ORACLE

TDE From A Non-Security Guy

Learnings from my baby steps

Daniel Overby Hansen
Senior Principal Product Manager

Daniel Overby Hansen

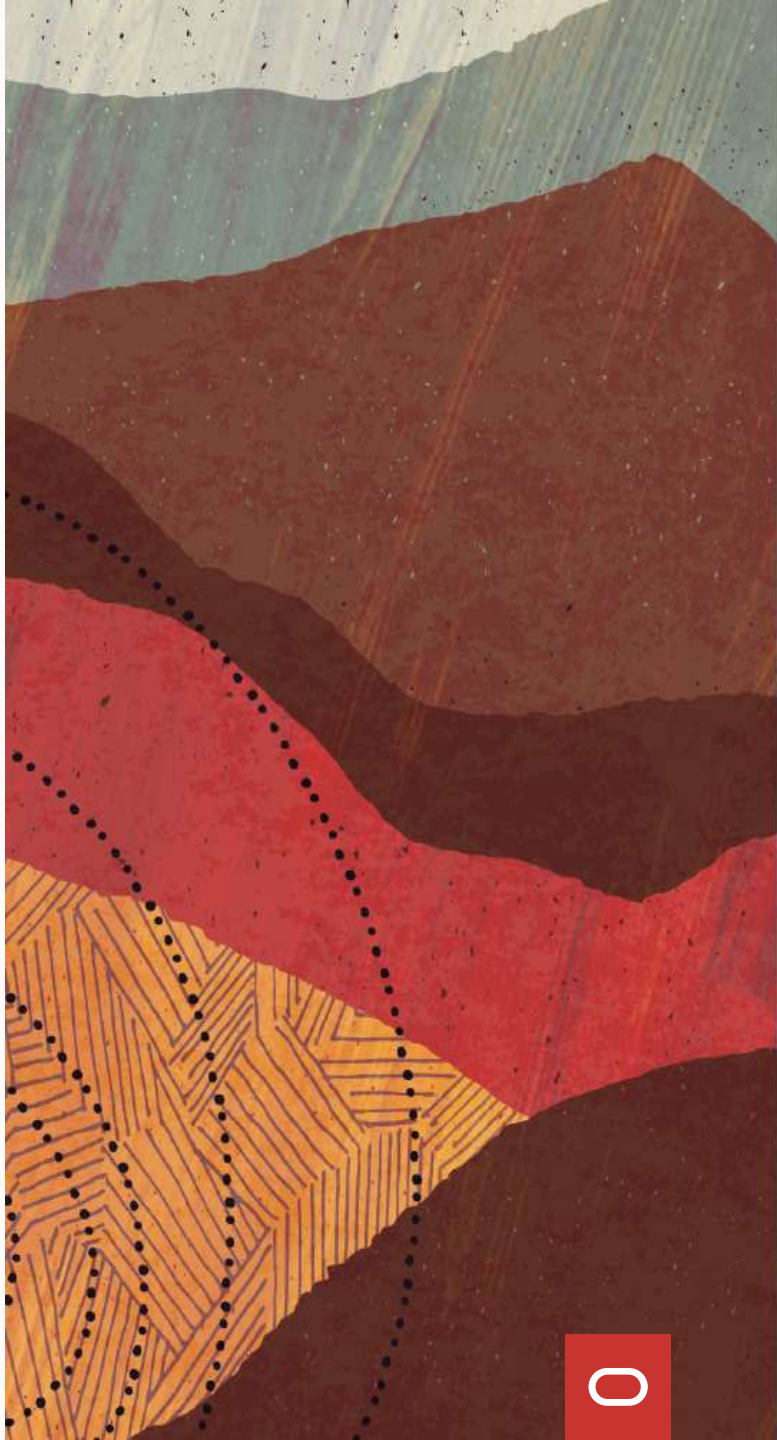
Senior Principal Product Manager
Cloud Migration

https://dohdatabase.com

@dohdatabase

dohdatabase

https://dohdatabase.com



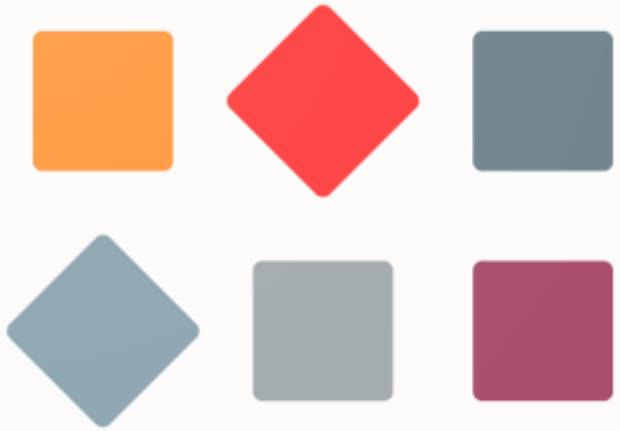
Safe Harbor

I am no security expert. I am just a simple Product Manager that was forced to expand my knowledge about TDE because I kept running into problems. I have worked as DBA for many years happily unaware about the wonders of encryption and TDE. But there is no cloud without encryption! The following is a summary of endless nights of reading about keys, algorithms, wallets and many other wonderful things.

Introduction

14

Introduction | Quick Poll



How long were you a DBA before you had to worry about encryption?

Transparent Data Encryption

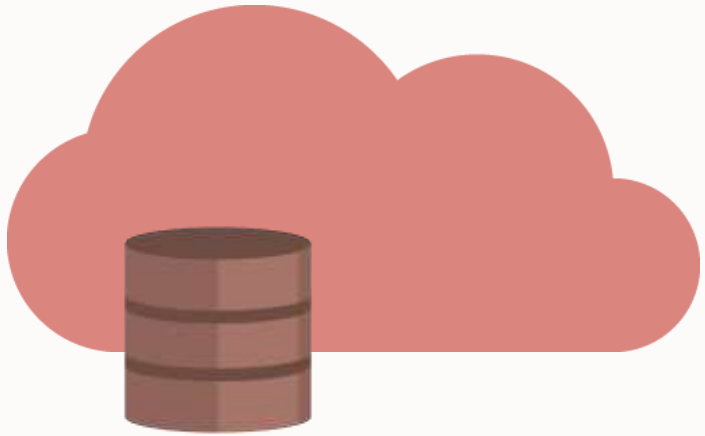
Introduction | Definition

TDE enables you to encrypt data so that only an authorized recipient can read it.

Use encryption to protect sensitive data in a potentially unprotected environment, such as data you placed on backup media that is sent to an off-site storage location. To use Transparent Data Encryption, you do not need to modify your applications. TDE enables your applications to continue working seamlessly as before. It automatically encrypts data when it is written to disk, and then automatically decrypts the data when your applications access it. Key management is built-in, eliminating the complex task of managing and securing encryption keys.

Introduction | Yesterday

Introduction | Today



The Basics

The Basics | Database

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- TDE available in all supported releases

The Basics | License

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

Feature / Option / Pack	SE2	EE	EE-ES	DBCS SE	DBCS EE	DBCS EE-HP	DBCS EE-EP	ExaCS	Notes
Column-Level Encryption	N	Y	Y	N	N	Y	Y	Y	**EE** and **EE-ES**: Requires the Oracle Advanced Security option
Tablespace Encryption	N	Y	Y	Y	Y	Y	Y	Y	**EE** and **EE-ES**: Requires the Oracle Advanced Security option
Oracle Advanced Security	N	Y	Y	N	N	Y	Y	Y	**EE** and **EE-ES**: Extra cost option
Oracle Database Vault	N	Y	Y	N	N	Y	Y	Y	**EE** and **EE-ES**: Extra cost option
Oracle Label Security	N	Y	Y	N	N	Y	Y	Y	**EE** and **EE-ES**: Extra cost option

The Basics | Keystore

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- Keystore type and location defined by
  - `WALLET_ROOT` and `TDE_CONFIGURATION` (initialization parameters)
  - `SQLNET.ENCRYPTION_WALLET_LOCATION` (sqlnet.ora) - deprecated
  - `$ORACLE_BASE/admin/`*`db_unique_name`*`/wallet` (software keystores only)
  - `$ORACLE_HOME/admin/`*`db_unique_name`*`/wallet` (software keystores only)

The Basics | Keystore Configuration

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- Set `WALLET_ROOT` parameter in CDB$ROOT

```
SQL> ALTER SYSTEM
     SET WALLET_ROOT='$ORACLE_BASE/admin/$ORACLE_SID/wallet'
     SCOPE=SPFILE;
```

- Set `TDE_CONFIGURATION` in CDB$ROOT

```
SQL> ALTER SYSTEM
     SET TDE_CONFIGURATION='KEYSTORE_CONFIGURATION=FILE'
     SCOPE=BOTH;
```

- Keystore must be stored in subfolder named *tde*

```
SQL> ADMINISTER KEY MANAGEMENT
     CREATE KEYSTORE '$ORACLE_BASE/admin/$ORACLE_SID/wallet/tde'
     IDENTIFIED BY "S3cr3t";
```

- Setting `TDE_CONFIGURATION` in a PDB enables isolated keystore mode

The Basics | Keystore

- Database
- License
- **Keystore**
- Encryption Key
- Algorithm
- Mode

- Software
  - Password protected
  - Auto-login
  - Local auto-login
- Hardware
  - Hardware Security Module
  - Oracle Key Vault

Pro Tip:
Wallets and keystores are the same.
Keystore is the *new* term.

The Basics | Password Protected Keystore

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- Required

```
SQL> ADMINISTER KEY MANAGEMENT
     CREATE KEYSTORE '$ORACLE_BASE/admin/$ORACLE_SID/wallet/tde'
     IDENTIFIED BY <keystore-password>;
```

- Encrypted file - protected by keystore password
- File name is always `ewallet.p12`
- Restrict access to the files (`chmod 500`)
- Keep your keystore password safe - don't forget it

Pro Tip:
You can read more about the keystore types in the documentation.

The Basics | Password Protected Keystore

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- DBA must enter keystore password when database starts
- Keystore folder must be created in advance
- Never delete keystore files

Pro Tip:
Keystore file follows PKCS #12 and PKCS #5 format

The Basics | Auto-login Keystore

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- Optional

```
SQL> ADMINISTER KEY MANAGEMENT
     CREATE AUTO_LOGIN KEYSTORE
     FROM KEYSTORE '$ORACLE_BASE/admin/$ORACLE_SID/wallet/tde'
     IDENTIFIED BY S3cr3t;
```

- Allows the database to start without DBA intervention
- Auto-login keystore is protected by password generated by the database
- File name is always `cwallet.sso`

The Basics | Local Auto-login Keystore

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- Optional

```
SQL> ADMINISTER KEY MANAGEMENT
     CREATE LOCAL AUTO_LOGIN KEYSTORE
     FROM KEYSTORE '$ORACLE_BASE/admin/$ORACLE_SID/wallet/tde'
     IDENTIFIED BY S3cr3t;
```

- Similar to auto-login keystore
- Can only be used on server where it was generated

The Basics | Encryption Key

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- TDE Master Encryption Key

- Generated by database

```
SQL> ADMINISTER KEY MANAGEMENT
     SET KEY IDENTIFIED BY <keystore password>
     WITH BACKUP USING 'creating initial key';
```

- "Bring-your-own-key"
  - allow you to set a TDE master encryption key yourself

Pro Tip:
Also referred to as *master key* or *master encryption key*

The Basics | Algorithm

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- Default and recommended: AES128

Algorithm	Key Size	Parameter Name
Advanced Encryption Standard (AES)	• 128 bits (default for tablespace encryption) • 192 bits (default for column encryption) • 256 bits	• AES192 • AES128 • AES256
ARIA	• 128 bits • 192 bits • 256 bits	• ARIA128 • ARIA192 • ARIA256
GOST	256 bits	GOST256
SEED	128 bits	SEED128
Triple Encryption Standard (DES)	168 bits	3DES168

Pro Tip:
To change default algorithm visit
My Oracle Support Document ID 2654121.1

The Basics | Offline Encryption

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

- Syntax:

```
SQL> ALTER TABLESPACE users OFFLINE NORMAL;
     ALTER TABLESPACE users ENCRYPTION OFFLINE DECRYPT;
     ALTER TABLESPACE users ONLINE;
```

- System tablespaces done in `MOUNT` mode

- Encryption is done serially.
  Parallelize by using multiple sessions on multiple tablespaces

NEW IN 12.2

The Basics | Online Encryption

- Database
- License
- Keystore
- Encryption Key
- Algorithm
- Mode

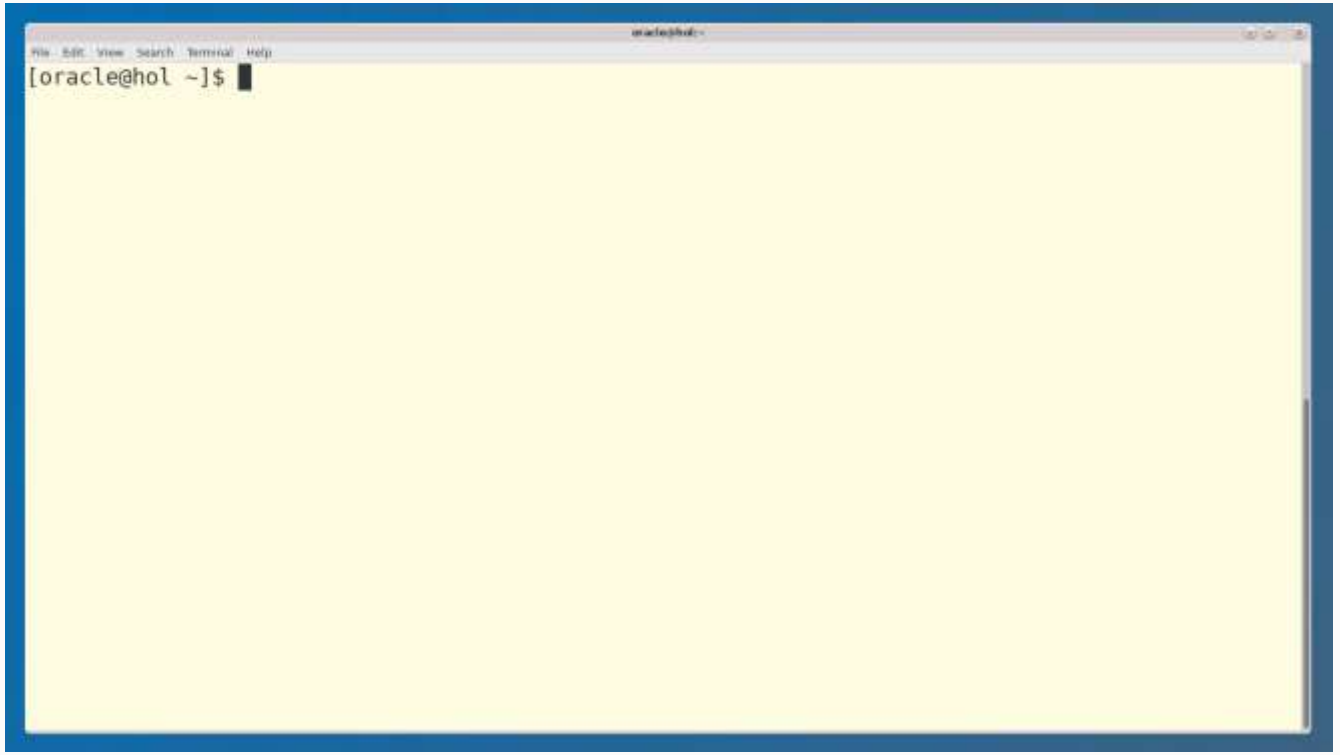
- Syntax:

```
SQL> ALTER TABLESPACE system ENCRYPTION ONLINE ENCRYPT;

SQL> ALTER TABLESPACE system ENCRYPTION ONLINE ENCRYPT
     FILE_NAME_CONVERT=('system01.dbf','system01_enc.dbf');
```

- Works on all data tablespaces, including UNDO and SYSTEM
- Parallelize by using multiple sessions on multiple tablespaces
- Don't parallelize SYSTEM and UNDO
- Requires additional disk space - 2x

The Basics | Demo



Architecture

Architecture | Privileges

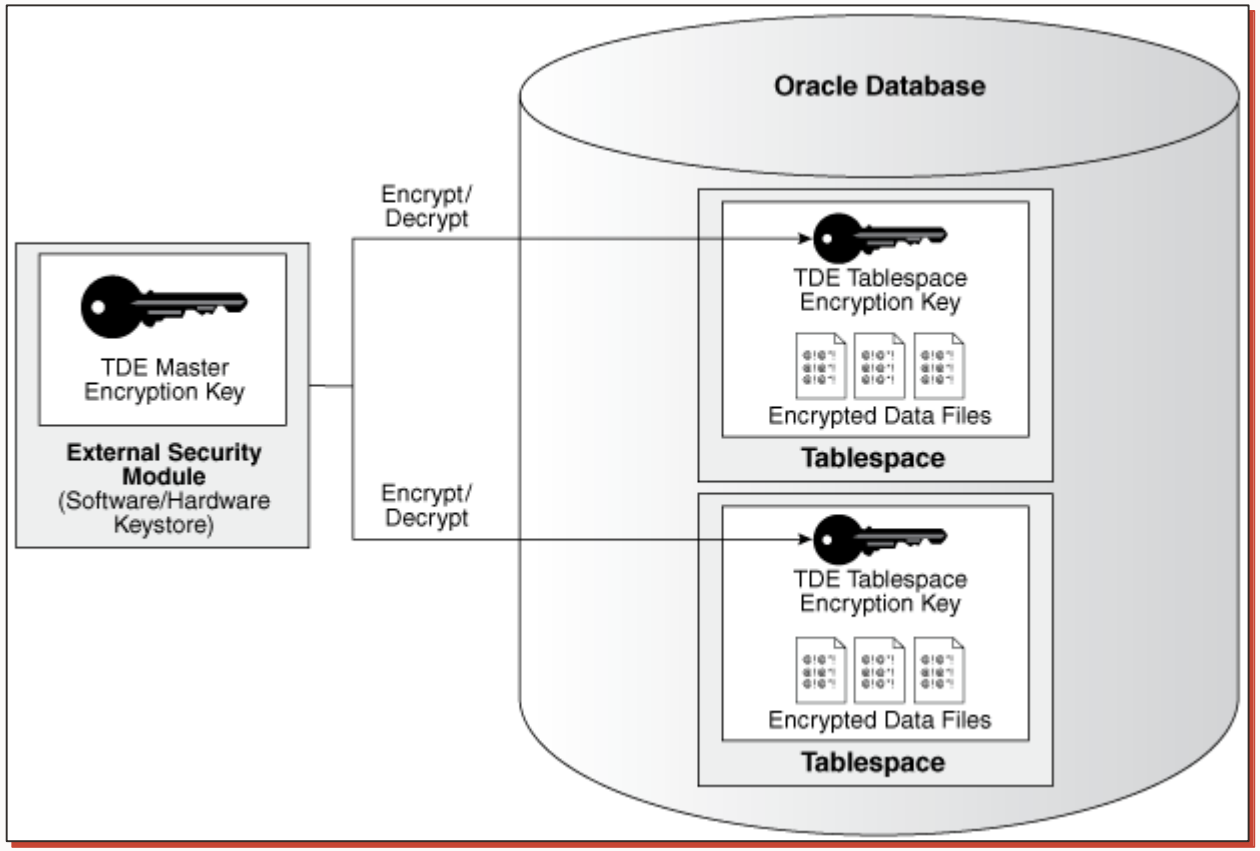
- System privilege: ADMINISTER KEY MANAGEMENT
  - Allows use of encryption commands

- Administrative privilege: SYSKM
  - Allows connection to database while closed
  - ADMINISTER KEY MANAGEMENT
  - CREATE SESSION
  - SELECT on a few views

Architecture | Performance

- Performance overhead? Yes!
  - Expect low single digit numbers
- Minimize overhead
  - Newer database version
  - Newer hardware
  - Newer operating systems
  - CPUs with crypto acceleration capabilities
- Storage overhead? No!

Architecture | Two-tiered Key-based



Pro Tip:
Check Advanced Security Guide for more information.

Architecture | Data Guard

- If TDE is in use, redo is automatically encrypted
- Each standby database must have a copy of the keystore
- Hybrid Data Guard to OCI
  - On-premises primary not encrypted
  - OCI standby database is encrypted
  - Technical Brief

Pro Tip:
To encrypt with data guard, minimize downtime by encrypting standby first, switching over, and then encrypt primary

Architecture | RAC

- All nodes must have access to the same keystore

- Store keystore in:
  - ASM (recommended)
  - ACFS
  - Local file system

Architecture | Multitenant

- Two keystore modes
  - Unified
  - Isolated (OCI-only)
- Can be combined
- To determine current mode:

```
SQL> SELECT con_id, keystore_mode FROM v$encryption_wallet;

 CON_ID    KEYSTORE_MODE
_________ _______________
         1 NONE
         2 UNITED
         3 UNITED
         4 ISOLATED
```

Architecture | Multitenant

- Setting `WALLET_ROOT` and `TDE_CONFIGURATION` in CDB$ROOT enables TDE
- Default is united keystore mode
- Setting `TDE_CONFIGURATION` in a PDB enables isolated keystore mode for that specific PDB

```
SQL> ALTER SESSION SET CONTAINER=PDB1;
SQL> ALTER SYSTEM
     SET TDE_CONFIGURATION='KEYSTORE_CONFIGURATION=FILE'
     SCOPE=BOTH;

SQL> ALTER SESSION SET CONTAINER=PDB2;
SQL> ALTER SYSTEM
     SET TDE_CONFIGURATION='KEYSTORE_CONFIGURATION=OKV'
     SCOPE=BOTH;
```

Operations

Operations | Temporary tablespaces

- Not possible to encrypt or decrypt
- Always create new tablespaces and drop existing

```
SQL> CREATE TEMPORARY TABLESPACE temp2 ENCRYPTION USING 'AES128' ENCRYPT;
SQL> ALTER DATABASE DEFAULT TEMPORARY TABLESPACE temp2;
SQL> DROP TABLESPACE temp INCLUDING CONTENTS AND DATAFILES;
```

- Needed?

> *"The encrypted data is protected during operations such as JOIN and SORT. This means that the data is safe when it is moved to temporary tablespaces. Data in undo and redo logs is also protected."*
>
> Advanced Security Guide 19c, chapter 2, Introduction to Transparent Data Encryption

Operations | Backup

- Backup the keystore
- Don't store keystore together with backup
- Backup current and all previous versions of the keystore
- No RMAN configuration required

Operations | Restore

- First, restore keystore
- Next, use RMAN to restore database

```
RMAN> RESTORE CONTROLFILE FROM '...';

RMAN> ALTER DATABASE MOUNT;

RMAN> sql 'ADMINISTER KEY MANAGEMENT
           SET KEYSTORE OPEN FORCE KEYSTORE IDENTIFIED BY S3cr3t';

RMAN> sql "ADMINISTER KEY MANAGEMENT
           CREATE LOCAL AUTO_LOGIN KEYSTORE
           FROM KEYSTORE ''...'' IDENTIFIED BY S3cr3t";

RMAN> RESTORE DATABASE;
```

- No keystore, no database!

NEW IN 12.2

Operations | Secure External Password Store

- Existed since 10g – however since 12.2 you can use it for storing keystore credentials
- Allows delegation of tasks and enforces separation of duties
- Some keystore actions still require the keystore password
- It is an (local) auto-login keystore - protected by system generated password

Pro Tip:
SEPS can also hold password to Oracle Key Vault

Operations | Secure External Password Store

- Without SEPS

```
SQL> .... KEYSTORE IDENTIFIED BY S3cr3t;
```

- With SEPS

```
SQL> .... KEYSTORE IDENTIFIED BY EXTERNAL STORE;
```

- For security reasons some `ADMINISTER KEY MANAGEMENT` commands still requires keystore password

Operations | Secure External Password Store

- Configuration

```
SQL> ALTER SYSTEM
     SET EXTERNAL_KEYSTORE_CREDENTIAL_LOCATION =
     '$ORACLE_BASE/admin/$ORACLE_SID/wallet/tde_seps'
     SCOPE=SPFILE;
SQL> SHUTDOWN IMMEDIATE
SQL> STARTUP
```

- Default location is `$ORACLE_BASE/admin/$ORACLE_SID/wallet/tde_seps`

- If you are using `WALLET_ROOT` – then `EXTERNAL_KEYSTORE_CREDENTIAL_LOCATION` is ignored
  You must use the default location

Pro Tip:
You have to create the directory in advance

Operations | Secure External Password Store

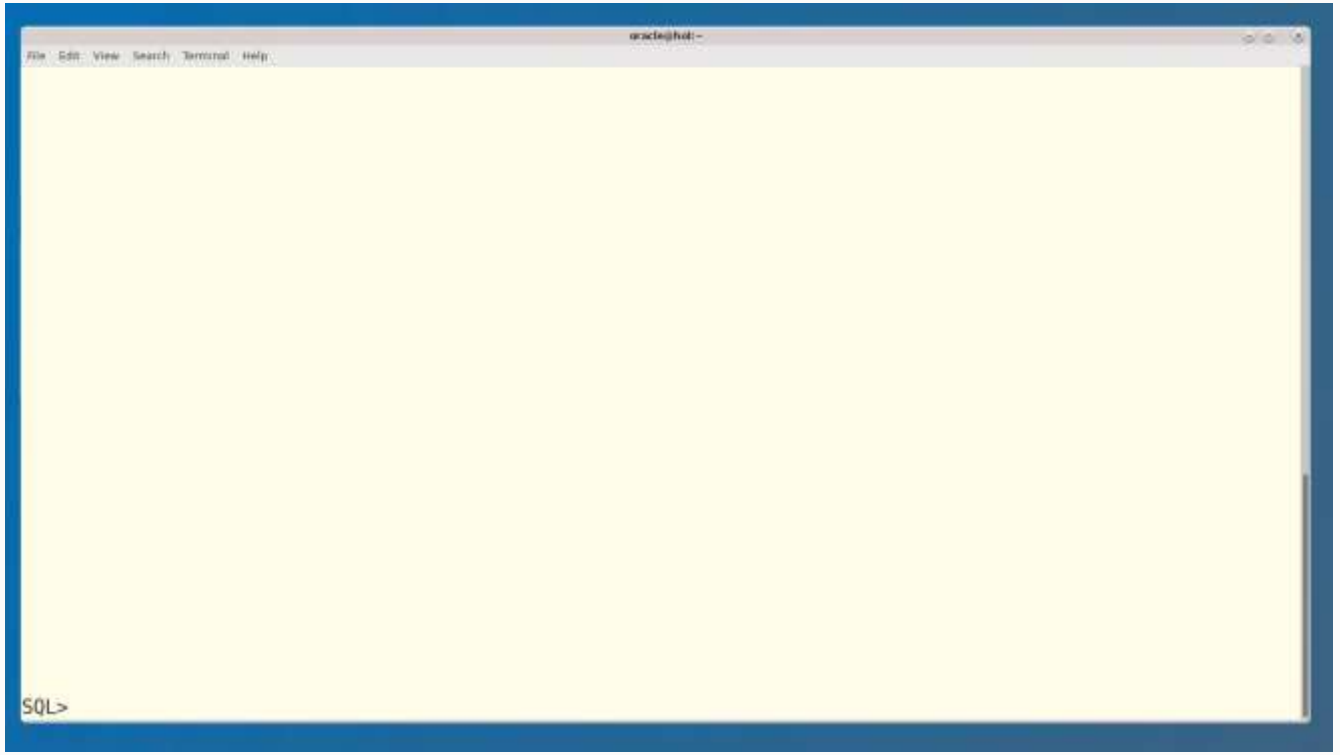
- Adding keystore password to SEPS

```
SQL> ADMINISTER KEY MANAGEMENT
     ADD SECRET 'S3cr3t' FOR CLIENT 'TDE_WALLET'
     USING TAG 'TDE keystore password'
     TO LOCAL AUTO_LOGIN KEYSTORE '$ORACLE_BASE/admin/$ORACLE_SID/wallet/tde_seps';
```

- Use `UPDATE SECRET` when you change keystore password

Pro Tip:
To store Oracle Key Vault password use `CLIENT 'OKV_PASSWORD'`

Operations | Secure External Password Store



Wrapping Up

Wrapping Up | Best Practice

- Use Secure External Password Store
- Use `WALLET_ROOT`
- Always use `WITH BACKUP USING`

```
SQL> ADMINISTER KEY MANAGEMENT
     SET KEY IDENTIFIED BY <keystore password>
     WITH BACKUP USING 'quarterly-rotate';

SQL> host ls $ORACLE_BASE/admin/$ORACLE_SID/wallet/tde
cwallet.sso ewallet_2020092311160618_quarterly-rotate.p12 ewallet.p12
```

- Never delete keystore files
- Use AES128

Wrapping Up | Old And New Syntax

Behavior	ALTER SYSTEM or orapki	ADMINISTER KEY MANAGEMENT
Creating a keystore	For software keystores (called wallets in previous releases): `ALTER SYSTEM SET ENCRYPTION KEY ["certificate_ID"] IDENTIFIED BY keystore_password;` For hardware keystores, the keystore is available after you configure the hardware security module.	For software keystores: `ADMINISTER KEY MANAGEMENT CREATE KEYSTORE 'keystore_location' IDENTIFIED BY software_keystore_password` For hardware keystores, the keystore is available after you configure the hardware security module.
Creating an auto-login keystore	`orapki wallet create -wallet wallet_location -auto_login [-pwd password]`	For software keystores: `ADMINISTER KEY MANAGEMENT CREATE [LOCAL] AUTO_LOGIN KEYSTORE FROM KEYSTORE 'keystore_location' IDENTIFIED BY software_keystore_password;` This type of keystore applies to software keystores only.

Advanced Security Guide 19c, chapter 7, How ALTER SYSTEM and orapki Map to ADMINISTER KEY MANAGEMENT

Wrapping Up | Further Reading

- Advanced Security Guide
- AskTOM Database Security Office Hours
- YouTube videos:
  - Transparent Data Encryption - Advanced Use Cases - Part 1
  - Transparent Data Encryption - Advanced Use Cases - Part 2

Wrapping Up | Questions

Thank you