

SQL*Net for Alpha OpenVMS

Configuration and User's Guide

Release 8.0.5

October 1999

Part No. A76987-01

This manual describes how to configure and maintain SQL*Net on Alpha OpenVMS.

ORACLE®

SQL*Net for OpenVMS Configuration and User's Guide, Release 8.0.5

Part No. A76987-01

Release 8.0.5

Copyright © 1996, 1999 Oracle Corporation. All rights reserved.

Primary Author: Anjana Suparna Sriram

Contributing Authors: Matthew Vincent, DJCover, Gary Huffmann

Contributors: Thomas Leah-Martin, Pierre Krabbendam, John Sobecki and Nancy Wajda.

The Programs (which include both the software and documentation) contain proprietary information of Oracle Corporation; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent and other intellectual and industrial property law. Reverse engineering, disassembly or decompilation of the Programs is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. Oracle Corporation does not warrant that this document is error free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Oracle Corporation

If the Programs are delivered to the U.S. Government Agency or anyone licensing or using the Programs on behalf of the US Government, the following notice is applicable:

Restricted Rights Notice Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication and disclosure of the Programs shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are "restricted computer software" and use, duplication and disclosure of the Programs shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be licensee's responsibility to take all appropriate fail-safe, back up, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and Oracle disclaims liability for any damages caused by such use of the Programs.

Contents

Send Us Your Comments	vii
Preface.....	ix
Audience.....	x
How This Guide is Organized	x
Conventions Used in this Guide	xi
Syntax	xi
Related Publications	xiv
OpenVMS Publications.....	xiv
Product-Specific Publications	xiv
Customer Support Information	xv
 1 Introduction to SQL*Net on Alpha OpenVMS	
Configuration Overview	1-2
Centralized	1-3
Client/Server.....	1-4
Distributed Database.....	1-4
SQL*Net Installations.....	1-4
SQL*Net and the Transparent Network Substrate (TNS).....	1-5
Using SQL*Net.....	1-5
SQL*Net Architecture	1-7
 2 The Protocol Adapters	
Mailbox Adapter	2-2

Syntax	2-2
Example.....	2-3
TCP/IP Adapter	2-3
Syntax	2-3
Example.....	2-4
Bequeath Adapter	2-4
Bequeath Listener	2-5
Bequeath Listener Privileges.....	2-8

3 TNS Listener

Introduction	3-2
LSNRCTL	3-2
Listener Privileges	3-3
Process Quotas	3-4
ORASRV_NETV2 Command File	3-5
General Connections	3-6

4 Oracle Names

Introduction	4-2
NAMESCTL	4-2
Names Server Privileges	4-3
Requirements	4-3
SDNS.ORA.....	4-3
Optional Files	4-4

5 Oracle Intelligent Agent and SNMP Support

Introduction	5-2
Installing the Oracle Intelligent Agent	5-2
Oracle Intelligent Agent Setup and Discovery Option	5-2
Creating the Startup Scripts	5-3
Agent Parameter Files and Discovery Option.....	5-3
Setting the Preferred Credentials	5-6
Oracle Intelligent Agent Startup, Shutdown, and Status Query	5-6
Startup of the Agent	5-6

Shutdown of the Agent.....	5-7
Status Query of the Agent	5-7
Oracle Intelligent Agent Maintenance	5-7
SNMP Support	5-7

6 Advanced Networking Option

Documentation Set	6-2
Requirements.....	6-2
What's in this Release?.....	6-2
Installation Requirements	6-2
Installation	6-4
Installation Warning	6-4
Installation Tasks	6-5
De-Installation.....	6-7
De-Installation Warning	6-7
De-Installation Tasks.....	6-7
Usage Notes for the Authentication Adapters	6-9
General Information.....	6-9
SecurID	6-9
Kerberos5	6-10

A Major Changes and Enhancements

List of Changes.....	A-2
Enhancements.....	A-2

B ORASRV_NETV2 Command Procedure Sample File

C ORASRV_BEQ_<sid> Command Procedure Sample File

Index

Send Us Your Comments

SQL*Net for Alpha OpenVMS Configuration and User's Guide, Release 8.0.5

Part No. A76987-01

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:

- Email - infodec@us.oracle.com
- FAX - 1.650.506.7361 Attn: Technical Publications Manager
- Postal service:
Technical Publications Manager
Compaq Products Division
Oracle Corporation
500 Oracle Parkway, Mailstop 1op5
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 World Wide Support Center.

Preface

The *SQL*Net for Alpha OpenVMS Configuration and User's Guide, Release 8.0.5* describes how to configure and maintain SQL*Net on Alpha OpenVMS. The purpose of this guide is to explain how to configure and use SQL*Net 8.0.5 for Alpha OpenVMS, including the following components:

- TNS (Transparent Network Substrate) Listener
 - Oracle Names
 - Oracle Advanced Networking Option
 - Oracle Intelligent Agent and SNMP support
- The configuration and use of the protocols, TCP/IP and Alpha OpenVMS Mailboxes, are detailed in this guide.

Oracle8 and Oracle8 Enterprise Edition

Unless noted otherwise, features and functionality described in this document are common to both Oracle8 and Oracle8 Enterprise Edition.

The following topics are covered in this Preface:

- Audience
- How This Guide is Organized
- Conventions Used in this Guide
- Related Publications
- Customer Support Information

Audience

This guide is intended for users who configure and use SQL*Net for Alpha OpenVMS 8.0.5.

This guide assumes that the user has a fundamental knowledge of the Alpha OpenVMS operating system; it does not document any features of Alpha OpenVMS except when they affect or are affected by SQL*Net.

Only those SQL*Net features that are specific to the Alpha OpenVMS environment are documented in this guide. For general, non-platform-specific information about SQL*Net, refer to the documents listed in the section "Related Publications" later in this chapter.

How This Guide is Organized

This guide is divided into the following chapters and appendices:

Chapter 1, "Introduction to SQL*Net on Alpha OpenVMS"

This chapter provides general conceptual information about SQL*Net release 8.0.5 in the Alpha OpenVMS environment. It presents a configuration overview, describes the possible SQL*Net installations, and explains the role of the Transparent Network Substrate.

Chapter 2, "The Protocol Adapters"

This chapter provides information about the Mailbox, TCP/IP, and bequeath protocol adapters on Alpha OpenVMS.

Chapter 3, "TNS Listener"

This chapter provides information about the TNS Listener on Alpha OpenVMS.

Chapter 4, "Oracle Names"

This chapter provides Alpha OpenVMS-specific information about Oracle Names. It describes the requirements and tells how to start Oracle Names from the DCL prompt.

Chapter 5, "Oracle Intelligent Agent and SNMP Support"

This chapter provides Alpha OpenVMS-specific information about installing and running the Oracle Intelligent Agent.

Chapter 6, "Advanced Networking Option"

This chapter provides Alpha OpenVMS-specific information about installing the Advanced Networking Option (ANO).

Appendix A, "Major Changes and Enhancements"

This appendix lists the major changes from previous releases of SQL*Net to SQL*Net 8.0.5 or Net8.

Appendix B, "ORASRV_NETV2 Command Procedure Sample File"

This appendix shows the contents of the ORASRV_NETV2_COM.SAMPLE file.

Appendix C, "ORASRV_BEQ_<sid> Command Procedure Sample File"

This appendix shows the contents of the ORASRV_BEQ_COM.SAMPLE file.

Conventions Used in this Guide

This section explains the following:

- Syntax
- SQL*Net Terms Used in this Guide

Syntax

This guide uses the following conventions:

Convention	Meaning
Monospace text	Represents command line text. Type this text exactly as shown.
< >	Text enclosed in angle brackets represents a variable. Substitute an appropriate value for the variable text. Do not type the brackets.
[]	Square brackets enclose optional items or indicate a function key. Do not type the brackets.
	A vertical bar represents an “or” option among several options. You must enter only one of the options. Do not type the vertical bar.
/directory	A slash before a directory name indicates that it is a subdirectory. The path name may be either uppercase or lowercase.
\$	Represents the OpenVMS command prompt. Your prompt may differ.
\	In examples of commands you type online, a backward slash at the end of a line signifies that you must type the entire command on one line. Do not type the backslash.

Special notes alert you to particular information within the body of the manual.

Additional Information:	Refers you to portions of this manual, another manual, or the online documentation. All references to other manuals refer to the most recent version of that manual, unless otherwise noted.
Attention:	Alerts you to important information that will help you use the system.
Note:	Highlights helpful hints and practical tips that can save time and make installation or other procedures easier.
Warning:	Warns of actions which, if not carried out properly, could be damaging or destructive to your operations.

SQL*Net Terms Used in this Guide

Readers of this guide should be familiar with the following terms as they are used in Oracle networking documents.

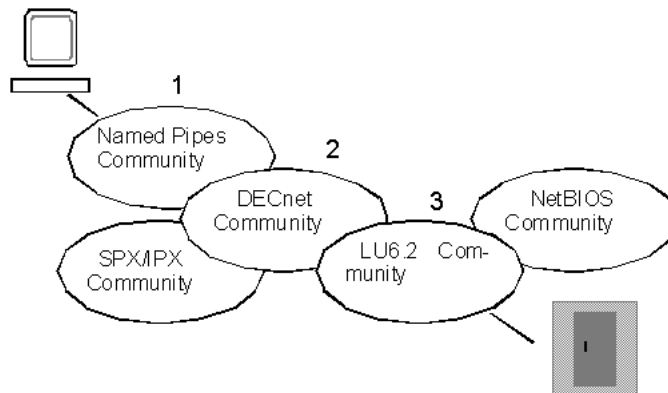
- **TNS-based product**—A product based on the Transparent Network Substrate (TNS) refers to software that directly calls or uses TNS. An example would be SQL*Net 8.0.5.
- **TNS community**—A TNS community corresponds to what is typically called a network. It consists of several clients or nodes that fulfill the following conditions:
 - Connected and can communicate using one common network transport protocol (regardless of media or distance)
 - Run TNS-based products, such as SQL*Net 8.0.5

Communities are given names, but because they are based on a single protocol they may also be referred to using phrases such as the TCP/IP community.

- **TNS network/application network**—A TNS network (also called an application network) is a network that consists of one or more TNS communities running TNS-based products. A simple example is one TNS community in which two nodes are running SQL*Net release 8.0.5.
- **TNS client**—A TNS client is an application that is a member of a TNS community and that is running TNS-based software such as SQL*Net release 8.0.5. A TNS client may be either a requestor or a provider of a service.
- **Initiator**—An initiator is a TNS client application that requests a connection to another client in a TNS network. Every TNS connection is originated by an

initiator. An example is an Oracle Forms application that accesses an Oracle8 database using SQL*Net release 8.0.5.

- **TNS connection**—A TNS connection is an application-level connection between two TNS clients on the same TNS network. The initiator and destination can belong to the same TNS community or to different TNS communities. A TNS connection between two clients in the same community requires neither navigation nor an Interchange. A TNS connection that spans multiple communities requires an Interchange to navigate and establish the connection path. The figure below shows a logical TNS connection that spans three different transport protocol communities.



- **Connect descriptor**—An initiator requests a TNS connection using a connect descriptor, which identifies the destination application and usually contains initial application connection information (such as a database system ID). Usually the connect descriptor is identified by a service name, a short alias for the connect descriptor. The connect descriptors and their corresponding service names are stored in the Oracle Names server for the network (if one exists) or in the TNSNAMES.ORA file.
- **Path**—A path is the series of nodes traversed by a connection, beginning with the start point (the initiator) to the end point (the destination).

Related Publications

OpenVMS Publications

- Oracle8 for Alpha OpenVMS Installation Guide
- Release notes associated with Oracle product on Alpha OpenVMS
- Oracle8 for Alpha OpenVMS Server and Tools Administrator's Guide

Product-Specific Publications

- Net8 Administrator's Guide
- SQL*Net V1 to SQL*Net V2 Migration Guide
- Oracle8 Server Concepts
- Oracle8 Server Administrator's Guide
- Oracle8 Server Tuning
- Oracle8 Server Application Developer's Guide
- Oracle8 Server SQL Reference
- Oracle8 Server Utilities
- PL/SQL User's Guide and Reference
- Oracle8 Server Distributed Systems I:Distributed Data
- Oracle8 Server Distributed Systems II:Replicated Data
- Oracle8 Parallel Server
- Product-specific README and DOC files in the product directories (for example, ORA_RDBMS:README.DOC)

Customer Support Information

(Please copy this page and distribute within your organization as necessary.)

For Oracle Support Services (OSS), contact your local number. (The hours are detailed in your support contract.)

Please prepare the following information before you call, using this page as a checklist:

- Your Customer Support Identification (CSI) number if applicable, or full contact details, including any special project information
- The complete release numbers of the Oracle8 Enterprise Server and associated products (for example, Oracle8 Enterprise Server release 8.0.5 or Oracle Forms release 4.5.6.3.2). _____
- The hardware type on which the problem occurs (for example, Compaq Alpha)

- The operating system name and release number (for example, Alpha OpenVMS 7.1)

- Details of error codes and associated descriptions. Please write these down as they occur, since they are critical in helping OSS to quickly resolve your problem. _____
- A full description of the issue, including:
- **What** - What happened? For example, the command used and result obtained.
- **When** - When did it happen? For example, time of day, or after a certain command, or after an O/S upgrade.
- **Where** - Where did it happen? For example, on a particular system or within a certain procedure or table.
- **Extent** - What is the extent of the problem? For example, production system unavailable, or moderate impact but increasing with time, or minimal impact and stable.

Note: Keep in mind what *did not* happen, as well as what *did* happen. This type of information can help OSS to more quickly resolve your problem.

- Keep copies of any trace files, core dumps, and redo log files recorded at or near the time of the incident, since OSS will need these to further investigate your problem.

For installation-related problems please have the following information available:

- Error returned by the installation procedure and/or Alpha OpenVMS _____

Introduction to SQL*Net on Alpha OpenVMS

This chapter provides general conceptual information about SQL*Net 8.0.5 in the Alpha OpenVMS environment. It covers the following topics:

- Configuration Overview
- SQL*Net Installations
- SQL*Net and the Transparent Network Substrate (TNS)

Configuration Overview

SQL*Net is a communications software product that allows you to create a data management environment to share information stored in Oracle databases. SQL*Net uses the communications protocols supported by various operating systems to provide a distributed processing and distributed database environment for Oracle. SQL*Net also refers to a set of products or adapters that support industry-standard protocols such as TCP/IP.

An Oracle database management system can be configured in one of the following ways:

- Centralized
- Client/Server
- Distributed Database

In the figure below Ethernet is an example only. SQL*Net works with other network types.

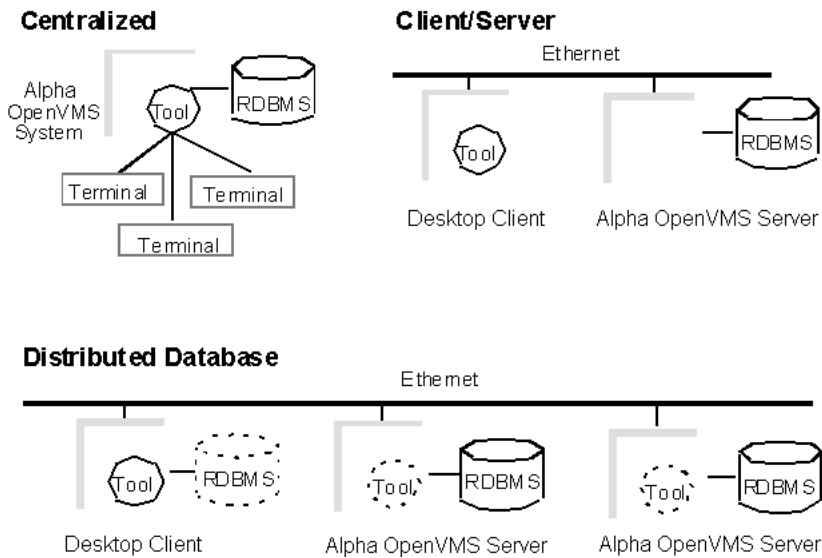


Figure 1-1 Configuration Options

Centralized

In a centralized configuration, the Oracle8 Enterprise Server and Oracle tool are located on the same machine. This machine is not necessarily on a network and users access the application through terminals. If you use a centralized configuration, you may use a simple SQL*Net adapter called the *bequeath adapter*, which requires no SQL*Net configuration. However, if you wish to use pre-spawned servers or multithreaded servers, you must configure SQL*Net even in centralized configurations.

Client/Server

In a client/server configuration, the Oracle8 Enterprise Server resides on a multi-tasking server system, and the client side of the applications resides on another computer, such as a workstation or personal computer. Both the client and server are connected by a physical network and communicate via a network protocol such as TCP/IP. In a client/server environment, the Oracle application built with an application development tool makes database requests to the server over the network.

Distributed Database

In a distributed database configuration, users query separate databases as a single database. The major advantage of a distributed database is that users and applications are not required to know where data resides. You can query database tables by name, regardless of how the network protocols work together to access the appropriate remote database containing the table. Therefore, SQL*Net users can communicate and share database information stored in different locations, on different computers, with different operating systems. Distributed databases allow local administration of data and can reduce network traffic if the data that is accessed most often at a location can be stored locally.

SQL*Net allows the client and server to communicate over a variety of media and protocols. A client/server configuration allows DBAs to distribute CPU-intensive user interfaces to low-cost workstations. It also allows application users to be greeted with the graphical user interface (GUI) with which they are most familiar.

SQL*Net Installations

When installing SQL*Net on Alpha OpenVMS, you can choose to install the SQL*Net TCP/IP adapter.

In addition, the Alpha OpenVMS Mailbox adapter is installed automatically, as is the bequeath adapter, which allows mailbox connections without a network configuration or running listener.

See the *Oracle8 for Alpha OpenVMS Installation Guide* for instructions on installing SQL*Net. Also see the file `ORA_RDBMS:READMEVMS.DOC` and `ORA_NETCONFIG:README_NETCONFIG.DOC`.

SQL*Net and the Transparent Network Substrate (TNS)

This section introduces SQL*Net in general terms and describes the components that make up SQL*Net for Alpha OpenVMS release 8.0.5.

Using SQL*Net

SQL*Net connects dissimilar networks together and allows client/server transactions to occur transparently. An end user does not have to know that a network exists, because SQL*Net hides the complexity of machine-level interactions by presenting a layer of interconnectivity to the user through its client/server architecture. This layer is called the Transparent Network Substrate, or TNS.

Figure 1-2 shows a client/server configuration using SQL*Net.

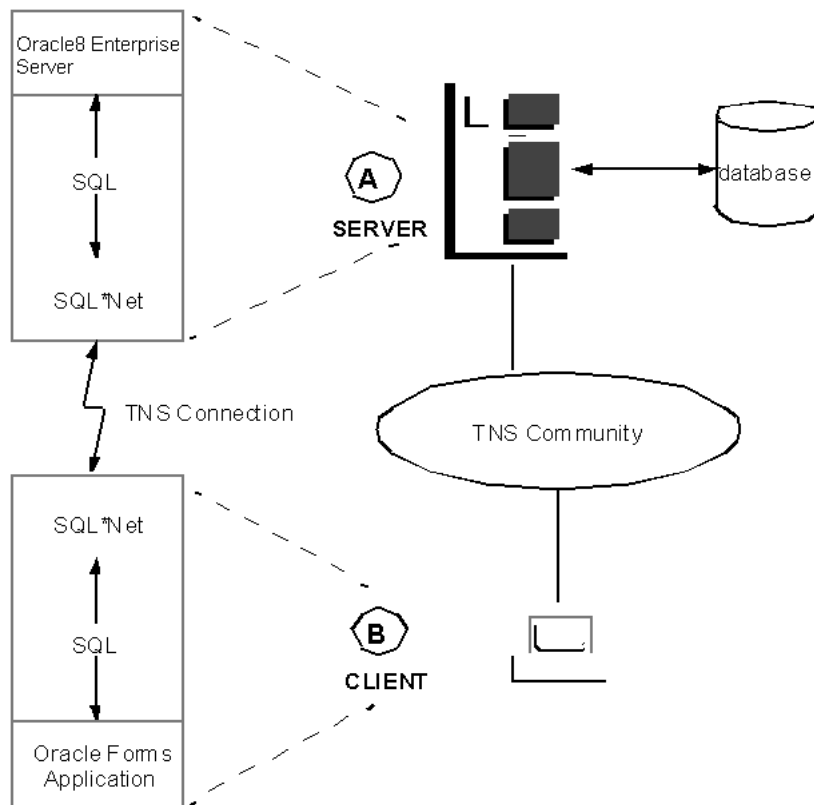


Figure 1-2 Client/Server Configuration

Figure 1-2 shows an Alpha OpenVMS computer that holds the physical Oracle8 database and a client workstation with an Oracle Forms application that needs to access the Oracle8 database. The Alpha OpenVMS computer (A in the figure) is the server and the workstation (B in the figure) is the client.

The transaction proceeds as follows:

1. The client requests some data.
2. SQL*Net packages the request and sends it to the Transparent Network Substrate (TNS).
3. TNS routes the packaged request to the server.

4. SQL*Net on the server side unpackages the request and sends it to Oracle8.
5. Oracle8 processes the request and sends the requested data to SQL*Net.
6. SQL*Net packages the data and sends it to TNS.
7. TNS routes the data to the client.
8. SQL*Net on the client side unpackages the data and sends it to the application.

SQL*Net Architecture

SQL*Net consists of the following components:

- SQL*Net Interface
- Transparent Network Substrate
- Oracle Protocol Adapters

SQL*Net Interface

The SQL*Net interface bundles or unbundles messages received from TNS. The SQL*Net interface code resides on all nodes that use SQL*Net. On the client (application program) side, the interface bundles the messages received from the application and passes them to TNS for delivery. On the Oracle8 Enterprise Server side, the interface unbundles the messages received from TNS and passes them to the Oracle8 Enterprise Server.

Transparent Network Substrate

TNS allows peer-to-peer connectivity where no machine-level connectivity can occur. It provides a user-transparent layer that enables a heterogeneous network consisting of different protocols to function as a homogeneous network. TNS forms a transparent layer to which different network protocols are connected. It provides a network of applications above the existing networks of computers.

Oracle Protocol Adapters

The Oracle Protocol Adapters allow TNS and its services to communicate over existing network *communication protocols*. The Protocol Adapters map the functions of the underlying protocol into the equivalent functions within TNS. This mapping of communication functions allows calls to or from TNS to be protocol nonspecific.

Figure 1-3 shows how TNS and the Oracle Protocol Adapters interface with existing network protocols. For any TNS client running an industry-standard protocol, the Oracle Protocol Adapter interfaces between the unique API of the underlying protocol and the consistent interface of Oracle's TNS.

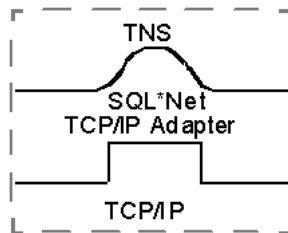


Figure 1–3 TNS, Adapters, and Protocols

A single machine can support multiple protocols and protocol adapters simultaneously. A node that supports multiple protocols and protocol adapters is said to be a member of multiple TNS communities, one for each protocol installed.

A TNS client belonging to multiple communities is common in two cases:

- As a client application that needs to access other applications in more than one network. Installing two protocols and protocol adapters allows a client to connect to any server application in either community.
- As a server application that is being accessed by clients from multiple TNS communities. Installing two protocols and the protocol adapters would allow all clients from both communities to access a server application on that machine.

For more information about SQL*Net, see the following manuals:

- *Oracle Network Products Messages Manual*
- *Oracle Network Manager Administrator's Guide*
- *Understanding SQL*Net*
- *Oracle Names Administrator's Guide*
- *Oracle SNMP Support Reference Guide*
- *Oracle Advanced Networking Option Administrator's Guide*

The Protocol Adapters

This chapter gives information about the following protocol adapters on Alpha OpenVMS:

- Mailbox Adapter
- TCP/IP Adapter
- Bequeath Adapter
- Bequeath Listener

Note: This chapter is a supplement to the *Net8 Administrator's Guide* guide.

Mailbox Adapter

The Mailbox protocol adapter, or IPC adapter, is automatically configured for use when you install SQL*Net. It can be used for client/server connections when both client and server are on the same Alpha OpenVMS node. If the client and server are on different machines, then the connection must take place using TCP/IP.

When configuring the TNS listener to listen for mailbox connections, you need to specify a **KEY** value in LISTENER.ORA for the IPC protocol. The listener then creates a mailbox which listens for connections and creates a system-wide logical name (the same as the **KEY** value) which translates to this mailbox device. It is via this logical name that clients find the listener's mailbox.

When the SQL*Net parameter **AUTOMATIC_IPC** is not turned off, connect descriptors specifying the TCP/IP protocols for a server on the same node are converted to mailbox connections. If you wish to prevent this, you must include the following line in the TNS_ADMIN:SQLNET.ORA file:

```
AUTOMATIC_IPC = OFF
```

Syntax

The following fields must be defined:

```
(PROTOCOL=IPC)
(KEY=<IPC logical name>)
```

where:

PROTOCOL

The keyword that identifies the specific protocol adapter used; for this protocol, the value is **IPC**. The value can be entered in either uppercase or lowercase.

KEY

The logical name used to connect to the listener via the Mailbox adapter.

Example

This example shows the two fields for the Alpha OpenVMS Mailbox adapter.

```
(PROTOCOL=IPC)
(KEY=ORA_IPC)
```

Note: A full example of a Mailbox connect descriptor can be found in the file TNS_ADMIN:LISTENER_ORA.SAMPLE.

TCP/IP Adapter

The TCP/IP protocol adapter provides support for client/server connections using TCP/IP as a protocol. You can turn SQL*Net support for TCP/IP on or off via the NetConfig configuration screen (please refer to the *Oracle8 for Alpha OpenVMS Installation Guide*).

SQL*Net 8.0.5 on Alpha OpenVMS is developed and certified using Compaq's TCP/IP Services for OpenVMS (UCX). If you wish to use the TCP/IP protocol adapter for SQL*Net, you should have Version 4.2 or higher of TCP/IP Services for Alpha OpenVMS installed. TCP/IP protocol stacks from other vendors may work with Oracle, but customers use these products at their own risk. Any TCP/IP problems that can not be reproduced using TCP/IP Services for Alpha OpenVMS will simply be referred to the TCP/IP vendor.

Syntax

The following fields must be defined:

```
(PROTOCOL=TCP)
(HOST=hostname)
(PORT=port#)
```

The following field is optional:

```
(QUEUE SIZE=n)
```

where:

PROTOCOL

Keyword that identifies the specific protocol adapter used; for this protocol, the value is TCP. The value can be entered in either uppercase or lowercase.

HOST

Host name or IP address.

PORT#

TCP/IP port number.

QUEUESIZE

Parameter to increase the queue size. This parameter is optional; if it is not specified, the default value of 20 is used. If simultaneous connections are made to the listener, some connection requests may not be received if the listener socket queue size is too small.

Example

In this example, the TCP/IP connect descriptor specifies a listener on the ALPHA1 host.

```
(PROTOCOL=TCP)
(HOST=ALPHA1)
(PORT=1526)
```

Note: A full example of a TCP/IP connect descriptor can be found in the file TNS_ADMIN:TNSNAMES_ORA.SAMPLE.

Bequeath Adapter

Each database that you wish to connect to with the bequeath adapter must have a command file named ORASRV_BEQ_<sid>.COM in ORA_ROOT:[NETWORK.ADMIN]. For databases created with the Oracle7 RDBMS release 7.3.2 or later, this command file is generated when you create the database. You must create this command procedure manually for pre-existing databases. Edit the ORA_ROOT:[NETWORK.ADMIN]ORASRV_BEQ_COM.SAMPLE file or execute the command procedure ORA_NETWORK:CREATE_ORASRV_BEQ.COM as follows:

```
$ @ORA_NETWORK:CREATE_ORASRV_BEQ <ora_db> <sid> <dbname>
```

where:

<ora_db> is the database administration directory;

<sid> is the SID of the database, and

`<dbname>` is the NAME of the database.

For example:

```
$ @ORA_NETWORK:CREATE_ORASRV_BEQ DKA400:[ORACLE805.DB_PROD] -  
    PROD PRODDB
```

You can find a sample ORASRV_BEQ_COM.SAMPLE file in Appendix C of this manual.

Bequeath Listener

The Bequeath Listener, running as a detached process, creates detached server processes to service clients on the same machine, using the bequeath adapter. This allows the Oracle server to run in a suitably privileged process. The alternative would be to have the server installed with privileges and run in a subprocess of the client. However, that would require the server to be linked without traceback information, making server trace information unusable if problems are encountered.

For each request from the client, the Bequeath Listener creates a detached server process and two mailboxes. It then sends the mailbox names to the client and the client establishes a connection to the server using these mailboxes.

By default, these mailboxes are created with a buffer quota of 8192 bytes and a maximum message size of 2048 bytes. You can change these parameters by defining logical names with other values. For example:

```
$ define ORA_BEQ_MBXSIZE n  
$ define ORA_BEQ_MBXBFQ n
```

The maximum value for the mailbox buffer quotas is 60000 bytes. You should adjust these values carefully, and you should adjust them for performance reasons only.

The Bequeath Listener uses a known mailbox name to listen for client requests. This mailbox name is in the format:

```
ORA_BEQ_READ_MBX_XXXXXXXXXX_n
```

where:

`XXXXXXXXXX` is the Oracle image ID unique to the system (padded with zeroes).

`n` is a single-digit number (0-9) that is the Bequeath Listener number.

Starting up the Bequeath Listener

The Bequeath Listener starts automatically when INSORACLE is invoked (at installation time or later, usually during system startup). Unless you decide to invoke the REMORACLE command, the Bequeath Listener should be up and running all the time.

If the Bequeath Listener is down and you want to start it, execute the command:

```
BEQLSNR START
```

Bequeath Listener Status

You can issue a status command to determine whether the Bequeath Listener is up and running. Issue the command:

```
BEQLSNR STATUS [n]
```

If you do not provide the optional numeric parameter, then Bequeath Listener 0 is queried. To query Bequeath Listeners 1 through 9, if they exist, supply the number on the command line.

Shutting Down the Bequeath Listener

To stop the Bequeath Listener issue the command:

```
BEQLSNR STOP [n]
```

If you do not provide the optional numeric parameter, then all Bequeath Listeners for the installation are stopped. To stop a particular Bequeath Listener, provide its number in the command line.

Problem Resolution

Writing trace information

The Bequeath Listener writes some trace information, but because the output of the detached processes is set to the null device (NL:), normally you will not see it.

To get the trace information from the Bequeath Listener, you should do the following:

1. Stop the Bequeath Listener.
2. Edit the STARTUP_BEQLSNR.COM.
3. Change the NL: to a file name.

4. Restart the Bequeath Listener.

Changing the quota for a Server Process that is created by the Bequeath Listener

To change the quota, modify the file BEQLSNR.COM and remove the comments for the quota parameter that you want to change. Be sure to STOP/START the Bequeath Listener after modifying this file.

For all ORA-12203 Problems

Be sure that the image identifier string is present in the ORA_BEQ_READ_MBX system logical name. It must be the same as the equivalence - name for the ORA_BEQ process logical.

To verify this, issue the command:

```
$ show logical *beq*
```

The results displayed will look similar to the following:

```
(LNM$PROCESS_TABLE)
      "ORA_BEQ" = "V805100000"
(LNM$SYSTEM_TABLE)
      "ORA_BEQ_READ_MBX_V805100000_0" = "MBA6839:"
```

Problem: ORA-12203: TNS:unable to connect to destination

If you experience this problem, issue the command BEQLSNR STATUS to determine whether the Bequeath Listener is up and running. If the Bequeath Listener does not respond, use the command BEQLSNR STOP to stop the Bequeath Listener and use the command BEQLSNR START to restart it.

Client Problem: ORA-12203: TNS:unable to connect to destination

Choose one of the following solutions:

- Change the logical ORA_BEQ_TIMEOUT to something greater than 120 seconds (for example: 300 seconds). Before running the client program, define this logical also in the ORA_NETWORK:BEQLSNR.COM file

or

- Define the logical ORA_BEQ_NUM_OF_LISTENERS to a value between 1 and 10 to increase the capacity when a number of clients are connecting at the same time to the Bequeath Listener.

With this method, you can increase the number of connections that the Bequeath Listeners can handle at one time. Each time that a client requests a connection, it will randomly pick one of the Bequeath Listeners that are running to serve it with the connection request. Note that you do not need to STOP/START the Bequeath Listener after defining this logical name. This logical name determines the number of Bequeath Listeners.

Bequeath Listener Privileges

The process in which the Bequeath Listener runs must have the OpenVMS privileges in the table below to be able to perform the associated function:

Table 2–1 Privileges and Their Functions

Privilege	Function
CMKRNL	Pass this privilege to server processes that the Listener creates.
DETACH	Create detached processes.
LOG_IO	Perform certain I/O functions.
PRMMBX	Create a permanent mailbox on which to listen. (The mailbox is permanent so that the logical name associated with it goes into the SYSTEM logical name table.)
SYSLCK	Lock system wide resources.
SYSNAM	Create SYSTEM logical names and shared logical name tables.
SYSPRV	May access objects via system protection.
TMPMBX	Create temporary mailboxes.
WORLD	Allow the Listener to get information about and to control processes that it may not have created, such as dispatchers and shared server processes.

Note: Before attempting to start the Bequeath Listener, the process that starts the Bequeath Listener must have the privileges in this table or be able to have them set.

TNS Listener

This chapter gives information about the TNS Listener on Alpha OpenVMS. It covers the following topics:

- Introduction
- LSNRCTL
- Listener Privileges
- Process Quotas
- ORASRV_NETV2 Command File
- General Connections

Introduction

The function of the TNS Listener is to receive connection requests from local or remote clients and to provide the client with a Server process to which to connect. The Listener can service multiple instances. For each instance, the Listener keeps a list of services that provide access to that instance. If multithreaded servers are being used, the Listener may direct a client connection to a dispatcher. Otherwise, for dedicated servers, the Listener will direct the client connection to an existing prespawnd server or will create a new server process to service the connection.

General information about the Listener and its configuration can be found in the generic SQL*Net documentation. This chapter provides only information about the Listener that is specific to Alpha OpenVMS.

LSNRCTL

The LSNRCTL utility is used to start and stop the Listener and to query its status or services. The LSNRCTL command executes the command procedure `ORA_NETCONFIG:LSNRCTL.COM`, which provides a shell to the executable program `ORA_NETCONFIG:LSNRCTL.EXE`.

The main function of the command procedure is to check that the privileges required to start the Listener are present (see the section "Listener Privileges"). If a LSNRCTL START command is entered and the required privileges are not present, an error is displayed and LSNRCTL exits.

Note: Start the listener using the Oracle account.

Warning: If you enter the LSNRCTL interactive mode by giving the LSNRCTL command without a subcommand and you have received a warning about inadequate privileges, **DO NOT attempt to start the Listener. The Listener process may still start, depending on the privileges you do have, but it may not function properly.**

Warning: Also, DO NOT start the Listener from a process that has a UIC in the system group, for example a group less than or equal to MAXSYSGROUP. If you enter a LSNRCTL START command from such a process, an error is displayed and LSNRCTL exits. If you enter a LSNRCTL command with no arguments, you are warned not to start the Listener from within the LSNRCTL utility. If the Listener did run in a system group, any Server processes it creates would be in the system group and the Server would abort, because it does not allow itself to run in privileged groups.

Listener Privileges

The process in which the Listener runs must have the Alpha OpenVMS privileges in the table below to be able to perform the associated function.

Table 3–1 Privileges and Their Functions

Privilege	Function
CMKRNL	Pass this privilege to server processes that the Listener creates.
DETACH	Create detached processes.
LOG_IO	Perform certain I/O functions.
PRMMBX	Create a permanent mailbox on which to listen. (The mailbox is permanent so that the logical name associated with it goes into the SYSTEM logical name table.)
SYSLCK	Lock system wide resources.
SYSNAM	Create SYSTEM logical names and shared logical name tables.
TMPMBX	Create temporary mailboxes.
WORLD	Allow the Listener to get information about and to control processes that it may not have created, such as dispatchers and shared server processes.

Note: Before attempting to start the Listener, the process that starts the Listener must have the privileges in this table or be able to have them set. As noted above, the LSNRCTL command file will attempt to set these privileges and warn the user if it was unable to do so.

Process Quotas

Process quotas for the Listener and for the Server processes which the Listener creates can be controlled by logical names. The logical names are:

`ORA_LSNR_<quotaname>`

where:

`<quotaname>` is ASTLM, BIIOLM, BYTLM, CPULM, DIOLM, FILLM, PGFLQUOTA, PRCLM, TQELM, WSQUOTA, WSDEFAULT, ENQLM, WSEXTENT, or JTQUOTA.

Several of the logical names are defined in `LSNRCTL.COM` and control the quotas of the Listener process. They are defined in user mode so that they are not present after exiting `LSNRCTL`. If your Listener supports an especially large number of services, some of these quotas may need to be increased. For the quotas you determine to be deficient or under direction of Oracle Support, you can edit the quota values in `LSNRCTL.COM`.

To control the quotas of the processes that the Listener creates, specify the logical names in the file `ORA_NETWORK:TNSLSNR.COM`, the command file that runs in the Listener process. Statements to define these logical names are in `TNSLSNR.COM`, but are commented out.

If, for example, a very large file backed SGA requires that Server processes have larger quotas, you can uncomment the appropriate logical name definition in `TNSLSNR.COM` and specify the quota value. Starting with release 7.3.2.3.2 and the VLM feature, a file backed SGA is created when the `INIT.ORA` parameter `VLM_BACKING_STORAGE_FILE` is set to `TRUE`.

Quotas can also be specified for the Server processes in the `LISTENER.ORA` file on a SID-by-SID basis. This is done in the `SID_DESC` section for a Listener. For example:

```
SID_LIST_LISTENER =
  (SID_DESC =
    (SID_NAME = <name>)
    (PROGRAM = <disk:>[<directory>]ORASRV_NETV2.COM)
    (OSDS=
      (PRIORITY=<number>)
      (QUOTA=
        (ASTLM=<number>)
        (BYTLM=<number>)
        (PGFLQUOTA=<number>)
      )
    )
  )
```

)

There are no restrictions on the number of quotas that you can specify in the QUOTA list. However, if any quota is specified in the QUOTA list, then none of the quotas specified by logical name will be used and quotas that are not specified in the list will assume the system default.

Warning: The process priority of the Server process can also be specified, as in the example above, but this is not recommended.

ORASRV_NETV2 Command File

In the LISTENER.ORA file, you provide a SID_LIST for each defined Listener, and for each SID you provide a program name. The program name is the full physical path, without Oracle logical names, to a command file, based on the file TNS_ADMIN:ORASRV_NETV2_COM.SAMPLE.

When you edit the file TNS_ADMIN:ORASRV_NETV2_COM.SAMPLE for a particular SID, it may have any name. The recommendation is to name it:

TNS_ADMIN:ORASRV_NETV2_<sid>.COM

For example:

```
(SID_LIST_LISTENER =  
  (SID_DESC =  
    (SID_NAME = PROD)  
    (PROGRAM = DK: [O805.NETWORK.ADMIN] ORASRV_NETV2_PROD.COM)  
  )  
)
```

When the Listener starts a prespawnd or dedicated server process, the process runs this command file. To create this file for a particular SID, change the following three lines in the sample command file:

```
$ ora_db = "database administration directory"  
$ sid    = "sid name"  
$ dbname = "database name"
```

where:

“database administration directory” is the directory that contains the startup, shutdown, and orauser command files for the database. Specify a full directory path, without using Oracle logical names.

“sid name” and “database name” are the names provided for the database when it was created.

You can find a sample command file in Appendix B of this document.

General Connections

Make sure that your SQL*Net task file defines any logical names used by the INIT.ORA parameters USER_DUMP_DEST and BACKGROUND_DUMP_DEST (if defined).

If you define these logical names by calling ORA_DB:ORAUSER_<database_name>.COM, make sure that ORA_UTIL:ORAUSER.COM runs RDBMSUSER.COM **AFTER** PROGINTUSER.COM.

PROGINTUSER.COM also defines ORA_SLAX, making it into a search list with ORA_PROGINT_MESG and ORA_RDBMS.

Oracle Names

This chapter provides information about Oracle Names on Alpha OpenVMS. It covers the following topics:

- Introduction
- NAMESCTL
- Names Server Privileges
- Requirements

Note: This chapter assumes that Oracle Names and all related SQL*Net products have been installed at your site.

Introduction

The function of the Names Server is to resolve connection addresses in a homogeneous and centralized location. As a client issues a connection request, the Names Server is responsible for directing the client connection request to the appropriate listener for the specified SID. TNSNAMES.ORA can also resolve the listener address. However, the benefits of the centralized list of connection addresses that Oracle Names provides greatly eases the maintenance of large network definitions.

NAMESCTL

The NAMESCTL utility is used to start and stop the Names Server and to query its status or services. The NAMESCTL command executes the command procedure `ORA_NETCONFIG:NAMESCTL.COM`, which provides a shell to the executable program `ORA_NETCONFIG:NAMESCTL.EXE`.

The main function of the command procedure `ORA_NETCONFIG:NAMESCTL.COM` is to check that the privileges required to start the Names Server are present (see the section "Names Server Privileges"). If a `NAMESCTL START` command is entered and the required privileges are not present, an error is displayed and NAMESCTL exits.

Note: Start the Names Server using the Oracle account.

Warning: If you enter the NAMESCTL interactive mode by giving the NAMESCTL command without a subcommand and you have received a warning about inadequate privileges, DO NOT attempt to start the Names Server. The Names Server process may still start, depending on the privileges you do have, but it may not function properly.

Warning: DO NOT start the Names Server from a process that has a UIC in the system group, for example, a group less than or equal to MAXSYSGROUP. If you give a NAMESCTL START command from such a process, an error is displayed and NAMESCTL exits. If you enter a NAMESCTL command with no arguments, you are warned not to start the Names Server from within the NAMESCTL utility.

Names Server Privileges

The process in which the Names Server runs must have the Alpha OpenVMS privileges in the table below to be able to perform the associated function.

Table 4–1 Privileges and Their Functions

Privilege	Function
CMKRNL	Facilitate kernel mode processing
PRMMBX	Create a permanent mailbox on which to listen (The mailbox is permanent so that the logical name associated with it goes into the SYSTEM logical name table.)
SYSNAM	Create SYSTEM logical names and shared logical name tables
TMPMBX	Create temporary mailboxes

Note: Before attempting to start the Names Server, the process that starts the Names Server must have the privileges in this table or be able to have them set. As noted above, the NAMESCTL command file will attempt to set these privileges and warn the user if it was unable to do so.

Requirements

Before you can use Oracle Names, the following configuration files must be in the ORA.ROOT:[NETWORK.NAMES] directory of the system that calls Oracle Names:

- SDNS.ORA

SDNS.ORA

The SDNS.ORA file contains a list of preferred names servers. This file is updated to reflect the names and addresses of all the known servers that Names Server is

running. The initial startup requires that SDNS.ORA identifies, at a minimum, the specifics for its own address. For example, to start a Names Server to set up at port 1575 the SNDS.ORA file would contain the following:

```
NAMES.PREFERRED_SERVERS =  
  (ADDRESS_LIST =  
    (ADDRESS =  
      (PROTOCOL = TCP)  
      (HOST = avms04)  
      (Port = 1575)  
    )  
  )  
)
```

The easiest way to facilitate this is to simply modify the provided SDNS_ORA.SAMPLE file to suit your purposes. Additionally, the NAMESCTL REORDER_NS will generate the SDNS.ORA file.

Optional Files

The Names Server environment can be more tightly controlled through the following (optional) files in the TNS_ADMIN directory:

- SQLNET.ORA
- NAMES.ORA

SQLNET.ORA

The SQLNET.ORA file is read by both the client, as part of a connection request, and by NAMESCTL, for all operations other than server startup. This file identifies the Names service to be used to resolve the connection. Within SQLNET.ORA, the NAMES.PREFERRED_SERVERS list specifies to the client the address of the Names Server with which to connect.

Note: The presence of the NAMES.PREFERRED_SERVERS clause in SQLNET.ORA supersedes the presence of TNSNAMES.ORA in TNS_ADMIN. Therefore, if the use of a Names Server has been specified with SQLNET.ORA, then even if TNSNAMES.ORA exists and has a complete list of connection addresses, the client will not read TNSNAMES.ORA.

NAMES.ORA

The NAMES.ORA file describes the Names Server and is read by NAMESCTL.EXE at Server startup. NAMES.ORA identifies, among other things:

- Address of the running server
- Trace information
- Protocols serviced
- Indication of whether the network definition is stored in file format or in a database
- SID of the Names database (if applicable)

Oracle Intelligent Agent and SNMP Support

This chapter provides information about installing and running the Oracle Intelligent Agent (hereafter referred to as “the Agent”) on Alpha OpenVMS. Read this chapter carefully and completely before starting to install and use the Agent and SNMP Support on Alpha OpenVMS.

This chapter covers the following topics:

- Introduction
- Installing the Oracle Intelligent Agent
- Oracle Intelligent Agent Setup and Discovery Option
- Oracle Intelligent Agent Startup, Shutdown, and Status Query
- Oracle Intelligent Agent Maintenance

Introduction

The Agent is a backend server process that communicates with the Oracle Enterprise Manager (OEM) running on a Windows 95 or Windows NT machine.

Installing the Oracle Intelligent Agent

The Agent requires that a supported TCP/IP implementation be installed on your Alpha OpenVMS system. In addition, you must enable TCP/IP support for SQL*Net in the NetConfig configuration screen.

For more information, see Chapter 3, "TNS Listener".

The Agent may be installed at the same time as other products or it may be installed later.

Installation of the Agent creates the directory ORA_ROOT:[OEMAGENT] as an installation directory. It also creates a directory structure under the network subdirectory ORA_ROOT:[NETWORK.AGENT...] where most of the Agent files will reside.

If you are using the same Oracle Installation from more than one node in a Alpha OpenVMS cluster, you can only run the agent from this installation on one of the nodes. If you attempt to run the agent on multiple nodes from the same installation, there will be file name and file usage conflicts. This is a generic limitation of the Oracle Intelligent Agent on Clusters for all platforms.

For each additional node on which you wish to run the Agent, you must perform a client-only installation for the Agent (installing AGENT, NETCONFIG, and UTIL) and run the Agent from this client-only installation.

Oracle Intelligent Agent Setup and Discovery Option

To correctly set up the Agent environment, the following two kinds of files need to be created:

- Startup command procedures
- Parameter (*.ORA) files

Creating the Startup Scripts

Once the Agent has been successfully installed, create the following three files:

- `ORA_ROOT:[NETWORK.AGENT]AGENT_START.COM`
- `ORA_ROOT:[NETWORK.AGENT]DBSNMPC.COM`
- `ORA_ROOT:[NETWORK.AGENT]DBSNMPI.COM`

To create these three files, use the sample files `AGENT_START_COM.SAMPLE`, `DBSNMPC_COM.SAMPLE`, and `DBSNMPI_COM.SAMPLE` provided in `ORA_ROOT:[NETWORK.AGENT]`. Correctly fill in the values for `ORA_ROOT` in these three files.

When you startup the Agent, `AGENT_START.COM` is run as a detached process. `DBSNMPC.COM` is invoked, again as a detached process, whenever the OEM console requests a connection on the address specified by the `dbsnmp.spawn_address` parameter in `SNMP_RW.ORA`.

Agent Parameter Files and Discovery Option

At startup, and when requested by the OEM console thereafter, the agent runs a `Tcl` script called `NMICONF.TCL`, which resides in the `ORA_ROOT:[NETWORK.AGENT.CONFIG]` directory. This script starts by reading `ORA_RDBMS:ORA_RDBMS_SIDS.DAT` to discover any locally installed Oracle databases and instances. Then it reads an optional `ORATAB.ORA` file, if present, in the `TNS_ADMIN` directory.

The `ORATAB.ORA` file should be as follows:

```
<sid_name>,<network admin directory>
```

For example:

```
ORA805,disk$dl:[ORACLE8.NETWORK.ADMIN]
```

Note that you can specify any SID that you want the Agent to monitor that exists on this node.

Then, for each database instance found in `ORATAB.ORA`, the `tnsnames` list is searched for an address on the local host with the appropriate SID in the `CONNECT_DATA`. The key corresponding to the first matching address in the list becomes the name of the database. The `listener.ora` found in those same directories

is searched for the SID of the database. Again, the first listener that matches our SID becomes the listener active for that database.

Note: This generic discovery phase is impossible if the local database names are stored in Oracle Names instead of in a local TNSNAMES.ORA file. We cannot do the backwards SID-to-name matching through Names. As a result, if Oracle Names is in use for the host, an old-style `SNMP.ORA` must still be in `TNS_ADMIN`, with the parameter `nmi.register_with_names` set to `FALSE`. If this flag is detected at start-up, none of the generic discovery occurs. Instead, the information in the old-style `snmp.ora` is used to construct the new configuration files.

The configuration files `SNMP_RO.ORA` and `SNMP_RW.ORA` are created.

The file `SNMP_RO.ORA` should reside at `TNS_ADMIN`, the same location as the `TNS` config files. This file contains lines from the old `SNMP.ORA` that should never be touched by the user:

```
SNMP.VISIBLESERVICES=(SERVICE-1,...SERVICE-N)
SNMP.SID.<service name>=<SID for the database>
SNMP.ORACLEHOME.<service name>=<oracle_home>
```

The file `SNMP_RW.ORA` should reside at `TNS_ADMIN`, the same location as the `TNS` config files. This file contains the following lines from the old `SNMP.ORA` that are automatically generated, but the user may want to modify or add to them:

```
SNMP.INDEX.<service name>=<integer index>
```

Note: The integer must be unique on the host.

```
SNMP.CONTACT.<service name>=<free form text giving contact info>
NMI.REGISTER_WITH_NAMES=[TRUE|FALSE]
```

Note: The choice of `TRUE` or `FALSE` determines whether the agent should try to register itself with Oracle Names.

```
NMI.TRACE_LEVEL=0
NMI.TRACE_MASK=(106)
```


DBSNMP.ADDRESS=<TNS address on which the agent can listen>
DBSNMP.SPAWNADDRESS=<TNS address for the services.ora>

The following lines are not automatically generated, but may be added to the `SNMP_
RW.ORA` file:

```
SNMP.CONNECT.<service name>.NAME=<user name for the subagent to  
use>  
SNMP.CONNECT.<service name>.PASSWORD=<password for the subagent  
to use>
```

Note: This is optional for ALL databases.

```
SNMP.DBPOLLTIME=<interval for polling the database, in seconds>
```

Note: This line governs the time that the subagent polls the database to determine whether it has gone down, or the interval between retries if the database has gone down or was never connected. The default is 30 seconds.

```
NMI.TRACE_DIRECTORY=<directory to which NMD will have write  
access>  
NMI.TRACE_FILE=<file name for the trace>  
NMI.LOG_DIRECTORY=<directory to which NMD will have write access>  
NMI.LOG_FILE=<file name for the log file>
```

Note: The address chosen is a reserved TCP port granted to Oracle by the IANA (Internet Assigned Number Authority). Changing this port will likely make the agent undetectable by the EM Console and force a manual configuration step!).

The `Tcl` script `NMICONF.TCL` can execute other `Tcl` scripts written specifically to discover other Oracle services. If these other scripts exist, they should be installed with `NMICONF.TCL` in `ORA_ROOT: [NETWORK.AGENT.CONFIG]`, and their names should be listed in a file in the same directory called '`NMICONF.LST`', one script per line.

The file `ORA_OEMAGENT:SERVICES.ORA` is created during the discovery phase, and will be used to tell the OEM which services the Agent is monitoring.

Setting the Preferred Credentials

The preferred credentials are supported from the OEM console. To run a job on the HOST database, you **MUST** supply username/password in the preferred credentials fields in the OEM console. To check that the username/password is valid, login to the HOST node where the Intelligent Agent is running and issue the command:

```
$ show process/right
```

to see that the account is not disabled and that it has the `ORA_AGENT_ID` identifier.

Oracle Intelligent Agent Startup, Shutdown, and Status Query

This section explains how to startup, shutdown, and status query of the Agent.

Startup of the Agent

The Agent consists of the following two processes:

- WORK process, which runs as a background detached process
- COMM process, which also runs as a detached process.

Additionally, a third JOB process is created by the WORK process whenever it is needed to execute jobs.

Use the following command to start the Agent:

```
$ LSNRCTL DBSNMP_START
```

This command creates a detached process with a process name of the form `ORA_AGENTWORK`, which will then create the COMM process.

If a nonzero trace level is specified in `SNMP_RW.ORA`, two trace files with the names `DBSNMPW.TRC` and `DBSNMPC.TRC` will be created in `ORA_ROOT:[NETWORK.TRACE]`.

Whenever a job is executed, a trace file of the form DBSNMPJ_<pid>.TRC will also be created.

Note: The process that starts up the Agent must have the GROUP and GRPNAM privileges.

Note: The creation of the processes and initialization of the Agent takes about one to two minutes. This includes connection to the listener and database if specified in the visible services parameter of SNMP_RO.ORA. It also includes initialization of the queue files.

Shutdown of the Agent

Use the following command to shutdown the Agent:

```
$ LSNRCTL DBSNMP_STOP
```

Note: Use the Oracle8 account to stop or start the Agent.

Status Query of the Agent

Use the following command to verify whether the Agent is running:

```
$ LSNRCTL DBSNMP_STATUS
```

Oracle Intelligent Agent Maintenance

Unlike the listener process, the Agent processes are in a continuous loop, polling for incoming connections in each loop. This means that trace information is continuously being generated. Therefore, it is advisable to turn off tracing during normal operation and to turn it on only when a problem is encountered.

SNMP Support

SNMP is supported by using Intelligent Agent as an Oracle subagent. To enable SNMP support, choose Y in the SNMP support line when configuring the Agent.

PEER master agent is currently supported. To run PEER master agent, change the following two files, as follows:

1. Modify the three lines in the PEERAGENT_COM.SAMPLE file that could be found in ORA_NETWORK directory and save it as PEERAGENT.COM in the same directory.
2. Modify the CONFIG_MASTER.SAMPLE file that could be found in the TNS_ADMIN directory, and save it as CONFIG.MASTER in the same directory.

Then connect to Oracle using the SYS account to run the CATSNMP.SQL script from the ORA_OEMAGENT directory, unless it was already run when the Agent was configured.

To use SNMP Support, start the SNMP master agent before starting the TNS Listener and before starting the Intelligent Agent

- To start PEER master agent, execute the command:

```
@ORA_NETWORK:STARTUP_PEERAGENT
```

- To query the status of the PEER master agent, execute the command:

```
@ORA_NETWORK:STATUS_PEERAGENT
```

- To stop PEER master agent, execute the command:

```
@ORA_NETWORK:SHUTDOWN_PEERAGENT
```

For additional information, see the *Oracle SNMP Support Reference Guide*.

Advanced Networking Option

This chapter provides Alpha OpenVMS-specific installation information for the current release of Advanced Networking Option (ANO) for Security and Single Sign-On.

Note: A separate license is required to use ANO.

The topics covered are as follows:

- Documentation Set
- Requirements
- Installation
- De-Installation
- Usage Notes for the Authentication Adapters

Documentation Set

Use this section to install ANO, then see the Administrator's Guide for operating instructions. For further information about installing Oracle SQL*Net products, see the *Oracle8 for Alpha OpenVMS Installation Guide*.

Requirements

This section details installation requirements for ANO on Alpha OpenVMS.

The topics covered in this section are:

- What's in this Release?
- Installation Requirements

What's in this Release?

The Advanced Networking Option for Security and Single Sign-On (ANO) is the new name for the product released earlier under the name: Secure Network Services. This release of ANO Alpha OpenVMS supports the following features:

- Encryption (to RSA and DES standards)
- Checksumming (MD5)
- Authentication (SecurID and Kerberos5 adapters)

Note: At this time, there is NO support for SQL*Net/DCE and Native Naming Adapters.

Installation Requirements

This section summarizes all the requirements necessary before installing ANO Alpha OpenVMS.

System Requirements

This section summarizes the hardware and software requirements for installing ANO Alpha OpenVMS.

For complete information on hardware and software requirements for Oracle8, see the *Oracle8 for Alpha OpenVMS Installation Guide*.

Hardware: See Chapter 1 of the *Oracle8 for Alpha OpenVMS Installation Guide*.

Software: OpenVMS Version 7.1 (minimum)

Oracle Software Requirements

The table below specifies the software requirements for ANO.

Table 6–1 ANO Software Requirements

Software Requirements	Version	State During Installation
Oracle8 Enterprise Edition	8.0.5	Installed
SQL*Net	8.0.5	Installed (see Note)

Note: At least one network protocol adapter must be installed.

Server Authentication Adapter Requirements

The table below specifies the software requirements for Authentication Adapters.

Table 6–2 Adapter Requirements for ANO

Adapter	Version
MIT Kerberos5	Kerberos v5.4.2 or higher. The Kerberos authentication server must be installed on a physically secure machine.
SecurID	ACE/Server v1.2.4 or higher

Note: No additional authentication adapter software is required to relink Oracle products. However, Oracle does not provide an authentication server for Kerberos5, or SecurID. You must separately install and configure the appropriate authentication server.

Installation

This section describes the steps necessary to install ANO Alpha OpenVMS.

The topics covered in this section are:

- Installation Warning
- Installation Tasks

For more information about installing Oracle products using the Installer, see also the *Oracle8 for Alpha OpenVMS Installation Guide*.

Any reference to ANO in the following pages signifies one or more of the following options while choosing to build NETCONFIG using the Oracle Installer:

- Install ANO encryption
- Install SecurID Authentication Adapter
- Install Kerberos5 Authentication Adapter

Installation Warning

When you install ANO, the Installer automatically relinks all Oracle products.

- NETCONFIG (lsnrctl, tnslnr, names, namesctl)
- RDBMS (srv, imp, exp, sqlldr, ...)
- SVRMGR
- UTIL
- PROGIN
- SQLPLUS
- OEMAGENT (if installed)
- and the rest

If you do not wish to relink these executables, do not choose the options to install ANO.

Installation Tasks

- Task 1: Responding to Installer Prompts
- Task 2: Using with Oracle Names
- Task 3: Manual Steps for the Authentication Adapters

Task 1: Responding to Installer Prompts

The following build option screen is displayed:

```
NETCONFIG.DEF Configuration Options
Option                                     Current Value

1. System or Group Installation? [S/G]      S
2. Install TCP/IP adapter? [Y/N]           Y
3. Build Oracle Names Server? [Y/N]        N
4. Install ANO encryption? [Y/N]          N
5. Install SecurID Authentication Adapter? [Y/N] N
6. Install Kerberos5 Authentication Adapter? [Y/N] N

Enter (A)LL to select all options.
Enter (E)XIT to exit this menu with selected options.
Enter (Q)UIT to quit this menu with no action.
```

Enter the number of the option that you want to change:

Options 4, 5, and 6 are related to ANO.

Task 2: Using with Oracle Names

The Oracle Names executables are automatically relinked during the ANO build. To use ANO with Oracle Names, modify the file TNS_ADMIN:NAMES.ORA by adding an entry for the SQLNET.CRYPTO_SEED parameter. You can do this by copying the line that begins with "SQLNET.CRYPTO_SEED=" from your TNS_ADMIN:SQLNET.ORA file into your TNS_ADMIN:NAMES.ORA file.

Note: The complete line must be copied exactly or you will not be able to start the Oracle Names Server using ANO.

Task 3: Manual Steps for the Authentication Adapters

In the database server's local INIT.ORA file, set the following parameters:

```
remote_os_authent = false
os_authent_prefix = ""
```

For SecurID Adapter The logical ORA_VAR_ACE should point to the directory where the configuration file SDCONF.REC is available. By default, this logical will point to the [NETWORK.ACE] directory under ORA_ROOT. If your configuration file is located somewhere else, modify the logical definition in ORA_ROOT:[NETCONFIG]SECURID_USER.COM to point to the correct directory.

Make sure that the directory is readable by all Oracle Server processes.

For Kerberos5 Adapter The following file is required on the client side:

- KRB.CONF (configuration file that specifies the default realm of the client and maps all known realms to Key Distribution Centers (KDCs))

The following files are required on the server side:

- KRB.REALMS (maps hostnames and domains into realms)
- V5SRVTAB (contains key that the KDC uses to encrypt a service ticket for the client)

The location of all of the above files **MUST** be specified using corresponding parameters in SQLNET.ORA.

Additionally, the SQL*Net client also creates a credential cache file whose location needs to be specified in SQLNET.ORA on the client side.

The following is an example of the parameters in SQLNET.ORA for an installation that can act as both client and server:

```
SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=ORACLE
SQLNET.AUTHENTICATION_SERVICES = (BEQ,KERBEROS5)
SQLNET.KERBEROS5_KEYTAB = DISK: [TST7323.NETWORK.ETC] V5SRVTAB.
SQLNET.KERBEROS5_CONF = DISK: [TST7323.NETWORK.KRB5] KRB.CONF
SQLNET.KERBEROS5_REALMS = DISK: [TST7323.NETWORK.KRB5] KRB.REALMS
SQLNET.KERBEROS5_CC_NAME = DISK: [TST7323.NETWORK.CCACHE] CCFILE.DAT
```

De-Installation

This section describes the steps necessary to de-install ANO from your system.

The topics covered in this section are:

- De-Installation Warning
- De-Installation Tasks

Note: The de-install process will NOT modify any of the .ORA files under the TNS_ADMIN directory. This means that if your SQLNET.ORA file contained parameters to enable authentication or encryption, they may no longer work after the de-install. Remember to do the same actions on any of the client installs, if necessary, to retain the compatibility.

De-Installation Warning

Warning: The de-install script does NOT automatically relink any of the executables linked during ANO install. You need to use ORACLEINS to relink all of these executables.

De-Installation Tasks

- Task 1: Preparing Your System
- Task 2: De-install

Task 1: Preparing Your System

To prepare your system to de-install ANO, do the following:

1. Shut down all running database instances normally.
2. Shut down all SQL*Net listener processes.
3. Login as the 'oracle' software owner, for example:

Username: ORACLE8
Password: <password>

4. ORAUSER.COM in your UTIL directory under ORA_ROOT. This will define the symbols and logicals for your oracle installation environment.

Task 2: De-install

De-installing ANO does NOT result in automatic relinking of the executables that were linked during ANO install. You need to relink these using ORACLEINS.

1. At the command prompt, type:

```
$ ORACLEINS
```

2. Choose option 3 to go to the Main Menu.
3. Choose option 1 to go to the "Software Installation and Upgrade Menu".
4. Choose option 2 "Select Build Configuration Options". Then select product "NetConfig". Your previous install options are remembered by ORACLEINS.

The following build option screen is displayed:

NETCONFIG.DEF Configuration Options Option	Current Value
1. System or Group Installation? [S/G]	S
2. Install TCP/IP adapter? [Y/N]	Y
3. Build Oracle Names Server? [Y/N]	N
4. Install ANO encryption? [Y/N]	N
5. Install SecurID Authentication Adapter? [Y/N]	N
6. Install Kerberos5 Authentication Adapter? [Y/N]	N

Enter (A)LL to select all options.

Enter (E)XIT to exit this menu with selected options.

Enter (Q)UIT to quit this menu with no action.

Enter the number of the option that you want to change:

Options 4, 5, and 6 are related to ANO. Choose N for the options that you want to de-install.

5. Exit back to the "Software Installation and Upgrade Menu" and choose option 4 to build the selected products. This causes the following products to be relinked:

- NetConfig (lsnrctl, tnslnr, names, namesctl, ...)

- RDBMS (srv, imp, exp, sqlldr, ...)
- SVRMGR
- UTIL
- PROGINT
- SQLPLUS
- OEMAgent (if installed)
- and the rest.

Usage Notes for the Authentication Adapters

The usage notes are categorized into the following areas.

- General Information
- SecurID
- Kerberos5

General Information

Include the following line in your LISTENER.ORA file:

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

The listener should not participate in the authentication service.

It is recommended that you always include BEQ as one of the authentication services in your SQLNET.ORA file. Here is an example:

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

In this way, connections within the Server machine through the default bequeath adapter do not have to go through the authentication. This is especially important during database startups and shutdowns.

SecurID

If you expect excessive delays in your relink to access the ACE server from your client machine, use the following syntax to connect to the database, for example:

```
$ SQLPLUS USERNAME/"<nnnn><pppppp>+<qqqqqq>"@DATABASE
```

where:

<nnnn> is the PIN number of your SecurID card.

<pppppp> and <qqqqqq> are two successive codes displayed on the card.

Kerberos5

1. Make sure that the clock skew between the client machine and the machine running the KDC is less than one minute.
2. Oracle client and server processes use the Coordinated Universal Time (UTC) format (time elapsed since 00:00:00 Jan. 1, 1970 in records). Make sure that your system is set to the correct time zone in terms of deviation from Greenwich Mean Time (GMT). Otherwise you will get the error "Clock skew too great" in your SQL*Net trace file.
3. Make sure that the value of the parameter `SQLNET.AUTHENTICATION_KERBEROS5_SERVICE` that you specify in `SQLNET.ORA` matches exactly, including case, with the value specified in the KDC.

Major Changes and Enhancements

This appendix lists the major changes from previous releases of SQL*Net 2 to SQL*Net 8 as follows:

- List of Changes
- Enhancements

List of Changes

Most changes are specific to Alpha OpenVMS customers, except for the first item listed below, which is the most significant one. Generic enhancements have been made to SQL*Net functionality and these are explained in the “*Understanding SQL*Net*” documentation.

- SQL*Net version 1 is no longer supported, starting with Oracle7 release 7.3.2 on all platforms. You should remove any SQL*Net Version 1 aliases from your TNSNAMES.ORA file and replace any database links which use version 1 connect strings with links that use version 2 connect descriptors.

The Network Manager (NETMAN) is no longer supported on server platforms. You must now configure your network with NET8 Assistant on a Windows desktop system.

- Linking in “Single-task” is no longer supported on Alpha OpenVMS. All database client programs must use SQL*Net to connect to a server process.
- There are no longer separate adapters for the different TCP/IP vendors. Therefore, there is no need to specify the TCP/IP vendor on the NETCONFIG configuration screen. The Oracle TCP/IP adapter for Alpha OpenVMS is now coded for UCX (Compaq TCP/IP Services for Alpha OpenVMS) and uses the UCX emulation mode provided by the other TCP/IP vendors.
- The bequeath adapter now provides the capability to connect to a local database without configuring SQL*Net. Please refer to Chapter 2, “The Protocol Adapters” for more information.
- The process quotas for the listener, dedicated servers, and prespawnd servers can now be controlled via logical names. Please refer to Chapter 3, “TNS Listener”, for more information.
- DECnet is not supported.
- Connection Manager is not supported.

Enhancements

Enhancements for Oracle8 release 8.0.5 are:

- Enabling nonblocking read/write for the Bequeath/IPC adapters
- Increasing the default message size for the Bequeath Adapter from 512 to 2048 bytes
- Modifying process names to reflect the type of process

- TNS Listener process name in the format `ORA_XXXXXXXXnnn`
where:
`XXXXXXXX` is the last eight characters of the Listener name
`nnn` is a unique number
- Bequeath process name is in the format `ORA_XXXXXXBnnnn`
where:
`XXXXXX` is the SID name (maximum of six characters)
`nnnn` is a unique number
- Dedicated server process in the format `ORA_XXXXXXCnnnn`
where:
`XXXXXX` is the SID name (maximum of six characters)
`nnnn` is a unique number
- Prespawnd server process in the format `ORA_XXXXXXPnnnn`
where:
`XXXXXX` is the SID name (maximum of six characters)
`nnnn` is a unique number
- Names process in the format `ORA_NAMESnnnn`
where:
`nnnn` is a unique number
- Implementing valid node verification using `PROTOCOL.ORA` file
A sample file can be found in `TNS_ADMIN:PROTOCOL_ORA.SAMPLE`
- Creating a server trace file in the trace directory when using TNS Listener

Note: In the previous release, it was created in the directory:
`sys$login`.

- Trapping messages like 'process quota exceeded' in a trace file
If a process exited abnormally, a trace file will be created in the trace directory under the DB directory.

- Modifying the default key for AUTOMATIC_IPC to something other than SID

Note: In the previous release, the SID name was used to create a system logical name that sometimes interfered with other products. Now the prefix `ORA_IPC_` is being used before the SID name to get a unique logical name that relates to an Oracle product.

- Increasing 'LSNRCTL STARTUP' time when using TCP/IP protocols.

Note: In the previous release, the 'LSNRCTL STARTUP' command using TCP/IP took almost two minutes. Now, to startup the listener with any of these protocols should take about twenty seconds.

- Adding SQL*Net OPEN

Note: Refer to the SQLNETOPEN.TXT file in the installation for a complete explanation.

ORASRV_NETV2 Command Procedure Sample File

This appendix shows the contents of the ORASRV_NETV2_COM.SAMPLE file.

You can find a copy of this file in the TNS_ADMIN directory.

Use this sample file as a base for creating an ORASRV_NETV2 command procedure which contains correct information about your database environment.

```

$! NAME: ORASRV_NETV2.COM sample file
$!
$!
$! USAGE: In (PROGRAM=...) argument in LISTENER.ORA
$!
$! FUNCTION: Setup environment for RDBMS shadow processes and start them for
$!           SQL*Net V2.
$!
$! MODIFIED:
$! Rgutherz 10/06/97 - Replace SRV.EXE by ORACLE.EXE
$! SRavindh 07/30/97 - Set default to trace directory to create trace files
$! Rgutherz 07/28/97 - Add authentication support
$! Rgutherz 06/24/97 - Change process name for PRESPAWN/DEDICATED server proc
$! Rgutherz 06/24/97 - Grant all privileges needed to server process
$! DSchwab 08/03/96 - rewritten (using suggestions from DHayter)
$!
$!-----
$ on error then goto CLEANUP
$!
$! This file serves as a template for the actual file that will be created on a
$! per instance basis to setup environment for, and startup RDBMS shadow
$! processes for SQL*Net V2 connections.
$!
$! Name of the actual .COM file will be specified in (PROGRAM='...') name-value
$! pair in the SID entry of LISTENER.ORA.
$!
$! To create ORASRV_NETV2.COM files for different databases, edit the following
$! lines to substitute the SID of the database and its administration directory.
$!
$ ora_db = "database administration directory"
$ sid    = "sid name"
$ dbname = "database name"
$!
$ define/job/nolog ora_sid 'sid'
$ lnm = "ORA_RDBMS_LNM_"+sid
$!
$!
$! When a database is started up, Server Manager creates and populates
$! a logical name table ORA_RDBMS_LNM_<sid>. If this logical name
$! table doesn't (yet) exist then pick up the required logical names
$! from the ORAUSER_<dbname>.COM command procedure. This should only be
$! the case if the database has not yet been started.
$!
$! If the logical name table DOES exist, then it should contain all of
$! the logicals required for the server process to access message files

```

```

$! etc.
$!
$ lnm = f$strlnm(lnm,"LNM$SYSTEM_DIRECTORY",,,, "TABLE")
$ if lnm
$ then
$   ora_rdbms = f$strlnm("ORA_RDBMS",lnm)
$   if ora_rdbms .eqs. "" then lnm = "FALSE"
$ endif
$ if lnm
$ then
$   if f$strlnm("ORA_SYSTEM",lnm) .eqs. "" then define ora_system 'ora_rdbms'
$   if f$strlnm("ORA_PLSQL", lnm) .eqs. "" then define ora_plsql 'ora_rdbms'
$   lnm = lnm + ","
$ else
$   v1 = f$verify(0)    ! We don't want to see ORAUSER stuff
$   @'ora_db'orauser_'dbname'    ! Pick up the logicals this way
$   v2 = f$verify(v1)    ! Switch VERIFY back on if it was originally on
$   lnm = ""            ! Blank out logical name table
$ endif
$!
$ old_priv = f$setprv("SYSLOCK,PRMMBX,LOG_IO,TMPMBX,SYSNAM,DETACH,CMKRNL,WORLD")
$!
$ orasrv := $ORA_RDBMS:ORACLE.EXE
$!
$! Form name of our logical name table
$!
$ pid = f$getjpi("", "PID")
$ tab = "TNS_" + pid
$ define/table=lnm$process_directory lnm$file_dev -
    'lnm' 'tab', lnm$process, lnm$job, lnm$group, lnm$system
$ arg1 = f$strlnm("arg1")
$ arg2 = f$strlnm("arg2")
$ arg3 = f$strlnm("arg3")
$ arg4 = f$strlnm("arg4")
$ arg5 = f$strlnm("arg5")
$ arg6 = f$strlnm("arg6")
$ arg7 = f$strlnm("arg7")
$ arg8 = f$strlnm("arg8")
$!
$! _____
$!
$! Fix for bug 636327
$ rmbx = "ORA_NTP_IR_" + pid
$ rmbxdev = "' 'f$strlnm(rmbx)'"
$ define 'rmbx' 'rmbxdev'

```

```

$ wmbx = "ORA_NTP_IW_" + pid
$ wmbxdev = "'f$trlnm(wmbx)'"
$ define 'wmbx' 'wmbxdev'
$ tnsdev = f$parse("'f$trlnm("TNS_ADMIN")'"',,, "DEVICE", "NO_CONCEAL")
$ tnsdir = f$parse("'f$trlnm("TNS_ADMIN")'"',,, "DIRECTORY")
$ define TNS_ADMIN 'tnsdev' 'tnsdir'
$ netdev = f$parse("'f$trlnm("ORA_NETWORK")'"',,, "DEVICE", "NO_CONCEAL")
$ netdir = f$parse("'f$trlnm("ORA_NETWORK")'"',,, "DIRECTORY")
$ define ORA_NETWORK 'netdev' 'netdir'
$!
$! _____
$! Execute the Authentication command scripts, if present
$ oraroot = f$trlnm("ORA_NETWORK") - "NETWORK]"
$ if f$search("'ORAROOT'netconfig]SECURID_USER.COM") .nes. "" then -
    @'ORAROOT'netconfig]SECURID_USER.COM
$ if f$search("'ORAROOT'netconfig]KERBEROS_USER.COM") .nes. "" then -
    @'ORAROOT'netconfig]KERBEROS_USER.COM
$!
$!
$! Check/Change process name (Rgutherz)
$! Check for prespawn/dedicated server process
$ len = f$length(arg1)
$ if f$locate("prespawn",arg1) .ne. len
$ then
$   type = "P"
$ else
$   type = "C"
$ endif
$ sname = f$edit(sid,"upcase")
$!
$ i = 1
$retry:
$ if i .gt. 100
$ then
$   write sys$output "Error changing process name after 100 times"
$   exit
$ endif
$ on error then goto retry
$ time = f$time()
$ time = f$cvtime(time)
$ time = f$extract(17,5,time)
$ time = time - "."
$ i = i + 1
$ set process/name="ORA_'sname''type''time'"
$ on error then goto CLEANUP
$!

```

```

$! Set default to trace directory
$ oranetwork = f$trnlnm("ORA_NETWORK") - "]"
$ set default 'oranetwork'.TRACE]
$
$ if arg1 .eqs. ""
$ then
$   orasrv "(LOCAL=NO)"
$ else
$   orasrv 'arg1' 'arg2' 'arg3' 'arg4' 'arg5' 'arg6' 'arg7' 'arg8'
$ endif
$!
$ CLEANUP:
$ sts = $status
$ severity = f$message(sts,"severity")
$!
$ if severity .eqs. "%E" .or. severity .eqs. "%F" .or. severity .eqs. "%W"
$ then
$   status = sts
$   trace_dir = ora_db - "]" + ".trace]
$   open/write trace_file 'trace_dir'orasrv_netv2_'pid'.trc
$   write trace_file "This trace file was generated by ORASRV_NETV2.COM"
$   write trace_file "exit with STATUS = ''status'"
$   msg = f$message("''status'")
$   write trace_file "''msg'"
$   close trace_file
$ endif
$!
$ on error then continue
$ deassign/user/table=lnm$system_directory 'tab'
$ exit 'sts'

```

ORASRV_BEQ_<sid> Command Procedure Sample File

This appendix shows the contents of the ORASRV_BEQ_COM.SAMPLE file. You can find a copy of this file in the TNS_ADMIN directory.

```

$!
$! NAME: ORASRV_BEQ_<sid>.COM template
$!
$!
$! FUNCTION: Setup environment for RDBMS shadow processes and start it for
$!           SQL*Net V2 bequeath adapter.
$!
$! MODIFIED:
$! Rgutherz 05/01/98 - Fix for bug 636327
$! Rgutherz 10/06/97 - Replace SRV.EXE by ORACLE.EXE
$! Rgutherz 04/18/97 - Mods for BEQ listener
$! DSchwab  07/23/95 - Mods so that this will run in spawned process
$! DHayter  07/19/96 - Avoid running ORAUSER_<dbname>.COM if possible
$! DSchwab  07/16/96 - Created
$!-----
$!
$! This file serves as a template for the actual file that will be created on a
$! per instance basis to setup environment for, and startup RDBMS shadow
$! processes for SQL*Net V2 bequeath connections. It must exist as
$! ORA_ROOT: [NETWORK.ADMIN] ORASRV_BEQ_<sid>.COM.
$!
$ ON ERROR THEN GOTO CLEANUP
$!
$! Edit the following lines to substitute the SID of this database
$! and its administration directory:-
$!
$! Substitute <sid> with the actual sid of the database
$ sid    = "<sid>"                ! e.g. PROD
$!
$! Substitute <dbname> with the name of the database
$ dbname = "<dbname>"              ! e.g. PRODDB
$!
$! Substitute <ora_db> with the location of the database admin. directory
$ ora_db = "<ora_db>"              ! e.g. DKB300:[ORACLE7.DB_PROD]
$!
$!
$!
$ define ora_sid 'sid'            ! Need the ORA_SID logical
$ lnm = "ORA_RDBMS_LNM_'"sid'"    ! The database's own logical name table
$!
$!
$! If the logical name table doesn't (yet) exist then pick up the required
$! logical names from the ORAUSER_<dbname>.COM command procedure.
$! This should only be the case if the database has not yet been started.
$!
$! If the logical name table DOES exist, then it should contain all of the

```

```

$! logicals required for the server process to access message files etc.
$!
$ lnm = f$trnlrm(lnm,"LNM$SYSTEM_DIRECTORY",,,, "TABLE")
$ if lnm                                ! If ORA_RDBMS_LNM_<sid> exists...
$ then
$   ora_rdbms = f$trnlrm("ORA_RDBMS",lnm)      ! If ORA_RDBMS isn't there...
$   if ora_rdbms .eqs. "" then lnm = "FALSE"    ! ...pretend the table isn't
$ endif
$!
$! If the ORA_RDBMS_LNM_<sid> logical name table exists, and has ORA_RDBMS
$! defined within it, then we should be safe in assuming that it will have
$! all the other required logicals. ORA_SYSTEM and ORA_PLSQL may be
$! exceptions to this before Oracle7 Server release 7.3.2.2.0 so we'll
$! make SURE that they get defined somewhere.
$!
$ if lnm                                ! If it REALLY exists...
$ then
$   if f$trnlrm("ORA_SYSTEM",lnm) .eqs. "" then define ora_system 'ora_rdbms'
$   if f$trnlrm("ORA_PLSQL", lnm) .eqs. "" then define ora_plsql  'ora_rdbms'
$   lnm = lnm+", "
$ else
$!
$! If we get to this point then we are forced to use ORAUSER_<dbname>.COM
$! to define the necessary logicals for us - but this maybe rather slow.
$!
$   v1 = f$verify(0)    ! If logging output we don't want to see ORAUSER stuff
$   @'ora_db'orauser_ 'dbname' 'sid' ! Pick up the required logicals this way
$   v2 = f$verify(v1)   ! Switch VERIFY back on if it was originally on
$   lnm = ""            ! Blank out logical name table as it doesn't yet exist
$ endif
$!
$ orasrv := $ORA_RDBMS:ORACLE.EXE
$!
$! Form name of our logical name table
$!
$ pid = f$getjpi("", "PID")
$ tab = "TNS_" + pid
$!
$ define/table=lnm$process_directory lnm$file_dev -
$   'tab', 'lnm' lnm$process, lnm$job, lnm$group, lnm$system
$!
$!
$ set noon
$ arg0 = f$trnlrm("arg0")
$ arg1 = f$trnlrm("arg1")

```

```

$ arg2 = f$trmlnm("arg2")
$ arg3 = f$trmlnm("arg3")
$ arg4 = f$trmlnm("arg4")
$ arg5 = f$trmlnm("arg5")
$ arg6 = f$trmlnm("arg6")
$ arg7 = f$trmlnm("arg7")
$ arg8 = f$trmlnm("arg8")
$!
$! _____
$!
$! Fix for bug 636327
$ rmbx = "ORA_NTP_IR_" + pid
$ rmbxdev = "'f$trmlnm(rmbx)'"
$ define 'rmbx' 'rmbxdev'
$ wmbx = "ORA_NTP_IW_" + pid
$ wmbxdev = "'f$trmlnm(wmbx)'"
$ define 'wmbx' 'wmbxdev'
$ tnsdev = f$parse("'f$trmlnm("TNS_ADMIN")'",,,,"DEVICE","NO_CONCEAL")
$ tnsdir = f$parse("'f$trmlnm("TNS_ADMIN")'",,,,"DIRECTORY")
$ define TNS_ADMIN 'tnsdev' 'tnsdir'
$ netdev = f$parse("'f$trmlnm("ORA_NETWORK")'",,,,"DEVICE","NO_CONCEAL")
$ netdir = f$parse("'f$trmlnm("ORA_NETWORK")'",,,,"DIRECTORY")
$ define ORA_NETWORK 'netdev' 'netdir'
$! Execute the Authentication command scripts, if present
$ oraroot = f$trmlnm("ORA_NETWORK") - "NETWORK]"
$ if f$search("'ORAROOT'netconfig]SECURID_USER.COM") .nes. "" then -
    @'ORAROOT'netconfig]SECURID_USER.COM
$ if f$search("'ORAROOT'netconfig]KERBEROS_USER.COM") .nes. "" then -
    @'ORAROOT'netconfig]KERBEROS_USER.COM
$ if f$search("'ORAROOT'netconfig]KERBEROS_USER.COM") .nes. "" then -
    define SYS$SCRATCH 'ORAROOT'NETWORK.KRB5]
$!
$!
$!Change process name (Rgutherz)
$ sname = f$edit(sid,"upcase")
$ i = 1
$retry:
$ if i .gt. 100
$ then
$     write sys$output "Error changing process name after 100 times"
$     exit
$ endif
$ on error then goto retry
$ time = f$time()
$ time = f$cvtime(time)

```

```

$ time = f$extract(17,5,time)
$ time = time - "."
$ i = i + 1
$ set process/name="ORA_''sname'B''time'"
$ on error then goto CLEANUP
$!
$ old_priv = f$setprv("SYSLCK,SYSPRV,PRMMBX,LOG_
IO,TMPMBX,SYSNAM,DETACH,CMKRNL,WORLD")
$ if arg1 .eqs. ""
$ then
$   orasrv "(LOCAL=NO)"
$ else
$   orasrv 'arg1' 'arg2' 'arg3' 'arg4' 'arg5' 'arg6' 'arg7' 'arg8'
$ endif
$!
$CLEANUP:
$ sts = $status
$ severity = f$message(sts,"severity")
$!
$ if severity .eqs. "%E" .or. severity .eqs. "%F" .or. severity .eqs. "%W"
$ then
$   status = sts
$   trace_dir = ora_db - "]" + ".trace]
$   open/write trace_file 'trace_dir'orasrv_beq'pid'.trc
$   write trace_file "This trace file was generated by ORASRV_BEQ_''sid'.COM"
$   write trace_file "exit with STATUS = ''status'"
$   msg = f$message("''status'")
$   write trace_file "''msg'"
$   close trace_file
$ endif
$!
$ on error then continue
$ deassign/user/table=lnm$system_directory 'tab'
$ exit 'sts'

$

```

Index

A

Advanced Networking Option, 6-1
AGENT_START.COM, 5-3
ANO, 6-1
 de-installation, 6-7
 requirements, 6-2
 with Oracle Names, 6-5
authentication, 6-2
 usage notes, 6-9
AUTOMATIC_IPC, 2-2

B

BACKGROUND_DUMP_DEST, 3-6
Bequeath Adapter, 2-4
Bequeath Listener, 2-5
 privileges, 2-8
 problem resolution, 2-6
 shutting down, 2-6
 starting, 2-6

C

Centralized configuration, 1-3
checksumming, 6-2
Client-server configuration, 1-4
Communities
 protocol-dependent groups, 1-8
Configuration, 1-2
 centralized, 1-3
 client-server, 1-4
 distributed, 1-4

D

DBSNMPC.COM, 5-3
DBSNMPJ.COM, 5-3
Discovery option, 5-3
distributed database, 1-4
Distributed database configuration, 1-4

E

encryption, 6-2
Ethernet, 1-2

K

Kerberos5, 6-2

L

LISTENER.ORA, 2-2, 3-5

M

Mailbox Adapter, 2-2
Mailbox protocol adapter, 2-2
Major Changes and Enhancements, A-1
Multiple protocols
 on single machine, 1-8
multithreaded servers, 1-3

N

Names Server
 Oracle Names
 introduction, 4-2

- privileges, 4-3
- namesctl, 4-2
- NAMES.ORA, 4-5
- NMICONF.TCL, 5-3

O

- openObjectId ch1.fm
 - 1 1001351, 2-1
- ORA_LSNR, 3-4
- Oracle Enterprise Manager (OEM), 5-2
- Oracle Intelligent Agent, 5-2
 - maintenance, 5-7
 - shutdown, 5-7
 - startup, 5-6
- Oracle Intelligent Agent and SNMP Support, 5-1
- Oracle Names, 4-1
 - with ANO, 6-5
- Oracle Protocol Adapters, 1-7
 - role in TNS network, 1-7
- Oracle Worldwide Technical Support
 - how to contact, xv
- ORASRV
 - command procedure, B-1
- ORASRV_BEQ, C-1
- ORASRV_BEQ_Command Procedure Sample File, C-1
- ORASRV_NETV2, B-1
- ORASRV_NETV2 Command Procedure Sample File, B-1
- ORASRV_NETV2_COM.SAMPLE file, B-1

P

- PEER master agent, 5-8
- preface
 - PT PrefaceTitle, ix
- preferred credentials, 5-6
- pre-spawned servers, 1-3
- Protocol Adapters, 1-7
 - Bequeath Adapter, 2-4
 - Mailbox Adapter, 2-2
 - TCP/IP Adapter, 2-3
- Protocol API differences
 - TNS implementation, 1-8

- Protocol implementations, 1-8
- PT PrefaceTitle, ix

S

- SDNS.ORA, 4-3
- SecurID, 6-2
- SID_LIST, 3-5
- SNMP support, 5-7
- SQL*Net
 - architecture, 1-7
 - changes from previous releases, A-1
- SQLNET.ORA, 4-4

T

- TCP/IP Adapter, 2-3
- The Protocol Adapters, 2-1
- TNS
 - Oracle network layer, 1-7
- TNS client
 - community member
 - common cases, 1-8
- TNS Listener, 3-1
 - general connections, 3-6
 - introduction, 3-2
 - privileges, 3-3
 - process quotas, 3-4
- TNS, Adapters, and Protocols
 - Protocol Adapters, 1-8
- trace information, writing, 2-6
- Transparent Network Substrate, 1-8
- Transparent Network Substrate (TNS), 1-5

U

- USER_DUMP_DEST, 3-6

V

- VMS Mailbox adapter, 1-4

W

- Windows 95, 5-2
- Windows NT, 5-2