41
starting, 11-9
testing, 11-15
trace files, 12-40
configuring, 12-40
Oracle Connection Manager Control Utility (CMCTL)
See CMCTL utility, 11-9
Oracle Context
creating, 6-18, 6-22
described, 2-41
Oracle Customer Support, contacting, 12-59
Oracle Database Configuration Assistant
directory naming, 2-43, 6-22
Oracle Enterprise Manager
connect-time failover, 2-21, 5-8
SID_LIST_*listener_name* parameter in listener.ora file, 7-13, C-84
static service information in listener.ora file, 2-21, 5-8
Oracle home directory
configuring on the listener, 7-13

Oracle Home Directory field in Net8
    Assistant, 7-13, 8-47, 8-49
Oracle home directory, defined, xxii
Oracle Internet Directory, 2-39
Oracle Names, 2-36
  administrative regions, 4-12
  advantages and disadvantages, 5-4
  architecture, 4-10
  client cache, 11-6
  configuring, 6-47 to 6-74
    clients and servers, 6-60
    region database, 6-50
  connecting with, 6-74
  creating Oracle Names servers, 6-49
  database as a repository for information, 4-13
  default network domains, 4-17
  delegated
    administrative regions, 4-21
  described, 2-36, 4-10
  differences between versions, 4-21
  discovering Oracle Names servers, 6-59, 6-60
  discovery, 4-24
  domains, 4-12
    administering multiple, 6-53, 6-58
  Dynamic Discovery Option, 4-22
  establishing a connection with, 2-35, 2-36, 6-74
  exporting objects to directory
      server, 6-35 to 6-46, A-30
  hierarchical naming model, 4-17
  installing, 6-48
  listener configuration for, 6-64
  log files, 12-21
  multiple domains, 4-18
  multiple regions, 4-18
  namesini.sql script, 6-51
  namesupg.sql script, 6-51
  naming network components, 4-16
  querying objects, 11-14
  recommended for, 5-4
  region database, 4-13
  regions, 4-12
  registering
    global database link qualifiers, 6-68
    global database links, 6-68
    net service names, 6-65
    objects, 6-65
  replicated data, 4-13, 6-55
  root administrative region, 4-19
  .sdns.ora file, 4-24
  sdns.ora file, 4-24
  single domain naming mode, 4-16
  starting
    from Net8 Assistant, 11-14
    from the command line, 11-14
    from the Control Panel, 11-9
  starting from the command line, 11-14
  storing
    aliases, 4-15
    database global database names and
        addresses, 4-14
    database links, 4-15
    net service names, 4-14
    Oracle Connection Manager
        information, 4-15
    other Oracle Names server names and
        addresses, 4-14
  trace files, 12-39
  version 1, 4-21
  version 2, 4-22
  version 8, 4-23
Oracle Names Control Utility (NAMESCTL)
  See NAMESCTL utility
Oracle Names servers
  ckcfg.ora file, 6-49
  ckpcch.ora file, 6-49
  ckpreg.ora file, 6-49
  client cache, 11-6
  configuration parameter reference, C-59 to C-69
  configuring
    multiple domains, 6-53, 6-58
    protocol addresses, B-1
  creating, 6-49
    region database, 6-51
    replicated data, 6-55
  data stored in, 4-14
  default configuration, 6-50

delegated
domains, 6-73
delegated administrative regions, 4-19
described, 4-10
discovery of, 6-59
log files, 12-21
configuring, 12-22
preferred, 4-21, 8-26
registering
global database link qualifiers, 6-68, 6-70
global database links, 6-68
net service names, 6-65
objects, 6-65
replicated data, 6-55
root administrative regions, 4-19
starting, 6-54, 11-5
client cache, 11-6
from Net8 Assistant, 11-14
from the command line, 11-14
from the Control Panel, 11-9
storing
aliases, 4-15
database global database names and addresses, 4-14
database links, 4-15
net service names, 4-14
Oracle Connection Manager information, 4-15
other Oracle Names server names and addresses, 4-14
testing, 11-13
TNSPING utility, 11-19
trace files, 12-39
well-known, 4-22
Oracle Parallel Server
connect-time failover, 2-21, 8-4, C-39
FAILOVER parameter, 8-4, C-39
FAILOVER_MODE parameter, 8-13, C-43
INSTANCE_NAME parameter, C-46, C-87
LOAD_BALANCE parameter, C-40, C-87
Transparent Application Failover, 8-12
Oracle Program Interface (OPI). described, 3-9
Oracle protocols
Bequeath, 4-4, B-3
configuring addresses, B-3
described, 3-9, 4-2
IPC, B-3
LU6.2, B-3, 4-4
Named Pipes, 4-3, B-3
SPX, 4-3, B-4
TCP/IP, 4-2, B-4
TCP/IP with SSL, 4-2, B-4
Oracle Rdb database
configuring for connection to, 8-52
described, 8-52
Oracle RDB Database field in Net8 Assistant, 8-9
Oracle schema
attributes, D-1 to D-5
creating, 6-22
described, 2-54
object classes, D-1 to D-5
Oracle System Identifier, configuring on the listener, 7-13
ORACLE_HOME parameter, 7-13, 8-47, 8-49, C-51
ORACLE_HOME, defined, xxii
Oracle8*i* JServer
connections, 3-16
listener configuration, 7-8
Oracle8*i* JServer connections, 7-8
OracleDBCreators group, 2-43, 2-44, 2-53, 6-22
OracleNetAdmins group, 2-44, 2-52, 6-22, 6-23, 6-31
OracleSecurityAdmins group, 2-53, 6-22
orclDBServer object class, 2-54, D-2
orclDescList attribute, D-5
orclDescName attribute, D-5
orclLoadBalance attribute, D-5
orclNetAddress object class, 2-54, D-4
orclNetAddressList object class, 2-54, D-4
orclNetAddrList attribute, D-5
orclNetAddrString attribute, D-5
orclNetConnParamList attribute, D-5
orclNetDescription object class, 2-54, D-3
orclNetDescriptionList object class, 2-54, D-3
orclNetFailover attribute, D-5
orclNetInstanceName attribute, D-5
orclNetProtocolStack attribute, D-5

orclNetSdu attribute, D-5
orclNetService object class, 2-54, D-2
orclNetServiceName attribute, D-5
orclNetSourceRoute attribute, D-5
orclProtocol attribute, D-5
orclSid attribute, D-5
orclVersion attribute, D-5
OSI model
  client application layer, 3-4
  network layer, 3-4
  physical, 3-4
  presentation layer, 3-4
  session layer, 3-4
  transport layer, 3-4
OSS.MY.WALLET parameter, C-13, C-55
ou attribute, 2-41
OUT_OF_RELAY, A-94

P

packets
  example of trace data, 12-44
  types of, 12-44
parameters, B-3
  cman.ora configuration reference, C-70 to C-76
  ldap.ora configuration reference, C-78
  listener.ora configuration reference, C-48 to C-58
  names.ora configuration reference, C-59 to C-69
  obsolete for release 8.1, C-80
  Oracle protocols, B-3
  protocol.ora configuration reference, C-77
  sqlnet.ora configuration reference, C-5 to C-30
  tnsnames.ora configuration reference, C-31 to C-48
PARTNER_LU_LOCAL_ALIAS parameter, B-3
PARTNER_LU_NAME parameter, B-3
password
  for LSNRCTL access, A-3
  for NAMESCTL access, A-26
PASSWORD command, A-40
PASSWORDS_*listener_name* parameter, C-55
physical layer, described, 3-4
PING utility, 11-13, A-41
  testing Oracle Names servers with, 11-13
PIPE parameter, B-3
planning overview, 5-2
PLU parameter, B-3
PLU_LA parameter, B-3
PMON process, 2-22
POOL attribute, 9-6
POOL_SIZE parameter, 2-17, 7-17
port 1024, B-5
port 1521, B-4
  if not using, 7-10
port 1575, 4-24, 6-55, B-4, C-59
port 1630, 4-5, 8-28, 8-32, 8-35, B-4
port 1646, C-23
port 1830, 8-28, 8-32, B-4, C-85
port 2481, B-4
port 2482, B-4
port numbers, allowed, B-5
PORT parameter, B-4
Prefer Connection Manager Routing option in Net8 Assistant, 8-21
preferred Oracle Names servers, 4-21
presentation layer
  described, 3-4
  General Inter-Orb Protocol, 3-16
  GIOP, 3-16
PRESPAWN_DESC parameter, C-54
PRESPAWN_LIST parameter, C-53
PRESPAWN_MAX parameter, 2-17, 7-16, C-53
prespawned dedicated servers, 2-16
  advantages and disadvantages, 5-5
  configuring, 7-16
  number of, 7-17
  POOL_SIZE parameter, 2-17
  PRESPAWN_MAX parameter, 2-17
  recommended for, 5-5
  TIMEOUT parameter, 2-17
prestarted dedicated server process
  See prespawned dedicated server process
private database links, 4-15

profiles (sqlnet.ora)
- configuration parameter reference, C-5 to C-30
- configuring
  - advanced options, 8-23
  - default domains, 8-17
- configuring preferred Oracle Names servers, 8-26
- naming methods, specifying, 8-19
- prioritizing naming methods, 8-19
- routing connection requests, 8-21

Program Name field in Net8 Assistant, 8-47, 8-49

PROGRAM parameter, 8-47, 8-49, B-3, C-51

PROTOCOL (PRO or PROT) attribute, 8-29, 9-4

PROTOCOL parameter, 7-16, B-3

protocol.EXCLUDED_NODES parameter, C-77

protocol.INVITED_NODES parameter, C-77

protocol.ora file, C-78
- configuration parameter reference, C-77
- controlling access to database, C-77
- *protocol*.EXCLUDED_NODES parameter, C-77
- *protocol*.INVITED_NODES parameter, C-77
- *protocol*.VALIDNODE_CHECKING parameter, C-77
- TCP.NODELAY parameter, C-77

protocols, B-3
- Bequeath, 4-4, B-3
- configuration parameter reference, C-77, C-78
- configuring address with, B-1
- configuring addresses, B-3
- IPC, B-3
- LU6.2, 4-4
- Named Pipes, 4-3, B-3
- selecting for network layout, 5-2
- selecting multiple for network layout, 5-2
- SPX, 4-3, B-4
- TCP/IP, 4-2, B-4
- TCP/IP with SSL, 4-2, B-4

protocol.VALIDNODE_CHECKING parameter, C-77

public database links, 4-15

Q

QUERY command, 6-59, 11-14
- A.SMD record, 7-11, 11-14, A-42
- CNAME.SMD, 11-14, A-42
- DLCR.TDBMS.OMD record, 11-14, A-42
- DL.RDBMS.OMD, 11-14, A-42
- NS.SMD, 11-14, A-42
- of NAMESCTL, reference, A-42
- V1ADD.NPO.OMD, 11-14, A-42

queuesize, 5-6, 7-12

QUEUESIZE parameter, 7-12
- for adjusting listener queue size, 7-12
- for adjusting listener queuesize, 5-6

QUIT command
- of CMCTL, A-83
- of LSNRCTL, A-8
- of NAMESCTL, A-44

R

RADIUS authentication method
- *See* Oracle Advanced Security

randomizing requests among listeners, 2-20, C-40

RDB Database field in Net8 Assistant, 8-52

RDB_DATABASE parameter, 8-9, 8-52, C-46

redirect connection, 2-9

reference
- for cman.ora, C-70 to C-76
- for CMCTL commands, A-81 to A-97
- for ldap.ora, C-78
- for listener.ora, C-48 to C-58
- for LSNRCTL commands, A-4 to A-25
- for NAMESCTL commands, A-28 to A-79
- for names.ora, C-59 to C-69
- for protocol.ora, C-77
- for sqlnet.ora, C-5 to C-30
- for tnsnames.ora, C-31 to C-48

region database
- configure database server, 6-50, 6-54
- create Oracle Names server, 6-51
- defined, 4-13

regions
  delegated administrative, 4-19
  delegated administrative below root, 4-21
  in Oracle Names, 4-12, 4-18
  organizing multiple administrative network, 4-18
  root administrative, 4-19
register, 6-69
REGISTER command, 6-66, 6-69, A-44
Register Service with Oracle Names option, 6-64
registering
  aliases with Oracle Names, 6-72
  global database link qualifiers with Oracle Names, 6-68
  global database links with Oracle Names, 6-68
  net service names with Oracle Names, 6-65
  objects with Oracle Names, 6-65
Relative Distinguished Name (RDN), defined, 2-40
RELAY_STATISTICS parameter, C-74
RELOAD command
  of LSNRCTL, A-8
  of NAMESCTL, A-45
REMOTE_ADMIN parameter, C-74, C-89
REORDER_NS command, 6-59, 6-61, 8-26, A-46
REPEAT command, A-47
replicated data in Oracle Names, 4-13
replication of data in Oracle Names, 6-55
RESET_STATS command, A-48
resolving errors, *See* troubleshooting
RESTART command, A-48
RETRIES parameter, 8-13, C-44
root administrative regions
  described, 4-19
  domain requirements, 4-19
  requirements, 4-19
root domains, described, 4-19
routing connections, 1-3, 8-21
rules, syntax for configuration files, C-2

S

SAVE_CONFIG command, A-9
SAVE_CONFIG_ON_STOP_*listener_name* parameter, C-56
schema extension, in NDS, 6-83
.sdns.ora file, 4-24
sdns.ora file, 4-24
SDU
  See session data unit
SDU parameter, 8-9, C-40
SecurID authentication method
  See Oracle Advanced Security
security, A-3, A-26
  LSNRCTL utility, A-3
  NAMESCTL utility, A-26
server configuration
  allocating resources with multi-threaded server, 9-9
  connection concentration, 8-29
  discovering Oracle Names servers, 6-60
  external procedures, 8-48
  local naming, 6-6
  log files, 12-20
  multi-threaded server, 9-4
  prespawned dedicated server, 7-16
  region database for Oracle Names, 6-50
  trace files, 12-35
server connections
  methods of, 11-11
  syntax, 11-10
SERVER parameter, 2-9, B-3, C-46
server testing, 11-15
server troubleshooting, 12-3
servers
  dedicated
    multi-threaded contrasted with, 9-2
  disconnections, 2-6
  in stack communications, 3-9
  loopback tests, 12-4
  multi-threaded
    dedicated contrasted with, 9-2
  server-to-server interaction, described, 3-9
  starting databases, 11-8
SERVICE (SER or SERV) attribute, 9-6

service name
  and address storage, in NDS, 6-84
  character set keyword values, C-5
  described, 6-2
SERVICE parameter, B-4
service registration, xx
  benefits, 2-23
  configuration, 2-23
  connection load balancing, 2-24, 2-26
  connect-time failover, 2-23
  described, 2-22
  events in listener.log, 12-26
  SERVICE_DIED event, 12-26
  SERVICE_REGISTER event, 12-26
  SERVICE_UPDATE event, 12-26
SERVICE_DIED event, 12-26
SERVICE_NAME parameter, 6-3, C-47, C-88
SERVICE_NAMES parameter, 2-32, 6-3, 9-6
SERVICE_REGISTER event, 12-26
SERVICE_UPDATE event, 12-26
services, 1-5
  connecting, 1-5
  described, 1-5
  service names, 6-2
SERVICES command, 2-9, 2-14, A-9
session data unit
  adjusting to improve network performance, 5-8
  configuring, 8-9
Session Data Unit (SDU) Size field in Net8 Assistant, 8-9
session layer
  described, 3-4
SESSIONS (SES or SESS) attribute, 9-7
SET AUTHENTICATION_LEVEL command, A-84
SET CACHE_CHECKPOINT_INTERVAL command, A-50
SET command
  of CMCTL, A-83
  of LSNRCTL, A-11
  of NAMESCTL, A-49
SET CONNECT_TIMEOUT command, A-12
SET CURRENT_LISTENER command, A-12
SET DEFAULT_DOMAIN command, A-51
SET DISPLAYMODE command
  of CMCTL, A-85
  of LSNRCTL, A-13
SET FORWARDING_AVAILABLE command, A-51
SET LOG_DIRECTORY command, A-13
SET LOG_FILE command, A-14
SET LOG_FILE_NAME command, A-52
SET LOG_LEVEL command, A-85
SET LOG_STATS_INTERVAL command, A-53
SET LOG_STATUS command, A-14
SET NAMESCTL_TRACE_LEVEL command, A-54
SET PASSWORD command
  of LSNRCTL, 11-7, A-15
  of NAMESCTL, A-55
SET RELAY_STATISTICS command, A-86
SET REQUESTS_ENABLED command, A-56
SET RESET_STATS_INTERVAL command, A-56
SET SAVE_CONFIG_INTERVAL command, A-57
SET SAVE_CONFIG_ON_STOP command, A-15
  of LSNRCTL, A-15
  of NAMESCTL, A-58
SET SERVER command, A-59
SET STARTUP_WAITTIME command, A-16
SET TRACE_FILE_NAME command, A-60
SET TRACE_LEVEL command, A-60
SET TRC_DIRECTORY command, A-17
SET TRC_FILE command, A-17
SET TRC_LEVEL command, A-18
SET USE_PLUGANDPLAY command, A-18
shared servers
  advantages and disadvantages, 5-5
  recommended for, 5-5
  See also multi-threaded servers
SHOW ADDRESS command, A-87
SHOW ALL command, A-88
SHOW CACHE_CHECKPOINT INTERVAL command, A-62
SHOW command
  of CMCTL, A-86
  of LSNRCTL, A-19
  of NAMESCTL, A-61
SHOW CONNECT_TIMEOUT command, A-19
SHOW CURRENT_LISTENER command, A-19
SHOW DEFAULT_DOMAIN command, A-63

SHOW DISPLAYMODE command
of CMCTL, A-89
of LSNRCTL, A-19
SHOW FORWARDING_AVAILABLE command, A-64
SHOW LOG_DIRECTORY command, A-19
SHOW LOG_FILE command, A-19
SHOW LOG_FILE_NAME command, A-65
SHOW LOG_STATS_INTERVAL command, A-65
SHOW LOG_STATUS command, A-19
SHOW NAMESCTL_TRACE_LEVEL command, A-66
SHOW PASSWORD command, A-19
SHOW PROFILE command, A-89
SHOW RAW_MODE command, A-19
SHOW RELAY command, A-90
SHOW REQUESTS_ENABLED command, A-67
SHOW RESETS_STATS_INTERVAL command, A-67
SHOW RULES command, A-91
SHOW SAVE_CONFIG_INTERVAL command, A-68
SHOW SAVE_CONFIG_ON_STOP command, A-19, A-68
SHOW SERVER command, A-69
SHOW STARTUP_WAITTIME command, A-19
SHOW STATUS command, A-69
SHOW SYSTEM_QUERIES command, A-70
SHOW TRACE_FILE_NAME command, A-70
SHOW TRACE_LEVEL command, A-71
SHOW TRC_DIRECTORY command, A-19
SHOW TRC_FILE command, A-19
SHOW TRC_LEVEL command, A-19
SHOW USE_PLUGANDPLAY command, A-19
SHOW VERSION command, A-72
SHOW_TNS_INFO parameter, C-74
SHUTDOWN command
of CMCTL, A-92
of NAMESCTL, A-72
SID field in Net8 Assistant, 7-13, 8-47, 8-49
SID parameter, C-48
SID, configuring on the listener, 7-13
SID_DESC parameter, C-52
SID_LIST_*listener_name* parameter, C-52
external procedures, 8-47
Heterogeneous Services, 8-49
Oracle Enterprise Manager
listener.ora requirements, 7-13, C-50
SID_NAME parameter, 7-13, 8-47, 8-49, C-52
signal handling
requirements for Oracle's operating system dependent call, 10-11
UNIX considerations, 10-10
using in conjunction with Net8, 10-11
simple authentication for directory naming, 2-51
single domain naming model, 4-16
Single Sign-On, described, 4-27
SNA, 4-4
SOURCE_ROUTE parameter, 8-4, 8-7, 8-31, 8-34, 8-38, C-41
SPAWN command, A-20
SPX protocol
described, 4-3
parameters for addresses, B-4
SERVICE parameter, B-4
SQLNET_CRYPTO_CHECKSUM_TYPE_CLIENT parameter, C-15
SQLNET_CRYPTO_CHECKSUM_TYPE_SERVER parameter, C-15
SQLNET.ALTERNATE_PORT parameter, C-21
SQLNET.AUTHENTICATION_GSSAPI_ SERVICE parameter, C-14
SQLNET.AUTHENTICATION_KERBEROS5_ SERVICE parameter, C-14
SQLNET.AUTHENTICATION_SERVICES parameter, C-13
SQLNET.CLIENT_AUTHENTICATION parameter, C-23, C-56
SQLNET.CLIENT_REGISTRATION parameter, 8-23, C-14
SQLNET.CRYPTO_CHECKSUM_CLIENT parameter, C-14
SQLNET.CRYPTO_CHECKSUM_SERVER parameter, C-15
SQLNET.CRYPTO_SEED parameter, C-16
SQLNET.ENCRYPTION_CLIENT parameter, C-16
SQLNET.ENCRYPTION_SERVER parameter, C-17

SQLNET.ENCRYPTION_TYPES_CLIENT
  parameter, C-17
SQLNET.ENCRYPTION_TYPES_SERVER
  parameter, C-18
SQLNET.EXPIRE_TIME parameter, 8-23, C-18
SQLNET.IDENTIX_FINGERPRINT_DATABASE
  parameter, C-18
SQLNET.IDENTIX_FINGERPRINT_DATABASE_
  PASSWORD parameter, C-19
SQLNET.IDENTIX_FINGERPRINT_DATABASE_
  USER parameter, C-19
SQLNET.IDENTIX_FINGERPRINT_METHOD
  parameter, C-19
SQLNET.KERBEROS5_CC_NAME
  parameter, C-19
SQLNET.KERBEROS5_CLOCKSKEW
  parameter, C-20
SQLNET.KERBEROS5_CONF parameter, C-20
SQLNET.KERBEROS5_KEYTAB parameter, C-20
SQLNET.KERBEROS5_REALMS parameter, C-20
sqlnet.log file, 12-19
  sample client-side error stack, 12-18
sqlnet.ora file
  configuration parameter reference, C-5 to C-30
  parameters
    AUTOMATIC_IPC, C-80
    BEQUEATH_DETACH, 10-12, C-5
    DAEMON.TRACE_DIRECTORY, C-5
    DAEMON.TRACE_LEVEL, C-6
    DAEMON.TRACE_MASK, C-6
    DISABLE_OOB, C-6
    LOG_DIRECTORY_CLEINT, C-7
    LOG_DIRECTORY_SERVER, C-7
    LOG_FILE_CLIENT, C-7
    LOG_FILE_SERVER, C-7
    NAMES_DIRECTORY_PATH, C-8
    NAMESCTL_INTERNAL_ENCRYPT_
      PASSWORD, C-10
    NAMESCTL.INTERNAL_USE, C-11
    NAMESCTL.NO_INITIAL_SERVER, C-11
    NAMESCTL.NOCONFIRM, C-11
    NAMESCTL.SERVER_PASSWORD, C-11
    NAMESCTL.TRACE_DIRECTORY, C-12
    NAMESCTL.TRACE_FILE, C-12
    NAMESCTL.TRACE_LEVEL, C-12
    NAMESCTL.TRACE_UNIQUE, C-12
    NAMES.DCE_PREFIX, C-7
    NAMES.DEFAULT_DOMAIN, 8-17, C-8
    NAMES.DIRECTORY_PATH, 8-19
    NAMES.INITIAL_RETRY_TIMEOUT, C-8
    NAMES.MAX_OPEN_CONNECTIONS, C-9
    NAMES.MESSAGE_POOL_START_
      SIZE, C-9
    NAMES.NDS.NAME.CONTEXT, C-9
    NAMES.NIS.META_MAP, C-9
    NAMES.PREFERRED_SERVERS, 8-26, C-10
    NAMES.REQUEST_RETRIES, C-10
    OSS.MY.WALLET, C-13
    SQLNET_CRYPTO_CHECKSUM_TYPE_
      CLIENT, C-15
    SQLNET_CRYPTO_CHECKSUM_TYPE_
      SERVER, C-15
    SQLNET.ALTERNATE_PORT, C-21
    SQLNET.AUTHENTICATION_GSSAPI_
      SERVICE, C-14
    SQLNET.AUTHENTICATION_KERBEROS5_
      SERVICE, C-14
    SQLNET.AUTHENTICATION_
      SERVICES, C-13
    SQLNET.CLIENT_
      AUTHENTICATION, C-23
    SQLNET.CLIENT_REGISTRATION, C-14
    SQLNET.CRYPTO_CHECKSUM_
      CLIENT, C-14
    SQLNET.CRYPTO_CHECKSUM_
      SERVER, C-15
    SQLNET.CRYPTO_SEED, C-16
    SQLNET.ENCRYPTION_SERVER, C-17
    SQLNET.ENCRYPTION_TYPES_
      CLIENT, C-17
    SQLNET.ENCRYPTION_TYPES_
      SERVER, C-18
    SQLNET.ENCYRPTION_CLIENT, C-16
    SQLNET.EXPIRE_TIME, C-18
    SQLNET.IDENTIX_FINGERPRINT_
      DATABASE, C-18
    SQLNET.IDENTIX_FINGERPRINT_
      DATABASE_PASSWORD, C-19
    SQLNET.IDENTIX_FINGERPRINT_
      DATABASE_USER, C-19

SQLNET.IDENTIX_FINGERPRINT_
  METHOD, C-19
SQLNET.KERBEROS5_CC_NAME, C-19
SQLNET.KERBEROS5_CLOCKSKEW, C-20
SQLNET.KERBEROS5_CONF, C-20
SQLNET.KERBEROS5_KEYTAB, C-20
SQLNET.KERBEROS5_REALMS, C-20
SQLNET.RADIUS_ALTERNATE, C-21
SQLNET.RADIUS_ALTERNATE_
  RETRIES, C-21
SQLNET.RADIUS_
  AUTHENTICATION, C-21
SQLNET.RADIUS_AUTHENTICATION_
  INTERFACE, C-22
SQLNET.RADIUS_AUTHENTICATION_
  PORT, C-22
SQLNET.RADIUS_AUTHENTICATION_
  RETRIES, C-22
SQLNET.RADIUS_AUTHENTICATION_
  TIMEOUT, C-22
SQLNET.RADIUS_CHALLENGE_
  RESPONSE, C-23
SQLNET.RADIUS_SECRET, C-23
SQLNET.RADIUS_SEND_
  ACCOUNTING, C-23
SSL_CIPHER_SUITES, C-24
SSL_VERSION, C-24
TNSPING.TRACE_DIRECTORY, C-24
TNSPING.TRACE_LEVEL, C-25
TRACE_DIRECTORY_CLIENT, C-25
TRACE_DIRECTORY_SERVER, C-25
TRACE_FILE_CLIENT, C-25
TRACE_FILE_SERVER, C-25
TRACE_FILEN_CLIENT, C-26
TRACE_FILEN_SERVER, C-26
TRACE_FILENO_CLIENT, C-26
TRACE_FILENO_SERVER, C-27
TRACE_LEVEL_CLIENT, C-27
TRACE_LEVEL_SERVER, C-27
TRACE_TIMESTAMP_CLIENT, C-28
TRACE_TIMESTAMP_SERVER, C-28
TRACE_UNIQUE_CLIENT, C-28
USE_CMAN, 8-21, C-29
USE_DEDICATED_SERVER, 8-21, C-30

sqlnet*pid*.trc file, C-28
SQLNET.RADIUS_ALTERNATE parameter, C-21
SQLNET.RADIUS_ALTERNATE_RETRIES
  parameter, C-21
SQLNET.RADIUS_AUTHENTICATION
  parameter, C-21
SQLNET.RADIUS_AUTHENTICATION_
  INTERFACE parameter, C-22
SQLNET.RADIUS_AUTHENTICATION_PORT
  parameter, C-22
SQLNET.RADIUS_AUTHENTICATION_RETRIES
  parameter, C-22
SQLNET.RADIUS_AUTHENTICATION_TIMEOUT
  parameter, C-22
SQLNET.RADIUS_CHALLENGE_RESPONSE
  parameter, C-23
SQLNET.RADIUS_SECRET parameter, C-23
SQLNET.RADIUS_SEND_ACCOUNTING
  parameter, C-23
sqlnet.trc file, 12-34, C-25
square brackets, defined, xxii
SRC parameter, 8-36, C-76
SRV parameter, 8-36, C-76
SSL authentication for directory naming, 2-51
SSL authentication method
  See Oracle Advanced Security
SSL_CIPHER_SUITES parameter, C-24
SSL_VERSION parameter, C-24
stack communications
  client application, 3-6
  described, 3-2
  Network Interface, 3-8
  Network Session, 3-8
  NI, 3-8
  NS, 3-8
  OCI, 3-7
  Oracle Call Interface, 3-7
  Oracle protocols, 3-9
  server side interaction, 3-9
  TTC, 3-7
START CMAN command, 11-9
START command
  of CMCTL, A-92
  of LSNRCTL, 7-5, 11-7, A-20
  of NAMESCTL, 6-54, 6-59, 11-5, A-73

START_CLIENT_CACHE command, 11-6, A-75
starting
  an instance
    with dispatchers, 9-7
    with multi-threaded servers, 9-4
  databases, 11-8
  listeners, 11-7
  Net8 Assistant, 4-29
  Net8 components, 11-4
  Net8 Configuration Assistant, 4-28
  Oracle Connection Manager, 11-9
  Oracle Names client cache, 11-6
  Oracle Names servers, 6-54, 11-5
STARTUP command
  of NAMESCTL, A-75
STARTUP_WAITTIME_*listener_name* parameter, C-56
STATS command, A-93
STATUS command
  of CMCTL, A-95
  of LSNRCTL, 11-7, A-22
  of NAMESCTL, A-76
STOP command
  of CMCTL, A-96
  of LSNRCTL, 11-7, A-24
  of NAMESCTL, A-76
STOPNOW command
  of CMCTL, A-97
strong authentication for directory naming, 2-51
svr_*pid*.trc file, 12-34, C-25
syntax
  for CMCTL, 11-4
  for connect identifiers, 11-10
  for LSNRCTL, 11-3
  for NAMESCTL, 11-3
  rules for configuration files, C-2
System Network Architecture (SNA), 4-4

T

TAF
  See Transparent Application Failover (TAF)
TCP/IP protocol
  described, 4-2
  HOST parameter, B-4
  parameters for addresses, B-4
  PORT parameter, B-4
TCP/IP with SSL protocol
  described, 4-2
  HOST parameter, B-4
  parameters for addresses, B-4
  PORT parameter, B-4
TCP.NODELAY parameter, C-77
testing
  client configuration
    using applications, 11-11
    using OS, 11-11
    using special commands, 11-12
    with 3GL, 11-12
    with Net8 Assistant, 11-22
    with TCROUTE, 11-20
    with TNSPING, 11-17
  listener configuration, 11-15
  network connectivity, 11-17
  network objects, 11-14
  Oracle Connection Manager, 11-15
  Oracle Names servers, 11-14
  server configuration
    with Net8 Assistant, 11-15
  with control utilities, 11-3
tftp utility, 10-9
TICKS attribute, 9-7
TIMED_QUERY command, A-77
TIMEOUT parameter, 2-17, 7-17
timer initiated disconnect
  See dead connection detection
timestamping
  client trace files, C-28
  listener trace files, C-58
  Oracle Connection Manager trace files, C-75
  server trace files, C-28
Timout field in Net8 Assistant, 7-17

TNS
  benefits, 3-2
  described, 3-2
TNS Time Out Value option, 8-23
TNS_ADMIN environment variable, 12-4, 12-6
TNS_ADMIN parameter, 12-4, 12-6
TNSAPI.A file, 10-7
TNSAPI.DLL file, 10-7
TNSAPI.H file, 10-7
TNSAPI.LIB file, 10-7
tnsnames.ora file
  configuration parameter reference, C-31 to C-48
  configuring, 6-5
  exporting entries to directory server, 6-32
  parameters
    ADDRESS, C-38
    ADDRESS_LIST, C-38
    BACKUP, C-44
    BACKUP parameter, 8-13
    CONNECT_DATA, C-42
    DELAY, 8-13, C-44
    DESCRIPTION, C-37
    DESCRIPTION_LIST, C-37
    FAILOVER, 8-4, C-39
    FAILOVER_MODE, 8-13, C-43
    GLOBAL_NAME, 8-9, C-45
    HS, 8-9, C-45
    INSTANCE_NAME, 8-9, C-46, C-87
    LOAD_BALANCE, 8-4, C-40, C-87
    METHOD, 8-13, C-44
    RDB_DATABASE, 8-9, C-46
    RETRIES, 8-13, C-44
    SDU, 8-9, C-40
    SERVER, C-46
    SERVICE_NAME, C-47, C-88
    SID, C-48
    SOURCE_ROUTE, 8-4, C-41
    TYPE, 8-13, C-44
    TYPE_OF_SERVICE, 8-9, C-41
  release 8.1, changes in, C-86 to C-88
    FAILOVER parameter, C-83
    LOAD_BALANCE parameter, C-82
    nested addresses, C-84
    SERVICE_NAME parameter, C-81
    SOURCE_ROUTE parameter, C-83
TNSPING utility, 11-17
  compared to TRCROUTE utility, 11-20
TNSPING.TRACE_DIRECTORY parameter, 12-37, C-24
TNSPING.TRACE_LEVEL parameter, 12-37, C-25
TOTAL_REFUSED, A-94
TOTAL_RELAYS, A-94
TP_NAME parameter, B-3
TPN parameter, B-3
TRACE command, A-24
trace files
  cmadm_*pid*.trc, 12-40
  cmadm*pid*.trc, 12-40
  cman_*pid*.trc, 12-40
  cman*pid*.trc, 12-40
  cycling
    client, C-26
    listener, C-57
    Oracle Connection Manager, C-75
    server, C-27
  default names for, 12-34
  error message information, 12-45
  example, 12-55
  example of error messages, 12-45
  example of packet data, 12-44
  listener.trc, 12-38
  names*pid*.trc, 12-39
  names.trc, 12-39
  sqlnet*pid*.trc file, C-28
  sqlnet.trc, C-25
  sqlnet.trc for clients, 12-35
  svr_*pid*.trc for servers, 12-35, C-25
  timestamping
    client, C-28
    listener, C-58
    Oracle Connection Manager, C-75
    server, C-28
TRACE_DIRECTORY parameter, 12-38, 12-40, C-74
TRACE_DIRECTORY_CLIENT parameter, 12-35, C-25
TRACE_DIRECTORY_*listener_name* parameter, C-56
TRACE_DIRECTORY_SERVER parameter, 12-36, C-25

TRACE_FILE parameter, 12-38
TRACE_FILE_CLIENT parameter, 12-35, C-25
TRACE_FILE_*listener_name* parameter, C-57
TRACE_FILE_SERVER parameter, 12-36, C-25
TRACE_FILELEN parameter, C-74
TRACE_FILELEN_CLIENT parameter, C-26
TRACE_FILELEN_SERVER parameter, C-26
TRACE_FILEN_*listener_name* parameter, C-57
TRACE_FILENO_CLIENT parameter, C-26
TRACE_FILENO_*listener_name* parameter, C-57
TRACE_FILENO_SERVER parameter, C-27
TRACE_LEVEL parameter, 12-38
TRACE_LEVEL_CLIENT parameter, 12-35, C-27
TRACE_LEVEL_*listener_name* parameter, C-58
TRACE_LEVEL_SERVER parameter, 12-36, C-27
TRACE_TIMESTAMP parameter, C-75
TRACE_TIMESTAMP_CLIENT parameter, C-28
TRACE_TIMESTAMP_*listener_name* parameter, C-58
TRACE_TIMESTAMP_SERVER parameter, C-28
TRACE_UNIQUE_CLIENT parameter, C-28
tracing
  client, C-25
  cycling files
    client, C-26
    listener, C-57
    Oracle Connection Manager, C-75
    server, C-27
  server, C-25
  timestamping
    client trace files, C-28
    listener trace files, C-58
    Oracle Connection Manager trace files, C-75
    server trace files, C-28
TRACING parameter, 12-40, C-74
Transparent Application Failover (TAF)
  configuring, 8-12
  GLOBAL_DBNAME parameter in listener.ora file, 7-13, 8-14
  overview, 2-21
  parameters, C-43
Transparent Network Substrate (TNS)
  benefits, 3-2
  described, 3-2
transport layer, described, 3-4
TRCROUTE utility
  described, 11-20
  performance effects, 11-20
  requirements for, 11-20
  with earlier versions of SQL*Net, 11-20
troubleshooting, 12-1 to 12-59
  client, 12-4
  contacting Oracle Customer Support, 12-59
  log files, 12-16
  loopback tests, 12-4
  questions, 12-15
  server, 12-3
  trace files, 12-16
  using log file, 12-24
Turn Off UNIX Signal Handling option in Net8 Assistant, 8-23
Type of Service field in Net8 Assistant, 8-9, 8-52
TYPE parameter, 8-13, C-44
TYPE_OF_SERVICE parameter, 8-9, 8-52, C-41

U

UNIX
  disadvantages of signal handlers, 10-11
  Oracle's operating system dependent call for signal handling, 10-11
  using both signal handling and Net8, 10-11
  using signal handlers with, 10-10
UNREGISTER command, A-78
uppercase, defined, xxii
Use for Heterogeneous Services option in Net8 Assistant, 8-9, 8-51
USE_ASYNC_CALL parameter, C-75
USE_CMAN parameter, 8-21, 8-30, 8-33, 8-37, 8-38, C-29
USE_DEDICATED_SERVER parameter, 8-21, C-30
USE_PLUG_AND_PLAY_*listener_name* parameter, 6-64, C-58
user-initiated disconnect, 2-6

utilities
- control, 11-3
- finger, 10-9
- tftp, 10-9

V

V$SESSION table, 8-16
V1ADD.NPO.OMD record, 11-14, A-42
VERSION command
- of CMCTL, A-97
- of LSNRCTL, A-25
- of NAMESCTL, A-79

virtual circuits, 2-7

W

well-known Oracle Names servers, 4-22
Windows NT native authentication, 4-25

Y

ypserv program, 6-79