

Oracle TDE with PDBs and try not go mad again

Marcin Przepiorowski



TDE can help me to:

- A) protect my database from SQL ingestion in application server
- B) protect my database from access with stolen credentials
- C) protect my database from unauthorized admin access
- D) Protect my database from update outside of SQL engine
- E) protect my database from stolen hard drive or tape with backup
(only if you don't backup keystore with your datafiles)

How TDE works

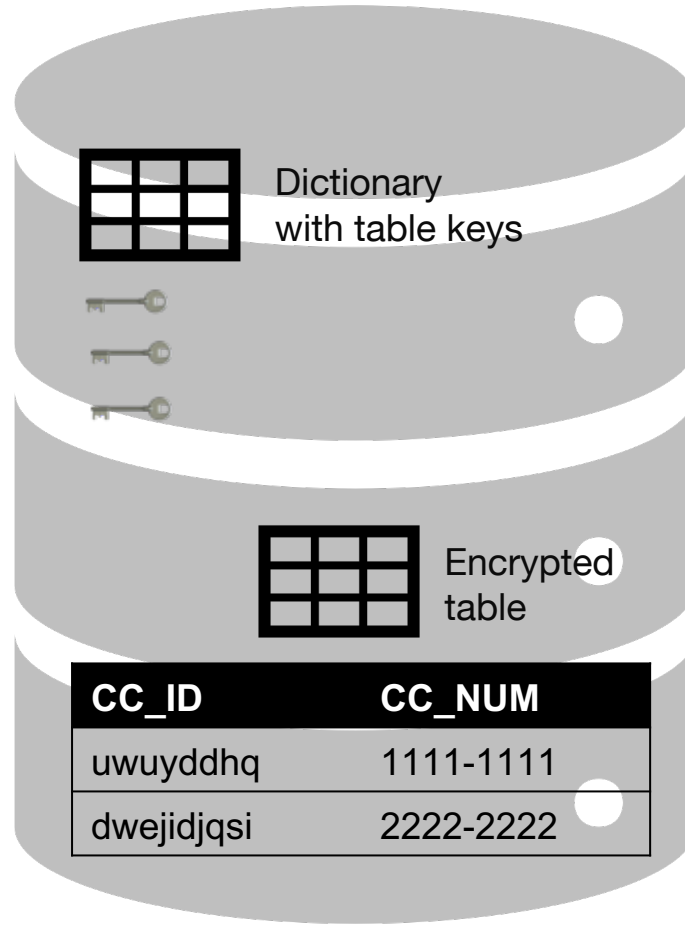
Master Key

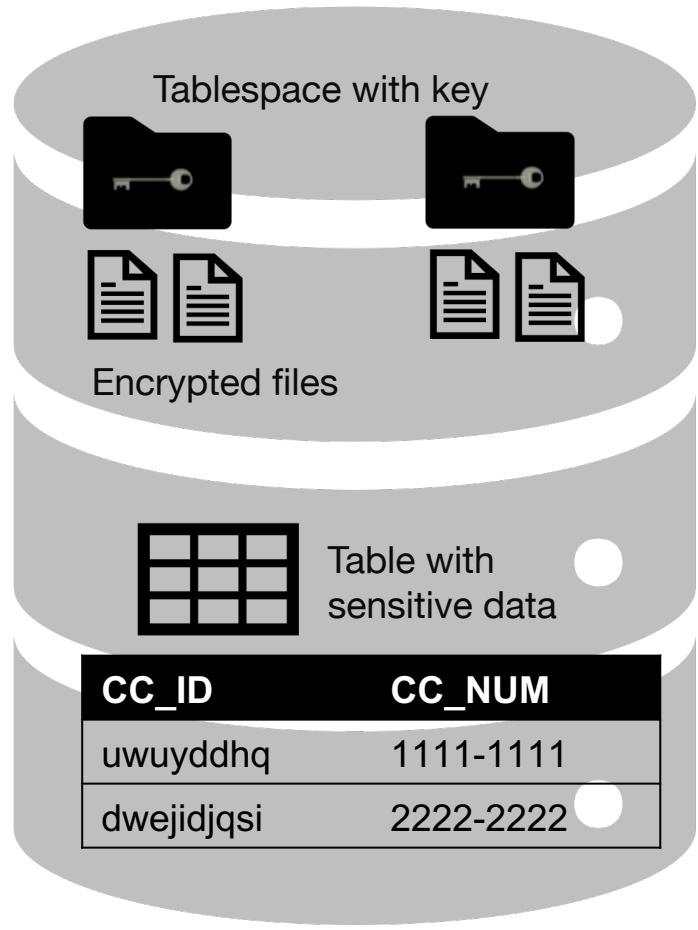
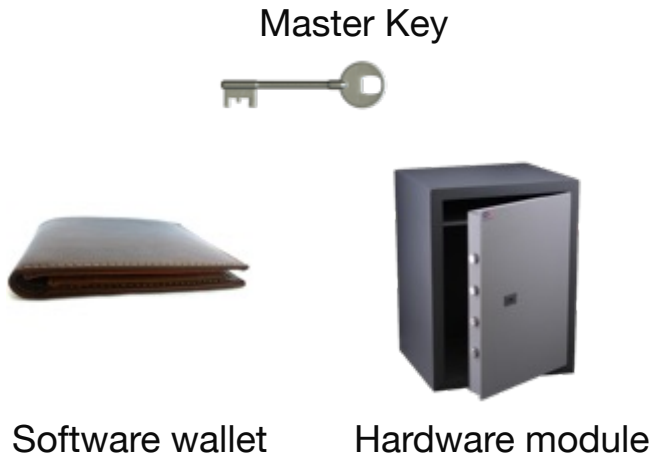


Software wallet

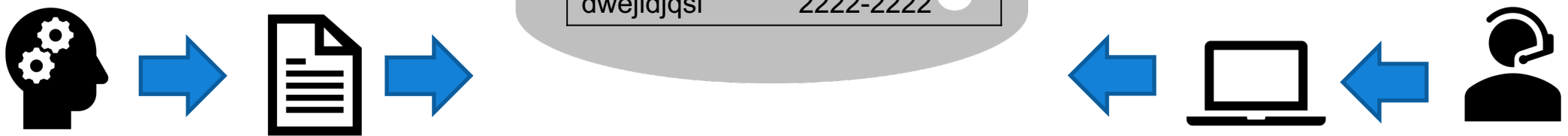


Hardware module





CC_ID	CC_NUM
uwuyddhq	1111-1111
dwejidqsi	2222-2222



Checking encryption

```
create table pioro.test_enc_column (id number, cc varchar2(50) encrypt) tablespace users;
Table created.
insert into pioro.test_enc_column values (1, 'marcin');
1 row created.
commit;
Commit complete.
```

```
create table pioro.test_noenc (id number, cc varchar2(50) ) tablespace users;
Table created.
insert into pioro.test_noenc values (1, 'przepiorowski');
1 row created.
commit;
Commit complete.
shutdown immediate
```

```
[oracle@hobbiton ~]$ grep -i przepiorowski /u01/app/oracle/oradata/PIORO19/devdb/users01.dbf
Binary file /u01/app/oracle/oradata/PIORO19/devdb/users01.dbf matches
```

```
[oracle@hobbiton ~]$ grep -i marcin /u01/app/oracle/oradata/PIORO19/devdb/users01.dbf
[oracle@hobbiton ~]$
```

Table vs Tablespace encryption – review

Table	Tablespace
No range scan on indexes	Unique and range scans on indexes
Space overhead (1 to 52 bytes per row)	No space overhead
Potential impact on execution plan 728292.1.	No impact on execution plan
Always encrypted	Decrypted in SGA
Key per table	Key per tablespace

TDE Master Note – MOS 1228046.1

TDE 12c FAQ - MOS 2253348.1

Quick TDE Setup and FAQ - MOS 1251597.1



Wallets

Multitenant keystores (wallet) types:

- United – TDE master encryption key for CDB and all PDBs are in same keystore
- Isolated – TDE master encryption key for CDB and PDB's are in individual PDB's keystores

```
SQL> select con_id, wrl_parameter, status, wallet_type, keystore_mode from v$encryption_wallet;
```

CON_ID	WRL_PARAMETER	STATUS	WALLET_TYPE	KEYSTORE
1	/u01/app/oracle/wallets/tde/pioro19/	OPEN	PASSWORD	NONE
2		OPEN	PASSWORD	UNITED
3		OPEN	PASSWORD	UNITED

Wallets

Locations:

- WALLET_ROOT (init.ora but also TDE_CONFIGURATION value)
- WALLET_LOCATION (sqlnet.ora)
- ENCRYPTION_WALLET_LOCATION (sqlnet.ora)
- \$ORACLE_BASE/admin/db_unique_name/wallet
- \$ORACLE_HOME/admin/db_unique_name/wallet

TNS_ADMIN can do a trick as well 😊

```
ls -l /u01/app/oracle/wallets/tde/pioro19/
```

```
-rw-----. 1 oracle oinstall 5835 Sep 12 16:23 ewallet_2022091220513916.p12  
-rw-----. 1 oracle oinstall 8507 Sep 12 16:53 ewallet_2022091220533242.p12  
-rw-----. 1 oracle oinstall 4171 Sep 13 11:23 ewallet_2022091315231259.p12  
-rw-----. 1 oracle oinstall 5835 Sep 13 11:23 ewallet.p12
```

Wallets

The use of PKI (orapki) encryption with Transparent Data Encryption is deprecated.
Use the ADMINISTER KEY MANAGEMENT SQL statement.

```
[oracle@hobbiton ~]$ $ORACLE_HOME/bin/orapki wallet display -wallet  
/u01/app/oracle/wallets/tde/pioro19/  
Oracle PKI Tool Release 19.0.0.0.0 - Production
```

```
..  
Enter wallet password:
```

```
..  
Oracle Secret Store entries:  
ORACLE.SECURITY.DB.ENCRYPTION.AVpZ8Ohu50/jv2uvQxp59rQAAAAAAAAAAAAAAAAAAAAAAAAAA  
ORACLE.SECURITY.DB.ENCRYPTION.AYtos6a6Ck+kv45nlnFP4EsAAAAAAAAAAAAAAAAAAAAAAAAAA  
ORACLE.SECURITY.DB.ENCRYPTION.MASTERKEY  
ORACLE.SECURITY.DB.ENCRYPTION.MASTERKEY.E7B331B4F01249C8E055020C29B28AD3  
ORACLE.SECURITY.ID.ENCRYPTION.
```

```
SQL> @keys
```

KEY_ID	TAG	CON_ID
AVpZ8Ohu50/jv2uvQxp59rQAAAAAAAAAAAAAAAAAAAAAAAAAA	new master 2	1
AYtos6a6Ck+kv45nlnFP4EsAAAAAAAAAAAAAAAAAAAAAAAAAA	new master devdb	3

Check if TDE was ever enabled

You can't recreate a wallet if database was encrypted before

Check table `x$kcbdbk`

Clean database (no tablespace encryption enabled)

```
11g  
BITAND (FLAGS,8) == 0
```

```
12c+  
mkloc == 0
```

Database setup

Multitenant setup

```
mkdir -p /u01/app/oracle/wallets/tde/pioro19
```

```
sqlnet.ora
```

```
WALLET_LOCATION=
```

```
(SOURCE=
```

```
  (METHOD=FILE)
```

```
  (METHOD_DATA=
```

```
    (DIRECTORY=/u01/app/oracle/wallets/tde/${ORACLE_SID}/)))
```

Multitenant setup

```
SQL> select con_id, wrl_parameter, status, wallet_type from v$encryption_wallet
```

CON_ID	WRL_PARAMETER	STATUS	WALLET_TYPE
1	/u01/app/oracle/wallets/tde/pioro19	NOT_AVAILABLE	UNKNOWN
2		NOT_AVAILABLE	UNKNOWN
3		NOT_AVAILABLE	UNKNOWN

```
SQL> select con_id, mkloc from x$kcdbk
```

CON_ID	MKLOC
1	0
2	0
3	0

Multitenant setup

```
administer key management create keystore /u01/app/oracle/wallets/tde/pioro19/' identified by xxx;
```

```
select con_id, wrl_parameter, status, wallet_type from v$encryption_wallet;
```

CON_ID	WRL_PARAMETER	STATUS	WALLET_TYPE
1	/u01/app/oracle/wallets/tde/pioro19/	CLOSED	UNKNOWN
2		CLOSED	UNKNOWN
3		CLOSED	UNKNOWN

```
administer key management set keystore open identified by xxx;
```

```
select con_id, wrl_parameter, status, wallet_type from v$encryption_wallet;
```

CON_ID	WRL_PARAMETER	STATUS	WALLET_TYPE
1	/u01/app/oracle/wallets/tde/pioro19/	OPEN_NO_MASTER_KEY	PASSWORD
2		CLOSED	UNKNOWN
3		CLOSED	UNKNOWN

Multitenant setup

```
administer key management set key using tag "first master key cdb" identified by delphix with backup
using 'backup before master cdb';
```

```
select con_id, wrl_parameter, status, wallet_type from v$encryption_wallet;
```

CON_ID	WRL_PARAMETER	STATUS	WALLET_TYPE
1	/u01/app/oracle/wallets/tde/pioro19/	OPEN	PASSWORD
2		CLOSED	UNKNOWN
3		CLOSED	UNKNOWN

```
select con_id, mkloc from x$kcdbk;
```

CON_ID	MKLOC
1	1
2	0
3	0

Multitenant setup

```
administer key management set keystore open identified by delphix container = all;
```

```
select con_id, wrl_parameter, status, wallet_type from v$encryption_wallet;
```

CON_ID	WRL_PARAMETER	STATUS	WALLET_TYPE
1	/u01/app/oracle/wallets/tde/pioro19/	OPEN	PASSWORD
2		CLOSED	UNKNOWN
3		OPEN_NO_MASTER_KEY	PASSWORD

```
administer key management set key using tag "first master key all" identified by delphix with backup using 'backup before master all' container=all;
```

```
select con_id, wrl_parameter, status, wallet_type from v$encryption_wallet;
```

CON_ID	WRL_PARAMETER	STATUS	WALLET_TYPE
1	/u01/app/oracle/wallets/tde/pioro19/	OPEN	PASSWORD
2		CLOSED	UNKNOWN
3		OPEN	PASSWORD

Multitenant setup

```
alter session set container = devdb;
```

```
select con_id, wrl_parameter, status, wallet_type from v$encryption_wallet;
```

CON_ID	WRL_PARAMETER	STATUS	WALLET_TYPE
3		OPEN	PASSWORD

```
create tablespace ts_pii datafile '/u01/app/oracle/oradata/PIORO19/devdb/ts_pii01.dbf' size 10M  
encryption using 'AES256' default storage (encrypt);
```

RAC

12.1

- non-shared TDE wallets are supported but not recommended

19c

- Oracle does not support the use of individual TDE wallets for each Oracle RAC node. Instead, use shared wallets for TDE in the Oracle RAC environment.
 - Oracle ACFS
 - ASM , +DATA/\$ORACLE_UNQNAME/WALLETS.
- Keystore operations (such as opening or closing the keystore, or rekeying the TDE master encryption key) can be issued on any one Oracle RAC instance. Internally, the Oracle database takes care of synchronizing the keystore context on each Oracle RAC node ?????

RAC

```
[oracle@hobbiton ~]$ sudo sysdig fd.name=/u01/app/oracle/wallets/tde/pioro19/ewallet.p12  
987253 07:55:47.619539508 1 oracle (33689.33689) < openat  
fd=16(<f>/u01/app/oracle/wallets/tde/pioro19/ewallet.p12) dirfd=-100(AT_FDCWD)  
name=/u01/app/oracle/wallets/tde/pioro19/ewallet.p12 flags=1(O_RDONLY) mode=0 dev=FD00
```

Daily actions

Encrypting system objects

```
SQL> create table test_enc_column (id number, cc varchar2(50) encrypt) tablespace users;  
create table test_enc_column (id number, cc varchar2(50) encrypt) tablespace users  
*
```

ERROR at line 1:

ORA-28336: cannot encrypt SYS owned objects

You can encrypt SYSTEM, SYSAUX, TEMP and UNDO tablespace but you can't close wallet manually anymore.
MOS 2393734.1

Tablespace level encryption

```
create tablespace ts_pii_pdb datafile '/u01/app/oracle/oradata/PIOR019/devdb/ts_pii.dbf' size 100M
encryption using 'AES256' encrypt;
```

ALTER SYSTEM SET ENCRYPT_NEW_TABLESPACES = value;

- CLOUD_ONLY transparently encrypts the tablespace in the Cloud using the AES128 algorithm CLOUD_ONLY is the default.
- ALWAYS automatically encrypts the tablespace using the AES128 algorithm if you omit the ENCRYPTION clause of CREATE TABLESPACE
- DDL encrypts the tablespace using the specified setting of the ENCRYPTION

Tablespace level encryption

```
alter system set encrypt_new_tablespaces = always;  
System altered.
```

```
create tablespace ts_enc2 datafile '/u01/app/oracle/oradata/PIOR019/devdb/ts_enc2_01.dbf' size 10M;  
Tablespace created.
```

```
SQL> select TABLESPACE_NAME, ENCRYPTED from dba_tablespaces;
```

TABLESPACE_NAME	ENC
-----	---
SYSTEM	NO
SYSAUX	NO
UNDOTBS1	NO
TEMP	NO
USERS	NO
TS_ENC	YES
TS_ENC2	YES

System tablespace encryption

```
SQL> shutdown
```

```
Pluggable Database closed.
```

```
SQL> show pdbs
```

CON_ID	CON_NAME	OPEN MODE	RESTRICTED
4	DEVDB	MOUNTED	

```
SQL> ALTER TABLESPACE SYSTEM ENCRYPTION OFFLINE ENCRYPT;
```

```
Tablespace altered.
```

```
Elapsed: 00:00:00.01
```

```
SQL> ALTER TABLESPACE SYSAUX ENCRYPTION OFFLINE ENCRYPT;
```

```
Tablespace altered.
```

```
Elapsed: 00:00:03.34
```

```
SQL> ALTER TABLESPACE UNDOTBS1 ENCRYPTION OFFLINE ENCRYPT;
```

```
Tablespace altered.
```

System tablespace encryption

```
SQL> select tablespace_name, encrypted from dba_tablespaces;
```

TABLESPACE_NAME	ENC
-----	---
SYSTEM	YES
SYSAUX	YES
UNDOTBS1	YES
TEMP	NO
USERS	NO
TS_PII_PDB	YES

```
[oracle@hobbiton trace]$ grep -i SLON /u01/app/oracle/oradata/PIOR019/devdb/system01.dbf  
Binary file /u01/app/oracle/oradata/PIOR019/devdb/system01.dbf matches
```

Restart

```
SQL> startup mount
Database mounted.
```

```
SQL> administer key management set keystore open force keystore identified by delphix container=all;
keystore altered.
```

```
SQL> show pdbs
```

CON_ID	CON_NAME	OPEN MODE	RESTRICTED
2	PDB\$SEED	MOUNTED	
3	DEVDB	MOUNTED	
4	SYSENC2	MOUNTED	

```
SQL> alter session set container = sysenc2;
Session altered.
```

```
SQL> @wallet
```

WRL_PARAMETER	STATUS	WALLET_TYPE
	CLOSED	UNKNOWN

Restart

```
SQL> alter database open;  
Database altered.
```

```
SQL> @wallet
```

WRL_PARAMETER	STATUS	WALLET_TYPE
/u01/app/oracle/wallets/tde/pioro19/	OPEN	PASSWORD
	CLOSED	UNKNOWN
	CLOSED	UNKNOWN
	CLOSED	UNKNOWN

```
2022-09-16T13:06:34.156315-04:00
```

```
DEVDB(3):Error 28365 during pluggable database DEVDB opening in read write
```

```
2022-09-16T13:06:34.156457-04:00
```

```
DEVDB(3):Errors in file /u01/app/oracle/diag/rdbms/pioro19/pioro19/trace/pioro19_p000_30796.trc:
```

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

```
DEVDB(3):Pluggable database DEVDB pseudo close cleaning up
```

Restart

```
SQL> administer key management set keystore open force keystore identified by delphix container=all;  
keystore altered.
```

```
SQL> @wallet
```

WRL_PARAMETER	STATUS	WALLET_TYPE
-----	-----	-----
/u01/app/oracle/wallets/tde/pioro19/	OPEN	PASSWORD
	CLOSED	UNKNOWN
	CLOSED	UNKNOWN
	CLOSED	UNKNOWN

Restart

```
SQL> administer key management set keystore close identified by delphix container=all;  
keystore altered.
```

```
SQL> @wallet
```

WRL_PARAMETER	STATUS	WALLET_TYPE
-----	-----	-----
/u01/app/oracle/wallets/tde/pioro19/	CLOSED	UNKNOWN
	CLOSED	UNKNOWN
	CLOSED	UNKNOWN
	CLOSED	UNKNOWN

Restart

```
SQL> administer key management set keystore open force keystore identified by delphix container=all;  
keystore altered.
```

```
SQL> @wallet
```

WRL_PARAMETER	STATUS	WALLET_TYPE
-----	-----	-----
/u01/app/oracle/wallets/tde/pioro19/	OPEN	PASSWORD
	OPEN	PASSWORD
	CLOSED	UNKNOWN
	CLOSED	UNKNOWN

Restart

```
SQL> alter session set container = sysenc2;  
Session altered.
```

```
SQL> @wallet
```

WRL_PARAMETER	STATUS	WALLET_TYPE
-----	-----	-----
	CLOSED	UNKNOWN

```
SQL> administer key management set keystore open force keystore identified by delphix;  
keystore altered.
```

```
SQL> @wallet
```

WRL_PARAMETER	STATUS	WALLET_TYPE
-----	-----	-----
	OPEN	PASSWORD

```
SQL> alter pluggable database sysenc2 open ;  
Pluggable database altered.
```

Autologin wallet

Creating a Auto login for wallet

```
administer key management create auto_login keystore from keystore '/path' identified by delphix;
```

```
administer key management create local auto_login keystore from keystore '/path' identified by delphix;
```

Wallet backup

- Backup your wallet
- Backup your wallet always and probably include with database backup
- Backing up using administer command is not enough – you need to backup a physical file to other location
- Did I say to backup your wallet ??????

REKEY master key

```
alter system flush buffer_cache; alter system checkpoint;
```

```
[oracle@oracle18 ~]$ dd if=/u01/app/oracle/oradata/PIOR019/devdb/ts_pii01.dbf of=block.dmp_enc bs=8k  
count=1 skip=131
```

```
administer key management set key using tag "rekey pdb" identified by delphix with backup using  
"backup";
```

```
keystore altered.
```

```
alter system flush buffer_cache; alter system checkpoint;
```

```
[oracle@oracle18 ~]$ dd if=/u01/app/oracle/oradata/PIOR019/devdb/ts_pii01.dbf of=block.dmp_enc_rekey  
bs=8k count=1 skip=131
```

```
[oracle@oracle18 ~]$ md5sum block.dmp_enc  
ab35da0dbe93a006424d94566c906db2 block.dmp_enc
```

```
[oracle@oracle18 ~]$ md5sum block.dmp_enc_rekey  
ab35da0dbe93a006424d94566c906db2 block.dmp_enc_rekey
```

Tablespace encryption key

```
SQL> select NAME, ENCRYPTEDTS, MASTERKEYID, STATUS, et.CON_ID from v$encrypted_tablespaces et,  
v$tablespace t where et.ts# = t.ts#;
```

NAME	ENC	MASTERKEYID	STATUS.	CON_ID
SYSTEM	YES	4AE87763092B4F45BFE2C5BB635733F4	NORMAL	4
SYSAUX	YES	4AE87763092B4F45BFE2C5BB635733F4	NORMAL	4
UNDOTBS1	YES	4AE87763092B4F45BFE2C5BB635733F4	NORMAL	4
TS_PII_PDB	YES	4AE87763092B4F45BFE2C5BB635733F4	NORMAL	4

```
SQL> select key_id, tag, con_id from v$encryption_keys;
```

KEY_ID	TAG	CON_ID
AUrod2MJK09Fv+LFu2NXM/QAAAAAAAAAAAAAAAAAAAAAAAAAAAA	new master devdb 2	4

Tablespace encryption key

MOS Doc ID 1228046.1

```
4AE87763092B4F45BFE2C5BB635733F4
```

```
01 4AE87763092B4F45BFE2C5BB635733F4
```

```
SQL> select utl_raw.cast_to_varchar2(utl_encode.base64_encode('014AE87763092B4F45BFE2C5BB635733F4'))  
from dual;
```

```
AUrod2MJK09Fv+LFu2NXM/Q=
```

```
AUrod2MJK09Fv+LFu2NXM/QAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Wallet operations during rekey

```
sudo sysdig fd.name=/u01/app/oracle/wallets/tde/pioro19/ewallet.p12
```

```
209682 07:59:04.674268440 0 oracle_33689_pi (33689.33689) < openat
fd=16(<f>/u01/app/oracle/wallets/tde/pioro19/ewallet.p12) dirfd=-100(AT_FDCWD)
name=/u01/app/oracle/wallets/tde/pioro19/ewallet.p12 flags=1(O_RDONLY) mode=0 dev=FD00
209687 07:59:04.674331466 0 oracle_33689_pi (33689.33689) > read
fd=16(<f>/u01/app/oracle/wallets/tde/pioro19/ewallet.p12) size=65536
209688 07:59:04.674337218 0 oracle_33689_pi (33689.33689) < read res=9371
data=0.$....0.$]..*.H.....$N..$J0.$F0.$B..*.H.....$30.$/...0.$(..*.H.....0W..*
```

```
218584 07:59:04.767724841 2 oracle_33689_pi (33689.33689) < openat
fd=16(<f>/u01/app/oracle/wallets/tde/pioro19/ewallet.p12) dirfd=-100(AT_FDCWD)
name=/u01/app/oracle/wallets/tde/pioro19/ewallet.p12 flags=263(O_TRUNC|O_CREAT|O_RDWR) mode=0666
dev=FD00
218589 07:59:04.767766495 2 oracle_33689_pi (33689.33689) > write
fd=16(<f>/u01/app/oracle/wallets/tde/pioro19/ewallet.p12) size=8192
218590 07:59:04.767825947 2 oracle_33689_pi (33689.33689) < write res=8192
data=0.$....0.$]..*.H.....$N..$J0.$F0.$B..*.H.....$30.$/...0.$(..*.H.....0W..*
```

REKEY tablespace online

```
alter session set container = pdb1;  
Session altered.
```

```
select key_version, status from V$ENCRYPTED_TABLESPACES;  
KEY_VERSION STATUS  
-----  
0 NORMAL
```

```
alter tablespace ts_pii encryption using 'AES192' rekey file_name_convert = ('ts_pii01.dbf',  
'ts_pii01_new.dbf');
```

Tablespace altered.

```
select key_version, status from V$ENCRYPTED_TABLESPACES;  
KEY_VERSION STATUS  
-----  
1 NORMAL
```


REKEY tablespace online

```
select name from v$datafile;
```

```
NAME
```

```
-----  
/u01/app/oracle/oradata/PIOR019/devdb/system01.dbf  
/u01/app/oracle/oradata/PIOR019/devdb/sysaux01.dbf  
/u01/app/oracle/oradata/PIOR019/devdb/undotbs01.dbf  
/u01/app/oracle/oradata/PIOR019/devdb/users01.dbf  
/u01/app/oracle/oradata/PIOR019/devdb/ts_pii01_new.dbf
```

```
[oracle@oracle18 ~]$ dd if=/u01/app/oracle/oradata/PIOR019/devdb/ts_pii01_new.dbf  
of=block.dmp_enc_rekey_new bs=8k count=1 skip=131
```

```
[oracle@oracle18 ~]$ md5sum block.dmp_enc  
ab35da0dbe93a006424d94566c906db2  block.dmp_enc
```

```
[oracle@oracle18 ~]$ md5sum block.dmp_enc_rekey_new  
c4f4537701ea28a7f48fc3fab8950c86  block.dmp_enc_rekey_new
```

Changing the Wallet password

Changing the Password-Protected wallet Password

```
administer key management alter keystore password [force keystore]  
identified by old_password set new_password  
with backup;
```

Plug / Unplug /
Clone

Unplug/plug PDB

```
alter pluggable database pdb1 close;  
Pluggable database altered.
```

```
alter pluggable database  pdb1 UNPLUG INTO '/tmp/pdb1.pdb';  
alter pluggable database  pdb1 UNPLUG INTO '/tmp/pdb1.pdb'  
*
```

ERROR at line 1:

ORA-46680: master keys of the container database must be exported

Unplug/plug PDB

```
administer key management export encryption keys with secret "delphix" to '/tmp/pdb1.p12' identified by  
xxx;
```

```
administer key management export encryption keys with secret "delphix" to '/tmp/pdb1.p12' identified by  
xxx
```

```
*
```

```
ERROR at line 1:
```

```
ORA-46658: keystore not open in the container
```

```
alter pluggable database pdb1 open read write;  
Pluggable database altered.
```

```
alter session set container = pdb1;  
Session altered.
```

```
administer key management export encryption keys with secret "password" to '/tmp/pdb1.key' force  
keystore identified by xxx;  
keystore altered.
```

Unplug/plug PDB

```
alter session set container = cdb$root;  
Session altered.
```

```
alter pluggable database pdb1 UNPLUG INTO '/tmp/pdb1.pdb';  
Pluggable database altered.
```

Unplug/plug PDB

```
create pluggable database pdb1 using '/tmp/pdb1_new.pdb'  
FILE_NAME_CONVERT=('/tmp/', '/u01/app/oracle/oradata/CLT18MT/PDB1/');  
Pluggable database created.
```

```
alter pluggable database pdb1 open read write;  
Warning: PDB altered with errors.
```

```
select message, status from PDB_PLUG_IN_VIOLATIONS where name = 'PDB1';
```

MESSAGE	STATUS
-----	-----
PDB needs to import keys from source.	PENDING

```
alter session set container=pdb1;  
Session altered.
```

```
administer key management import encryption keys with secret "password" from '/tmp/pdb1.key' force  
keystore identified by delphixclone with backup;  
keystore altered.
```

Unplug/plug PDB

```
alter session set container=cdb$root;  
Session altered.
```

```
alter pluggable database pdb1 close;  
Pluggable database altered.
```

```
alter pluggable database pdb1 open read write;  
Pluggable database altered.
```


Unplug/plug PDB

```
alter session set container=pdb1;  
Session altered.
```

```
select count(*) from pioro.test_ts_enc;  
select count(*) from pioro.test_ts_enc  
*
```

```
ERROR at line 1:  
ORA-28365: wallet is not open
```

```
administer key management set keystore open identified by xxxclone;  
keystore altered.
```

```
select count(*) from pioro.test_ts_enc;
```

```
COUNT(*)  
-----  
9999
```

Unplug/plug PDB

```
alter pluggable database pdb1 unplug into '/tmp/pdb1.xml' encrypt using myhiddenpassword;  
Pluggable database altered.
```

```
create pluggable database pdb1 using '/tmp/pdb1.xml' nocopy tempfile reuse decrypt using  
myhiddenpassword keystore identified by xxxclone;  
Pluggable database created.
```

```
alter pluggable database pdb1 unplug into '/tmp/pdb1.pdb' encrypt using myhiddenpassword;  
Pluggable database altered.
```

```
create pluggable database pdb1 using '/tmp/pdb1.pdb'  
file_name_convert=('/tmp/', '/u01/app/oracle/oradata/CLT18MT/PDB1/') decrypt using myhiddenpassword  
keystore identified by xxxclone;  
Pluggable database created.
```

No Keys ?!?

```
SQL> alter session set container = devdb;
```

```
SQL> @wallet
```

WRL_PARAMETER.	STATUS	WALLET_TYPE
-----	-----	-----
	OPEN	PASSWORD

```
SQL> create tablespace ts_pii_pdb datafile '/u01/app/oracle/oradata/PIORO19/devdb/ts_pii.dbf' size 10M
encryption using 'AES256' default storage(encrypt);
```

```
SQL> create table test (name varchar2(10)) tablespace ts_pii_pdb;
Table created.
```

```
SQL> insert into test values ('Marcin');
1 row created.
```

```
SQL> commit;
Commit complete.
```

No Keys ?!?

```
SQL> @keys
```

KEY_ID	TAG	CON_ID
Ac3KWSggv0/hv9ZDtD+qmpEAAAAAAAAAAAAAAAAAAAAAAAAAAAA	initial CDB key	1
AbEgNxflw09Sv4RWTMA2oM4AAAAAAAAAAAAAAAAAAAAAAAAAAAA	initial devdb key	3

```
SQL> show pdbs
```

CON_ID	CON_NAME	OPEN MODE	RESTRICTED
3	DEVDB	READ WRITE	NO

```
SQL> alter session set container = devdb;
```

```
Session altered.
```

```
SQL> ADMINISTER KEY MANAGEMENT EXPORT ENCRYPTION KEYS WITH SECRET "delphix" TO '/tmp/devdb.p12' FORCE  
KEYSTORE IDENTIFIED BY delphix;
```

```
keystore altered.
```

```
SQL> shutdown
```

```
Pluggable Database closed.
```

No Keys ?!?

```
SQL> alter pluggable database devdb unplug into '/tmp/devdb.xml';  
Pluggable database altered.
```

```
SQL> drop pluggable database devdb;  
Pluggable database dropped.
```

```
SQL> create pluggable database devdb using '/tmp/devdb.xml' nocopy;  
Pluggable database created.
```

```
SQL> show pdbs
```

CON_ID	CON_NAME	OPEN MODE	RESTRICTED
2	PDB\$SEED	READ ONLY	NO
4	DEVDB	MOUNTED	

```
SQL> alter session set container = devdb;  
Session altered.
```

No Keys ?!?

```
SQL> startup
```

```
Warning: PDB altered with errors.
```

```
Pluggable Database opened.
```

```
SQL> @keys
```

```
no rows selected
```

```
SQL> @wallet
```

```
WRL_PARAMETER
```

```
STATUS
```

```
WALLET_TYPE
```

```
-----
```

```
CLOSED
```

```
UNKNOWN
```

```
SQL> administer key management set keystore open force keystore identified by delphix;  
keystore altered.
```

```
SQL> @wallet
```

```
WRL_PARAMETER
```

```
STATUS
```

```
WALLET_TYPE
```

```
-----
```

```
OPEN
```

```
PASSWORD
```

No Keys !?!

```
SQL> @keys
```

```
no rows selected
```

```
SQL> select con_id, name, cause, message, status from PDB_PLUG_IN_VIOLATIONS where name = 'DEVDB';
```

CON_ID	NAME	CAUSE.	MESSAGE.	STATUS
4	DEVDB.	Wallet Key Needed.	PDB needs to import keys from source.	PENDING

```
SQL> administer key management import encryption keys with secret "delphix" from '/tmp/devdb.p12' FORCE  
KEYSTORE IDENTIFIED BY delphix with backup;
```

```
keystore altered.
```

```
SQL> shutdown
```

```
Pluggable Database closed.
```

```
SQL> startup
```

```
Pluggable Database opened.
```

```
SQL> select con_id, name, cause, message, status from PDB_PLUG_IN_VIOLATIONS where name = 'DEVDB';
```

CON_ID	NAME.	CAUSE.	MESSAGE.	STATUS
4	DEVDB.	Wallet Key Needed.	PDB needs to import keys from source.	RESOLVED

No Keys ?!?

```
SQL> @wallet
```

```
WRL_PARAMETER
```

```
STATUS
```

```
WALLET_TYPE
```

```
-----  
OPEN
```

```
-----  
PASSWORD
```

```
SQL> @keys
```

```
no rows selected
```

```
SQL> select * from test;
```

```
NAME
```

```
-----  
Marcin
```


No Keys ?!?

```
SQL> alter session set container = cdb$root;  
Session altered.
```

```
SQL> @keys
```

KEY_ID	TAG	CON_ID
Ac3KWSggv0/hv9ZDtD+qmpEAAAAAAAAAAAAAAAAAAAAAAAAAAAA	initial CDB key	1
AbEgNxflw09Sv4RWTMA2oM4AAAAAAAAAAAAAAAAAAAAAAAAAAAA	initial devdb key	3

```
SQL> show pdbs
```

CON_ID	CON_NAME	OPEN MODE	RESTRICTED
2	PDB\$SEED	READ ONLY	NO
4	DEVDB	READ WRITE	NO

No Keys ?!?

<https://docs.oracle.com/en/database/oracle/oracle-database/19/asoag/managing-keystores-encryption-keys-in-united-mode.html#GUID-34325A34-02A1-445E-BE5A-993CA5EDB926>

“After you create the cloned PDB, encrypted data is still accessible by the clone using the master encryption key of the original PDB. After a PDB is cloned, there may be user data in the encrypted tablespaces. This encrypted data is still accessible because the master encryption key of the source PDB is copied over to the destination PDB. Because the clone is a copy of the source PDB but will eventually follow its own course and have its own data and security policies, you should rekey the master encryption key of the cloned PDB.”

Also here Doc ID 2764277.1

Clone PDB

```
create pluggable database pdb2 from pdb1 file_name_convert=('PDB1','PDB2');  
create pluggable database pdb2 from pdb1 file_name_convert=('PDB1','PDB2')  
*
```

ERROR at line 1:

ORA-46697: Keystore password required.

```
create pluggable database pdb2 from pdb1 file_name_convert=('PDB1','PDB2') keystore identified by xxx;
```

Pluggable database created.

Clone PDB

```
@show_keys
```

CON_ID	KEY_ID	TAG
3	AZAsw6mosk/pv7DQnWwc2XMAAAAAAAAAAAAAAAAAAAAAAAAAAAA	rekey 1212
1	AZZzCkrae08Zv1Th20jqQ3MAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	rekey 1212
3	AWYbVyB7b08Iv2cLfpFuetcAAAAAAAAAAAAAAAAAAAAAAAAAAAA	rekey 1554

```
alter session set container = pdb2;  
Session altered.
```

```
select * from pioro.test_ts_enc where rownum < 10;  
select * from pioro.test_ts_enc where rownum < 10  
*
```

```
ERROR at line 1:  
ORA-28365: wallet is not open
```

Clone PDB

```
administer key management set keystore open force keystore identified by delphix;  
keystore altered.
```

```
administer key management set key using tag "rekey new pdb" identified by delphix with backup using  
"backup_0511_before_newpdb" container = all;
```

```
keystore altered.
```

```
@show_keys
```

CON_ID	KEY_ID	TAG
6	AZROuKlafk96v73nWf2gGtoAAAAAAAAAAAAAAAAAAAAAAAAAAAA	rekey new pdb
3	Ad9MKwByoU9yv1bOAnw2NhkAAAAAAAAAAAAAAAAAAAAAAAAAAAA	rekey new pdb
1	AeVcqaBYnU9+v+z9zwwarIEAAAAAAAAAAAAAAAAAAAAAAAAAAAA	rekey new pdb

Clone PDB – 19c

```
SQL> select ts#, ENCRYPTEDKEY, MASTERKEYID, CON_ID from v$encrypted_tablespaces order by ts#;
```

TS#	ENCRYPTEDKEY	MASTERKEYID	CON_ID
0	2B5EFD42D32C24190A401BE3CD2FBC32DDA2647F0CB1BA8683E4165E1A935025	4AE87763092B4F45BFE2C5BB635733F4	3
1	2B5EFD42D32C24190A401BE3CD2FBC32DDA2647F0CB1BA8683E4165E1A935025	4AE87763092B4F45BFE2C5BB635733F4	3
2	2B5EFD42D32C24190A401BE3CD2FBC32DDA2647F0CB1BA8683E4165E1A935025	4AE87763092B4F45BFE2C5BB635733F4	3
6	739737B12D4EC2A19F1E30D5C4B2D931BD2D9AA5810754E41D3ABA30A629D6D6	4AE87763092B4F45BFE2C5BB635733F4	3

```
SQL> select ts#, ENCRYPTEDKEY, MASTERKEYID, CON_ID from v$encrypted_tablespaces order by ts#;
```

TS#	ENCRYPTEDKEY	MASTERKEYID	CON_ID
0	2B5EFD42D32C24190A401BE3CD2FBC32DDA2647F0CB1BA8683E4165E1A935025	4AE87763092B4F45BFE2C5BB635733F4	5
1	2B5EFD42D32C24190A401BE3CD2FBC32DDA2647F0CB1BA8683E4165E1A935025	4AE87763092B4F45BFE2C5BB635733F4	5
2	2B5EFD42D32C24190A401BE3CD2FBC32DDA2647F0CB1BA8683E4165E1A935025	4AE87763092B4F45BFE2C5BB635733F4	5
6	2BEEFE9F5E9A9B0182382CFD9EB46E85F3D3A2B9F0FF8CA3970C5B9E94F95061	4AE87763092B4F45BFE2C5BB635733F4	5

Clone PDB 19c

“By default, during a PDB clone or relocate operation, the data encryption keys are rekeyed, which implies a re-encryption of all encrypted tablespaces. This rekey operation can increase the time it takes to clone or relocate a large PDB. With the optional `NO REKEY` clause, the data encryption keys are not renewed, and encrypted tablespaces are not re-encrypted.”

<https://docs.oracle.com/en/database/oracle/oracle-database/19/asoag/managing-keystores-encryption-keys-in-united-mode.html#GUID-34325A34-02A1-445E-BE5A-993CA5EDB926>

```
SQL> create pluggable database sysenc3 from devdb FILE_NAME_CONVERT=('devdb','sysenc3') keystore
identified by delphix no rekey;
```

```
Pluggable database created.
```

Clone PDB 19c

```
dd if=/u01/app/oracle/oradata/PIOR019/devdb/ts_pii.dbf of=block_devdb.dmp bs=8k count=8 skip=128
```

```
dd if=/u01/app/oracle/oradata/PIOR019/sysenc/ts_pii.dbf of=block_synenc_rk.dmp bs=8k count=8 skip=128
```

```
dd if=/u01/app/oracle/oradata/PIOR019/sysenc3/ts_pii.dbf of=block_synenc3.dmp bs=8k count=8 skip=128
```

```
[oracle@hobbiton ~]$ md5sum block_devdb.dmp
```

```
ed179bcbdaeb2df7e5deec7ccce31f9f  block_devdb.dmp
```

```
[oracle@hobbiton ~]$ md5sum block_synenc_rk.dmp
```

```
4e650dac8572b65a78b6466bbf30ac01  block_synenc_rk.dmp
```

```
[oracle@hobbiton ~]$ md5sum block_synenc3.dmp
```

```
ed179bcbdaeb2df7e5deec7ccce31f9f  block_synenc3.dmp
```


Non-prod
environments

Copy to non-prod

Is it Possible to Remove/Disable TDE?

NO

Doc ID 2488898.1 - Last update: 21-Jan-2019 !!!

Deleting the TDE wallet will not disable TDE. Once TDE wallet is configured, the wallet should never be deleted or recreated.

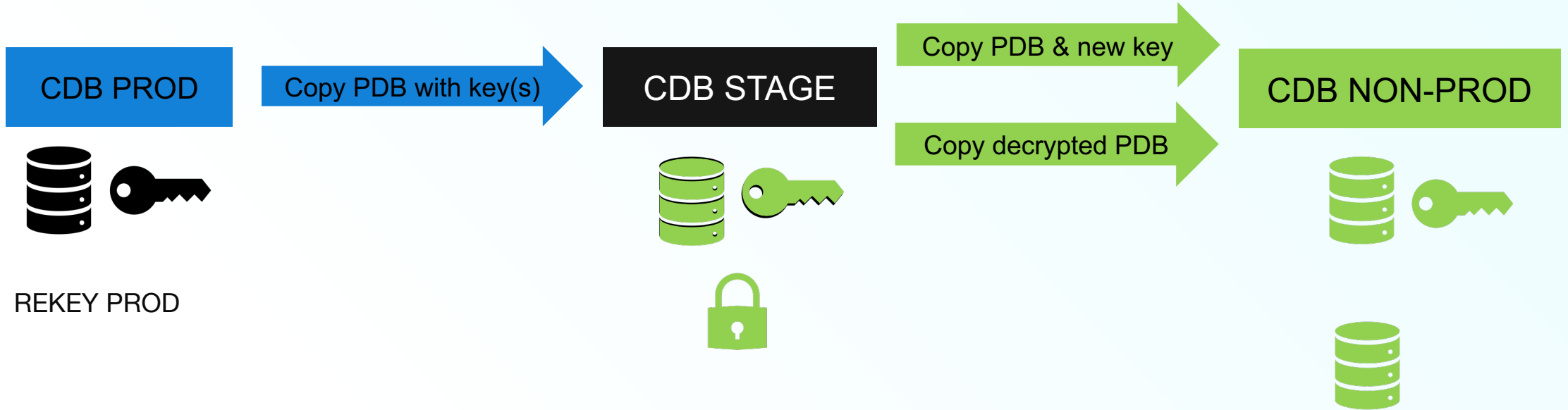
...

Even if there are no encrypted objects in the database, the TDE wallet has to be present in the wallet location. It does not cause any harm.

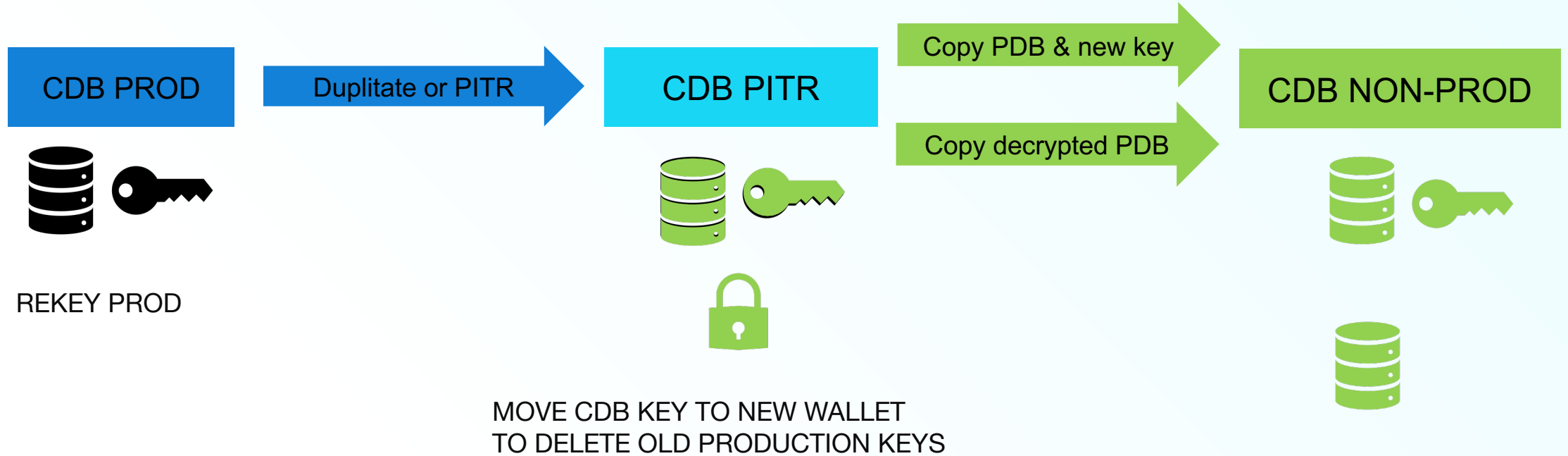
...

Recreating the wallet using any parameters is not supported. Oracle Support/Development team will not help in resolving any issues arising due to such operations.

How to move data to non-prod



How to move data to non-prod



Decrypt data

```
alter tablespace TS_PII offline normal;  
Tablespace altered.
```

```
alter tablespace TS_PII encryption offline decrypt  
*
```

```
ERROR at line 1:
```

```
ORA-28435: cannot decrypt data file  
/u01/app/oracle/oradata/PIOR019/devdb/ts_pii01.dbf which is not encrypted with  
the database key
```

ORA-28435: Cannot Decrypt Data File %s Which Is Not Encrypted With The Database Key (From ALTER DATABASE DATAFILE ... DECRYPT) (Doc ID 2406173.1)

This is expected behavior ...

Use:

```
alter table ... move tablespace ts_non_encrypted;
```

Decrypt data

...

LOGFILE

```
GROUP 1 '/prodb/delphix/VTEST/datafile/PROD/onlinelog/o1_mf_1_gc1cbf1b_.log' SIZE 52428800,  
GROUP 2 '/prodb/delphix/VTEST/datafile/PROD/onlinelog/o1_mf_2_gc1cbf4q_.log' SIZE 52428800,  
GROUP 3 '/prodb/delphix/VTEST/datafile/PROD/onlinelog/o1_mf_3_gc1cbf57_.log' SIZE 52428800
```

2019-04-30T12:14:54.876814+01:00

Clearing online redo logfile 1 complete

Clearing online redo logfile 2 complete

Clearing online redo logfile 3 complete

Online log /prodb/delphix/VTEST/datafile/PROD/onlinelog/o1_mf_1_gc1cbf1b_.log: Thread 1 Group 1 was previously cleared

Online log /prodb/delphix/VTEST/datafile/PROD/onlinelog/o1_mf_2_gc1cbf4q_.log: Thread 1 Group 2 was previously cleared

Online log /prodb/delphix/VTEST/datafile/PROD/onlinelog/o1_mf_3_gc1cbf57_.log: Thread 1 Group 3 was previously cleared

Online log /prodb/delphix/VTEST/datafile/VTEST/onlinelog/o1_mf_4_gdjcnr1_.log: Thread 1 Group 4 was previously cleared

Online log /prodb/delphix/VTEST/datafile/VTEST/onlinelog/o1_mf_5_gdjcnso_.log: Thread 1 Group 5 was previously cleared

Completed: alter database open resetlogs

2019-04-30T12:14:59.915339+01:00

Decrypt data

2019-04-30T12:15:01.117125+01:00

Shutting down instance (abort) (OS id: 2460)

...

2019-04-30T12:15:23.271945+01:00

ALTER DATABASE OPEN

2019-04-30T12:15:23.756549+01:00

Beginning crash recovery of 1 threads

parallel recovery started with 7 processes

2019-04-30T12:15:23.938460+01:00

Recovery of Online Redo Log: Thread 1 Group 2 Seq 2 Reading mem 0

Mem# 0: /proddb/delphix/VTEST/datafile/PROD/onlinelog/o1_mf_2_gc1cbf4q_.log

2019-04-30T12:15:23.938755+01:00

Completed redo application of 0.05MB

2019-04-30T12:15:23.979012+01:00

Slave encountered ORA-28365 exception during crash recovery

Slave exiting with ORA-28365 exception

2019-04-30T12:15:23.979455+01:00

Errors in file /product/oracle/12c/ora/diag/rdbms/VTEST/VTEST/trace/VTEST_p003_2601.trc:

ORA-28365: wallet is not open

Decrypt data

- Make sure there is no sign of encryption in:
 - redo logs (switch or recreate them)
 - undo tablespace (recreate ?)
 - archive log ?
- If you are cloning using snapshot / crash consistency tools – make sure you checkpoint your database

Move key between production / non-production

- No easy method in 12.1, 12.2 and 18c:
 - Doc ID 2216279.1 - How to delete old master keys from 12c TDE keystore (wallet).
 - Requires a open database but also requires to restart a database during a process

Move key – 19c

```
SQL> @keys
```

KEY_ID	TAG	CON_ID
AVpZ8Ohu50/jv2uvQxp59rQAAAAAAAAAAAAAAAAAAAAAAAAAAAA	new master 2	1
AchUP2/wAk+Uv0TcPpNFkTYAAAAAAAAAAAAAAAAAAAAAAAAAAAA	new master	1
AfF2WgXH0E+Vv8cG9hM7uacAAAAAAAAAAAAAAAAAAAAAAAAAAAA	new master	1
AUVjlyq+0E9kv9REfPVEhsIAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	new master	1

```
SQL> ADMINISTER KEY MANAGEMENT MOVE KEYS TO NEW KEYSTORE '/home/oracle/newkeys2' IDENTIFIED BY nowy  
FROM FORCE KEYSTORE IDENTIFIED BY delphix WITH IDENTIFIER IN  
'AchUP2/wAk+Uv0TcPpNFkTYAAAAAAAAAAAAAAAAAAAAAAAAAAAA', 'AfF2WgXH0E+Vv8cG9hM7uacAAAAAAAAAAAAAAAAAAAA  
AAAAAA', 'AUVjlyq+0E9kv9REfPVEhsIAAAAAAAAAAAAAAAAAAAAAAAAAAAAA' with backup;  
keystore altered.
```

```
SQL> @keys
```

KEY_ID	TAG	CON_ID
AVpZ8Ohu50/jv2uvQxp59rQAAAAAAAAAAAAAAAAAAAAAAAAAAAA	new master 2	1

Move key – 19c

```
SQL> @keys
```

KEY_ID	TAG	CON_ID
AZ/qG+GDmE/Tv7mpI8btokQAAAAAAAAAAAAAAAAAAAAAAAAAAAA	rekey pdb	5
Ad9nnfwUv0/Yv98C432fxoQAAAAAAAAAAAAAAAAAAAAAAAAAAAA	sysenc2-nonprod	5

```
SQL> select ts#, MASTERKEYID, CON_ID from v$encrypted_tablespaces order by ts# ;
```

TS#	MASTERKEYID	CON_ID
6	DF679DFC14BF4FD8BFDF02E37D9FC684	Ad9nnfwUv0/Yv98C432fxoQ

```
SQL> alter session set container = cdb$root;
```

```
Session altered.
```

```
SQL> ADMINISTER KEY MANAGEMENT MOVE KEYS TO NEW KEYSTORE '/home/oracle/newkeys2' IDENTIFIED BY nowy  
FROM FORCE KEYSTORE IDENTIFIED BY test WITH IDENTIFIER IN  
'AZ/qG+GDmE/Tv7mpI8btokQAAAAAAAAAAAAAAAAAAAAAAAAAAAA' with backup;
```

```
*
```

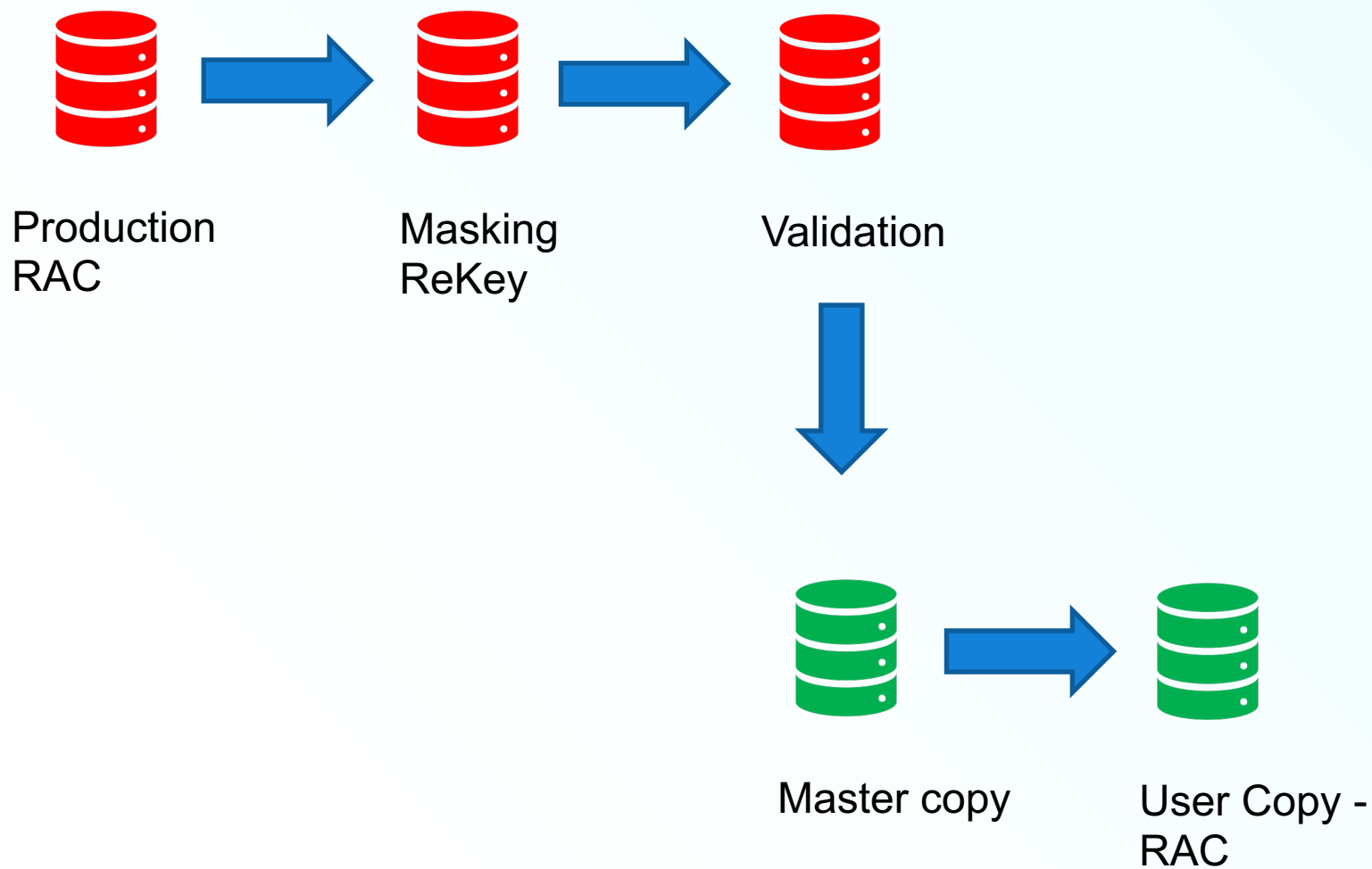
```
ERROR at line 1:
```

```
ORA-46688: cannot move a master key
```

Move key between production / non-production

- No easy method in 12.1, 12.2, 18c and 19c:
 - Doc ID 2216279.1 - How to delete old master keys from 12c TDE keystore (wallet).
 - Requires a open database but also requires to restart a database during a process

Real config



Solution based on recovery / unplug / plug / export / import

```
select con_id, status from v$encryption_wallet;
```

```
CON_ID      STATUS
-----
OPEN_NO_MASTER_KEY
```

```
select * from XXX.TEST;
select * from XXX.TEST;
```

```
ERROR at line 1:
ORA-28374: typed master key not found in wallet
```

```
select id from table;
```

```
ID
--
10
```

```
update table set id = 20;
```

```
ORA-28374: typed master key not found in wallet
```

```
alter pluggable database "test" open read write
ERROR at line 1: 8ORA-30013: undo tablespace
'\''BIGTBS'\'' is currently in use'
```

Set same key

```
SQL> ADMINISTER KEY MANAGEMENT CREATE KEY using tag 'sysenc2-nonprod' FORCE KEYSTORE identified by
test with backup;
keystore altered.
```

```
SQL> @keys
```

KEY_ID	TAG	ACTIVATION_TIME
AZ/qG+GDmE/Tv7mpI8btokQAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	rekey pdb	18-SEP-22 09.57.57.445127 AM
Ad9nnfwUv0/Yv98C432fxoQAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	sysenc2-nonprod	

```
SQL> administer key management use key 'Ad9nnfwUv0/Yv98C432fxoQAAAAAAAAAAAAAAAAAAAAAAAAAAAAA' force
keystore identified by test with backup;
Keystore altered.
```

```
SQL> @keys
```

KEY_ID	TAG	ACTIVATION_TIME
AZ/qG+GDmE/Tv7mpI8btokQAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	rekey pdb	18-SEP-22 09.57.57.445127 AM
Ad9nnfwUv0/Yv98C432fxoQAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	sysenc2-nonprod	18-SEP-22 02.32.39.984053 PM

Set same key

```
SQL> shutdown
```

```
Pluggable Database closed.
```

```
SQL> alter session set container = cdb$root;
```

```
Session altered.
```

```
SQL> drop pluggable database sysenc2 including datafiles;
```

```
Pluggable database dropped.
```

```
SQL> create pluggable database sysenc2 using '/tmp/sysenc2_fc.pdb' decrypt using delphix keystore  
identified by test FILE_NAME_CONVERT=('/tmp', '/u01/app/oracle/oradata/TGT19/sysenc2');
```

```
Pluggable database created.
```


Set same key

```
SQL> alter session set container = sysenc2;  
Session altered.
```

```
SQL> administer key management set keystore open force keystore identified by test;  
keystore altered.
```

```
SQL> startup  
Pluggable Database opened.
```

```
SQL> @keys  
no rows selected
```

```
SQL> administer key management use key 'Ad9nnfwUv0/Yv98C432fxoQAAAAAAAAAAAAAAAAAAAAAAAAAAAA' force  
keystore identified by test with backup;  
keystore altered.
```

```
SQL> @keys
```

KEY_ID	TAG	CON_ID
Ad9nnfwUv0/Yv98C432fxoQAAAAAAAAAAAAAAAAAAAAAAAAAAAA	sysenc2-nonprod	3

I am using TDE because:

- A) database is in the regulated industry, and I have to
- B) I want to feel safe
- C) I want to increase a risk of my own ransomware attack by losing my keystore or password to a keystore

Key take aways

- Upgrade to 19c
- Build a strategy to copy your production database to non-prod
- Make sure your protection needs are fulfilled by TDE

Q&A

@pioro