Transparent Data Encryption(TDE)

OS	Linux
DB	11g R2
Author	윤 현
Email	sensyh@dbworks.co.kr
HomePage	http://www.dbworks.co.kr http://support.dbworks.co.kr

DBWorks

1. TDE 란?

TDE는 테이블스페이스 및 테이블에 저장된 신용카드 번호와 같은 민감한 데이터를 암호화하는 기능입니다.
암호화된 데이터는 데이터베이스 데이터에 접근하는 사용자 또는 응용프로그램을 위해 투명하게 해독됩니다
TDE는 저장장치 또는 데이터파일의 도난으로 부터 데이터를 보호합니다.

2. TDE 개요

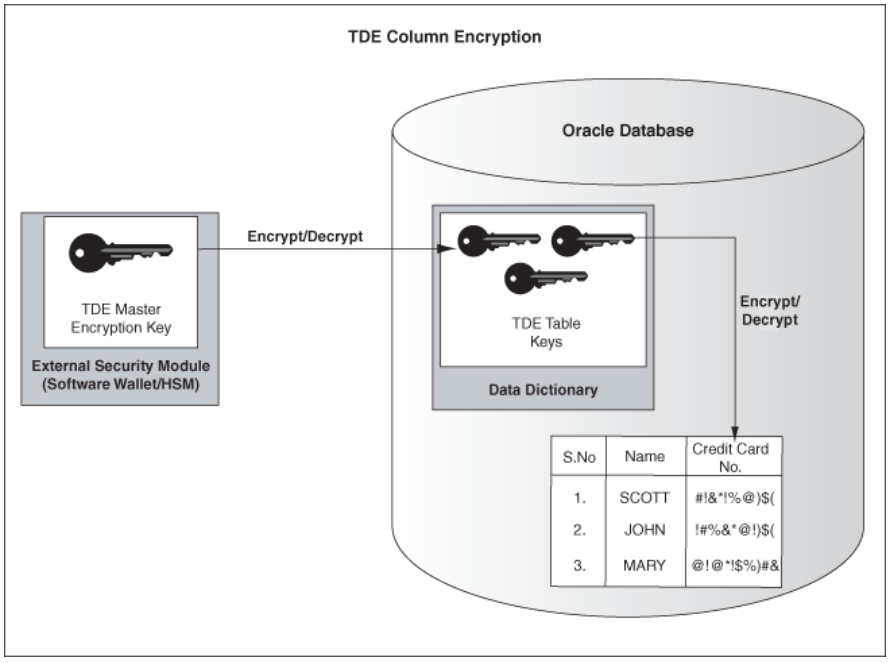
Oracle 데이터베이스는 데이터 보호를 위해 인증, 권한부여, 감사 메커니즘을 사용하지만
데이터가 저장된 OS data file에 대해서는 그러한 메커니즘이 적용되지 않습니다..
이러한 데이터 파일을 보호하기 위해 오라클 데이터베이스는 TDE를 제공합니다
TDE는 데이터파일에 저장된 민감한 데이터를 암호화합니다
TDE는 무단 해독을 방지하기 위하여 암호키를 데이터베이스 외부의 보안 모듈(HMS, Oracle Wallet) 안에 저장합니다.
데이터베이스 사용자와 어플리케이션은 key 저장소를 관리하거나 보조 테이블, 뷰, 트리거를 생성 할 필요가 없습니다.
또한 응용프로그램의 변경없이 강력한 데이터 암호화를 제공하는 TDE를 사용할 수 있습니다.
TDE의 사용은 신용카드 번호 나 주민번호 같은 기밀 데이터를 보호합니다.
전체 테이블 스페이스 암호화를 위해 TDE를 사용 할 수 있습니다.

2. TDE 이점

보안 관리자로서 저장 매체 또는 데이터파일을 도단 당한 경우에 민감한 데이터를 안전하게 보호할 수 있습니다.
TDE는 보완 관련 규제 준수 문제를 해결 할 수 있습니다.
승인된 유저 또는 어플리케이션은 데이터 해독을 위해 트리거 또는 뷰를 생성할 필요가 없습니다.
테이블에서 데이터는 유저 및 어플리케이션에게 투명하게 해독됩니다.
데이터베이스 사용자와 어플리케이션은 엑세스 하는 데이터를 암호화된 형태로 저장되어 있다는 사실을 알고 있을 필요가 없습니다 데이터는 투명하게 해독됩니다.
암호화된 데이터를 처리하기 위해 어플리케이션을 수정할 필요가 없습니다.
데이터 암화화 및 해독은 데이터베이스에 의해 관리됩니다.
key 관리 작업이 자동화 되어 있습니다. 사용자 또는 어플리케이션은 암호화 키를 관리할 필요가 없습니다.

1. TDE Column Encryption

TDE column 암호화 기법은 선택한 테이블의 컬럼의 데이터들을 암호화 합니다.
TDE column 암호화는 테이블 컬럼에 저장된 신용카드 및 주민번호 같은 기밀 데이터를 보호하는데 사용됩니다.
TDE column 암호화는 table column을 암호화 및 해독 하기 위해 두 계층, 키 기반 아키텍처를 사용합니다.
TDE master 암호화 키는 Oracle wallet 또는 Hardware Sercurity Module(HMS)같은 외부 보안 뮤듈에 저장됩니다.
master 암호화 키는 테이블컬럼을 암호화하고 해독하는데 사용되는 table key를 암호화하는데 사용됩니다.
master 암호화 키는 오직 보안관리자만이 접속할수 있는 데이터베이스 밖의 외부 보안 모듈에 저장됩니다.
이 외부 보안 모듈의 경우 oracle wallet 또는 HSM을 사용합니다.
master 암호화 키 저장방식은 무단 사용을 방지합니다.
외부 보안 모듈의 사용은 암호화 작업에서 데이터베이스 관리자와 보안관리자 사이의 의무를 나누는것을 가능하게 합니다.
테이블에 암호화된 컬럼을 포함하면 단일 테이블 키를 암호화된 열 개수에 관계 없이 사용합니다.
모든 테이블에 대한 테이블 키는 데이터베이스의 딕셔너리 테이블에 저장됩니다.



DBWorks

TDE Column Encryption에서 사용할 수 없는 기능들

- Index types other than B-tree
- Range scan search through an index
- External large objects (BFILE)
- Synchronous Change Data Capture
- Transportable Tablespaces
- Original import/export utilities
- 컬럼이 foreign key constraints를 사용하고 있다면 TDE column을 사용할 수 없다.
- 만약에 암호화 테이블의 컴럼에 인덱스를 사용하고자 한다면 no salt를 사용해야 한다.

TDE Colum Encryption 예제

Wallet 생성

```
<SQLNET.ORA>
ENCRYPTION_WALLET_LOCATION=
(SOURCE=(METHOD=FILE)(METHOD_DATA=
(DIRECTORY=/home/oracle/wallet)))
```

Master Encryption Key 생성

```
SQL>alter system set encryption key identified by "hyun";
```

Encryption tablespace 및 user생성

```
SQL> create tablespace encryption datafile  '
        '/home/oracle/oradata/TDE/encryp01.dbf' size 100M
         autoextend on
SQL> create user encryption identified by encryption default
        tablespace encryption account unlock;
SQL> grant resource, connect to encryption;
```

Encryption Column 생성

```
SQL> create table dbworks
        (name varchar2(10),
        passwd varchar2(10) encrypt no salt);
SQL> insert into dbworks values('younhyun','dbworks');
SQL> commit;;
```

TDE Colum Encryption 예제

Tablespace Offline (Disk에 Write 하기 위한 과정)

```
SQL>alter tablespace encryption offline;
SQL>alter tablespace encryption online;
```

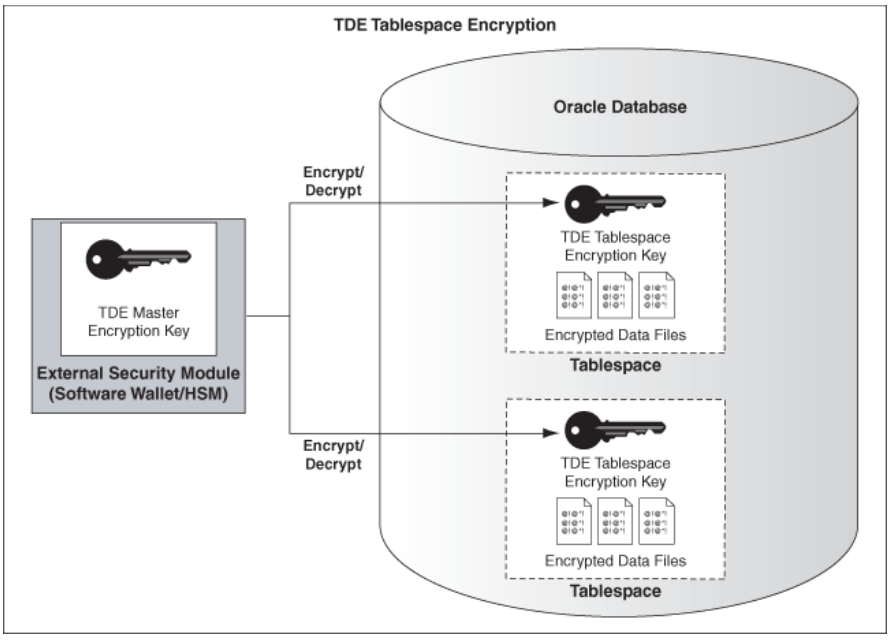
검증

```
$>strings encryp01.dbf |grep younhyun
younhyun$
$>strings encryp01.dbf |grep dbworks
```

* TDE Column 에서는 strings로 데이터 파일이 조회가 되지 않는걸 확인 할 수 있습니다.

1. TDE Tablespace Encryption

TDE tablespace 암호화기법은 테이블스페이스 전체를 암호화 합니다.
암화화 테이블스페이스 안에 생성된 모든 objects는 자동으로 암호화 됩니다.
TDE 테이블스페이스 암호화는 테이블안의 데이터를 보호하려 할때 유용합니다
또한 TDE 테이블스페이스 암호화는 향상된 성능을 제공하기 위한 대량 암호화와 caching에서 장점이 있습니다응용프로그램의 실제 성능에 미치는 영향이 다를수 있지만 성능 overhead는 5% ~8%로 추정됩니다.
BLOB 및 CLOBs 같은 LOB또한 포함됩니다.
Bfile column은 암호화 되지 않습니다.
암호화된 테이블스페이스의 모든 데이터들은 디스크에 암호화된 형식으로 저장됩니다.
데이터는 인증된 유저에게 투명하게 해독됩니다.
데이터베이스 유저 또는 어플리케이션은 특정 테이블의 데이터가 디스크게 암호화 되어 있는지 알 필요가 없습니다. 디스크 또는 백업 장치가 도난당하더라도 데이터는 보호됩니다.
TDE tablespace 암호화는 테이블 스페이스의 투명한 암호화와 해독을 위해 두계층 , 키기반 아키텍처를 사용합니다.
TDE master Key는 외부 보안 모듈에 저장됩니다
TDE master key는 TDE tablespace encyption key를 암호화 하는데 사용됩니다.
talbespace encyption key는 tablespace안의 데이터를 암호화하고 해독하는데 사용됩니다.
TDE 테이블스페이스 암호화는 또한 암호화된 테이블스페이스에서 index range 스캔을 사용할 수 있습니다.
이것은 TDE column encryption 에선 가능하지 않습니다.
oracle 11g release 2에서 tablespace space 암호화에 대한 다음 향상된 기능을 구현합니다.
통일 master 암호화 키를 TDE column 및 TDE tablespace 암호화 모두에서 사용합니다.
당신은 통일 master key 재설정을 할 수 있습니다.
이것은 향상된 보안을 제공하고 보안 및 규정 준수 요구 사항을 충족 도움이 됩니다.



TDE Tablespace의 새로운 기능

- 데이터를 보호하기 위해 영구적으로 모든 테이블 스페이스를 암호화 할 수 있습니다.
  모든 세그먼트 유형을 지원합니다. (table, clusters, indexes, LOBs, table and index partitions 기타 등등)
- 테이블 스페이스 암호화는 완전히 응용 프로그램에 투명하기 때문에 어플리 케이션을 수정하지 않아도 됩니다.
- exp의 사용은 지원되지 않으며 IMP는 지원됩니다.
- 비트맵 인덱스 사용 가능합니다.
- range scan 이 가능해졌습니다.
- LOB 데이터 타입을 지원한다
- TTS를 지원하지만 endian 변경은 안된다.
- TTS시 암화화 테이블스페이스를 옮기기전에 wallet을 타겟 디비로 복사 해야합니다.
- 또한 타겟 디비가 이미 wallet을 사용하고 있다면 data pump를 이용해야 합니다.

TDE Tablespace PUMP 사용 옵션

1. encryption - 덤프하기전에 데이터의 암호화 여부를 표시합니다.
[all | data_only |encryped_colums_only | metadata_only |none
    all - 모두 암호화
    data_only - data만 암호화
    encryped_colums_only - encryped_colums만 암호화
    metadata_only - metadata만 암호화
    none - 암호화 하지 않음

2. ENCRYPTION_ALGORITHM - 암호화를 사용하는데 사용하는 알고리즘입니다.
    오라클 10g의 RMAN과 동일하게 작동합니다.
[AES128 | AES192 | AES256]

3.ENCRYPTION_MODE - 암호화및 해독을 수행할 때 모드입니다.
[DUAL | PASSWORD | TRANSPARENT]
    password - 덤프 파일 셋트를 만드는 암호를 지정합니다. expdp와 impdp 모두 encryption_password절을 지정해줘야 합니다. passwd가 틀리면 impdp가 되지 않습니다.
    TRANSPARENT - expdp 와 impdp 모두 encryption_password 파라메터를 쓰지 않습니다.
    DUAL - expdp 때 encryption_password를 지정해주나 impdp때 encryption_password 절을 쓰지 않아도 됩니다.

TDE Tablespace 생성과 TDE Tablespace로의 table move(예제)

Wallet 생성

```
<SQLNET.ORA>
ENCRYPTION_WALLET_LOCATION=
(SOURCE=(METHOD=FILE)(METHOD_DATA=
(DIRECTORY=/home/oracle/wallet)))
```

Master Encryption Key 생성

```
SQL>alter system set encryption key identified by "hyun";
```

Encryption Tablespace 생성

```
SQL>CREATE TABLESPACE encryptedtbs
DATAFILE '/home/oracle/oradata/encryptedtbs01.dbf' SIZE 100M
ENCRYPTION USING 'AES256'
DEFAULT STORAGE(ENCRYPT);
```

ENCRYPTION는 USING절을 사용하여 알고리즘 속성을 지정합니다.
스토리지 절의 ENCRYPT가 테이블 스페이스를 암호화 합니다.
USING 절을 쓰지않으면 AES128가 DEFAULT로 적용됩니다.
AES128, AES192, AES256 3DES168 4개의 알고리즘을 사용할수 있습니다.

Encryption Tablespace 생성 확인 및 알고리즘 확인

```
SQL> select * from V$ENCRYPTED_TABLESPACES;

TS#  ENCRYPT    ENC
---- ------  -------
7    AES128    YES

SQL> select tablespace_name, encrypted from dba_tablespaces
where tablespace_name like 'ENCRY%';

TABLESPACE_NAME                ENC
------------------------------  ---
ENCRYPTEDBS                    YES
```

유저 생성 및 권한부여

```
SQL> create user hyun identified by hyun account unlock
     default tablespace users;
SQL> grant resource , connect to hyun;
```

Table 생성 (일반 Tablespace)

```
SQL> create table hyun(abc varchar(20));
SQL> insert into hyun values('DBWORKS');
```

Tablespace Offline (Disk에 Write 하기 위한 과정) (일반 Tablespace)

```
SQL> alter tablespace users offline;
SQL> alter tablespace users online
```

Test 검증 (일반 Tablespace)

```
$ strings users01.dbf |grep D
DBWORKS,
```

Table Move (암호화 Tablespace)

```
SQL>alter table hyun.hyun move tablespace encryptedtbs
```

Tablespace Offline (Disk에 Write 하기 위한 과정) (암호화 Tablespace)

```
SQL>alter tablespace users offline;
SQL>alter tablespace users online;
```

Test 검증 (암호화 Tablespace)

```
$ strings encryptedtbs01.dbf |grep D
```

* TDE tablespace 에서는 strings로 데이터 파일이 조회가 되지 않는걸 확인 할 수 있습니다.