20

Configuring Oracle Database Native Network Encryption and Data Integrity

You can configure native Oracle Net Services data encryption and data integrity for both servers and clients.

20.1 About Oracle Database Native Network Encryption and Data Integrity

Oracle Database enables you to encrypt data that is sent over a network.

20.1.1 How Oracle Database Native Network Encryption and Integrity Works

Oracle Database provides native data network encryption and integrity to ensure that data is secure as it travels across the network.

The purpose of a secure cryptosystem is to convert plaintext data (text that has not been encrypted) into unintelligible ciphertext (text that has been encrypted) based on a key, in such a way that it is very hard (computationally infeasible) to convert ciphertext back into its corresponding plaintext without knowledge of the correct key.

In a symmetric cryptosystem, the same key is used both for encryption and decryption of the same data. Oracle Database provides the Advanced Encryption Standard (AES) symmetric cryptosystem for protecting the confidentiality of Oracle Net Services traffic.

20.1.2 Advanced Encryption Standard

Oracle Database supports the Federal Information Processing Standard (FIPS) encryption algorithm, Advanced Encryption Standard (AES).

AES can be used by all U.S. government organizations and businesses to protect sensitive data over a network. This encryption algorithm defines three standard key lengths, which are 128-bit, 192-bit, and 256-bit. All versions operate in outer Cipher Block Chaining (CBC) mode. CBC mode is an encryption method that protects against block replay attacks by making the encryption of a cipher block dependent on all blocks that precede it; it is designed to make unauthorized decryption incrementally more difficult. Oracle Database employs outer cipher block chaining because it is more secure than inner cipher block chaining, with no material performance penalty.

> **Note:**
>
> The AES algorithms have been improved. To transition your Oracle Database environment to use stronger algorithms, download and install the patch described in My Oracle Support note 2118136.2.

20.1.3 Choosing Between Native Network Encryption and Transport Layer Security

Oracle offers two ways to encrypt data over the network, native network encryption and Transport Layer Security (TLS).

There are advantages and disadvantages to both methods.

Table 20-1 Comparison of Native Network Encryption and Transport Layer Security

-	Native Network Encryption	Transport Layer Security
Advantages	• It is configured with parameters in the `sqlnet.ora` configuration file. • In most cases, no client configuration changes are required. • No certificates are required. • Clients that do not support native network encryption can fall back to unencrypted connections while incompatibility is mitigated.	• It is an industry standard for encrypting data in motion. • It provides non-repudiation for server connections to prevent third-party attacks. • It can be used for database user authentication.
Disadvantages	• It uses a non-standard, Oracle proprietary implementation. • It provides no non-repudiation of the server connection (that is, no protection against a third-party attack).	• It requires client and server changes. • Certificates are required for server and are optional for the client. However, the client must have the trusted root certificate for the certificate authority that issued the server's certificate. • Certificates eventually expire.

20.2 Oracle Database Native Network Encryption Data Integrity

Encrypting network data provides data privacy so that unauthorized parties cannot view plaintext data as it passes over the network.

Oracle Database also provides protection against two forms of active attacks.

Table 20-2 provides information about these attacks.

Table 20-2 Two Forms of Network Attacks

Type of Attack	Explanation
Data modification attack	An unauthorized party intercepting data in transit, altering it, and retransmitting it is a data modification attack. For example, intercepting a $100 bank deposit, changing the amount to $10,000, and retransmitting the higher amount is a data modification attack.
Replay attack	Repetitively retransmitting an entire set of valid data is a replay attack, such as intercepting a $100 bank withdrawal and retransmitting it ten times, thereby receiving $1,000.

20.3 Data Encryption and Integrity sqlnet.ora Parameters

Oracle provides many parameters that you can set in the `sqlnet.ora` file for data encryption and integrity.

- About the Data Encryption and Integrity Parameters
  The data encryption and integrity parameters control the type of encryption algorithm you are using.

- Sample sqlnet.ora File
  The sample `sqlnet.ora` configuration file is based on a set of clients with similar characteristics and a set of servers with similar characteristics.

20.3.1 About the Data Encryption and Integrity Parameters

The data encryption and integrity parameters control the type of encryption algorithm you are using.

The `sqlnet.ora` file, which is where you set these parameters, is generated when you perform the network configuration. Also provided in this process are encryption and data integrity parameters. You can use the default parameter settings as a guideline for configuring data encryption and integrity.

The following table lists the data encryption and integrity parameters.

Table 20-3 Data Encryption and Integrity Parameters

Parameter	Description
`SQLNET.CRYPTO_CHECKSUM_CLIENT`	Specifies the checksum behavior for the client
`SQLNET.CRYPTO_CHECKSUM_SERVER`	Specifies the checksum behavior for the server
`SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT`	Specifies a list of crypto-checksum algorithms for the client to use
`SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER`	Specifies a list of crypto-checksum algorithms for the server to use
`SQLNET.ENCRYPTION_CLIENT`	Enables encryption for the client
`SQLNET.ENCRYPTION_SERVER`	Enables encryption for the server
`SQLNET.ENCRYPTION_TYPES_CLIENT`	Lists encryption algorithms the client to use
`SQLNET.ENCRYPTION_TYPES_SERVER`	Lists encryption algorithms the server to use

If you do not specify any values for Server Encryption, Client Encryption, Server Checksum, or Client Checksum, the corresponding configuration parameters do not appear in the `sqlnet.ora` file. However, the defaults are `ACCEPTED`.

For both data encryption and integrity algorithms, the server selects the first algorithm listed in its `sqlnet.ora` file that matches an algorithm listed in the client `sqlnet.ora` file, or in the client installed list if the client lists no algorithms in its `sqlnet.ora` file. If there are no entries in the server `sqlnet.ora` file, the server sequentially searches its installed list to match an item on the client side—either in the client `sqlnet.ora` file or in the client installed list. *If no match can be made and one side of the connection REQUIRED the algorithm type (data encryption or integrity), then the connection fails.* Otherwise, the connection succeeds with the algorithm type `inactive`.

Data encryption and integrity algorithms are selected independently of each other. Encryption can be activated without integrity, and integrity can be activated without encryption, as shown by Table 20-4:

Table 20-4 Algorithm Type Selection

Encryption Selected?	Integrity Selected?
Yes	No

Table 20-4 (Cont.) Algorithm Type Selection

Encryption Selected?	Integrity Selected?
Yes	Yes
No	Yes
No	No

Related Topics

- *Oracle Database Net Services Reference*
- Configuring Oracle Database Native Network Encryption and Data Integrity
  You can configure native Oracle Net Services data encryption and data integrity for both servers and clients.
- About Activating Encryption and Integrity
  In any network connection, both the client and server can support multiple encryption algorithms and integrity algorithms.

20.3.2 Sample sqlnet.ora File

The sample `sqlnet.ora` configuration file is based on a set of clients with similar characteristics and a set of servers with similar characteristics.

The file includes examples of Oracle Database encryption and data integrity parameters.

By default, the `sqlnet.ora` file is located in the *`ORACLE_HOME`*`/network/admin` directory or in the location set by the `TNS_ADMIN` environment variable. Ensure that you have properly set the `TNS_ADMIN` variable to point to the correct `sqlnet.ora` file.

Trace File Setup

```
#Trace file setup
trace_level_server=16
trace_level_client=16
trace_directory_server=/orant/network/trace
trace_directory_client=/orant/network/trace
trace_file_client=cli
trace_file_server=srv
trace_unique_client=true
```

Oracle Database Native Network Encryption

```
sqlnet.encryption_server=accepted
sqlnet.encryption_client=requested
sqlnet.encryption_types_server=(AES256)
sqlnet.encryption_types_client=(AES256)
```

> **Note:**
>
> The RC4_40 algorithm is deprecated in this release. To transition your Oracle Database environment to use stronger algorithms, download and install the patch described in My Oracle Support note 2118136.2.

Oracle Database Network Data Integrity

```
#ASO Checksum
sqlnet.crypto_checksum_server=requested
sqlnet.crypto_checksum_client=requested
sqlnet.crypto_checksum_types_server = (SHA256)
sqlnet.crypto_checksum_types_client = (SHA256)
```

Transport Layer Security

```
#SSL
WALLET_LOCATION = (SOURCE=
                   (METHOD = FILE)
                      (METHOD_DATA =
                         DIRECTORY=/wallet)

SSL_CIPHER_SUITES=(TLS_AES_128_CCM_SHA256)
SSL_VERSION= TLSv1.3
SSL_CLIENT_AUTHENTICATION=FALSE
```

Common

```
#Common
automatic_ipc = off
sqlnet.authentication_services = (beq)
names.directory_path = (TNSNAMES)
```

Kerberos

```
#Kerberos
sqlnet.authentication_services = (beq, kerberos5)
sqlnet.authentication_kerberos5_service = oracle
sqlnet.kerberos5_conf= /krb5/krb.conf
sqlnet.kerberos5_keytab= /krb5/v5srvtab
sqlnet.kerberos5_realms= /krb5/krb.realm
sqlnet.kerberos5_cc_name = /krb5/krb5.cc
sqlnet.kerberos5_clockskew=900
sqlnet.kerberos5_conf_mit=false
```

RADIUS

```
#Radius
sqlnet.authentication_services = (beq, RADIUS )
sqlnet.radius_authentication_timeout = (10)
sqlnet.radius_authentication_retries = (2)
sqlnet.radius_authentication_port = (1645)
sqlnet.radius_send_accounting = OFF
sqlnet.radius_secret = /orant/network/admin/radius.key
sqlnet.radius_authentication = radius.us.example.com
sqlnet.radius_challenge_response = OFF
sqlnet.radius_challenge_keyword = challenge
sqlnet.radius_challenge_interface =
oracle/net/radius/DefaultRadiusInterface
sqlnet.radius_classpath = /jre1.1/
```

20.4 Data Integrity Algorithms Support

Data integrity algorithms protect against third-party attacks and message replay attacks. Oracle recommends SHA-2, but maintains SHA-1 (deprecated) for backward compatibility.

These hashing algorithms create a checksum that changes if the data is altered in any way. This protection operates independently from the encryption process so you can enable data integrity with or without enabling encryption.

Related Topics

- Configuring Integrity on the Client and the Server
  You can use Oracle Net Manager to configure network integrity on both the client and the server.

20.5 Diffie-Hellman Based Key Negotiation

You can use the Diffie-Hellman key negotiation algorithm to secure data in a multiuser environment.

Secure key distribution is difficult in a multiuser environment. Oracle Database uses the well known Diffie-Hellman key negotiation algorithm to perform secure key distribution for both encryption and data integrity.

When encryption is used to protect the security of encrypted data, keys must be changed frequently to minimize the effects of a compromised key. Accordingly, the Oracle Database key management function changes the session key with every session.

The Diffie-Hellman key negotiation algorithm is a method that lets two parties communicating over an insecure channel to agree upon a random number known only to them. Oracle Database uses the Diffie-Hellman key negotiation algorithm to generate session keys.

The client and the server begin communicating using the session key generated by Diffie-Hellman. When the client authenticates to the server, they establish a shared secret that is only known to both parties. Oracle Database combines the shared secret and the Diffie-Hellman session key to generate a stronger session key designed to defeat a person-in-the-middle attack.

> **Note:**
>
> The use of the anonymous RC4 cipher suite for non-authenticated TLS connections was desupported in Oracle Database 21c (`SSL_DH_anon_WITH_RC4_128_MD5`). Oracle recommends that you use the more secure authenticated connections available with Oracle Database. If you use anonymous Diffie-Hellman with RC4 for connecting to Oracle Internet Directory for Oracle Enterprise User Security, then you must migrate to use a different algorithm connection. Oracle recommends that you use either TLS one-way, or mutual authentication using certificates. Note that Oracle Enterprise User Security has been deprecated starting with Oracle Database 23ai.

20.6 Configuration of Data Encryption and Integrity

Oracle Database native Oracle Net Services encryption and integrity presumes the prior installation of Oracle Net Services.

- About Activating Encryption and Integrity
  In any network connection, both the client and server can support multiple encryption algorithms and integrity algorithms.

- About Negotiating Encryption and Integrity
  The sqlnet.ora file on systems using data encryption and integrity must contain some or all the REJECTED, ACCEPTED, REQUESTED, and REQUIRED parameters.
- Configuring Encryption and Integrity Parameters Using Oracle Net Manager
  You can set up or change encryption and integrity parameter settings using Oracle Net Manager.

20.6.1 About Activating Encryption and Integrity

In any network connection, both the client and server can support multiple encryption algorithms and integrity algorithms.

When a connection is made, the server selects which algorithm to use, if any, from those algorithms specified in the sqlnet.ora files.The server searches for a match between the algorithms available on both the client and the server, and picks the first algorithm in its own list that also appears in the client list. If one side of the connection does not specify an algorithm list, all the algorithms installed on that side are acceptable. The connection fails with error message ORA-12650 if either side specifies an algorithm that is not installed.

Encryption and integrity parameters are defined by modifying a sqlnet.ora file on the clients and the servers on the network.

You can choose to configure any or all of the available encryption algorithms, and either or both of the available integrity algorithms. Only one encryption algorithm and one integrity algorithm are used for each connect session.

> **Note:**
>
> Oracle Database selects the first encryption algorithm and the first integrity algorithm enabled on the client and the server. Oracle recommends that you select algorithms and key lengths in the order in which you prefer negotiation, choosing the strongest key length first.

Related Topics

- Data Encryption and Integrity sqlnet.ora Parameters
  Oracle provides many parameters that you can set in the sqlnet.ora file for data encryption and integrity.
- *Oracle Database Advanced Security Guide*

20.6.2 About Negotiating Encryption and Integrity

The sqlnet.ora file on systems using data encryption and integrity must contain some or all the REJECTED, ACCEPTED, REQUESTED, and REQUIRED parameters.

- About the Values for Negotiating Encryption and Integrity
  Oracle Net Manager can be used to specify four possible values for the encryption and integrity configuration parameters.
- REJECTED Configuration Parameter
  The REJECTED value disables the security service, even if the other side requires this service.

- ACCEPTED Configuration Parameter
  The ACCEPTED value enables the security service if the other side requires or requests the service.
- REQUESTED Configuration Parameter
  The REQUESTED value enables the security service if the other side permits this service.
- REQUIRED Configuration Parameter
  The REQUIRED value enables the security service or preclude the connection.

20.6.2.1 About the Values for Negotiating Encryption and Integrity

Oracle Net Manager can be used to specify four possible values for the encryption and integrity configuration parameters.

The following four values are listed in the order of increasing security, and they must be used in the profile file (sqlnet.ora) for the client and server of the systems that are using encryption and integrity.

The value REJECTED provides the *minimum* amount of security between client and server communications, and the value REQUIRED provides the *maximum* amount of network security:

- REJECTED
- ACCEPTED
- REQUESTED
- REQUIRED

The default value for each of the parameters is ACCEPTED.

Oracle Database servers and clients are set to ACCEPT encrypted connections out of the box. This means that you can enable the desired encryption and integrity settings for a connection pair by configuring just one side of the connection, server-side or client-side.

So, for example, if there are many Oracle clients connecting to an Oracle database, you can configure the required encryption and integrity settings for all these connections by making the appropriate sqlnet.ora changes at the server end. You do not need to implement configuration changes for each client separately.

Table 20-5 shows whether the security service is enabled, based on a combination of client and server configuration parameters. If either the server or client has specified REQUIRED, the lack of a common algorithm *causes the connection to fail.* Otherwise, if the service is enabled, lack of a common service algorithm results in the service being *disabled*.

Table 20-5 Encryption and Data Integrity Negotiations

Client Setting	Server Setting	Encryption and Data Negotiation
REJECTED	REJECTED	OFF
ACCEPTED	REJECTED	OFF
REQUESTED	REJECTED	OFF
REQUIRED	REJECTED	Connection fails
REJECTED	ACCEPTED	OFF
ACCEPTED	ACCEPTED	OFF[1]
REQUESTED	ACCEPTED	ON

Table 20-5 (Cont.) Encryption and Data Integrity Negotiations

Client Setting	Server Setting	Encryption and Data Negotiation
REQUIRED	ACCEPTED	ON
REJECTED	REQUESTED	OFF
ACCEPTED	REQUESTED	ON
REQUESTED	REQUESTED	ON
REQUIRED	REQUESTED	ON
REJECTED	REQUIRED	Connection fails
ACCEPTED	REQUIRED	ON
REQUESTED	REQUIRED	ON
REQUIRED	REQUIRED	ON

[1] This value defaults to OFF. Cryptography and data integrity are not enabled until the user changes this parameter by using Oracle Net Manager or by modifying the sqlnet.ora file.

20.6.2.2 REJECTED Configuration Parameter

The REJECTED value disables the security service, even if the other side requires this service.

In this scenario, this side of the connection specifies that the security service is not permitted. If the other side is set to REQUIRED, the connection *terminates* with error message ORA-12650. If the other side is set to REQUESTED, ACCEPTED, or REJECTED, the connection continues without error and without the security service enabled.

20.6.2.3 ACCEPTED Configuration Parameter

The ACCEPTED value enables the security service if the other side requires or requests the service.

In this scenario, this side of the connection does not require the security service, but it is enabled if the other side is set to REQUIRED or REQUESTED. If the other side is set to REQUIRED or REQUESTED, and an encryption or integrity algorithm match is found, the connection continues without error and with the security service enabled. If the other side is set to REQUIRED and no algorithm match is found, the connection terminates with error message ORA-12650.

If the other side is set to REQUESTED and no algorithm match is found, or if the other side is set to ACCEPTED or REJECTED, the connection continues without error and without the security service enabled.

20.6.2.4 REQUESTED Configuration Parameter

The REQUESTED value enables the security service if the other side permits this service.

In this scenario, this side of the connection specifies that the security service is desired but not required. The security service is enabled if the other side specifies ACCEPTED, REQUESTED, or REQUIRED. There must be a matching algorithm available on the other side, otherwise the service is not enabled. If the other side specifies REQUIRED and there is no matching algorithm, *the connection fails.*

20.6.2.5 REQUIRED Configuration Parameter

The `REQUIRED` value enables the security service or preclude the connection.

In this scenario, this side of the connection specifies that the security service *must be enabled*. The connection *fails* if the other side specifies `REJECTED` or if there is no compatible algorithm on the other side.

20.6.3 Configuring Encryption and Integrity Parameters Using Oracle Net Manager

You can set up or change encryption and integrity parameter settings using Oracle Net Manager.

- Configuring Encryption on the Client and the Server
  Use Oracle Net Manager to configure encryption on the client and on the server.
- Configuring Integrity on the Client and the Server
  You can use Oracle Net Manager to configure network integrity on both the client and the server.
- Enabling Both Oracle Native Encryption and SSL Authentication for Different Users Concurrently
  Depending on the `SQLNET.ENCRYPTION_CLIENT` and `SQLNET.ENCRYPTION_SERVER` settings, you can configure Oracle Database to allow both Oracle native encryption and SSL authentication for different users concurrently.

20.6.3.1 Configuring Encryption on the Client and the Server

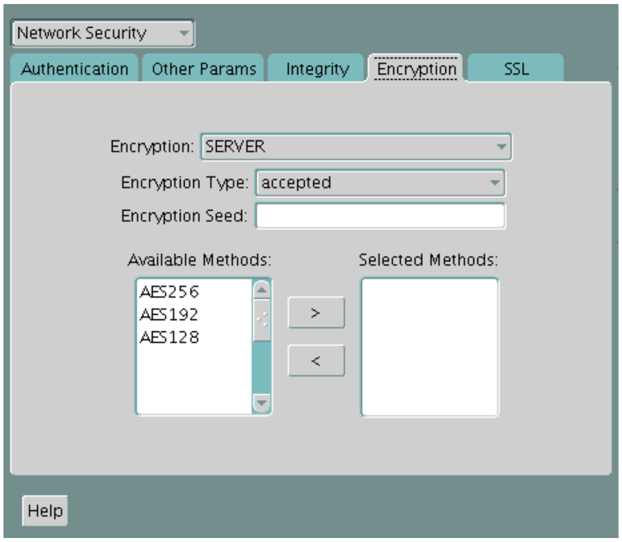
Use Oracle Net Manager to configure encryption on the client and on the server.

1. Start Oracle Net Manager.
   - (UNIX) From *`$ORACLE_HOME`*`/bin`, enter the following command at the command line:

     ```
     netmgr
     ```

   - (Windows) Select **Start**, **Programs**, **Oracle - HOME_NAME**, **Configuration and Migration Tools**, then **Net Manager**.
2. Expand **Oracle Net Configuration**, and from **Local**, select **Profile**.
3. From the **Naming** list, select **Network Security**.

   The Network Security tabbed window appears.
4. Select the **Encryption** tab.



Encryption: SERVER

Encryption Type: accepted

Encryption Seed:

5. Select **CLIENT** or **SERVER** option from the **Encryption** box.
6. From the Encryption Type list, select one of the following:
   - **REQUESTED**
   - **REQUIRED**
   - **ACCEPTED**
   - **REJECTED**
7. (Optional) In the **Encryption Seed** field, enter between 10 and 70 random characters. The encryption seed for the client should not be the same as that for the server.
8. Select an encryption algorithm in the **Available Methods** list. Move it to the **Selected Methods** list by choosing the right arrow (**>**). Repeat for each additional method you want to use.
9. Select **File**, **Save Network Configuration**. The `sqlnet.ora` file is updated.
10. Repeat this procedure to configure encryption on the other system. The `sqlnet.ora` file on the two systems should contain the following entries:
    - On the server:

      ```
      SQLNET.ENCRYPTION_SERVER = [accepted | rejected | requested | required]
      SQLNET.ENCRYPTION_TYPES_SERVER = (valid_encryption_algorithm
      [,valid_encryption_algorithm])
      ```

    - On the client:

      ```
      SQLNET.ENCRYPTION_CLIENT = [accepted | rejected | requested | required]
      SQLNET.ENCRYPTION_TYPES_CLIENT = (valid_encryption_algorithm
      [,valid_encryption_algorithm])
      ```

Table 20-6 lists valid encryption algorithms and their associated legal values.

Table 20-6 Valid Encryption Algorithms

Algorithm Name	Legal Value
AES 256-bit key	AES256
AES 192-bit key	AES192

Table 20-6 (Cont.) Valid Encryption Algorithms

Algorithm Name	Legal Value
AES 128-bit key	AES128

20.6.3.2 Configuring Integrity on the Client and the Server

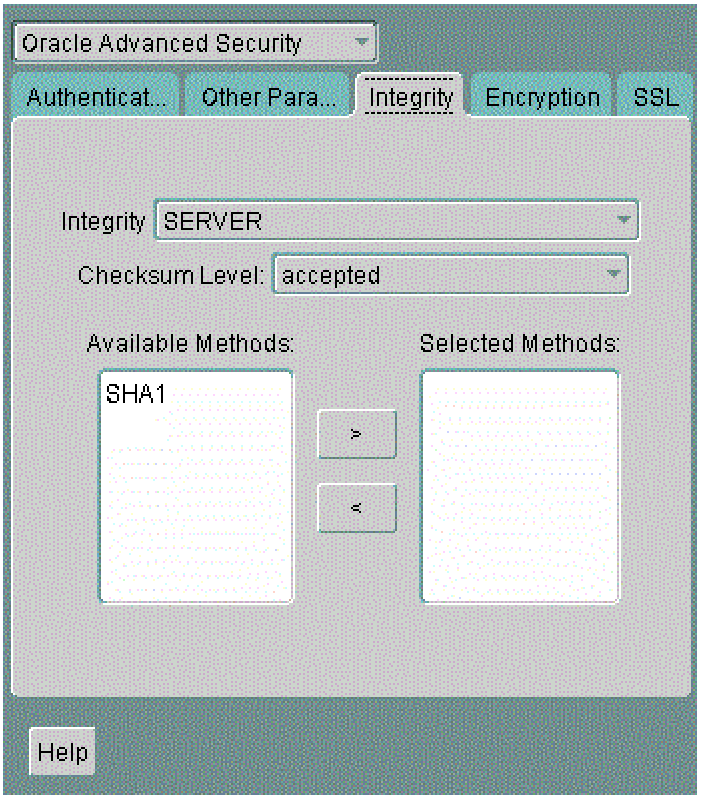
You can use Oracle Net Manager to configure network integrity on both the client and the server.

1. Start Oracle Net Manager.
   - (UNIX) From *$ORACLE_HOME*/bin, enter the following command at the command line:

     ```
     netmgr
     ```

   - (Windows) Select **Start**, **Programs**, **Oracle - HOME_NAME**, **Configuration and Migration Tools**, then **Net Manager**.
2. Expand **Oracle Net Configuration**, and from **Local**, select **Profile**.
3. From the **Naming** list, select **Network Security**.

   The Network Security tabbed window appears.
4. Select the **Integrity** tab.



Integrity SERVER

Checksum Level: accepted

Available Methods: SHA1

Selected Methods:

5. Depending upon which system you are configuring, select the **Server** or **Client** from the **Integrity** box.
6. From the **Checksum Level** list, select one of the following checksum level values:

  - **REQUESTED**
  - **REQUIRED**
  - **ACCEPTED**
  - **REJECTED**

7. Select an integrity algorithm in the **Available Methods** list. Move it to the **Selected Methods** list by choosing the right arrow (**>**). Repeat for each additional method you want to use.
8. Select **File**, **Save Network Configuration**.

   The `sqlnet.ora` file is updated.
9. Repeat this procedure to configure integrity on the other system.

   The `sqlnet.ora` file on the two systems should contain the following entries:

   - On the server:

     ```
     SQLNET.CRYPTO_CHECKSUM_SERVER = [accepted | rejected | requested | required]
     SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (valid_crypto_checksum_algorithm
     [,valid_crypto_checksum_algorithm])
     ```

   - On the client:

     ```
     SQLNET.CRYPTO_CHECKSUM_CLIENT = [accepted | rejected | requested | required]
     SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (valid_crypto_checksum_algorithm
     [,valid_crypto_checksum_algorithm])
     ```

Valid integrity/checksum algorithms that you can use are as follows:

- SHA1
- SHA256
- SHA384
- SHA512

Related Topics

- *Oracle Database Advanced Security Guide*

20.6.3.3 Enabling Both Oracle Native Encryption and SSL Authentication for Different Users Concurrently

Depending on the `SQLNET.ENCRYPTION_CLIENT` and `SQLNET.ENCRYPTION_SERVER` settings, you can configure Oracle Database to allow both Oracle native encryption and SSL authentication for different users concurrently.

- About Enabling Both Oracle Native Encryption and SSL Authentication for Different Users Concurrently
  By default, Oracle Database does not allow both Oracle native encryption and Transport Layer Security (SSL) authentication for different users concurrently.
- Configuring Both Oracle Native Encryption and SSL Authentication for Different Users Concurrently
  Use the `IGNORE_ANO_ENCRYPTION_FOR_TCPS` parameter to enable the concurrent use of both Oracle native encryption and Transport Layer Security (SSL) authentication.

20.6.3.3.1 About Enabling Both Oracle Native Encryption and SSL Authentication for Different Users Concurrently

By default, Oracle Database does not allow both Oracle native encryption and Transport Layer Security (SSL) authentication for different users concurrently.

The use of both Oracle native encryption (also called Advanced Networking Option (ANO) encryption) and TLS authentication together is called double encryption.

There are cases in which both a TCP and TCPS listener must be configured, so that some users can connect to the server using a user name and password, and others can validate to the server by using a TLS certificate. In these situations, you must configure both password-based authentication and TLS authentication. A workaround in previous releases was to set the `SQLNET.ENCRYPTION_SERVER` parameter to `requested`. If your requirements are that `SQLNET.ENCRYPTION_SERVER` be set to `required`, then you can set the `IGNORE_ANO_ENCRYPTION_FOR_TCPS` parameter in both `SQLNET.ENCRYPTION_CLIENT` and `SQLNET.ENCRYPTION_SERVER` to `TRUE`. By default, it is set to `FALSE`.

Setting `IGNORE_ANO_ENCRYPTION_FOR_TCPS` to `TRUE` forces the client to ignore the value that is set for the `SQLNET.ENCRYPTION_CLIENT` parameter for all outgoing TCPS connections. This parameter allows the database to ignore the `SQLNET.ENCRYPTION_CLIENT` or `SQLNET.ENCRYPTION_SERVER` setting when there is a conflict between the use of a TCPS client and when these two parameters are set to `required`.

20.6.3.3.2 Configuring Both Oracle Native Encryption and SSL Authentication for Different Users Concurrently

Use the `IGNORE_ANO_ENCRYPTION_FOR_TCPS` parameter to enable the concurrent use of both Oracle native encryption and Transport Layer Security (SSL) authentication.

On the server, you must set `IGNORE_ANO_ENCRYPTION_FOR_TCPS` in the `sqlnet.ora` file, and on the client, you can set it in either the `sqlnet.ora` file or the `tnsnames.ora` file.

1. Log in to the database server
2. Go to the location of the `sqlnet.ora` file.

   By default, `sqlnet.ora` is in the `*ORACLE_BASE*/network/admin` directory. The `sqlnet.ora` file can also be stored in the directory specified by the `TNS_ADMIN` environment variable.
3. In `sqlnet.ora`, check if the current `SQLNET.ENCRYPTION_SERVER` setting is `required` or `requested`.
4. If `SQLNET.ENCRYPTION_SERVER` is set to required, then add the `SQLNET.IGNORE_ANO_ENCRYPTION_FOR_TCPS` to `sqlnet.ora` and then set it to `TRUE`.

   ```
   IGNORE_ANO_ENCRYPTION_FOR_TCPS=TRUE
   ```
5. Save and exit `sqlnet.ora`.
6. Log in to the client.

   For the client, you can set the value in either the `sqlnet.ora` file or the `tnsnames.ora` file.

   - Setting the value in `sqlnet.ora`: Check if the `SQLNET.ENCRYPTION_CLIENT` parameter is set to `required`. If `SQLNET.ENCRYPTION_CLIENT`, then edit the `sqlnet.ora` file to have the following setting:

     ```
     IGNORE_ANO_ENCRYPTION_FOR_TCPS=TRUE
     ```

- Setting the value in tnsnames.ora: By default, tnsnames.ora is in the same location as sqlnet.ora. If SQLNET.ENCRYPTION_CLIENT is set to required in sqlnet.ora, then in the SECURITY portion of the TNS_ALIAS setting, set IGNORE_ANO_ENCRYPTION_FOR_TCPS=TRUE. For example:

  ```
  test_tls=
      (DESCRIPTION =
         (ADDRESS=(PROTOCOL=tcps)(HOST=)(PORT=1750))
         (CONNECT_DATA=(SID=^ORACLE_SID^))
         (SECURITY=(IGNORE_ANO_ENCRYPTION_FOR_TCPS=TRUE))
       )
  ```

20.7 Troubleshooting the Native Network Encryption Configuration

Oracle provides guidance for common native network encryption configuration problems.

- Checking if Native Network Encryption Is Enabled in the Current Session
  Depending on how the encryption parameters are set in the server and client sqlnet.ora file, you can check if native network encryption is enabled if the current session.
- ORA-12650 and ORA-12660 Errors in the Native Network Encryption Configuration
  Oracle provides several solutions for ORA-12650 and ORA-12660 errors that can occur in a native network encryption configuration.

20.7.1 Checking if Native Network Encryption Is Enabled in the Current Session

Depending on how the encryption parameters are set in the server and client sqlnet.ora file, you can check if native network encryption is enabled if the current session.

1. On the server, check ENCRYPTION_SERVER and ENCRYPTION_TYPES_SERVER parameters.

   For example:

   ```
   sqlnet.encryption_server = required
   sqlnet.encryption_types_server = AES256
   ```

   By default, sqlnet.ora is located in the $ORACLE_HOME/network/admin directory, for both the server and the client.
2. On the client, check the ENCRYPTION_SERVER and ENCRYPTION_TYPES_CLIENT parameters.

   For example:

   ```
   sqlnet.encryption_server = required
   sqlnet.encryption_types_client = AES256
   ```

3. From a client that has been configured with native network encryption for database connections, query the V$SESSION_CONNECT_INFO dynamic view.

   For example:

   ```
   set line 1000
   col NETWORK_SERVICE_BANNER for a100
   ```

```
SELECT NETWORK_SERVICE_BANNER FROM V$SESSION_CONNECT_INFO WHERE
SID=(SELECT SID FROM V$MYSTAT WHERE ROWNUM<2);
```

If the connection is unencrypted, then output similar to the following appears:

```
NETWORK_SERVICE_BANNER
---------------------------------------------------------------------------
-----
TCP/IP NT Protocol Adapter for Linux: Version version_number - Production
Authentication service for Linux: Version version_number - Production
KERBEROS5PRE Authentication service adapter for Linux: Version
version_number - Production
Encryption service for Linux: Version version_number - Production
Crypto-checksumming service for Linux: Version version_number - Production
```

However, if the connection is encrypted, then output similar to the following appears. The additional line in bold (`AES256 Encryption service adapter for Linux`) indicates that the connection is encrypted.

```
NETWORK_SERVICE_BANNER
---------------------------------------------------------------------------
-----
TCP/IP NT Protocol Adapter for Linux: Version version_number - Production
Authentication service for Linux: Version version_number - Production
KERBEROS5PRE Authentication service adapter for Linux: Version
version_number - Production
Encryption service for Linux: Version version_number - Production
AES256 Encryption service adapter for Linux: Version version_number -
Production
Crypto-checksumming service for Linux: Version version_number - Production
```

20.7.2 ORA-12650 and ORA-12660 Errors in the Native Network Encryption Configuration

Oracle provides several solutions for `ORA-12650` and `ORA-12660` errors that can occur in a native network encryption configuration.

The `ORA-12650: No common encryption or data integrity algorithm` and `ORA-12660: Encryption or crypto-checksumming parameters incompatible` errors are caused only when you set `SQLNET.ENCRYPTION_CLIENT` and `SQLNET.ENCRYPTION_SERVER` to `rejected` on each side (client and server). They can also occur if there is a misconfiguration in the `sqlnet.ora` file.

To remedy this problem, do the following

- Check the settings in the `sqlnet.ora` file on both the client and the server.
- If the `sqlnet.ora` settings look correct, then check the `PATH` and `TNS_ADMIN` environment variables.
- Look for any additional `sqlnet.ora` files that may be in the client and server directory tree.
- If the settings of `sqlnet.ora` and the actual behavior are different, and if you cannot find any specific incongruities in the `sqlnet.ora` file, then perform a net trace level 16 both in server side and client side.