

## Introduction

Oracle10g Release2 has introduced a new feature called **Transparent Data Encryption (TDE)**. TDE is beneficial for simple and easy encryption of sensitive data in table columns. Simple and easy because users or applications need not manage the encryption and decryption of data any more, it is handled by the database - so there is no need to manage views, tables, or triggers to decrypt data. The encryption keys are stored in a location (file named ewallet.p12) external to the database. This location can be either OS-specific or location specified in sqlnet.ora file using parameter WALLET\_LOCATION.

TDE can be used to protect confidential data like credit card information, social security number, etc. The data encrypted by TDE cannot be accessed unless authorized decryption occurs, which is automatic for users authorized to access the tables. For example, even if the disks are stolen, the data is safe as the encrypted table columns cannot be viewed unless the master key (stored in ewallet.p12) is provided. The master key is password provided while creating the wallet.

## Enabling Transparent Data Encryption

To use TDE, user must have "alter system" privilege and a valid password for Oracle wallet. If an Oracle wallet (ewallet.p12) does not exist, then a new one is created using the password specified in the SQL command.

```
SQL*Plus: Release 10.2.0.1.0 - Production on Sat Aug 6 17:24:38 2005
```

```
Copyright (c) 1982, 2005, Oracle. All rights reserved.
```

```
Connected to:
```

```
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
```

```
With the Partitioning, OLAP and Data Mining options
```

```
sys@ORCL.WORLD> alter system set encryption key identified by "zaq12wsx";
```

```
System altered.
```

This SQL command creates an encrypted wallet named ewallet.p12, opens the wallet and creates the database server master encryption key for TDE. If the wallet already exists it opens the wallet and creates or recreates the database server master encryption key for TDE.

For creating a wallet a folder named **wallet** should be present in the directory **\$ORACLE\_BASE/admin/\$ORACLE\_SID** otherwise Oracle complains with **ORA-28368: cannot auto-create wallet error**. After creating the folder wallet, in the above mentioned directory we create the wallet.

```
sys@ORCL.WORLD> alter system set encryption key identified by "zaq12wsx";
```

```
alter system set encryption key identified by "zaq12wsx"
```

```
*
```

```
ERROR at line 1:
```

ORA-28368: cannot auto-create wallet

## Using Transparent Data Encryption

Let's take an example of the EMP table from SCOTT schema.

We first create a copy of EMP table as

```
scott@ORCL.WORLD> create table emp1 as select * from emp;
```

Table created.

```
scott@ORCL.WORLD>
```

Then we encrypt the column DEPTNO using the SQL command

```
scott@ORCL.WORLD> alter table emp1 modify DEPTNO encrypt;
```

Table altered.

```
scott@ORCL.WORLD> select * from emp1;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	19-APR-87	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-87	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30

7902 FORD	ANALYST	7566 03-DEC-81	3000	20
7934 MILLER	CLERK	7782 23-JAN-82	1300	10

14 rows selected.

Here the data in column DEPTNO is encrypted in the database and is decrypted transparently when we fire a select query on the table. This can be tested by closing the wallet.

```
sys@ORCL.WORLD> alter system set wallet close;
```

System altered.

After closing the wallet, let's run the query again.

```
scott@ORCL.WORLD> select * from empl;
```

```
select * from empl
```

\*

ERROR at line 1:

ORA-28365: wallet is not open

```
scott@ORCL.WORLD> insert into empl values (1234,'AMAR','DBA',1111,'30-AUG-99', 1000,5000,10);
```

```
insert into empl values (1234,'AMAR','DBA',1111,'30-AUG-99', 1000,5000,10)
```

\*

ERROR at line 1:

ORA-28365: wallet is not open

We are not able to select or insert data into empl table column DEPTNO as the wallet is closed. This shows that the wallet should remain open when any DML operation has to be performed on the encrypted table column. This would not be the case if we perform DML operations on other columns while the wallet is closed, provided that the encrypted table column allows null values. After excluding the DEPTNO column from the select and insert statements we can retrieve data from columns which were not encrypted.

```
scott@ORCL.WORLD> select empno,ename,job,mgr,hiredate,sal,comm from empl;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM
7369	SMITH	CLERK	7902	17-DEC-80	800	

7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500
7566	JONES	MANAGER	7839	02-APR-81	2975	
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400
7698	BLAKE	MANAGER	7839	01-MAY-81	2850	
7782	CLARK	MANAGER	7839	09-JUN-81	2450	
7788	SCOTT	ANALYST	7566	19-APR-87	3000	
7839	KING	PRESIDENT		17-NOV-81	5000	
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0
7876	ADAMS	CLERK	7788	23-MAY-87	1100	
7900	JAMES	CLERK	7698	03-DEC-81	950	
7902	FORD	ANALYST	7566	03-DEC-81	3000	
7934	MILLER	CLERK	7782	23-JAN-82	1300	

14 rows selected.

```
scott@ORCL.WORLD> insert into empl values (1234,'AMAR','DBA', 1111,'30-AUG-99', 1000, 5000, NULL);
```

1 row created.

```
scott@ORCL.WORLD> commit;
```

Commit complete.

```
scott@ORCL.WORLD> select empno,ename,job,mgr,hiredate,sal,comm from empl;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM
7369	SMITH	CLERK	7902	17-DEC-80	800	
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500
7566	JONES	MANAGER	7839	02-APR-81	2975	
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400
7698	BLAKE	MANAGER	7839	01-MAY-81	2850	
7782	CLARK	MANAGER	7839	09-JUN-81	2450	

7788	SCOTT	ANALYST	7566	19-APR-87	3000	
7839	KING	PRESIDENT		17-NOV-81	5000	
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0
7876	ADAMS	CLERK	7788	23-MAY-87	1100	
7900	JAMES	CLERK	7698	03-DEC-81	950	
7902	FORD	ANALYST	7566	03-DEC-81	3000	
7934	MILLER	CLERK	7782	23-JAN-82	1300	
1234	AMAR	DBA	1111	30-AUG-99	1000	5000

15 rows selected.

All this encryption (during insert) and decryption (during select) of data is managed by the database without user or application intervention. Let's open the wallet again.

```
sys@ORCL.WORLD> alter system set encryption wallet open identified by "zaql2wsx";
```

System altered.

### Creating index on encrypted columns

By default columns are encrypted with Salt. Salt is a way to strengthen the security of encrypted data. It is a random string added to the data before it is encrypted, causing repetition of text in the clear to appear different when encrypted. Salt thus removes one method attackers use to steal data, namely, matching patterns of encrypted text.

```
scott@ORCL.WORLD> create index NU1_EMP1 on emp1 (deptno);
create index NU1_EMP1 on emp1 (deptno)
```

\*

ERROR at line 1:

ORA-28338: cannot encrypt indexed column(s) with salt

For indexing a column which has been encrypted, we need to remove salt from the column.

```
scott@ORCL.WORLD> alter table emp1 modify deptno encrypt no salt;
```

Table altered.

```
scott@ORCL.WORLD> create index NU1_EMP1 on emp1 (deptno);
```

Index created.

*Note: Salt cannot be added to already indexed columns.*

### **Supported Encryption Integrity for Transparent Data Encryption.**

Transparent data encryption uses the Advanced Encryption Standard with a 192-bit length cipher key (AES192). This is the default length. This length can be modified by using the REKEY command phrase.

```
scott@ORCL.WORLD> create table emp2 as select * from emp;
```

Table created.

```
scott@ORCL.WORLD> alter table emp2 modify DEPTNO encrypt;
```

Table altered.

```
scott@ORCL.WORLD> alter table emp1 rekey using 'AES256';
```

Table altered.

```
scott@ORCL.WORLD> select table_name,column_name,ENCRYPTION_ALG,SALT from user_encrypted_columns;
```

TABLE_NAME	COLUMN_NAME	ENCRYPTION_ALG	SAL
EMP1	DEPTNO	AES 256 bits key	NO
EMP2	DEPTNO	AES 192 bits key	YES

Other options available are 3DES168 and AES128.

### **Automatic Login Wallets**

By default, wallet closes when database instance is closed and has to be manually open when instance starts again using the set wallet open command.

```
sys@ORCL.WORLD> shutdown immediate
```

```
Database closed.
```

```
Database dismounted.
```

```
ORACLE instance shut down.
```

```
sys@ORCL.WORLD> startup
```

```
ORACLE instance started.
```

```
Total System Global Area 167772160 bytes
```

```
Fixed Size 1246852 bytes
```

```
Variable Size 62916988 bytes
```

```
Database Buffers 100663296 bytes
```

```
Redo Buffers 2945024 bytes
```

```
Database mounted.
```

```
Database opened.
```

```
sys@ORCL.WORLD> connect scott/tiger
```

```
Connected.
```

```
scott@ORCL.WORLD> select * from emp1;
```

```
select * from emp1
```

```
*
```

```
ERROR at line 1:
```

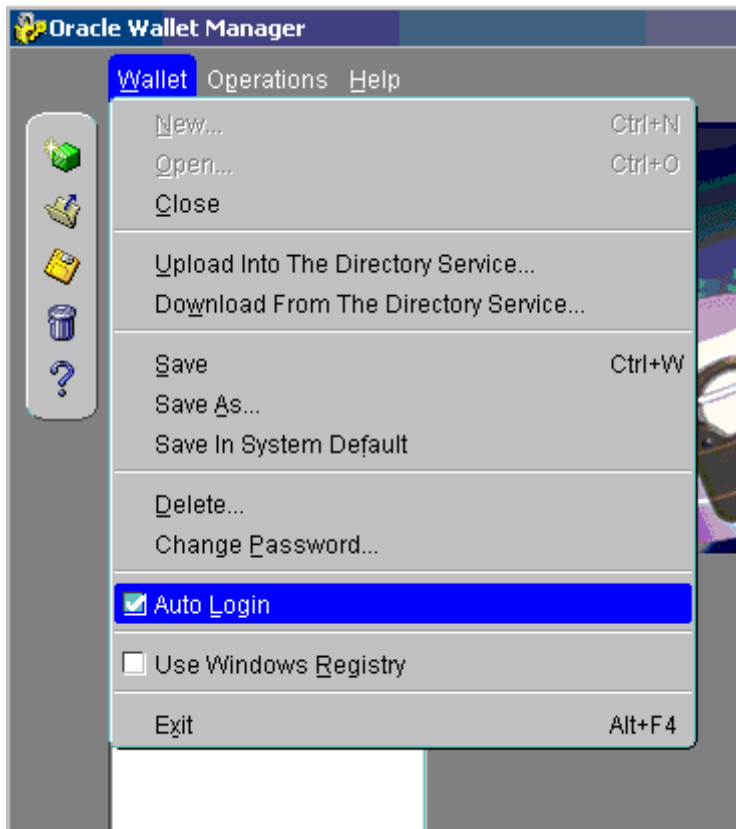
```
ORA-28365: wallet is not open
```

This can be overcome by using Auto Login Wallet. From the manual:

*Auto login wallets are protected by file system permissions. When auto login is enabled for a wallet, only the operating system user who created it can manage it, through the Oracle Wallet Manager.*

*You must enable auto login if you want single sign-on access to multiple Oracle databases: such access is normally disabled, by default. Sometimes the obfuscated auto login wallets are called "SSO wallets" because they support single sign-on capability.*

Auto Login can be enabled as shown below. From the wallet manager open the wallet (ewallet.p12) and again from file menu check the box for auto login and save the file. This will generate a **ewallet.sso** file. The wallet will remain open all the time.



### Location parameter in sqlnet.ora

Location of wallet can be changed if required. This can be done by entering the directory path for WALLET\_LOCATION parameter in sqlnet.ora file as shown below

```
WALLET_LOCATION = (SOURCE=
                    (METHOD = FILE)
                    (METHOD_DATA =
                     (DIRECTORY=/u02/wallets)
                    )))
```

After changing the directory path, make sure you copy the file/s ewallet.p12 and cwallet.sso (if auto login is enabled) to the new location.

### Performance Issues

During insert the data has to be encrypted and when retrieving, the data has to be decrypted before display. This will cause performance issues. Total performance effect will depend on the number of encrypted columns. Hence care should be taken and only the columns containing most sensitive data should be subjected to TDE.

### Conclusion:

TDE is to protect data at rest, so that even if someone steals the backup they cannot read the sensitive information as this

will require the original master key to decrypt the data. When wallet is open, the encrypted data is available to all, but when wallet is closed, it is available to none. Wallet cannot be opened for a specific user. If the requirement is to give access to confidential data to a select group of users, then the fine-grained access control of VPD can be combined with TDE to provide security holistically.

**For reference:**

- [Top Features for DBAs by Arup Nanda](#)
- [Oracle10g Rel2 Advanced Security Administrator's Guide](#)

**About the author:**

Amarjyoti Dewri has 5+ years of IT experience out of which he has been working as Oracle DBA for last 4 years. He is currently working for TCS, India. He has worked for the Oracle Support Group (TCS) as a technical consultant providing third-party technical support to various Oracle clients in India. He has worked on various Oracle projects within TCS providing consultancy on various database setup, backup strategies, migration strategies and performance tuning issues. He can be contacted at [amarjyoti.dewri@gmail.com](mailto:amarjyoti.dewri@gmail.com) or at <http://adewri.modblog.com>