**Net Services - Best Practices for Database Performance, Scalability and High-Availability**

Kuassi Mensah  
Oracle Corporation  
USA

**Keywords:**  
Net Manager, NetCA, Net Listener, IPv6, TCP, TNS, LDAP, Logon Storm.

**Introduction**

The ability to remotely connect to the Oracle database is a major architecture requirement. Oracle Net Services is a key component of several database components and services including: database listener, dedicated servers, shared servers, pooled servers, connection manager, instance registration, Data Guard, Streams, AQ notifications, and clusterware. It has a profound impact on system performance, high-availability, network scalability, network security.

This paper will first introduce the new features in Oracle database 11g R1 and R2 then discuss the best practices for tuning and managing Oracle Net Services through the operating system, the network, the database client, the Net listener, and the database server.

**Oracle Net Services in Oracle Database 11g R1 and R2**

![Figure 1 Oracle Net Services Architecture](image)

**Key Net Services features in Oracle Database 11g R2**
IPv6 support at the RDBMS level: complete IPv6 support for single-instance DB in 11.2; session layer abstraction to support listening across all IP interfaces (v4 and v6); fine-grained IP address family matching in listener.

IP address traversal for each hostname, during connect: applicable to all naming methods and timeouts enforced on each connect attempt (iow, each IP address).

TCP.CONNECT_TIMEOUT enabled by default (i.e., 60 sec)
  o SQLNET.OUTBOUND_CONNECT_TIMEOUT off by default.

RETRY_COUNT enabled by default (i.e., 0 sec): can be set at DESCRIPTION level in connect string; can be set globally at sqlnet.ora level; in case of a DESC list, the specified number of retries would be attempted within a DESC before the next DESC is tried; does not apply at DESCRIPTION_LIST level.

CONNECT_TIMEOUT for connect string: can be set at DESCRIPTION level in connect string; overrides the global SQLNET.OUTBOUND_CONNECT_TIMEOUT; does not apply at DESCRIPTION_LIST level.

Key Net Services features in Oracle Database 11g R1

- IPv6 support at the network level
- Secure Net Naming: authenticated LDAP lookup
- Performance & Scalability: optimized networking stacks for various data transfer needs (network Fast Path for SQL operations, zero copy I/O Path for bulk data transfers (SecureFile LOBs, DataGuard, Recovery Manager); support for Database Resident Connection Pool; support for Scalable Operating System Event Models.
- High Availability & Manageability: Easy Connect Naming enhancements; fast reconnects for HA; efficient dead-node detection for failover; integration with ADR framework; option for Default Service in Listener for ease of client-side configuration
- Network Security: various DoS mitigation features; authenticated LDAP name lookup - OID and Active Directory; protocol level access control for Listener administration

OS Tuning and Best Practices

The Operating System plays a key role in data transmission however:
  a) default OS configurations cannot handle modern Ethernet speeds; default buffer size not large enough to handle the amount of data on the “wire” (a.k.a. BDP)
  b) as a benevolent algorithm i.e., one size fits all, TCP is slow-start, small window sizes, exponential back-off, and some features may not be enabled by default.

TCP on Linux

  a) TCP auto-tuning in kernel (2.4.27, 2.6.7)
     /proc/sys/net/ipv4/tcp_moderate_rcvbuf (1=on)
  b) Tune TCP Max Memory
     /proc/sys/net/ipv4/tcp_rmem and tcp_wmem
     4096  87380 174760 <- Tune this to 2xBDP
  c) Tune the socket buffer sizes
     /proc/sys/net/core/rmem_max and wmem_max
     Set this to 2xBDP
  d) Ensure that TCP Performance features are enabled
     /proc/sys/net/ipv4/tcp_sack
     /proc/sys/net/ipv4/tcp_window_scaling
TCP on Windows

a) Vista / Server 2008 supports TCP auto-tuning
b) For other versions, tuning necessary under RegKey
   HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
   • Turn on Window Scaling and Timestamps
     Tcp1323Opts = 3
   • Set TCP Window Size to 2xBDP
     GlobalMaxTcpWindowSize = <2xBDP>
   • If desired, tune Window Size at the Interface Level
     Tcpip\Parameters\Interfaces\<interfaceGUID>\TcpWindowSize

Processor/Hardware Best Practices

a) Use Jumbo Frames for GigE networks
b) Use NICs with TCP off-loading capabilities
c) Monitor switches and OS for packet loss

Tuning and Managing Database Clients

In this paragraph we'll look at the tuning, the best practices and the management of database clients
including: sockets buffers, Session Data Unit (SDU), and Maximum Transfer Unit (MTU). But let’s
first review Net Services configuration files and where these fit

Net Services Configuration Files

a) sqlnet.ora is the main Oracle Net configuration file; defined on both database client and
database server hosts
b) listener.ora: for configuring the Net Listener, defined on database server only
c) tnsnames.ora: contains mapping of Connect Name to Descriptor; used by the TNSNames
   Naming adapter. Defined on both the database client and database server hosts.
d) ldap.ora: contains LDAP configuration information; used by the LDAP Naming adapter.
   Defined on both the database client and database server hosts.

Socket Buffers Tuning and Best Practices

The SEND and RECEIVE socket buffer sizes must be set in tnsnames.ora or sqlnet.ora to
accommodate the BDP (2x), using:
   • SEND_BUF_SIZE: OS send buffer size
   • RECV_BUF_SIZE: OS receive buffer size
Large buffer sizes help: (i) the application queue more data at the OS level, (ii) having more data on
the wire, (iii) better utilization of available bandwidth, (iv) WAN deployments.

Session Data Unit (SDU) Tuning and Best Practices

SDU controls SQL*Net packet size. The 11g default is 8k, pre-11g default is 2k, and max is 32k.
SDU is set in
a) sqlnet.ora: DEFAULT_SDU_SIZE
b) **tnsnames.ora**: SDU in address

Best practices suggest

a) Larger SDU gives: (i) better network throughput, (ii) fewer system calls to send and receive data, (iii) less CPU usage (system and user). However, with larger SDU, the network buffers will take up more memory.

b) Optimal SDU varies with application

- Increase SDU on both client and server: set SDU for a connection negotiated down to the lower of the two peers.
- Increase SDU to 8k: this is a good default value for most users.
- For bulk data transfer scenarios, increase to 32k; good for LOB and XML transfers.

**Maximum Transfer Unit (MTU)**

The MTU value is fixed and depends on the actual network implementation used. On standard Ethernet networks, the default MTU size is set to 1514 bytes. On standard token ring networks, the default MTU size is 4202. SDU and MTU are independent.

**Managing Database Clients – Net Naming**

Net Naming as illustrated hereafter, consists in mapping a **Connect Name** to a **Connect Descriptor**.

```
sales = ← this is the Connect Name
     (DESCRIPTION= ← this is the Connect Descriptor
          (ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
          (CONNECT_DATA=(SERVICE_NAME=sales)))
```

The mapping can be specified or resolved using the following methods: (i) host-based e.g., Easy Connect syntax; (ii) a file e.g., tnsnames.ora; (ii) a directory e.g., Oracle Internet Directory or Active Directory.

**Easy Connect**

Easy Connect is a simple, easy to use (i.e., no need for any client side configuration file) connect syntax for TCP/IP.

```
[//]host[:port][/service_name][:server] [/instance_name]
```

Example: `sqlplus scott/tiger@sales-server/sales`

Easy connect is useful when no connect descriptor customization is necessary.

```
sales-server/sales
```

is equivalent to

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

**Net Naming Best Practices**

a) Use Easy Connect whenever possible

b) If Descriptors do not change often (small deployments), use `tnsnames.ora` and `TNS_ADMIN` can be on a shared file system

c) If Descriptors change often or for large deployments, use a directory e.g., Oracle Internet Directory or Active Directory on Windows
Database Client High Availability

High availability for database client consists in fast database server death detection at connect-time and runtime and enabled by the following mechanisms: connection establishment timeouts, and connection failover.

Connection Establishment Timeouts
Helps detect database server death faster

- TCP.CONNECT_TIMEOUT (11g feature): it controls the TCP connection establishment duration (30 seconds is a good value to use)
- SQLNET.OUTBOUND_CONNECT_TIMEOUT (10gR2 feature): it controls the time taken to connect to a database server process. To be set if the session establishment takes a long time. These timeouts are not set by default, can be used individually or together however, Outbound_Connect_Timeout must be greater than Connect_Timeout.

Fast Connection Failover
Established client connections could hang when the database host crashes or the remote network fails. The detection of such failures could be long, unpredictable or faster depending on which mechanism is put in place and the state of the database session. For example, TCP timeouts take minutes to be enforced and long running queries could not interrupted quickly.

- Use/enable Receive Timeout: if your application is active and does not have long-running queries
- Use/enable Fast Application Notification (FAN); see RAC and Database APIs specific documentations for more details.

Net Listener Tuning and Best Practices

Net Listener is the first database process that clients talk to; as illustrated in figure 2, it renders many services including:
a) Brokering client requests and handing them off to service handlers including the dedicated servers, the dispatchers of shared servers configuration, and the connection broker of the Database Resident Connection Pool (DRCP).
b) Receiving load updates (load balancing advisories) from the database
c) Performing server side load-balancing across instances in RAC
d) Listening on multiple end-points or protocol addresses
e) Supporting other presentations including: HTTP, FTP, and IMAP

Listener Service Registration
Services can be registered either statically or dynamically. Static service registration consists in specifying SID_LIST configuration in listener.ora; this allows clients to connect immediately. Dynamic service registration also known simply as “service registration” consists in dynamically registering instance information with a listener. The PMON process provides the listener with the instance name, database service names, the type and addresses of service handlers, as well as the load balancing advisories. This information enables the listener to start the appropriate service handler when a client request arrives.

The benefits of service registration are:
a) Simplified configuration in init.ora: by removing the need for listener.ora, dynamic registration reduces the administrative overhead. By default, PMON connects to listener on port 1521 (see Easy Connect above).

SERVICE_NAMES lists the services an instance belongs to. On startup, each instance registers with the listeners of other instances belonging to the same services.

LOCAL_LISTENER: Address of listeners on local host

REMOTE_LISTENER: Address of listeners on remote hosts

b) Connect-time failover
c) Runtime Connection Load balancing: as illustrated by figure 3, consists in routing connection request to the least loaded node based on load balancing advisories.

d) Runtime Connection Load balancing is enabled by setting goals in init.ora
   - Long: for applications with long-lived connections (default)
   - Short: for applications with short-lived connections

Logon Storm
Logon storm consists in a sudden spike in incoming connection rate; which results in CPU starvation for existing sessions. Logon storm happens under normal circumstances, when middle-tiers reboot and all users try to get a connection. Logon storm happens also under abnormal circumstances, during a Denial of Service attack (DoS attack).

The Connection Rate Limiter protects from logon storm; it provides end-point level control of throttling and can be configured in LISTENER.ORA

```sql
LISTENER=(ADDRESS_LIST=
    (ADDRESS=(PROTOCOL=tcp)(HOST=sales)(PORT=1521)(RATE_LIMIT=3))
    (ADDRESS=(PROTOCOL=tcp)(HOST=lmgmt)(PORT=1522)(RATE_LIMIT=no)))
```

Set the Rate Limit to a value that matches your machine capabilities.

Scalable Event Models
Oracle uses the poll system call on most platforms however, poll does not scale well beyond 1000 connections. A new, more efficient polling methods is now supported on some platforms
- epoll on Linux – Kernel 2.6 (11g)
- /dev/poll on Solaris and other platforms (in the works)

This new event model furnishes excellent scalability for shared servers and DRCP. It is enabled by default for DRCP; it is enabled explicitly (for shared servers), in server-side sqlnet.ora

```sql
USE_ENHANCED_POLL = on
```

Database Server Tuning and Best Practices

Secure Connection Phase
Net Services provide server timeouts to limit the time taken for a client to connect and authenticate
- SQLNET.INBOUND_CONNECT_TIMEOUT: controls timeout for database server processes
- INBOUND_CONNECT_TIMEOUT_listener_name: controls timeout for the listener

These timeouts are independent of client-side timeouts and available from Oracle Database 10gR1 onwards; the default value in Oracle Database 10g R2 and up is 60 seconds.

TCP Valid Node Checks
Database security is further enforced through TCP node checks:
- Use TCP Invited Nodes: list of IPs or hostnames that are permitted to connect
- Use TCP Excluded Nodes: list of IPs or hostnames that are NOT permitted to connect

Note: Invited nodes takes precedence over excluded nodes.

Enable/set in sqlnet.ora

```sql
VALIDNODE CHECKING = YES
TCP.INVITED NODES = (hostname1, hostname2)
TCP.EXCLUDED NODES = (hostname3, hostname4)
```
Database Resident Connection Pool (D.R.C.P.)

As illustrated by figure 4, DRCP is a pool of dedicated servers shared across client systems and processes. The pooled server is “locked” on connect, and “released” on disconnect. The result is a low connect/disconnect costs, low-latency performance of dedicated servers, and extreme scalability.

Pooling is enabled by the DBA using

```
EXECUTE DBMS_CONNECTION_POOL.START_POOL ('SYS_DEFAULT_CONNECTION_POOL');
```

Change connect string on client in tnsnames.ora:

```
(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=sales-server)(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=sales)(SERVER=pooled))
```

Can use Easy Connect syntax too, e.g., `sqlplus joeuser@sales-server:1521/sales:POOLED`

In test environment, we were able to support more than 20,000 connections to a 2 GB Database Server

http://www.oracle.com/technology/tech/php/

Contact address:

**Kuassi Mensah**

Oracle Corporation

400 Oracle Parkway - 94065, Redwood City, USA

Phone: (+1) 650 607 2229

Fax: (+1) 650 506 7225

Email kuassi.mensah@oracle.com

Blog: http://db360.blogspot.com