INTEGRIGY

WHITE PAPER

Oracle Database Listener Security Guide

APRIL 2017

ORACLE DATABASE LISTENER SECURITY GUIDE

October 2002
March 2003 – Updated
January 2004 – Updated
July 2004 – Updated
March 2005 – Updated
April 2007 – Updated
March 2016 – Updated
April 2017 – Rewrite

Authors: Michael Miller and Stephen Kost

If you have any questions, comments, or suggestions regarding this document, please send them via e-mail to info@integrigy.com.

Table of Contents

OVERVIEW

The Oracle Database Listener is the database server software component that manages the network traffic between the Oracle Database and the client. The Oracle Database Listener listens on a specific network port (default 1521) and forwards network connections to the Database. The Listener is comprised of two binaries – (1) tnslsnr which is the Listener itself and (2) the Listener Control Utility (lsnrctl) which is used to administer the Listener on the server or remotely.

Through our security assessments, Integrigy has consistently identified poor Oracle Database Listener security as a significant security risk. The majority of Oracle Database Listeners are not properly secured as recommended by Oracle and security experts. Fortunately, in Oracle 11g and 12c, the default Listener configuration is much more secure than earlier versions.

The information contained in this paper is not new and is not obscure. However, it is not well known to many Oracle DBAs but is well known to security experts and hackers. This paper outlines the vulnerabilities in the Oracle Database Listener and provide recommendations for properly securing it. Providing minimal security for the Oracle Database Listener is simple and should be done for all Oracle installations – development, test, and production.

WHY PROTECT THE LISTENER?

One of the most misunderstood security issues with the Oracle Database is the security of the Listener. The Listener provides access to the database and is configured separately from the database thus the security of the database is highly dependent on the Listener being securely configured. Generally, DBAs are not aware how attackers can easily remotely attack and/or disrupt the Listener.

WHY READ THIS PAPER?

This paper presents Integrigy's recommended minimum of what we believe you should consider for securing your Listener as well as a roadmap of optional measures to take once you have met the recommended steps. Our recommendations are based both on our research and from working with our clients.

Earlier versions of the Oracle database (before version 10g) had numerous and serious security vulnerabilities with the Listener. Today, with version 11g and 12c of the Oracle Database, Oracle has both added functionality and changed a number of default settings to make the Listener more secure by default.

However, the default installation is still not sufficient for a properly secured Listener. To secure an Oracle Database and its Listener, action is required, especially if you have upgraded from an earlier version of the database. While the Oracle documentation concerning the security of the Listener is excellent, it is not complete. Recommended best practice security configurations for Oracle Listener security must also be considered. Besides numerous books on Oracle security, there are the hardening guides such as those

provided by the Center for Internet Security (CIS) and the US Department of Defense. Both the CIS and DoD standards both have recommendations for securing the Oracle Listener – see the reference section of this whitepaper.

Scope and Database Versions

This paper focuses only on the most widely deployed Listener configurations running supported versions of the Oracle Database and assumes the Listener has been configured to use TCP/IP as well as local naming (TNSNAMES.ORA) is used. No references will be made to Oracle Connection Manager, other naming methods (like LDAP), or advanced configurations like load balancing or Transparent Application Failover (TAF).

The focus of this whitepaper is on Oracle Database versions 12c (12.1 and 12.2), 11g (primarily 11.1.0.7 and 11.2.0.4), and 10g (primarily (10.2.0.5). Prior versions of this whitepaper address earlier versions of the Oracle database including 8.1.7.4, 9.0.1.4, and 9.2.0.7.

Terminology

For clarity and convenience, the Oracle Database Listener will be referred to as the "Listener" throughout this document. The Listener may also be referred to as the "Oracle Net Listener" or the "Oracle TNS Listener". Transparent Network Substrate (TNS) is the network protocol used by Oracle for connectivity to Oracle Databases.

LISTENER OVERVIEW

The Oracle Database Listener is the server process that provides basic network connectivity for clients, application servers, and other databases to an Oracle database. In addition to databases, the Listener can also be configured to execute custom and/or third party binary executables in the operating system.

For more information on the Listener, please see the Oracle documentation, specifically the Oracle Database Net Services Administrator's Guide 12c Release 1.

LISTENER DETAILS

The relevant files for the Listener are as follows –

$ORACLE_HOME/bin/lsnrctl	Listener control program
$ORACLE_HOME/network/admin/listener.ora	Configuration file for the Listener
$ORACLE_HOME/network/admin/sqlnet.ora	Configuration file for the Listener
$ORACLE_HOME/bin/tnslnsr	Server Listener process

The **lsnrctl** program is the mechanism for starting and stopping the listener process (**tnslsnr**). **tnslnsr** starts and reads the **listener.ora** and **sqlnet.ora** files for configuration information, such as port numbers and database service names.

The **tnslnsr** processes starts with the process owner of the **lsnrctl** program, usually the "oracle" account on UNIX or Linux. Any successful exploit will thus gain the privileges for this account.

Oracle TNS Default Ports

Port Number	Description
1521	The default port for the TNS Listener
1522 – 1540	Commonly used ports for the TNS Listener
1575	Default port for the Oracle Names Server
1630	Default port for the Oracle Connection Manager – client connections
1830	Default port for the Oracle Connection Manager – administrative connections
2483	TNS for TCP/IP
2484	TNS for TCP/IP with SSL

LISTENER EXPLOITS

Over the years, numerous exploits of the Listener have been reported. Oracle 12c, fortunately, is markedly improved. That said there are a number of exploits that need to be kept in mind even with Oracle 12c. Many of these exploits have been commoditized, if not weaponized, and are freely available in open source penetration tools and utilities – see Appendix A for a listing.

Denial-of-Service (DoS) Attacks

While the probability of an Oracle database residing behind a firewall being subject to a Denial of Service (DoS) attack via SQL*Net may be low, it is possible. Wherever open to direct physical access, databases are subject to DoS attacks. There are hacking tools and utilities that can flood the Oracle Listener with connection requests. These tools also come with attacks to attack well-known default accounts and passwords as well as SID guessing programs. Such attacks can inordinately monopolize CPU cycles and potentially "flood" local disk and/or storage with trace files and audit logs. Refer to Appendix A for a listing of a few of the Oracle hacking tools.

Remote Management Exploits

Earlier versions of Oracle allowed for remote management of the Listener. This allowed potentially anyone, if not secured, to reconfigure and/or shutdown the listener. Since Oracle 10g, the Listener by default cannot be remotely managed. Unless remote management is accidently or intentionally enabled, the Listener cannot be remotely managed and can only be managed locally by the owner of the **tnslsnr** process (usually oracle).

TNS Poisoning

In 2012, the details of a vulnerability in the Oracle Database listener were published that allows an attacker to register with the database listener and to intercept and modify TNS network traffic between the client and database server[1]. This "TNS Poison" attack allows an unauthenticated attacker with only network connectivity to compromise most database accounts. By default, Oracle 12c mitigates against TNS Poisoning, but this functionality can be disabled and/or turned off. For Oracle 11g, the same said functionality exists, but is not enabled by default and must be enabled.

Running Code in Operating System

The Oracle RDBMS is the preeminent relational database in the World, and its feature set is second to none. The standard functionality of the Oracle RDBMS allows the database to call external code. Whether or not this allowed, and if allowed, whether or not it is allowed to be done safely and security is an essential security control. If not properly secured by the Listener, external procedures can allow the database if not the server itself to be partially or wholely compromised.

Extract Information About the Database

Even if an attacker is not able to compromise the database through the Listener, if he or she has access to the database, he or she will be-be able to learn a great deal about the database to help them with additional attacks. Refer to Appendix A for a listing of several open source security penetration tools such as Metasploit and NMAP that can be used to research and enumerate Oracle Listener vulnerabilities.

[1] https://www.integrigy.com/files/Integrigy%20Oracle%20TNS%20Poisoning%20Attacks.pdf

LISTENER CONFIGURATION

The following recommendations are based on Integrigy's research. This research is a result of working with our clients and from Oracle database security best practices. The mandatory recommendations represent those steps that must be taken to secure an Oracle database. The optional steps should as well be seriously considered as part of a long-term roadmap for holistically securing an Oracle database.

MANDATORY RECOMMENDATIONS

Block SQL*Net on Firewalls and Isolate on Network

The number one recommendation Integrigy makes for database security is to, as much as possible, reduce physical access to the database. This is because most, if not essentially all, database exploits and vulnerabilities require a physical connection and, by physically isolating the database and barring direct physical access, these vulnerabilities can be largely mitigated. Ideally, databases, with their associated Listeners, should be physically isolated in network segments reserved for databases and/or for specific applications.

SQL*Net traffic should also not be allowed to pass through firewalls unless absolutely necessary. Firewall filters should be designed to only allow SQL*Net traffic from known application and web servers. SQL*Net traffic from application servers in the DMZ should be permitted only to access specific database servers.

Few applications require direct SQL*Net connections to a database from the Internet. SQL*Net performs poorly over high latency networks, thus is seldom used in Internet applications. If applications do require direct SQL*Net access, configure firewall filters based on a specific host and port number.

Only allow local administration

By default, the Listener cannot be remotely managed and can only be managed locally by the owner of the tnslsnr process (usually oracle) since Oracle 10G - this is the default. Do not turn local administration off.

LOCAL_OS_AUTHENTICATION_<*listener name*> = ON

Apply Latest Security Patches

Apply the latest Critical Patch Update. Critical Patch Updates are cumulative, therefore, the latest patch will contain all previous security patches for the RDBMS, inclusive of the Listener.

Turn on Logging

Turn on logging for all listeners to capture Listener commands and brute force password attacks.

```
LSNRCTL> set current_listener <listener name>
LSNRCTL> set log_directory <oracle_home path>/network/admin
LSNRCTL> set log_file <sid name>.log
LSNRCTL> set log_status on
LSNRCTL> save_config
```

Set ADMIN_RESTRICTIONS

When enabled (set to ON) ADMIN_RESTRICTIONS_<listener name> rejects SET commands issued at runtime or from remote systems to change the Listener's configurations. When enabled it only allows changes to the Listener's configuration through a lsnrctl reload command on the local system initiated by user who has write privilege on the **listener.ora** file. The reload command will read the configurations from the **listener.ora** file. The default is OFF, set it to ON.

ADMIN_RESTRICTIONS_<listener name> = ON

Configure Valid Node Checking Registration (VNCR)

In 2012, the details of a vulnerability in the Oracle Database listener were published that allows an attacker to register with the database listener and to intercept and modify TNS network traffic between the client and database server. This "TNS Poison" attack allows an unauthenticated attacker with only network connectivity to compromise most database accounts.

VNCR should not be confused with Valid Node Checking (described later) and is the replacement of Class of Secure Transport (COST). From 11.2.0.4 onwards, VNCR needs to be enabled to prevent TNS poisoning and ensure that instance registration (e.g. new instances added to the listener) are only performed from known and trusted servers. When VNCR set to LOCAL (recommended), then registration can only be done from the local server. Depending on the version of Oracle, the following options are available:

TNS Poisoning Options			
Database Version	***SSL Encrypt with Cert*** See ASO	***COST*** *class of secure transport* 1453883.1 1340831.1 (RAC)	***VNCR*** *Valid node checking registration* 1600630.1
8.1.7.x – 10.2.0.3	✓		
10.2.0.3 – 10.2.0.5	✓	✓	
11.1.0.x	✓	✓	
11.2.0.1 – 11.2.0.3	✓	✓	
11.2.0.4	✓	✓	✓ (Not enabled by default)
12.1.0.x*	✓	✓	✓ (Enabled by default)
12.2.0.x*	✓	✓	✓ (Enabled by default)

Disable Default Listener

Do not name the Listener 'Listener', use a unique name.

An additional step is to create a dummy Listener named 'listener' as depicted below. This configuration will throw errors and prevent a Listener named 'listener' from starting.

```
LISTENER=(DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST=)(PORT = 0)))
```

Secure the $TNS_ADMIN Directory

The file permissions on the **listener.ora**, **sqlnet.ora**, and **protocol.ora** files in the $TNS_ADMIN directory (usually $ORACLE_HOME/network/admin) should be read/write/execute for only the primary oracle account and no permissions for any other account (for UNIX and Linux 0600). The **tnsnames.ora** file permissions should be set to 0644 on UNIX and Linux.

Default installations of Oracle will secure the $TNS_ADMIN directory, but its permissions be checked regardless.

Secure tnslsnr and lsnrctl

The **tnslsnr** and **lsnrctl** executables in the $ORACLE_HOME/bin directory should be protected, and file permissions should be set to 0751 on UNIX and Linux as recommended by Oracle. It is possible to change the file permissions to 0700 which would be more secure, although this should be thoroughly tested in your environment.

Default installations of Oracle will secure tnslsnr and lsnrctl, but the permissions should be checked regardless.

Remove Unused Services

Since **listener.ora** files are sometimes copied between instances, and they may contain old and unused entries. Check all services in the **listener.ora** to determine if they are used. Remove any services not actively used.

Secure External Procedures

External procedures are written in C, C++, Java or another language. External procedures are compiled and stored outside the database. When external procedures are called, the Oracle RDBMS uses an external agent by default named 'ExtProc.' How to secure the Listener for external procedures depends on the version of Oracle, however, simplistically straightforward, if not needed, remove ExtProc from the **listener.ora**.

Oracle 10g

If external procedures are NOT used remove ExtProc references from the **tnsnames.ora** file by following the instructions in Oracle Support Note 244523.1 https://support.oracle.com/rs?type=doc&id=244523.1 . If there are any ExtProc custom or third party binaries in the respective directories, be sure to remove them as well.

If external procedures ARE required, follow the instructions in Oracle Support Note 244523.1 https://support.oracle.com/rs?type=doc&id=244523.1 to create a second Listener, such that two listeners are defined – one for the database and one for external procedures.

Oracle 11g

The Oracle 11g default configuration, unlike earlier versions, does not require the ExtProc agent to be configured in the **listener.ora** or **tnsnames.ora** files. Oracle 11g security has been increased to by default directly spawn the ExtProc agent thus eliminating several vulnerabilities where the ExtProc agent could be spawned unexpectedly. To utilize the new Oracle 11g ExtProc functionality, the external procedures must be configured in the **extproc.ora** file located in the $ORACLE_HOME/hs/admin directory.

If your database has been upgraded to 11g, check the listener entry for PL/SQL External Procedures (ExtProc). The entry name is usually ExtProc or PLSExtProc. Often ExtProc is installed by default but is not used. If found, Check with your application development team or application documentation to determine if ExtProc is used. If it is, migrate to the new 11g **extproc.ora** functionality and remove ExtProc references from the **listener.ora** file.

If external procedures are used, ensure that in the **extproc.ora** file the environment variable EXTPROC_DLL is set to whitelist only allow approved external code rather than allowing ANY code to be run.

Good example:	Bad example:
EXTPROC_DLLS=ONLY:/home/xyz/mylib.so	EXTPROC_DLLS=ANY

If the EXTPROC_DLLS environment variable is not set, the ExtProc agent will load and run code (DLLs) from the ORACLE_HOME/lib directory on UNIX operating or the ORACLE_HOME\bin directory on Windows. Integrigy strong recommends that the EXTPROC_DLLS environment variable is set. The following values are allowed with the recommended value of 'ONLY.'

- ***Colon-separated list of DLLs*** Syntax: "DLL:DLL" To only to load the identified "whitelisted" code by name as well as any code and/or DLLs from the ORACLE_HOME/lib directory on UNIX or the ORACLE_HOME\bin directory on Windows. To whitelist, the complete directory path and file name of the DLLs must be referenced.
- ***ONLY*** (Recommended) Syntax: "ONLY:DLL:DLL" This will only allow the whitelisted code and/or DDLs to be run regardless of whether or not they exist in the ORACLE_HOME/lib directory on UNIX or the ORACLE_HOME\bin directory on Windows. To whitelist, specify the complete directory path and file name of the code and/or DLLs.
- ***ANY*** Syntax: "ANY" This value allows the ExtProc agent to load and runs any code and/or DLL (not recommended).

Oracle 12c

Check with your application development team or application documentation to determine if ExtProc is used. If it is, migrate to the new 11g **extproc.ora** functionality using the recommended 'ONLY' whitelisting. If ExtProc is not used, remove it from the **listener.ora** file.

New with Oracle 12c, the ExtProc agent can be configured to run as a designated operating system account instead of using the operating system privileges of the listener user or the Oracle server process. This feature enables the definition of a database credential to be associated with the ExtProc process, which then can

authenticate and impersonate (that is, run on behalf of the supplied user credential) before loading a user-defined shared library and executing the function.

To configure the ExtProc user credential, use the new PL/SQL package, DBMS_CREDENTIAL. As part of this new feature, the CREATE LIBRARY statement has been enhanced to enable the association of the ExtProc user credential with a library.

Integrigy strongly recommends using dedicated accounts (credentials) for ExtProc code as well as setting the new ENFORCE_CREDENTIAL parameter to be used in the **extproc.ora** file to require credentials to be used. The default value of the parameter is FALSE, Integrigy recommends setting it to TRUE. Another new environment variable, GLOBAL_EXTPROC_CREDENTIAL is also available, this is the default credential to be used if none is specified. For more information on this topic, refer to the Oracle documentation http://docs.oracle.com/database/121/DBSEG/app_devs.htm#DBSEG758

Optional Recommendations

The following are measures that can be taken to further secure the Listener.

Monitor the Logfile

The Listener's logfile may contain TNS-01169, TNS-01189, TNS-01190, or TNS-12508 errors, which may signify attacks or inappropriate activity. Using a simple shell script or management tools, monitor the logfile and generate an alert whenever these errors reaches are encountered. If possible, send the listener log to Splunk or a similar such log aggregator.

By default, logging is not enabled (LOG_STATUS=OFF). When logging is enabled, the default directory is $ORACLE_HOME/network/admin and the log file default is <sid>.log. The logfile contains a history of listener commands issued both locally and remotely.

The logfile shows a timestamp, command issued, and result code. If an Oracle error is returned, it will include the error message. The logfile does not contain passwords or other significant information. The logfile does NOT show any information related to IP address, client name, or other identifying information for remote connections. It may show the client's current user name, but this can easily be spoofed or not provided.

The following are TNS errors that may signify an attack or inappropriate activity –

Error Code	Message	Comments
TNS-01169	The listener has not recognized the password	An attempt was made to issue a command, but a password is set
TNS-01189	The listener could not authenticate the user	Local OS Authentication is enabled, and attempt was made to manage the Listener remotely or locally by another user
TNS-01190	The user is not authorized to execute the requested listener command	Local OS Authentication is enabled, and attempt was made to manage the Listener locally by another user
TNS-12508	The listener could not resolve the COMMAND given	This error occurs when an invalid command is an issue (e.g., **statusx** instead of **status**) or when a **set** command is issued, and **ADMIN_RESTRICTIONS** is set to no.
ORA-12525	Listener has not received client's request in time allowed	SET INBOUND_CONNECT_TIMEOUT exceeded by client and connection terminated – most likely this would not be detected by Imperva/Guardium as an error is not returned to client
ORA-28040	No matching authentication protocol error or an ORA-03134	Connections to this server version are no longer supported error -- SQLNET.ALLOWED_LOGON_VERSION
ORA-12170	Connect timeout occurred	SQLNET.INBOUND_CONNECT_TIMEOUT time exceed – will see only in the log – client does not see this error.

Set Login Notification Banners

In the **sqlnet.ora** file set the following parameters to display appropriate disclosures and warnings to users:

- SEC_USER_AUDIT_ACTION_BANNER – this specifies a text file containing the banner that warns users they will be audited
- SEC_USER_UNAUTHORIZED_ACCESS_BANNER – this specifies a text file containing the banner warns user about unauthorized access

Protect Against TNS Protocol Attacks

While not explicitly **listener.ora** configurations, the following database startup parameters govern how the database responds to malformed network packets. These parameters were added in Oracle version 11g to better harden the Listener –

- SEC_PROTOCOL_ERROR_TRACE_ACTION
- SEC_PROTOCOL_ERROR_FURTHER_ACTION

SEC_PROTOCOL_ERROR_TRACE_ACTION

Specifies the action that the database should take when bad packets are received from a possibly malicious client. The options are below with TRACE being the default (recommended):

- **NONE**: The database server ignores the bad packets and does not generate any trace files or log messages. (Not recommended)
- **TRACE**: A detailed trace file is generated when bad packets are received, which can be used to debug any problems in client/server communication.
- **LOG**: A minimal log message is printed in the alert logfile and the server trace file. A minimal amount of disk space is used.
- **ALERT**: An alert message is sent to a DBA or monitoring console.

SEC_PROTOCOL_ERROR_FURTHER_ACTION

Specifies the further execution of a server process when receiving bad packets from a possibly malicious client. The default for 11.2.0.4 is CONTINUE. The default for 12.1.0.2 is DROP,3 (recommended). The options are below:

- **CONTINUE**: The server process continues execution. The database server may be subject to a Denial of Service (DoS) if bad packets continue to be sent by a malicious client. (Not recommended)
- **DELAY,integer:** The client experiences a delay of integer seconds before the server process accepts the next request from the same client connection. Malicious clients are prevented from excessive consumption of server resources while legitimate clients experience degradation in performance but can continue to function.
- **DROP,integer**: The server forcefully terminates the client connection after reaching the target number of bad packets. The server protects itself at the expense of the client (for example, a client transaction may be lost). The client may reconnect and attempt the same operation.

Protect Against Brut-Force Login Attacks

Oracle 11g delivered a new startup parameter SEC_MAX_FAILED_LOGIN_ATTEMPTS[2]. This parameter determines whether or not to drop and ignore failed authentication attempts against one or multiple accounts from the same network connection. If the threshold is exceeded, the connection will be automatically dropped by the database.

The startup parameter SEC_MAX_FAILED_LOGIN_ATTEMPTS is commonly confused with the password profile FAILED_LOGON_ATTEMPTS. The purpose of SEC_MAX_FAILED_LOGIN_ATTEMPTS is to be a connection "throttle' to help prevent potential intruders from executing brut-force login attacks (e.g. password guesser utilities). A key point is that SEC_MAX_FAILED_LOGIN_ATTEMPTS is looking at attempted sessions at a network level to login to the database as whole whereas the password profile FAILED_LOGON_ATTEMPTS is concerned with only one database user account. SEC_MAX_FAILED_LOGIN_ATTEMPTS will not lock database accounts and will only drop the network connection whereas FAILED_LOGON_ATTEMPTS will lock the account.

The default for SEC_MAX_FAILED_LOGIN_ATTEMPTS for 11.2.0.4 is ten (10). The default for 12.1.0.2 is three (3). If changed, testing is required to ensure that there are no performance ramifications.

Protect Against Denial-of-Service Attacks

To prevent Denial-of-Service attacks from flooding the database with idle login attempts, there are two (2) different INBOUND_CONNECT_TIMEOUT parameters respectively set in the **sqlnet.ora** and the **listener.ora** files. These two parameters work together to drop idle connection and login requests.

In the **listener.ora** file the parameter INBOUND_CONNECT_TIMEOUT_*<listener_name>* specifies the time, in seconds, for a user to complete their connection to the listener after the network connection is established. If the listener does not receive the client request within the specified number of seconds (default is 60), then the connection is terminated, and the listener logs the IP address of the client and an ORA-12525 error.

In the **sqlnet.ora**, file the parameter INBOUND_CONNECT_TIMEOUT specifies the time, in seconds (default 60), for a client to provide their authentication information. If the period is exceeded, the connection is terminated and the **sqlnet.ora** log file will record an ORA-12170: TNS: Connect timeout occurred. The client will also receive an ORA-12547 error: TNS: lost contact as well as potentially an ORA-12637: TNS: Packet receive failed error message.

The default values for both parameters of sixty (60) seconds is sufficient for most organizations. If, however per organizational requirements the value needs to be changed, set the **listener.ora** INBOUND_CONNECT_TIMEOUT_<listener_name> parameter to a smaller value than the **sqlnet.ora** INBOUND_CONNECT_TIMEOUT.

---

[2] Init.ora Parameter "SEC_MAX_FAILED_LOGIN_ATTEMPTS (Doc Id 567117.1)
https://support.oracle.com/rs?type=doc&id=567117.1

Encrypt Oracle SQL*Net Traffic

Consider encrypting SQL*Net traffic between clients and the database server. SQL*Net Encryption was previously only allowed through licensing Oracle Advanced Security Option (ASO). This is no longer the case, and for Oracle 11g and Oracle 12c, SQL*Net encryption is now available through the standard enterprise edition license and supports the following Ciphers:

- AES (128, 192, 256) – outer CBC only
- 3DES (112, 168) – CBC only

To see if enabled:

```
SELECT NETWORK_SERVICE_BANNER
FROM V$SESSION_CONNECT_INFO;
```

View the SQLNET.ORA file for configurations and refer to the Oracle Advanced Security Guide for detailed configuration instructions. The key to implementing SQL*Net encryption is whether the server and/or client(s) require encryption or opportunistically request it. The table below outlines to possible combinations -

SQL*Net Encryption Connection		
Client Setting	**Server Setting**	**Encryption Result**
REJECTED	REJECTED	OFF
ACCEPTED *	REJECTED	OFF
REQUESTED	REJECTED	OFF
REQUIRED	REJECTED	Connection fails
REJECTED	ACCEPTED *	OFF
ACCEPTED *	ACCEPTED *	OFF
REQUESTED	ACCEPTED *	ON
REQUIRED	ACCEPTED *	ON
REJECTED	REQUESTED	OFF
ACCEPTED *	REQUESTED	ON
REQUESTED	REQUESTED	ON
REQUIRED	REQUESTED	ON
REJECTED	REQUIRED	Connection fails
ACCEPTED *	REQUIRED	ON

SQL*Net Encryption Connection		
Client Setting	**Server Setting**	**Encryption Result**
REQUESTED	REQUIRED	ON
REQUIRED	REQUIRED	ON

Secure Sockets Layer (SSL) Encryption

Similar to SQL*Net encryption as previously only available with the purchase of the Advanced Security Option (ASO), encryption of SQL*Net traffic using SSL is now available as part of the standard Enterprise Edition for Oracle 11g and 12c. This feature is an alternative to native SQL*Net encryption and is useful if the identity of the client(s) and/or the server is required to be verified and/or that a higher amount of encryption is required.

Refer to Oracle's security documentation for more information; however, it is Integrigy's experience that setting up SSL encryption is rarely attempted due to its complexity. If interested in using SSL encryption of SQL*net traffic, Integrigy does not recommend SSL certifications for authentication. Integrigy recommends continuing to use password-based multi-factor authentication.

Consider Valid Node Checking

The number one recommendation Integrigy makes for database security is to, as much as possible, reduce physical access to the database. Setting up Valid Node Checking will whitelist those IP addresses allowed to physically connect to the database. Ideally, this is feasible only if the whitelist is between four to eight addresses. For example, such a whitelist could be comprised of the IP address of a jump box (aka a Bastion host) used by DBAs and developers and the web and/or application middle tier servers. Maintaining a whitelist of more than eight to twelve IP addresses usually does not prove practical.

The simplest method to determine valid IP addresses for node checking is through database auditing. We recommended you always have session level auditing enabled.

For Oracle 11g, the valid node checking lines are added to the $ORACLE_HOME/network/admin/**sqlnet.ora** file.

```
tcp.validnode_checking = yes
tcp.invited_nodes = (x.x.x.x | name, x.x.x.x | name)
tcp.excluded_nodes=( x.x.x.x | name, x.x.x.x | name)
```

Invited and/or excluded nodes are only used when Valid Node Checking is enabled. Also, include either the invited_nodes or excluded_nodes, but do not use both. Wildcards, subnets, etc. are not valid, only individual IP addresses or host names are allowed. For more sophisticated checking, use Oracle Connection Manager.

Change the TNS Port Number from 1521

To help stop automated attacks and detection of the Listener in networks, the default NTS port number should be changed from 1521 to a port outside of the 1521-1550 and 1600-1699 ranges. This will provide only minimal

additional security through obscurity, but may thwart an automated attack or simply scanning for Oracle Databases on port 1521.

The port number can be changed using Oracle Net Manager (**netmgr**) or editing the **listener.ora** file directly. All **tnsnames.ora** files on the database server and any clients must be updated to reflect the change in the port number. The database initialization parameter **LOCAL_LISTENER** must be set so that the database can dynamically register with the Listener. See Metalink Note ID 359277.1 "Changing Default Listener Port Number" for more information - https://support.oracle.com/rs?type=doc&id=359277.1

Please note that third party tools which assume and/or are hardcoded for Oracle being on 1521 may break if another port is used.

Meter Incoming Connection Requests

It is possible to set a governor for the number of new connections the Listener can open in a given second. To set a governor, use the CONNECTION_RATE_LISTENER parameter. However, at the time of this writing, there are a number of bugs with this feature. If required and/or attempt to use, careful and thorough testing will be needed.

Set Minimum Client Version – Oracle 12c only

Oracle 12c database servers have limited to no support for 10g clients – refer to Oracle Support Not 207303.1 for more information: Client / Server Interoperability Support Matrix for Different Oracle Versions (Doc ID 207303.1 https://support.oracle.com/rs?type=doc&id=207303.1).

- Oracle Client 10.2.0 – limited for 12.1.0 server
- Oracle Client 10.1.0 – no support for 12.1.0 server

Before Oracle 12c, to control what Oracle Middleware (TNS) clients are allowed to the database server the Oracle SQL*NET parameter SQLNET.ALLOWED_LOGON_VERSION specified the AUTHENTICATION PROTOCOL (for example SHA-1) that clients are allowed to use. This parameter (SQLNET.ALLOWED_LOGON_VERSION) is deprecated in Oracle Database 12c.

With Oracle 12c the SQLNET.ALLOWED_LOGON_VERSION parameter has been replaced with two new Oracle Net Services parameters:

- SQLNET.ALLOWED_LOGON_VERSION_SERVER – for clients connecting to a database server
- SQLNET.ALLOWED_LOGON_VERSION_CLIENT – when the database server is acting as a client (e.g. database links)

The effect of the new default value of 11 for SQLNET.ALLOWED_LOGON_VERSION_SERVER in Oracle Database 12*c* is that clients using Oracle Database release 10*g* and later can connect to the Oracle Database 12*c* server.

By default, Oracle12c authenticates user connections using all three verifiers, the 10G verifier, the 11G (SHA-1) verifier, and the 12C verifier. Note that the 10G verifier is not case sensitive, but the 11G and 12C verifiers are

case sensitive. What setting the SQLNET.ALLOWED_LOGON_VERSION_SERVER parameter should be set to depend on the specific balance of security and interoperability required for your organization.

ALLOWED_LOGON_VERSION_SERVER	Generated Password Version	Meaning for Clients
12a	12C	Only Oracle Database 12*c* Release 12.1.0.2 or later clients can connect to the server.
12	11G, 12C	Only clients who have applied critical patch update CPUOct2012 or later, or release 11.2.0.3 clients with an equivalent update can connect to the server.
11	10G, 11G, 12C	Clients using Oracle Database 10*g* and later can connect to the server. Clients using releases earlier than Oracle Database release 11.2.0.3 that have not applied critical patch update CPUOct2012, or later patches must use the 10G password version.
10	10G, 11G, 12C	Clients using Oracle Database 10*g* and later can connect to the server. Clients using releases earlier than Oracle Database release 11.2.0.3 that have not applied critical patch update CPUOct2012, or later patches must use the 10G password version.
9	10G, 11G, 12C	Oracle9*i* Database or later clients can connect to the server.
8	10G, 11G, 12C	Oracle8*i* Database and later clients can connect to the server.

SQL-Net FIPS-140 Encryption – Oracle 12c Only

The Federal Information Processing Standard (**FIPS**) Publication **140-2**, (**FIPS** PUB**140-2**), is a U.S. government computer security standard used to approve cryptographic modules. Oracle supports FIPS 140-2 Oracle Network encryption and it is configured in the **sqlnet.ora** file.

Please note FIPS-140 within SQL*Net is different than configuring SSL to use FIPS 140-2 within the **fips.ora** file. These are two different configuration files for two different protocols (communication types). To configure network data encryption to run in FIPS mode by setting the FIPS_140 parameter to TRUE in the **sqlnet.ora** file.

Ensure that the sqlnet.ora file is either located in the $ORACLE_HOME/network/admin directory or is in a location pointed to by the TNS_ADMIN environment variable

When FIPS_140 is set to TRUE in the sqlnet.ora file the network data encryption cryptographic operations take place in the embedded RSA library in FIPS mode. These cryptographic operations are accelerated by the CPU when hardware acceleration is available and properly configured in the host hardware and software.

If FIPS_140 to FALSE, the network data encryption cryptographic operations take place in the embedded RSA library in non-FIPS mode, and as with the TRUE setting, the operations are accelerated if possible.

After configuring the FIPS 140-2 settings, you must verify the following permissions in the operating system:

Set execute permissions on all Oracle executable files to prevent the execution of Oracle Cryptographic Libraries by users who are unauthorized to do so, by the system security policy.
Set read and write permissions on all Oracle executable files to prevent accidental or deliberate reading or modification of Oracle Cryptographic Libraries by any user.

Once instructed to FIPS 140-2 ciphers, there are specific sqlnet.ora configurations are set on both the client and server to enable Oracle Network encryption. Each side can ask that encryption be: REJECTED, ACCEPTED, REQUESTD OR REQUIRED. By using different combinations of these configurations, different clients can be forced to ask or not ask for network encryption. The server respectively can then opportunistically encrypt according to if the client supports encryption or not.

Additional notes on setting up Oracle Network encryption can be found in Chapter 13 “How to Enable Data Encryption and Integrigy” in the “Oracle Database Security Guide 12c Release 1 (12.1)”, Oracle Corporation, E48135-09, July 2014 http://docs.oracle.com/database/121/DBSEG/E48135-09.pdf.

Service-Level Access Control List (ACL) for TCP – Oracle 12.2 Only

Service-Level ALCs is a new feature of the 12.2 Listener that allows every database service to have its own ACL. The ACL must be based on IP addresses and this feature allows multitenant pluggable databases (PDBs) to each have an ACL enforced by the Listener. This is because each PDB is a unique service registered in the Listener.

To implement this feature a new parameter **FIREWALL** must be used and has the following options:

- **(FIREWALL=ON)** - This enables strict ACL validation (whitelist-based approach) of all connections based on the ACLs. If no ACLs are configured for a service, all connections are rejected.
- **FIREWALL** is not set (defined for service) – This is a mixed mode. If an ACL is configured for a service, it will be enforced. If no ACL is defined, all connections will be accepted.
- **(FIREWALL=OFF)** No validation (No ACLs enforced) and all connections are accepted

For more information refer to:
http://docs.oracle.com/database/122/NETAG/configuring-and-administering-oracle-net-listener.htm - NETAG0102

APPENDIX A - THIRD PARTY TOOLS

There are a number of third-party tools available that can remotely control or obtain information from the listener.

- ***Integrigy AppSentry Listener Check (Recommended)***
  A simple Windows GUI tool that checks a number of Listener security settings
  http://www.integrigy.com/security-resources/downloads/lsnrcheck-tool

- ***Metasploit***
  Open source penetration testing inclusive of Oracle vulnerabilities
  https://www.metasploit.com/

- ***NMAP***
  Open source network discovery and security auditing with Oracle capabilities
  https://nmap.org/

- ***Oracle Database Attack Tool (ODAT)***
  Open source Oracle audit and penetration testing tool
  https://github.com/quentinhardy/odat

- ***Oracle Auditing Tools*** **– cqure.net**
  A set of Java-based tools that can query the Listener for information
  http://www.cqure.net/wp/tools/database/test/

- ***Getsids SID Enumeration*** **– cqure.net**
  A Windows command-line tool to get the available databases from the Listener
  http://www.cqure.net/wp/?page_id=13

- ***SidGuesser*** **– cqure.net**
  A Windows command-line tool used to find databases using a dictionary attack
  http://www.cqure.net/wp/?page_id=41

REFERENCES

General

- Integrigy - Oracle Database TNS Poisoning Attacks
  https://www.integrigy.com/files/Integrigy%20Oracle%20TNS%20Poisoning%20Attacks.pdf
- Oracle TNS Listener Exploits
  http://www.red-database-security.com/exploits/oracle_exploit_listener.html

Documentation and Support Notes

- Oracle Database Net Services Administrator's Guide 11g Release 2
  https://docs.oracle.com/cd/E18283_01/network.112/e10836/advcfg.htm
- Oracle Database Net Services Administrator's Guide 12c Release 1
  https://docs.oracle.com/database/121/NETAG/E17610-11.pdf
- Oracle Security Guide12*c* Release 1 https://docs.oracle.com/database/121/DBSEG/toc.htm
- Deprecation of Listener Password in Oracle Database 11g Release 2 (Doc ID 1328725.1)
  https://support.oracle.com/rs?type=doc&id=1328725.1
- How To Network Secure Your Oracle Database Listener in Intranet / Internet (Doc Id 364388.1)
  https://support.oracle.com/rs?type=doc&id=364388.1
- What is TCP_VALIDNODE_CHECKING (462933.1)
  https://support.oracle.com/rs?type=doc&id=462933.1
- Setting Listener Passwords With an Oracle 10g or Newer (Note 260986.1)
  https://support.oracle.com/rs?type=doc&id=260986.1

Security Standards

- Center for Internet Security (CIS) Oracle Security Benchmarks https://benchmarks.cisecurity.org
- US Department of Defense Security Technical Implementation Guide (STIG)
  http://iase.disa.mil/stigs/Pages/index.aspx

About Integrigy

Integrigy Corporation (www.integrigy.com)

Integrigy Corporation is a leader in application security for enterprise mission-critical applications. AppSentry, our application and database security assessment tool, assists companies in securing their largest and most important applications through detailed security audits and actionable recommendations. AppDefend, our enterprise web application firewall is specifically designed for the Oracle E-Business Suite. Integrigy Consulting offers comprehensive security assessment services for leading databases and ERP applications, enabling companies to leverage our in-depth knowledge of this significant threat to business operations.

INTEGRIGY
Integrigy Corporation
P.O. Box 81545
Chicago, Illinois 60681 USA
888/542-4802
www.integrigy.com