Yann Neuhaus, dbi services

Oracle Net encryption – How to

This article presents how to configure the Oracle*Net encryption with Oracle 11g. It summarizes and consolidates several sources of information found on the web and in Oracle documentation in order to reach easily a successful setup which can be easily tested between an Oracle client and a Server.

It is worth to mention that Oracle does not require anymore to license the Advanced Security Option (ASO) in order to encrypt the client/server connection to the database, see:

http://docs.oracle.com/cd/E11882_01/license.112/e10594/editions.htm#DBLIC119

"Strong authentication services (Kerberos, PKI and RADIUS) and network encryption (native network encryption and SSL/TLS) are no longer part of Oracle Advanced Security and are available in all licensed editions of all supported releases of the Oracle database."

"Oracle supports both SSL/TLS and a native network encryption capability that can be turned on by simply modifying the Oracle sqlnet.ora configuration file. Supported encryption algorithms include AES and 3DES with key sizes up to 256."

Basics and theory

In order to protect data transferred through the of Oracle layer, Oracle Advanced Security provides the Advanced Encryption Standard algorithm (AES) with three standard key lengths: 128-bit, 192-bit and 256-bit. It also proposes the Triple-DES (3DES) symmetric cryptosystems algorithm.

Triple-DES Support provides a high degree of message security, but with a performance penalty.

Since Oracle Advanced Security 11g Release 1 (11.1) an RC4 implementation with 40-bit, 56-bit, 128-bit and 256-bit key lengths is provided. This algorithm offers backward-compatibility and strong encryption, with no material performance compromise.

Oracle Advanced Security also guarantees data integrity by providing hash algorithms which create a checksum that changes if the data is altered in any way.

When encryption is used to protect the security of encrypted data, keys must be changed frequently to minimize the effects of a compromised key. Oracle Advanced Security uses the well-known Diffie-Hellman key negotiation algorithm to perform secure key distribution for both encryption and data integrity.

Setup

In order to perform an easy setup, not disturbing the current configuration we decided to create a “separated” Oracle “environment” using a TNS_ADMIN variable pointing to a temporary Oracle*Net configuration directory. In this directory we are able on client and server side to modify the sqlnet.ora, tnsnames.ora and listener.ora files without disturbing the existing configuration.

Add the new environment in the oratab on both Oracle client and server machines:

```
rdbms11204t:/u01/app/oracle/product/11.2.0.4/db_1:D
```

Create a new Oracle*Net directory structure, for instance under $HOME from Oracle:

```
racle@client:/home/oracle/ [rdbms11204] mkdir -p network/admin
oracle@client:/home/oracle/ [rdbms11204] mkdir network/log
oracle@client:/home/oracle/ [rdbms11204] mkdir network/trc
```

Configure the environment framework to set TNS_ADMIN to a separate directory (using the dbi services Database Management Kit (DMK), this work with a configuration file):

```
oracle@client:/u01/app/oracle/local/dmk/ [rdbms11204] vi etc/dmk.conf
```

Add:

```
[rdbms11204t]
var::TNS_ADMIN::=::nowarn::"/home/oracle/network/admin"::# location of the
temporary Oracle*Net config (ASO tests)
```

Source the environment for rdbms11204t :

```
oracle@client:/u01/app/oracle/local/dmk/ [rdbms11204] . dmk.bash
oracle@client:/u01/app/oracle/local/dmk/ [rdbms11204] rdbms11204t
oracle@client:/u01/app/oracle/local/dmk/ [rdbms11204t]
```

Verify your settings:

```
oracle@client:/home/oracle/ [rdbms11204t] echo $TNS_ADMIN
/home/oracle/network/admin
oracle@client:/home/oracle/ [rdbms11204t] cd $TNS_ADMIN
oracle@client:/home/oracle/network/admin/ [rdbms11204t] ls
```

On the client side a tnsnames.ora and sqlnet.ora files will be created
On the server side a listener.ora and sqlnet.ora files will be created

OracleNet encryption and integrity configuration and negotiation

On both client and server side you can define four levels of encryption and integrity check: REJECTED, ACCEPTED, REQUESTED, REQUIRED

REJECTED is the less secure (non encryption/integrity check will be activated), REQUIRED is the most secure (only encrypted connections are possible).

The default value is ACCEPTED (when nothing is set), meaning that per default clients do "accept" encrypted connections when the listener is configured with encryption.

These 4 levels can also be configured on the server side.

Client / Server	Rejected	Accepted	Requested	Required
Rejected	OFF	OFF	OFF	ORA-12660
Accepted	OFF	OFF	ON	ON
Requested	OFF	ON	ON	ON
Required	ORA-12660	ON	ON	ON

In order to configure the encryption two main parameters have to be set on the client and server side:

```
SQLNET.ENCRYPTION_CLIENT = [accepted | rejected | requested | required]
```

This parameter sets the level of “restriction” concerning the activation of an encrypted session.

```
SQLNET.ENCRYPTION_TYPES_CLIENT = (valid_encryption_algorithm [,valid_
encryption_algorithm])
```

This parameter sets the possible algorithm to be used to encrypt the session.

On the server side, to make sure only encrypted connection will be established, you will have to set “SQLNET.ENCRYPTION_CLIENT = required” in the sqlnet.ora to force the listener to accept only encrypted connections.

Let's perform a test from the client side, while setting a secured connection with the RC4_256 algorithm. Oracle recommends RC4_256 for performance reason, some tests found on the web show that this algorithm impacts the less the performances.

From the Oracle doc : “Oracle Advanced Security 11g Release 1 (11.1) provides an RC4 implementation with 40-bit, 56-bit, 128-bit and 256-bit key lengths. This provides backward-compatibility and strong encryption, with no material performance compromise.”

Client configuration (extract of the sqlnet.ora on client side) :

```
#####################################################################
## Encryption and integrity configuration - client side
#####################################################################

SQLNET.ENCRYPTION_TYPES_SERVER = RC4_256
SQLNET.ENCRYPTION_CLIENT = accepted

SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (MD5)
SQLNET.CRYPTO_CHECKSUM_CLIENT = accepted
```

The only proposed algorithm for encryption is RC4_256. The client will accept (default) the encryption depending on the server behavior (see table above).

On the server side, configure the same algorithm and for the encryption and checksum usage:

```
oracle@server:/home/oracle/network/admin [rdbms11204t] cat sqlnet.ora

........
#####################################################################
## Encryption
#####################################################################
SQLNET.ENCRYPTION_TYPES_SERVER = RC4_256
SQLNET.ENCRYPTION_SERVER = required

SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (MD5)
SQLNET.CRYPTO_CHECKSUM_CLIENT = required
```

The listener is a simple listener configuration without particular parameters:

```
#####################################################################
## Listener 11.2.0.4 for encryption tests
#####################################################################
LISTENER_11204S =
  (ADDRESS_LIST =
    (ADDRESS = ( PROTOCOL = TCP )( Host = server.customer.ch )( Port =
1741 ))
  )

SID_LIST_LISTENER_11204S =
    (SID_LIST =
        (SID_DESC =
            (ORACLE_HOME    = /u01/app/oracle/product/11.2.0.4/db_1 )
            (GLOBAL_DBNAME = SLEQAS1_SITE1_DGMGRL.customer.ch )
            (SID_NAME       = SLEQAS1 )
        )
    )
```

Let's try to connect:

```
oracle@client:/home/oracle/network/admin/ [rdbms11204t]
sqlplus scott@SLEQAS1_SITE1.CUSTOMER.CH

SQL> select * from emp;
```

On the server side we check with tcpdump if the connection is really encrypted :

```
server:~ # tcpdump -nnvvXSs0 -i eth0 host 172.26.92.11 and port 1741
.....
        0x00f0:  e23f 14e5 005f 2054 4473 21ea aa0a e218  .?..._.TDs!.....
        0x0100:  ae64 44b3 6170 be69 4963 e736 38e0 5e88  .dD.ap.iIc.68.^.
        0x0110:  f303 a33e d9d5 956d 6437 9f0d 9476 1f5c  ...>...md7...v.\
        0x0120:  4574 1f45 ce9c 4bc8 348f 0dca 8114 9cfc  Et.E..K.4.......
        0x0130:  fd9d 56bc e43e 8690 60d6 1c02 d358 a4cc  ..V..>..`....X..
        0x0140:  ada8 c4de c0e9 3d76 69bb f365 96e7 a8c5  ......=vi..e....
        0x0150:  d17e d26d b1a0 7c16 7d15 7dd6 2fee 50cc  .~.m..|.}.}./.P.
        0x0160:  ad20 df3e 57a2 7660 edf1 a086 0233 b460  ...>W.v`.....3.`
        0x0170:  4886 a0d4 b191 1e0c 6d26 386b b15e e8ac  H.......m&8k.^..
        0x0180:  9454 f69d 0cd2 ee25 df0b 5b18 1ad5 28e8  .T.....%..[...(.
        0x0190:  b5fe 9ce4 cf0c acea e172 1c9a e0cd e38a  .........r......
        0x01a0:  6336 add6 072c 2039 5939 8e84 b4e7 8f21  c6...,.9Y9.....!
        0x01b0:  b564 28f7 4a40 d2ea e44a e25a 0552 740f  .d(.J@...J.Z.Rt.
        0x01c0:  6205 1c63 e8f4 e57c a61f 0454 4dbe 4086  b..c...|...TM.@.
        0x01d0:  2f3c 4fcb 2cc7 4ae2 bde9 83a1 f482 6f39  /<O.,.J.......o9
        0x01e0:  7c12 dd6c 0260 4c11 e333 3187 4267 f80c  |..l.`L..31.Bg..
        0x01f0:  2fb0 7851 08e0 f606 b82d b282 eb95 dc52  /.xQ.....-.....R
        0x0200:  0b11 6eb3 52a4 996d 71cc 34e0 b9ae f170  ..n.R..mq.4....p
        0x0210:  0d83 7e87 b854 eb82 b104 ea05 46c2 7057  ..~..T......F.pW
        0x0220:  dbf9 15ed 4926 afef 2767 a199 d100 ccb5  ....I&..'g......
        0x0230:  d3e7 eb3c 2369 c1d5 542f 00c1 4525 ed59  ...<#i..T/..E%.Y
        0x0240:  4c6b 9e76 9907 098d 8558 4303 e2f9 9e60  Lk.v.....XC....`

.....
```

The following tcpdump options have been used:

-n : Don't convert addresses (i.e., host addresses, port numbers, etc.) to names
-v, -vv, -vvv : Increase the amount of packet information you get back
-X : Show the packet's contents in both hex and ASCII
-s : Define the snaplength (size) of the capture in bytes. Use -s0 to get everything, unless you are intentionally capturing less
-S : Print absolute sequence numbers

Another way to make sure the data is encrypted, with the "SCOTT" session:

```
SQL> select NETWORK_SERVICE_BANNER
     from v$session_connect_info
     where SID = sys_context('USERENV','SID')
     /

NETWORK_SERVICE_BANNER
----------------------------------------------------------------------
TCP/IP NT Protocol Adapter for Linux: Version 11.2.0.4.0 - Production
Oracle Advanced Security: encryption service for Linux: Version 11.2.0.4.0
- Production
Oracle Advanced Security: RC4_256 encryption service adapter for Linux:
Version 11.2.0.4.0 - Produc
Oracle Advanced Security: crypto-checksumming service for Linux: Version
11.2.0.4.0 - Production
```

When de-activating the encryption on the server side, the data is sent in “plain-text” and any hacker can read it with tcpdump :

```
server:~ # tcpdump -nnvvXSs0 -i eth0 host 172.26.92.11 and port 1741

.......
        0x00f0:  4cff 02c1 1507 2d2c 0108 03c2 4d37 064d  L.....-,....M7.M
        0x0100:  4152 5449 4e08 5341 4c45 534d 414e 03c2  ARTIN.SALESMAN..
        0x0110:  4d63 0777 b509 1c01 0101 03c2 0d33 02c2  Mc.w.........3..
        0x0120:  0f02 c11f 0729 2c01 0803 c24d 6305 424c  .....),....Mc.BL
        0x0130:  414b 4507 4d41 4e41 4745 5203 c24f 2807  AKE.MANAGER..O(.
        0x0140:  77b5 0501 0101 0103 c21d 33ff 02c1 1f07  w.........3.....
        0x0150:  292c 0108 03c2 4e53 0543 4c41 524b 074d  ),....NS.CLARK.M
        0x0160:  414e 4147 4552 03c2 4f28 0777 b506 0901  ANAGER..O(.w....
        0x0170:  0101 03c2 1933 ff02 c10b 0728 2c01 0803  .....3.....(,...
        0x0180:  c24e 5905 5343 4f54 5407 414e 414c 5953  .NY.SCOTT.ANALYS
        0x0190:  5403 c24c 4307 77bb 0413 0101 0102 c21f  T..LC.w.........
        0x01a0:  ff02 c115 0726 2c01 0803 c24f 2804 4b49  .....&,....O(.KI
        0x01b0:  4e47 0950 5245 5349 4445 4e54 ff07 77b5  NG.PRESIDENT..w.
        0x01c0:  0b11 0101 0102 c233 ff02 c10b 072b 2c01  .......3.....+,.
        0x01d0:  0803 c24f 2d06 5455 524e 4552 0853 414c  ...O-.TURNER.SAL
        0x01e0:  4553 4d41 4e03 c24d 6307 77b5 0908 0101  ESMAN..Mc.w.....
        0x01f0:  0102 c210 0180 02c1 1f07 262c 0108 03c2  ..........&,....
        0x0200:  4f4d 0541 4441 4d53 0543 4c45 524b 03c2  OM.ADAMS.CLERK..
        0x0210:  4e59 0777 bb05 1701 0101 02c2 0cff 02c1  NY.w............
        0x0220:  1507 262c 0108 02c2 5005 4a41 4d45 5305  ..&,....P.JAMES.
        0x0230:  434c 4552 4b03 c24d 6307 77b5 0c03 0101  CLERK..Mc.w.....
        0x0240:  0103 c20a 33ff 02c1 1f07 272c 0108 03c2  ....3.....',....
        0x0250:  5003 0446 4f52 4407 414e 414c 5953 5403  P..FORD.ANALYST.
        0x0260:  c24c 4307 77b5 0c03 0101 0102 c21f ff02  .LC.w...........
        0x0270:  c115 0727 2c01 0803 c250 2306 4d49 4c4c  ...',....P#.MILL
        0x0280:  4552 0543 4c45 524b 03c2 4e53 0777 b601  ER.CLERK..NS.w..
.......
```

Conclusion

Reading the first documentation around Oracle*Net encryption, creating a test infrastructure, performing some tests and writing this article cost me about 1/2 day of work. I can state that activating this feature is not so complex and difficult, hopefully Oracle stopped to require a license for this feature. Nowadays, customers have to encrypt data transiting on the networks without additional costs. ■

Sources	
Oracle doc :	http://docs.oracle.com/cd/E11882_01/network.112/e10746/asoconfg.htm#ASOAG020
How do I encrypt network traffic to an Oracle instance?	https://kb.berkeley.edu/page.php?id=23274
Check encryption :	http://dba.stackexchange.com/questions/31847/verifying-oraclenet-network-encryption
Check trafic with tcpdump :	http://jonathanmanning.com/2010/04/08/how-to-tcpdump-specific-ip-address-and-port-number/
Dump all , encrypted :	http://www.thegeekstuff.com/2010/08/tcpdump-command-examples/
All about tcpdump	http://danielmiessler.com/study/tcpdump/
Oracle Advanced Security, Client Access Control and SSH Tunnelling	http://www.easysoft.com/products/data_access/odbc_oracle_driver/security.html http://www.orafaq.com/wiki/Network_Encryption => Performance figures of the algorithms.

Contact

dbi services

Yann Neuhaus
E-Mail:
yann.neuhaus@dbi-services.com