Oracle

Oracle Database

19C

Integration Guide

CryptoServer HSM

SecurityServer 4.45.3, 4.45.5

utimaco®

Imprint

Copyright 2026	Utimaco IS GmbH Krefelder Straße 220 52070 Aachen Germany
Phone	AMERICAS +1-844-UTIMACO (+1 844-884-6226) EMEA +49 800-627-3081 APAC +81 800-919-1301
Internet	https://support.hsm.utimaco.com/
e-mail	support@utimaco.com
Document Version	1.0.0
Date	2026-03-20
Status	PUBLISHED
Document No.	IG-2026-0034

Table of Contents

1 Introduction 5
1.1 About This Guide 5
1.1.1 Target Audience for This Guide 5
1.1.2 Document Conventions 5
1.1.3 Abbreviations 6
2 Overview 9
2.1 Oracle TDE 9
2.2 Utimaco CryptoServer HSM 9
3 Integration Requirements and Prerequisites 10
3.1 Tested Versions 10
3.2 Software Requirements 10
3.3 Hardware Requirements 11
3.4 Prerequisites 11
4 Installing and Configuring Utimaco SecurityServer Software 12
4.1 On Linux 12
4.2 On Windows 13
4.3 Update cs_pkcs11_R3.cfg 13
5 Integrating Oracle Database for Windows/Linux 15
5.1 Copying Utimaco PKCS#11 Library File 15
5.2 Configuring Oracle DB to use Utimaco HSM 16
5.3 Generate Master Encryption Key (MEK) on to the HSM 16
5.4 Verify Encrypting Column in Table 19
5.5 Create an Encrypted Tablespace 21
5.6 Encrypting an Existing Tablespace with Online Conversion 23
5.7 Decrypting an Existing Tablespace with Online Conversion 27
5.8 Rekeying an Existing Tablespace with Online Conversion 28
5.9 Encrypting an Existing Tablespace with Offline Conversion 29
5.10 Decrypting an Existing Tablespace with Offline Conversion 32
5.11 Configure Auto-login for Hardware Keystore 34
5.12 Migrating Master Encryption Key from Software Keystore to the HSM Keystore 37
5.12.1 Create Software Keystore 37

5.12.2 Migrate the Software Keystore to the Utimaco HSM 44
5.13 Configure TDE with Pluggable DB using HSM 47
5.13.1 About Containers in a CDB 47
5.13.2 About PDBs 48
5.13.3 Configure TDE with Pluggable DB 48
5.13.4 To verify the Master Encryption Key is Encrypting the PDB 53
6 Integrating Oracle RAC with Utimaco HSM 57
6.1 Copying Utimaco PKCS#11 Library File to all Oracle RAC Instances 57
6.2 Generate Master Encryption Key (MEK) on the HSM 57
7 FIPS Requirements 61
8 Troubleshooting 62
9 Further Information 64
10 References 65
11 Contact and Support Information 66

1 Introduction

This guide is part of the information and support provided by Utimaco. Additional documentation produced to support your Utimaco SecurityServer product can be found in the document directory of the Utimaco SecurityServer product bundle. All Utimaco SecurityServer product documentation is available from Utimaco's web site at https://utimaco.com/.

1.1 About This Guide

This guide provides an integration explaining how to integrate an Utimaco CryptoServer Hardware Security Module (HSM) with Oracle database. Utimaco HSM is used to securely store the master encryption keys used by oracle database.

1.1.1 Target Audience for This Guide

This guide is intended for Oracle database administrators and Utimaco HSM administrators.

1.1.2 Document Conventions

The following conventions are used in this guide:

Convention	Use	Example
Bold	Items of the Graphical User Interface (GUI), e.g., menu options	Select **Details** and click on **Properties** button
`Monospaced`	Code that is given for explanation or as an example, file paths	`certreq.exe -new request.inf IISCertRequest.csr`
Italic	References and important terms	Operating system listed in *Tested Versions*

Document conventions

We use special icons to highlight the most important notes and information.

Here you will find important safety information that should be followed.

Here you will find additional notes or supplementary information.

This message marks the result expected after the successful execution of an instruction.

1.1.3 Abbreviations

The following abbreviations are used in this guide:

Abbreviation	***Meaning***
CDB	Container Database
CSADM	CryptoServer Command-line Administration Tool
CSAR	Cloud Service Architecture
DB	Database
DBA	Database Administration
DBCA	Database Configuration Assistant
EMP	Employee
FIPS	Federal Information Processing Standards

Abbreviation	***Meaning***
GUI	Graphical User Interface
GP HSM	General Purpose Hardware Security Module
HSM	Hardware Security Module
HMAC	Hash Message Authentication Code
IDs	Identity
IP	Internet Protocol
JRE	Java Runtime Environment
LAN	Local Area Network
MBK	Master Backup Key
MEK	Master Encryption Key
P11CAT	PKCS#11 CryptoServer Administration Tool
Abbreviation	*Meaning*
PDB	Pluggable Database
PKCS#11	Public-Key Cryptography Standard #11
PCIe	PCI Express Interface

Abbreviation	***Meaning***
RAC	Real Application Clusters
RSA	Rivest, Shamir, Adleman (cryptosystem)
SO	Security Officer
SQL	Structured Query Language
SYS	System
SYSDBA	System Database Administrator
TDE	Transparent Data Encryption
URL	Uniform Resource Locator

Table 1: List of abbreviations

2 Overview

2.1 Oracle TDE

Transparent Data Encryption (TDE) enables you to encrypt sensitive data that you store in tables and tablespaces.

After the data is encrypted, this data is transparently decrypted for authorized users or applications when they access this data. TDE helps protect data stored on media (also called data at rest) if the storage media or data file is stolen. Oracle Database uses authentication, authorization, and auditing mechanisms to secure data in the database, but not in the operating system data files where data is stored. To protect these data files, Oracle Database provides Transparent Data Encryption (TDE). TDE encrypts sensitive data stored in data files. To prevent unauthorized decryption, TDE stores the encryption keys in a security module external to the database, called a keystore.

Oracle provides a straightforward method of managing database credentials across multiple domains by using Oracle Wallets. These enable users to update the database credentials, without the need to change specific data source definitions, since the database connection string in the data source definition is resolved by an entry in the wallet.

If the security of the wallets and cryptographic material they contain needs to be enhanced, the Oracle Database needs to be configured to use a Hardware Security Module (HSM). When the HSM module is enabled with the Oracle Database, this strengthens the protection of the wallets.

2.2 Utimaco CryptoServer HSM

CryptoServer is a hardware security module developed by Utimaco IS GmbH. CryptoServer is a physically protected specialized computer unit designed to perform sensitive cryptographic tasks and to securely manage as well as store cryptographic keys and data. It can be used as a universal, independent security component for heterogeneous computer systems.

3 Integration Requirements and Prerequisites

Ensure that the system environment you will be using, meets the following hardware and software requirements.

This guide assumes that the user has already installed and configured Oracle database.

3.1 Tested Versions

This guide assumes that the user has already installed and configured Oracle database.

Operating System	***Oracle***	***Utimaco Security Server Version***	***Utimaco HSM***
Windows Server 2019 RedHat Linux 7.7	Oracle 19C	SecurityServer 4.45.3	CryptoServer CSe-Series/ SeSeries
Oracle Linux 8	Oracle 19C RAC	SecurityServer 4.45.5	CryptoServer CSe-Series/ SeSeries

Table 2: List of tested versions

3.2 Software Requirements

Software	***Software Requirements***
Oracle	Oracle 19c
JRE 8	Java Runtime Environment 8
HSM Interface	SecurityServer PKCS#11 Provider

Table 3: List of software requirements

3.3 Hardware Requirements

Hardware	***Hardware Requirements***
Utimaco LAN HSM	CryptoServer CSe-Series/Se-Series LAN with firmware SecurityServer 4.45 or higher
Utimaco PCI-e HSM	CryptoServer CSe-Series/Se-Series PCI-e with firmware SecurityServer 4.45 or higher

Table 4: List of hardware requirements

3.4 Prerequisites

Before you begin, please ensure that you have installed/setup:

- Operating system listed in Tested Versions.
- SecurityServer listed in Tested Versions.
- CryptoServer Default Admin should be replaced with a new admin user.
- MBK must be created and stored onto each HSM. Refer the CryptoServer documentations to setup the MBK.
- CryptoServer is setup and configured. Refer the CryptoServer documentations to setup the HSM.
- PKCS#11 library is setup and configured as per your environment. Refer the CryptoServer documentations to setup and configure the PKCS#11 library.
- The user with admin privileges is required for installing few packages on to the Oracle Database server.
- Oracle Database must be installed on the target machine to carry on with the integration process. For a detailed installation procedure of Oracle Database, refer to the Oracle Database documentation according to the desired version of oracle.

4 Installing and Configuring Utimaco SecurityServer Software

4.1 On Linux

1. Copy the downloaded software at the appropriate location on the Oracle Database Server
2. Create utimaco folder under /opt directory and further create 2 directories `/opt/utimaco/bin and /opt/utimaco/lib` .

›_ Console

```
# mkdir -p  /opt/utimaco/bin
# mkdir /opt/utimaco/lib
```

3. Copy pkcs11 library file libcs_pkcs11_R3.so from Utimaco CryptoServer software to the /opt/utimaco/lib directory and make the file executable.

›_ Console

```
# cp ~/path_to_application_folder/lib/libcs_pkcs11_R3.so /opt/utimaco/lib
```

4. Copy the csadm and p11tool2 files from Utimaco CryptoServer software to `/opt/utimaco/bin` directory and make both the files executable.

›_ Console

```
# cd ~/path_to_application_folder
# cp csadm p11tool2 /opt/utimaco/bin
# chmod +x /opt/utimaco/bin/csadm /opt/utimaco/bin/p11tool2
```

5. Create the directory `/etc/utimaco` . Locate the Utimaco PKCS#11 configuration file in your SecurityServer directory, `Linux/x86-64/Crypto_APIS/PKCS11_R3/sample` . Copy

the Utimaco PKCS#11 configuration file `cs_pkcs11_R3.cfg into /etc/utimaco` directory

›_ Console

```
# mkdir /etc/utimaco
# cd <install directory>/Software/Linux/x86-
64/Crypto_APIs/PKCS11_R3/sample
# cp cs_pkcs11_R3.cfg /etc/utimaco # cd /etc/utimaco
```

4.2 On Windows

On windows, as part of CryptoServer software installation, `cs_pkcs11_R3.cfg` will get automatically created and will be available under `C:\ProgramData\Utimaco\PKCS11_R3` folder.

4.3 Update cs_pkcs11_R3.cfg

Edit the `cs_pkcs11_R3.cfg` file and make the appropriate changes to the file.

Example Values:

example.file

```
[Global]

# For unix:
Logpath = /tmp

# For windows:
# Logpath = C:/ProgramData/Utimaco/PKCS11_R3
# Loglevel (0 = NONE; 1 = ERROR; 2 = WARNING; 3 = INFO; 4 = TRACE)
Logging = 1
# Prevents expiring session after inactivity of 15 minutes
KeepAlive = true

# Set the Device to connect with
#[CryptoServer]
# Device specifier
Device = <HSM_IP>
```

For more information regarding the commands and command parameters please check the Utimaco CryptoServer documentation. The device may be a CryptoServer (PCIe or LAN) device. The device line will follow one of these patterns, based on the HSM form-factor: ***Device = 288@<HSM IP address>*** *Hardware (LAN) HSM*

OR

Device = /dev/cs2.0 *Hardware (PCIe) HSM*

To make your testing easier, it would be good to enable the PKCS#11 log file. That can be enabled by editing the ***Logging*** *Loglevel. Set the* ***LogPath*** *and Logging* ***Loglevel*** *to* ***1****. For testing you may want to increase it to* ***4****.*

The added ***LogPath*** *points to a writable directory, not to a file.*

If you encounter problems, check the log file named ***cs_pkcs11_R3.log*** *in the* ***LogPath*** *defined directory. When you are done testing, you should change Logging to* ***1*** *or* ***2****. This will limit the logging to only critical and important messages.*

5 Integrating Oracle Database for Windows/Linux

5.1 Copying Utimaco PKCS#11 Library File

Depending on whether your host server as 32-bit or 64-bit architecture, make sure the following directory already exists or else create it.

Linux:

Parameter	***Definition***
[32 bit]	$ORACLE_BASE/extapi/32/hsm[/hsm-manufacturer/library-version/]
[64 bit]	$ORACLE_BASE/extapi/64/hsm[/hsm-manufacturer/library-version/]

Table 5: List of parameters and definitions

Make ownership and permissions on the above directory as:

owner=oracle; group=oinstall; permissions=775

Windows:

Parameter	***Definition***
[32 bit]	C:\oracle\extapi\32\hsm[\hsm-manufacturera\library-version\]
[64 bit]	C:\oracle\extapi\64\hsm[\hsm-manufacturera\library-version\]

Table 6: List of parameters and definitions

Make sure the 'oracle' user can access the above Windows folder Valid directory examples on 64-bit are:

Parameter	***Definition***
Linux	/opt/oracle/extapi/64/hsm/utimaco/4.45.3/

Parameter	***Definition***
Windows	C:\oracle\extapi\64\hsm\utimaco\4.45.3\

Table 7: List of parameters and definitions

Please copy the Utimaco PKCS#11 (`cs_pkcs11_R3.dll` or `cs_pkcs11_R3.so`) library for your host architecture to the folder mentioned as above to allow the Oracle database to access the cryptographic library.

Oracle Database should now be able to access The Utimaco PKCS#11 HSM provider.

5.2 Configuring Oracle DB to use Utimaco HSM

To start using HSM based encryption, you need a Master Encryption Key (MEK) that will be stored inside the HSM. The MEK is used to encrypt or decrypt the Oracle database table columns or tablespace.

HSM separates ordinary program functions from encryption operations, making it possible to divide duties between database administrators and security administrators. Security is enhanced because the wallet password can be unknown to the database administrator, requiring the security administrator to provide the password.

The key is secure from unauthorized access attempts as the HSM is a physical device and not an operating system file. All encryption and decryption operations that use the master encryption key are performed inside the HSM. This means that the master encryption key is never exposed in insecure memory.

While performing this integration we have used Linux path in the SQL commands, change the path according to the appropriate operating system.

5.3 Generate Master Encryption Key (MEK) on to the HSM

1. Create a wallet directory located in the $ORACLE_BASE/admin/db_unique_name directory e.g., wallet.

2. Log in to the database instance as a user who has been granted the SYSDBA administrative privilege.

›_ sqlplus console

```
SQL> connect / as sysdba
```

3. Set WALLET_ROOT parameter.

›_ sqlplus console

```
SQL> alter system set wallet_root='<path to the oracle wallet directory>'
scope=spfile;
```

4. Shutdown and startup database.

›_ sqlplus console

```
SQL> shutdown immediate;
SQL> startup;
```

5. Set TDE_CONFIGURATION parameter.

›_ sqlplus console

```
SQL> alter system set TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=HSM" SCOPE=both;
```

6. Grant the ADMINISTER KEY MANAGEMENT or SYSKM privilege to SYSTEM and any user that you want to use.

›_ sqlplus console

```
SQL> grant ADMINISTER KEY MANAGEMENT to system;
SQL> commit;
```

7. Connect to the database as system user.

›_ sqlplus console

```
SQL> connect system/<password>
```

8. Run the ADMINISTER KEY MANAGEMENT SQL statement to open the HSM based keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

9. Set the MEK in HSM keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY <hsm_password>;
```

10. You can verify the key gets generated onto the HSM using following command.

›_ console

```
p11tool2 LoginUser=<hsm_password> ListObjects
```

Example:

›_ console

```
p11tool2 LoginUser=<hsm_password> ListObjects
CKO_DATA:
+ 1.1
  CKA_LABEL                       =
ORACLE.SECURITY.KM.ENCRYPTION.3036413334414244463744443446344634374246
6393441303231303345424535433 1

+ 1.2
  CKA_LABEL                       = DATA_OBJECT_SUPPORTED_IDEN
```

5.4 Verify Encrypting Column in Table

1. Create a SCIENTISTS table into DB.

›_ sqlplus console

```
SQL> create table SCIENTISTS (SCID NUMBER(4), FirstName VARCHAR2(128),  LastName
VARCHAR2(128), Salary NUMBER(6));
```

2. Enter data into the SCIENTISTS table.

›_ sqlplus console

```
SQL> insert into SCIENTISTS values (0001, 'Albert', 'Einstein', 850000);
SQL> insert into SCIENTISTS values (0002, 'Isaac', 'Newton', 750000);
SQL> insert into SCIENTISTS values (0003, 'Charles', 'Darwin', 650000);
SQL> insert INTO SCIENTISTS values (0004, 'Curie', 'Einstein', 550000);
SQL> commit;
```

3. Verify inserted data into SCIENTISTS table.

›_ sqlplus console

```
SQL> select * from SCIENTISTS;
```

4. Encrypt the 'Salary' column from SCIENTISTS.

›_ sqlplus console

```
SQL> alter table SCIENTISTS modify (Salary ENCRYPT);
```

5. The Transparent Data Encryption decrypts the encrypted column automatically and returns the data in clear format.

›_ sqlplus console

```
SQL> select salary from SCIENTISTS;
```

6. Verify the column is encrypted in your DB.

›_ sqlplus console

```
SQL> select * from DBA_ENCRYPTED_COLUMNS;
```

7. View the information of HSM keystore.

›_ sqlplus console

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

5.5 Create an Encrypted Tablespace

1. To create an encrypted tablespace, you must use the `CREATE TABLESPACE` statement with the `ENCRYPTION USING` clause.

›_ sqlplus console

```
SQL> create tablespace SECURE_TS DATAFILE '/u01/app/oracle/admin/utimacodb/
SECURETS_01.DBF' SIZE 10M ENCRYPTION USING 'AES256' ENCRYPT;
```

2. Create EMP table inside the SECURE_TS tablespace.

›_ sqlplus console

```
SQL> create table EMP (EMPID NUMBER(4),NAME VARCHAR(100),SALARY NUMBER(6))
tablespace SECURE_TS;
```

3. Insert data into EMP table.

›_ sqlplus console

```
SQL> insert into EMP values (0001, 'Michael Jackson', 999999);
SQL> insert into EMP values (0002, 'Lady Gaga', 888888);
SQL> insert into EMP values (0003, 'Freddie Mercury', 777777);
SQL> insert into EMP values (0004, 'Steven Tyler', 666666); SQL> commit;
```

4. View the data from EMP table.

›_ sqlplus console

```
SQL> select * from EMP;
```

5. Close the keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY <hsm_password>;
```

6. Now try to view the contents of EMP table.

›_ sqlplus console

```
SQL> select * from EMP;
```

As the keystore is closed, you will get an error message "ORA-28365: wallet is not open" and hence you cannot view the data from EMP table.

7. Open the Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

8. Now view the data from EMP table.

›_ sqlplus console

```
SQL> select * from EMP;
```

9. List all the Key IDs.

›_ sqlplus console

```
SQL> select KEY_ID from V$ENCRYPTION_KEYS;
```

5.6 Encrypting an Existing Tablespace with Online Conversion

To encrypt an existing tablespace with online conversion, use `ALTER TABLESPACE` with the `ONLINE` and `ENCRYPT` clauses.

1. Login to DB instance as a system user.

›_ sqlplus console

```
SQL> connect system/<password>;
```

2. Open HSM Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

3. Create a tablespace.

›_ sqlplus console

```
SQL> create tablespace NONSECURE_TS DATAFILE '/u01/app/oracle/oradata/utimacodb/
NONSECURETS_01.DBF' SIZE 10M;
```

4. Create EMP table inside the NONSECURE_TS tablespace.

›_ sqlplus console

```
SQL> create table EMP (EMPID NUMBER(4),NAME VARCHAR(100),SALARY NUMBER(6))
tablespace NONSECURE_TS;
```

5. Insert data into EMP table.

›_ sqlplus console

```
SQL> insert into EMP values (0001, 'Michael Jackson', 999999); SQL> insert into
EMP values (0002, 'Lady Gaga', 888888);

SQL> insert into EMP values (0003, 'Freddie Mercury', 777777);

SQL> insert into EMP values (0004, 'Steven Tyler', 666666); SQL> commit;
```

6. View the data from EMP table.

›_ sqlplus console

```
SQL> select * from EMP;
```

7. Check the COMPATIBLE parameter is set correctly according to DB version.

›_ sqlplus console

```
SQL> show PARAMETER COMPATIBLE;
Example

SQL> show PARAMETER COMPATIBLE;

NAME TYPE VALUE
------------ ----------- ------
compatible string 19.0.0
```

You must change the `COMPATIBLE` parameter if value is not 19.0.0, then complete the remaining steps in this procedure. To change the `COMPATIBLE` parameter, edit the initialization parameter file to use the new `COMPATIBLE` setting.

8. Encrypt the `NONSECURE_TS` tablespace.

›_ sqlplus console

```
SQL> alter tablespace NONSECURE_TS ENCRYPTION ONLINE using 'AES192' ENCRYPT
FILE_NAME_CONVERT = ('NONSECURETS_01.DBF','SECURETS_02.DBF');
```

9. View the data from EMP table.

›_ sqlplus console

```
SQL> select * from EMP;
```

10. Verify the `NONSECURE_TS` tablespace got encrypted.

›_ sqlplus console

```
SQL> Select TABLESPACE_NAME, ENCRYPTED FROM DBA_TABLESPACES;
```

11. Close the keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY <hsm_password>;
```

Now try to view the contents of EMP table, nothing is displayed as the wallet is closed.

As the keystore is closed, you will get an error message "ORA-28365: wallet is not open" and hence you cannot view the data from EMP table.

12. Open the keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

13. Verify EMP table.

›_ sqlplus console

```
SQL> select * from EMP;
```

5.7 Decrypting an Existing Tablespace with Online Conversion

To decrypt an existing tablespace with online conversion, you can use the `ALTER TABLESPACE SQL` statement with `DECRYPT` clause.

1. Login to DB instance as a system user.

›_ sqlplus console

```
SQL> connect system/<password>;
```

2. List encrypted tablespace.

›_ sqlplus console

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

3. Open a Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

4. Decrypt the existing tablespace `NONSECURE_TS`.

›_ sqlplus console

```
SQL> alter tablespace NONSECURE_TS ENCRYPTION ONLINE DECRYPT  FILE_NAME_CONVERT
= ('SECURETS_02.DBF', 'NONSECURETS_02.DBF');
```

5. Verify that the tablespace no longer encrypted

›_ sqlplus console

```
SQL> select TABLESPACE_NAME, ENCRYPTED from DBA_TABLESPACES WHERE
TABLESPACE_NAME='NONSECURE_TS';
```

5.8 Rekeying an Existing Tablespace with Online Conversion

To rekey an existing tablespace that is online, you can use the `REKEY` clause of the `ALTER TABLESPACE SQL` statement.

1. Login to DB instance as a system user.

›_ sqlplus console

```
SQL> connect system/<password>;
```

2. List the encrypted tablespace.

›_ sqlplus console

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

3. Open a Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

4. Rekey the existing tablespace SECURE_TS.

›_ sqlplus console

```
SQL> alter tablespace SECURE_TS ENCRYPTION USING 'AES192' REKEY
FILE_NAME_CONVERT = ('SECURETS_01.DBF','SECURETS_03.DBF');
```

5. Verify rekey is successful.

›_ sqlplus console

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

5.9 Encrypting an Existing Tablespace with Offline Conversion

You can encrypt a data file of an existing tablespace when the tablespace is offline.

1. Login to DB instance as a system user.

›_ sqlplus console

```
SQL> connect system/<password>;
```

2. Open the HSM Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

3. Create a tablespace.

›_ sqlplus console

```
SQL> create tablespace NONSECURE01_TS DATAFILE '/u01/app/oracle/admin/utimacodb/
NONSECURETS_01.DBF' SIZE 10M;
```

4. Create EMP table inside the `NONSECURE01_TS` tablespace.

›_ sqlplus console

```
SQL> create table EMP (EMPID NUMBER(4),NAME VARCHAR(100),SALARY NUMBER(6))
tablespace NONSECURE01_TS;
```

5. Insert data into EMP table.

›_ sqlplus console

```
SQL> insert into EMP values (0001, 'Michael Jackson', 999999); SQL> insert into
EMP values (0002, 'Lady Gaga', 888888);

SQL> insert into EMP values (0003, 'Freddie Mercury', 777777);

SQL> insert into EMP values (0004, 'Steven Tyler', 666666); SQL> commit;
```

6. View the data from EMP table.

›_ sqlplus console

```
SQL> select * from EMP;
```

7. Bring the NONSECURE01_TS tablespace offline.

›_ sqlplus console

```
SQL> alter tablespace NONSECURE01_TS OFFLINE NORMAL;
```

8. Encrypt the NONSECURE01_TS tablespace.

›_ sqlplus console

```
SQL> alter tablespace NONSECURE01_TS ENCRYPTION OFFLINE ENCRYPT;
```

Alternatively, to encrypt individual data files within a tablespace, use the ALTER DATABASE DATAFILE SQL statement.

›_ sqlplus console

```
SQL> alter database DATAFILE ' NONSECURETS01_TS.DBF' ENCRYPT;
```

9. Bring the tablespace online.

›_ sqlplus console

```
SQL> alter tablespace NONSECURE01_TS ONLINE;
```

10. Close the keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY <hsm_password>;
```

11. Now try to view the contents of EMP table, nothing is displayed as the wallet is closed.

›_ sqlplus console

```
SQL> select * from EMP;
```

5.10 Decrypting an Existing Tablespace with Offline Conversion

To decrypt an existing tablespace with online conversion, you can use the `ALTER TABLESPACE SQL` statement with `DECRYPT` clause.

1. Login to DB instance as a system user.

›_ sqlplus console

```
SQL> connect system/<password>;
```

2. List encrypted tablespace.

›_ sqlplus console

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

3. Open a Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

4. Bring the tablespace `NONSECURE01_TS` offline.

›_ sqlplus console

```
SQL> alter tablespace NONSECURE01_TS OFFLINE NORMAL;
```

5. Decrypt the existing tablespace `NONSECURE01_TS`.

›_ sqlplus console

```
SQL> alter tablespace NONSECURE01_TS ENCRYPTION OFFLINE DECRYPT;
```

6. Bring the tablespace `NONSECURE01_TS` online.

›_ sqlplus console

```
SQL> alter tablespace NONSECURE01_TS ONLINE;
```

7. Verify the tablespace got decrypted.

›_ sqlplus console

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

5.11 Configure Auto-login for Hardware Keystore

The auto login feature for wallets does not require human intervention to supply the necessary passwords it can enable PKI-based access to services. Enabling auto login creates an obfuscated copy of the wallet, which is then used automatically until the auto login feature is disabled for that wallet.

By Default, auto login feature is disabled. You must enable auto login if you want single signon access to multiple Oracle databases. When auto login is enabled then .sso file gets created under wallet directory.

1. Close the Hardware Keystore if it is opened.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY <hsm_password>;
```

2. Set the `WALLET_ROOT` parameter.

›_ sqlplus console

```
SQL> connect / as sysdba

SQL> alter system set wallet_root='<path to the oracle wallet directory>'
 scope=spfile
```

3. Shut down and start up database.

›_ sqlplus console

```
SQL> shutdown immediate;

SQL> startup;
```

4. Set the `TDE_CONFIGURATION` parameter.

›_ sqlplus console

```
SQL> alter system set TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=FILE"
 SCOPE=both;
```

5. Create the software keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT CREATE KEYSTORE IDENTIFIED BY
<software_keystore_password>;
```

Skip this step if software keystore already exist.

6. Open the Software Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY
<software_keystore_password>;
```

7. Add HSM password as a client to the Software Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT ADD SECRET '<hsm_password>' FOR CLIENT
'HSM_PASSWORD' IDENTIFIED BY <software_keystore_password> WITH BACKUP;
```

8. Close the Software Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY
<software_keystore_password>;
```

9. Create Auto-Login keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT CREATE AUTO_LOGIN KEYSTORE FROM KEYSTORE
IDENTIFIED BY <software_keystore_password>;
```

10. Set `TDE_CONFIGURATION` parameter.

›_ sqlplus console

```
SQL> alter system set TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=HSM|FILE"
 SCOPE=both;
```

11. At this stage, close the database and open it one more time and the next time when a TDE operation executes, the hardware security module auto-login keystore opens automatically.

›_ sqlplus console

```
SQL> shutdown immediate;

SQL> startup;
```

12. Check the status of the wallet.

›_ sqlplus console

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

Now you have a software wallet that contains the HSM password. And the software wallet password is protected by Oracles auto-login feature.

5.12 Migrating Master Encryption Key from Software Keystore to the HSM Keystore

This section describes how software wallet will be migrated to use a hardware-based wallet.

While performing this integration we have used Linux path in the SQL commands, change the path according to the appropriate operating system.

5.12.1 Create Software Keystore

This example starts with creating a database protected by a software wallet. If you already have an existing database protected by software wallet you can skip this section.

1. Create a wallet directory located in the `$ORACLE_BASE/admin/db_unique_name` directory e.g., wallet.

2. Log in to the database instance as a user who has been granted the SYSDBA administrative privilege.

›_ sqlplus console

```
SQL> connect / as sysdba
```

3. Set `WALLET_ROOT` parameter.

›_ sqlplus console

```
SQL> alter system set wallet_root='<path to the oracle wallet directory>'
 scope=spfile
```

4. Shut down and start up database.

›_ sqlplus console

```
SQL> shutdown immediate;

SQL> startup;
```

5. Set `TDE_CONFIGURATION` parameter.

›_ sqlplus console

```
SQL> alter system set TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=FILE"
 SCOPE=both ;
```

6. Grant the `ADMINISTER KEY MANAGEMENT` or `SYSKM` privilege to `SYSTEM` and any user that you want to use.

›_ sqlplus console

```
SQL> grant ADMINISTER KEY MANAGEMENT to system;

SQL> commit;
```

7. Connect to the database as system user.

›_ sqlplus console

```
SQL> connect system/<password>
```

8. Run the `ADMINISTER KEY MANAGEMENT SQL` statement to create the keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT CREATE KEYSTORE IDENTIFIED BY
<software_keystore_password>;
```

9. Run the `ADMINISTER KEY MANAGEMENT SQL` statement to open the software based keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY
<software_keystore_password >;
```

10. Set the master encryption key in the software keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY <software_keystore_password
> WITH BACKUP USING 'backupdb';
```

11. Create a `SCIENTISTS` table into the DB.

›_ sqlplus console

```
SQL> create table SCIENTISTS (SCID NUMBER(4), FirstName VARCHAR2(128),  LastName
VARCHAR2(128), Salary NUMBER(6));
```

12. Enter data into the `SCIENTISTS` table.

›_ sqlplus console

```
SQL> insert into SCIENTISTS values (0001, 'Albert', 'Einstein', 850000);

SQL> insert into SCIENTISTS values (0002, 'Isaac', 'Newton', 750000);

SQL> insert into SCIENTISTS values (0003, 'Charles', 'Darwin', 650000); SQL>
insert INTO SCIENTISTS values (0004, 'Curie', 'Einstein', 550000); SQL> commit;
```

13. Verify inserted data into `SCIENTISTS` table.

›_ sqlplus console

```
SQL> select * from SCIENTISTS;
```

14. Encrypt the 'Salary' column from `SCIENTISTS`.

›_ sqlplus console

```
SQL> alter table SCIENTISTS modify (Salary ENCRYPT);
```

15. The Transparent Data Encryption decrypts the encrypted column automatically and returns the data in clear format.

›_ sqlplus console

```
SQL> select salary from SCIENTISTS;
```

16. Verify the column is encrypted in your DB.

›_ sqlplus console

```
SQL> select * from DBA_ENCRYPTED_COLUMNS;
```

17. View the information of software keystore.

›_ sqlplus console

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

18. Create an encrypted tablespace.

›_ sqlplus console

```
SQL> create tablespace SECURETS DATAFILE '/u01/app/oracle/oradata/utimacodb/
SECURETDB_01.DBF' SIZE 10M ENCRYPTION USING 'AES256' ENCRYPT;
```

19. Create EMP table inside the `SECURETS` tablespace.

›_ sqlplus console

```
SQL> create table EMP (EMPID NUMBER(4),NAME VARCHAR(100),SALARY NUMBER(6))
tablespace SECURETS;
```

20. Insert data into the `EMP` table.

›_ sqlplus console

```
SQL> insert into EMP values (0001, 'Michael Jackson', 999999); SQL> insert into
EMP values (0002, 'Lady Gaga', 888888);

SQL> insert into EMP values (0003, 'Freddie Mercury', 777777);

SQL> insert into EMP values (0004, 'Steven Tyler', 666666); SQL> commit;
```

21. View the data from `EMP` table.

›_ sqlplus console

```
SQL> select * from EMP;
```

22. Close the software keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY
<software_keystore_password>;
```

23. Now try to view the contents of `EMP` table.

›_ sqlplus console

```
SQL> select * from EMP;
```

As the keystore is closed, you will get an error message "ORA-28365: wallet is not open" and hence you cannot view the data from EMP table

24. Open the Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <
software_keystore_password >;
```

25. Now view the data from EMP table.

›_ sqlplus console

```
SQL> select * from EMP;
```

26. List all the Key IDs.

›_ sqlplus console

```
SQL> select KEY_ID from V$ENCRYPTION_KEYS;
```

5.12.2 Migrate the Software Keystore to the Utimaco HSM

1. Log in to the database instance as a user who has been granted the SYSDBA administrative privilege.

›_ sqlplus console

```
SQL> connect / as sysdba
```

2. Set `WALLET_ROOT` parameter.

›_ sqlplus console

```
SQL> alter system set wallet_root='<path to the oracle wallet directory>'
 scope=spfile
```

3. Shut down and start up the database.

›_ sqlplus console

```
SQL> shutdown immediate;

SQL> startup;
```

4. Set `TDE_CONFIGURATION` parameter.

›_ sqlplus console

```
SQL> alter system set TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=HSM|FILE"
 SCOPE=both ;
```

5. Connect to the database as system user.

›_ sqlplus console

```
SQL> connect system/<password>
```

6. Now Migrate the wallet to HSM using the below command.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY

<hsm_password> MIGRATE USING <software_keystore_password> WITH BACKUP USING
'backupdb';
```

7. Now verify the wallet is moved to HSM wallet now using command below.

›_ sqlplus console

```
select * from v$encryption_wallet;
```

8. The Transparent Data Encryption decrypts the encrypted column automatically and returns the data in clear format.

›_ sqlplus console

```
SQL> select salary from SCIENTISTS;

SQL> select salary from EMP;
```

9. Verify the column is encrypted in your DB.

›_ sqlplus console

```
SQL> select * from DBA_ENCRYPTED_COLUMNS;
```

10. View the information of keystore.

›_ sqlplus console

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

11. Change the password of software keystore to be the same as HSM password.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT ALTER KEYSTORE PASSWORD IDENTIFIED BY
<software_keystore_password> SET <hsm_password> WITH BACKUP USING 'backupdb';
```

12. Close the keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY
<software_keystore_password>;
```

13. Open the keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY
<software_keystore_password>;
```

14. View the information of keystore.

›_ sqlplus console

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

5.13 Configure TDE with Pluggable DB using HSM

5.13.1 About Containers in a CDB

A container is a collection of schemas, objects, and related structures in a multitenant container database (CDB). Within a CDB, each container has a unique ID and name.

A CDB includes zero, one, or many customer-created pluggable databases (PDBs) and application containers. A PDB is a portable collection of schemas, schema objects, and nonschema objects that appears to an Oracle Net client as a separate database. An application container is an optional, user created CDB component that stores data and metadata for one or more application back ends. A CDB includes zero or more application containers.

5.13.2 About PDBs

A PDB is a user created set of schemas, objects, and related structures that appears logically to a client application as a separate database.

Every PDB is owned by SYS, regardless of which user created the PDB. SYS is a common user in the CDB, which means that this user that has the same identity in the root and in every existing and future PDB within the CDB. PDBs can be plugged into to CDBs. A CDB can contain multiple PDBs. Each PDB appears on the network as a separate database.

5.13.3 Configure TDE with Pluggable DB

There are several ways to create a PDB, but we recommend using the DBCA utility. It is assumed that you have already created PDBs.

While performing this integration we have used Linux path in the SQL commands, change the path according to the appropriate operating system.

For the purpose of this guide, we are using the PDB "utimacopdb".

1. Edit the tnsnames.ora file to add a new service for the PDB. By default, the `tnsnames.ora` file is in the `$ORACLE_HOME/network/admin` directory or in the location set by the `TNS_ADMIN` environment variable. Ensure that you have properly set the `TNS_ADMIN` environment variable to point to the correct `tnsnames.ora` file.

›_ console

```
UTIMACOPDB =

(DESCRIPTION =

 (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521))

  (CONNECT_DATA =

   (SERVER = DEDICATED)

   (SERVICE_NAME = utimacopdb.localdomain)

)

)
```

2. Restart the Listener Service.

›_ console

```
#lsnrctl stop

#lsnrctl start

#lsnrctl status
```

3. Log in to the database instance as a user who has been granted the SYSDBA administrative privilege.

›_ sqlplus console

```
SQL> connect system/<password>
```

4. Set the `WALLET_ROOT` parameter.

›_ sqlplus console

```
SQL> alter system set wallet_root='<path to the oracle wallet directory>'
 scope=spfile;
```

5. Shutdown and startup database.

›_ sqlplus console

```
SQL> shutdown immediate;

SQL> startup;
```

6. Set the `TDE_CONFIGURATION` parameter.

›_ sqlplus console

```
SQL> alter system set TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=HSM"
 SCOPE=both ;
```

7. Open the hardware keystore in the `CDB$ROOT` container.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

8. Set the master encryption key in the `CDB$ROOT` container onto HSM. Skip this step if the master encryption key is already generated onto HSM.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEY FORCE KEYSTORE IDENTIFIED BY
<hsm_password>;
```

9. Connect as sysdba.

›_ sqlplus console

```
SQL> connect / as sysdba
```

10. Open the pdb in read write mode.

›_ sqlplus console

```
SQL> alter pluggable database <PDB_NAME> open read write;
```

11. Set the container to the pdb.

›_ sqlplus console

```
SQL> alter session set container = <pdb_name>;
```

12. Grant the following privileges to the PDB Admin.

›_ sqlplus console

```
SQL> grant administer key management to <pdb_admin>;

SQL> grant create session to <pdb_admin>;

SQL> grant connect to <pdb_admin>;

SQL> grant dba to <pdb_admin>;

SQL> grant create any table to <pdb_admin>;

SQL> grant unlimited tablespace to <pdb_admin>;

SQL> alter user <pdb_admin> profile default;  SQL> commit;
```

13. Connect to the PDB using the PDB username.

›_ sqlplus console

```
SQL> Connect <pdb_admin>/<system_password>@<Pluggable Database Name>
```

14. Run the `ADMINISTER KEY MANAGEMENT` SQL statement to open PDB database.

›_ sqlplus console

```
SQL> administer key management set keystore open identified by "<hsm_password>";
```

15. Create the PDB Master Key onto the HSM.

›_ sqlplus console

```
SQL> administer key management set key identified by "<hsm_password>";
```

5.13.4 To verify the Master Encryption Key is Encrypting the PDB

1. Create a `SCIENTISTS` table in the PDB.

›_ sqlplus console

```
SQL> create table SCIENTISTS (SCID NUMBER(5), Name VARCHAR(42), SALARY
NUMBER(10));
```

2. Enter some values in the `SCIENTISTS` table.

›_ sqlplus console

```
SQL> insert into SCIENTISTS values (001, 'George Bailey', 10000);  SQL> insert
into SCIENTISTS values (002, 'Denial Vettory', 20000); SQL> commit;
```

3. Encrypt the Salary column of the `SCIENTISTS` table.

›_ sqlplus console

```
SQL> alter table SCIENTISTS modify (Salary Encrypt);
```

4. List the values in the encrypted column. Transparent Data Encryption decrypts them automatically and the values are returned in clear text.

›_ sqlplus console

```
SQL> select salary from SCIENTISTS;
```

5. List encrypted columns in your databases.

›_ sqlplus console

```
SQL> select * from DBA_ENCRYPTED_COLUMNS;
```

6. Create an encrypted tablespace.

›_ sqlplus console

```
SQL> create tablespace SECURETS DATAFILE '/u01/app/oracle/oradata/utimacodb/
SECURETS01.DBF' SIZE 10M ENCRYPTION DEFAULT STORAGE (ENCRYPT);
```

7. Create EMP table inside the `NONSECURE_TS` tablespace.

›_ sqlplus console

```
SQL> create table EMP (EMPID NUMBER(4),NAME VARCHAR(100),SALARY NUMBER(6))
tablespace SECURETS;
```

8. Insert data into `EMP` table.

›_ sqlplus console

```
SQL> insert into EMP values (0001, 'Michael Jackson', 999999); SQL> insert into
EMP values (0002, 'Lady Gaga', 888888);

SQL> insert into EMP values (0003, 'Freddie Mercury', 777777);

SQL> insert into EMP values (0004, 'Steven Tyler', 666666); SQL> commit;
```

9. View the data from `EMP` table.

›_ sqlplus console

```
SQL> select * from EMP;
```

10. Close the keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY <hsm_password>;
```

11. Now try to view the contents of `EMP` table.

›_ sqlplus console

```
SQL> select * from EMP;
```

As the keystore is closed, you will get an error message "ORA-28365: wallet is not open" and hence you cannot view the data from EMP table

12. Open the Keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

13. Now view the data from `EMP` table.

›_ sqlplus console

```
SQL> select * from EMP;
```

14. List all the Key IDs.

›_ sqlplus console

```
SQL> select KEY_ID from V$ENCRYPTION_KEYS;
```

6 Integrating Oracle RAC with Utimaco HSM

6.1 Copying Utimaco PKCS#11 Library File to all Oracle RAC Instances

1. Create the following directory `/opt/oracle//extapi/64/hsm/<hsm-manufacturername>/<library-version>` / if it does not exist on all the oracle RAC instances.

›_ console

```
# mkdir -p /opt/oracle/extapi/64/hsm/utimaco/4.45.5
```

2. Copy the Utimaco PKCS#11 library to above directory.

›_ console

```
# cp cs_pkcs11_R3.so /opt/oracle/extapi/64/hsm/utimaco/4.45.5/
```

3. Make ownership and permissions on the /opt/oracle directory as: `owner=oracle; group=oinstall; permissions=775`.

›_ console

```
# chown -R oracle:oinstall /opt/oracle

# chmod -R 755 /opt/oracle
```

6.2 Generate Master Encryption Key (MEK) on the HSM

1. Create a wallet directory located in `$ORACLE_BASE/admin/db_unique_name` directory e.g., wallet.

2. Log in to the database instance as a user who has been granted the SYSDBA administrative privilege.

›_ sqlplus console

```
SQL> connect / as sysdba
```

3. Set the `WALLET_ROOT` parameter.

›_ sqlplus console

```
SQL> alter system set wallet_root='<path to the oracle wallet directory>'
 scope=spfile;
```

4. Shutdown and startup the database.

›_ sqlplus console

```
SQL> shutdown immediate;

SQL> startup;
```

5. Set the `TDE_CONFIGURATION` parameter.

›_ sqlplus console

```
SQL> alter system set TDE_CONFIGURATION="KEYSTORE_CONFIGURATION=HSM" SCOPE=both;
```

6. Verify the `WALLET_ROOT` and the `TDE_CONFIGURATION` parameter are set.

›_ sqlplus console

```
SQL> show parameter WALLET_ROOT;

SQL> show parameter TDE_CONFIGURATION;
```

7. Grant the `ADMINISTER KEY MANAGEMENT` or `SYSKM` privilege to `SYSTEM` and any user that you want to use.

›_ sqlplus console

```
SQL> grant ADMINISTER KEY MANAGEMENT to system;

SQL> commit;
```

8. Connect to the database as system user.

›_ sqlplus console

```
SQL> connect system/<password>
```

9. Run the `ADMINISTER KEY MANAGEMENT` SQL statement to open the HSM based keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <hsm_password>;
```

10. Set the MEK in HSM keystore.

›_ sqlplus console

```
SQL> ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY <hsm_password>;
```

11. You can verify the key gets generated onto the HSM using following command.

›_ console

```
p11tool2 LoginUser=<hsm_password> ListObjects
```

```
[root@ol8-19-rac2 bin]# p11tool2 LoginUser=123456 listobjects


CKO_DATA:

+ 1.1
  CKA_LABEL                     = DATA_OBJECT_SUPPORTED_IDEN

+ 1.2
  CKA_LABEL                     =
ORACLE.SECURITY.KM.ENCRYPTION.303633323642384339433332344434464244424639324539363244 3
9303738343233


CKO_SECRET_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_AES
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_FALSE
  CKA_LABEL                     =
ORACLE.TDE.HSM.MK.06326B8C9C324D4FBDBF92E962D9078423
  CKA_ID                        =

[root@ol8-19-rac2 bin]#
```

Figure 1 : p11tool2 listobjects output

Most of the use cases for various types of tables and tablespace encryption are already covered in previous chapters. Use the following link to perform them on Oracle RAC instances. In case of Oracle RAC make sure to use shared location for wallet, software keystore and tablespace files which are accessible by all RAC instances.

7 FIPS Requirements

All the steps are identical for the HSM in FIPS 140-2 approved mode. The only difference is that the backup of the entire key database using csadm or CAT is not possible. In this case the P11CAT or the p11tool2 are used for backing up the keys.

8 Troubleshooting

Error	***Diagnosis***
ORA-12154: TNS:could not resolve the connect identifier specified	open tnsnames.ora file and append following line LISTENER_<PDBNAME> = (ADDRESS = (PROTOCOL = TCP)(HOST = FQDN)(PORT = 1521)) <PDBNAME> = (DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = FQDN)(PORT = 1521)) (CONNECT_DATA = (SERVER = DEDICATED) (SERVICE_NAME = <PDBNAME>)))
LoginUser= failed: 05.12.2021 23:45:45 src/p11adm_R2.c[429] p11_login: C_Login [type=1] returned Error 0x00000102 (CKR_USER_PIN_NOT_INITIALIZED)	PKCS#11 Slot is not initialized.

Error	***Diagnosis***
The CryptoServer PKCS#11 Library R3 is not initialized. Error CKR_CRYPTOKI_NOT_INITIALIZED occurred	PKCS#11 Slot is not initialized.
ORA-46661: keystore not open in root container	Check the sqlnet.ora file for the configuration, first open wallet in root container.
The listener supports no services	Check the tnsnames.ora file and listener.ora file for configuration
TNS-12541: TNS:no listener TNS-12560: TNS:protocol adapter error TNS-00511: No listener 64-bit Windows Error: 61: Unknown error	Start the Listener service by using lsnrctl start command
ORA-28365: wallet is not open	ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <password>
ORA-01031: insufficient privileges	Either grant the permission to user or run the command using sysdba user

Table 8: List of errors and its diagnosis

9 Further Information

This document forms a part of the information and support which is provided by the Utimaco IS GmbH. Additional documentation can be found on the product CD in the Documentation directory.

All CryptoServer product documentation is also available at the Utimaco IS GmbH website:

https://utimaco.com/

10 References

Reference	*Title/Company*	*Document No.*
[CSADMIN]	CryptoServer – csadm Manual/Utimaco IS GmbH	2009-0003
[CSTrSh]	CryptoServer Troubleshooting/Utimaco IS GmbH	M011-0008-en
[CSADMIN2]	CryptoServer_csadm_Manual_Systemadministrators.pdf	2009-0003
[CSP11Tool2]	CryptoServer_p11tool2_Manual.pdf	2012-0004
[CSPKCSM]	CryptoServer - PKCS#11 P11CAT Manual	M013-0001-en
[CSLAN5]	CryptoServerLAN_Manual_Systemadministrators.pdf	2018-0004

11 Contact and Support Information

You can reach us from Monday to Friday, 09.00 a.m. to 05.00 p.m., Central European Time (CET).

Utimaco IS GmbH
Krefelder Str. 220
52070 Aachen
Germany

RMA Query

If you need to send the device back to Utimaco IS GmbH, please open a new RMA case (Return Merchandise Authorization). We request that you use the following web address. RMA cases cannot be opened by email or phone.

https://support.hsm.utimaco.com/support/rma/new

Other Support Queries

- Mail (preferred contact method)
  support@utimaco.com
  Attach the diagnostic information to your email.
- Web portal
  https://support.hsm.utimaco.com/support/cases/new/
  The diagnostic information will be requested in our response if necessary.
- By phone
  AMERICAS +1-844-UTIMACO (+1 844-884-6226)
  EMEA +49 800-627-3081
  APAC +81 800-919-1301
  The diagnostic information will be requested in our response if necessary.