Oracle Transparent Data Encryption

Web Services Option Pack (WSOP) Integration Guide

2024-10-21

Table of Contents

Chapter 1. Introduction

This guide describes how to integrate and use Entrust Web Services Option Pack (WSOP) with an Oracle database. The Oracle feature Transparent Data Encryption (TDE) provides data-at-rest encryption for sensitive information held by the Oracle database, while at the same time allowing authorized clients to use the database.

Oracle database software, and Entrust Web Services Option Pack (WSOP), can be independently installed on separate host servers. They can then be configured to interoperate through a single library interface. It is possible to support multiple database instances on the same WSOP server, while each database instance is restricted to access only its own encryption keys. Oracle cluster technology is also supported.

Integrated Oracle and Entrust technology has been tested to support Oracle TDE for tablespace encryption, or column encryption, or concurrently for both. Entrust nShield HSMs are certified to FIPS 140 (level 3) to deliver a high grade of security assurance and were used by the WSOP configuration during testing. Functionality includes protection of sensitive encryption keys and support for offload of encryption and key management operations.

1.1. Using this guide

This *Integration Guide* covers UNIX/Linux based systems. Any time the term HSM protection is used in this guide, it refers to the HSM being used in the configuration of the WSOP system used in the integration. It provides:

- An overview of how the Oracle database software and Entrust WSOP software work together to enhance security.
- Configuration and installation instructions.
- Depending on your current Oracle setup, how to:
  - Migrate encryption from an existing Oracle wallet or keystore to HSM protection.
  - Begin using HSM protection immediately if no Oracle software wallet or keystore already exists.
- Examples and advice on how the product may be used.
- Troubleshooting advice.

It is assumed the reader has a good knowledge of Oracle database technology.

Assuming you already have your Oracle database installed, after installing and configuring the Entrust WSOP software with the Security World software and HSM, there is no other software required. However, some minor configuration changes will be needed.

This guide cannot anticipate all configuration requirements a customer may have. Examples shown in this guide are not exhaustive, and may not necessarily show the simplest or most efficient methods of achieving the required results. The examples should be used to guide integration of the Entrust WSOP with an Oracle database, and should be adapted to your own circumstances.

Entrust accepts no responsibility for loss of data, or services, incurred by use of examples, or any errors in this guide. For your own reassurance, it is recommended you thoroughly check your own solutions in safe test conditions before committing them to a production environment. If you require additional help in setting up your system, contact Entrust Support.

Entrust accepts no responsibility for information in this guide that is made obsolete by changes or upgrades to the Oracle product.

This guide assumes that you have read the WSOP, Security World and HSM documentation, and are familiar with the documentation and setup processes for Oracle database TDE.

1.2. Product configuration

Entrust has successfully tested WSOP integration with the in the following configurations:

1.2.1. Oracle Server

OS Version	Kernel	Oracle Version	WSOP Client Version
Red Hat Enterprise Linux release 9.4 (Plow)	Linux 5.14.0-427.16.1.el9_4.x86_64	Oracle Database 23ai 23.4.0.24.05	3.3 - wsop-p11-3.3.0-714-5306bc4 Client

1.2.2. WSOP Server

OS Version	Kernel	Security World	WSOP Version
Red Hat Enterprise Linux release 8.10 (Ootpa)	Linux 4.18.0-513.24.1.el8_9.x86_64	13.4.5	3.3 wsop-p11-3.3.0-714-5306bc4

1.2.3. Supported nShield hardware and software versions

Entrust has successfully tested with the following nShield hardware and software versions:

WSOP currently only supports softcard protection, so OCS and module protection are not supported.

1.2.3.1. Oracle 23ai 23.4.0.24.05

HSM	Security World Software	Firmware	Image	Softcard	FIPS Level 3
Connect XC	13.4.5	12.50.11 (FIPS 140-2 certified)	12.80.4	✓	
Connect XC	13.4.5	12.72.1 (FIPS 140-2 certified)	13.4.5	✓	✓
nShield 5C	13.4.5	13.2.2	13.4.5	✓	✓
nShield 5C	13.2.2	13.2.2	13.4.5	✓	✓

1.3. Conventions used in this document

1.3.1. Database connections

You must be a user with correct permissions to access a database, and also have the correct privileges to perform the required operations when connected to that database. Your system administrator should be able to create users and grant suitable permissions and privileges according to your organization's security policies. Example 2

- <database-user> is the user identity making the connection.
- <database-identifier> is the database to make the connection to.

For the purpose of examples in this guide, the following database users and database identifiers should be sufficient.

This guide uses the FREE database container that comes installed with Oracle 23ai.

- <database-user>. This guide will use one following users for connecting to databases:
  - sysdba, Oracle's standard sysdba user.
  - system, Oracle's standard system user.
    - C##TESTER, as a common user for container (FREE) and the PDBs it contains.
    - FREEPDB<k>TESTER, as a local user for a PDB<k> within container FREE.

      Where <k> is a distinguishing digit.
- <database-identifier>. This guide will use one following database identifies during a connection:
  - FREE, to connect to the CDBROOT for the FREE container database.
  - FREEPDB<k>, to connect to PDB<k> within FREE database.

  For example:

```
CONNECT sysdba@FREE
CONNECT C##TESTER@FREE
CONNECT C##TESTER@FREEPDB1
```

The connection implies that you must alter a session if you are not already connected to the required container. For example:

- Example 1:

  CONNECT C##TESTER@FREE

  This implies that, if you are not already connected to FREE, then alter the session:

  ALTER SESSION SET CONTAINER = FREE$ROOT;

- Example 2:

CONNECT FREEPDB<k>TESTER@FREEPDB<k>

This implies that, if you are not already connected to FREEPDB<k>, then alter the session:

ALTER SESSION SET CONTAINER = FREEPDB<k>;

Examples of sqlplus connection syntax for different users:

- sqlplus / as sysdba
- sqlplus / as sysdba@CDB1ROOT
- sqlplus FREEPDB1TESTER/Tester@//localhost:1521/FREEPDB1.interop.com

1.3.2. Key migration and legacy keys

Encryption master keys may be migrated from an existing Oracle keystore to Entrust WSOP, or vice versa. In this case, and as used in this document, the term 'key migration' means that the responsibility for holding the master keys is being migrated. The encryption keys themselves are not copied (or imported) between a software keystore and WSOP. Fresh master key(s) are created within the software keystore or WSOP that is to become the new key protector as a result of the migration. Subsidiary keys that are being protected are re-encrypted using the fresh master key(s). Thereafter, any new master keys are created in the current key protector you have migrated to.

During rekey, the previous master keys, or legacy keys, remain in the software keystore or WSOP where they were created. After you have performed a key migration, you can retain access to the legacy keys in the software keystore or WSOP you have migrated away from by making its passphrase the same as the current key protector's. This allows both to be open at the same time allowing access to encryption keys they both contain. If you do not do this, you will only be able to access keys in the current key protector. If you are using both a software keystore and WSOP at the same time, whichever is the current key protector is called the primary.

Oracle uses the HSM term when the key is not protected by the software keystore. This integration uses WSOP to play the role of the HSM in the oracle setup. WSOP uses an HSM to protect the keys.

1.4. Overview

Transparent Data Encryption (TDE) is used to encrypt an entire database in a way that does not require changes to existing queries and applications. A database encrypted with TDE is automatically decrypted when the database loads it into memory from disk storage, which means that a client can query the database within the server environment without having to perform any decryption operations. The database is encrypted again when saved to disk storage. When using TDE, data is not protected by encryption whilst in memory. The encryption keys that are used to encrypt the database are typically held as part of the database, but these keys are themselves encrypted using a master encryption key in order to protect them. Using an Entrust WSOP allows the master encryption keys to be kept physically separate from the database it is protecting, and also provides a hardware protected boundary from which encryption keys can never leave in plaintext. Additionally, the encryption keys are held by the WSOP server in the Security World which is also encrypted and is useless to anyone who does not possess the authorized means to access them.

Entrust recommends that you allow only unprivileged connections unless you are performing administrative tasks.

Other benefits of using the nShield WSOP include:

- Ability to store keys from all across an enterprise in one place for easy management.
- Key Retention (rotate keys while keeping the old ones).
- FIPS and Common Criteria compliance.

Chapter 2. Preparatory requirements

Before installing the software, Entrust recommends that you familiarize yourself with:

- The Oracle database TDE documentation and setup process.
- The Entrust documentation.

Entrust also recommends you have an agreed organizational Certificate Practices Statement and a Security Policy/Procedure in place covering administration of the HSM. In particular, these documents should include the following aspects of HSM administration:

- Whether the Security World must comply with FIPS 140 Level 3 or Common Criteria restrictions.
    - If you want to use a FIPS 140 Level 3 Security World, then you must create an OCS card set for FIPS authorization. This is true even for Softcard protection which is the only protection method currently supported by WSOP.
    - If you are running multiple database instances on the same host, the same FIPS authorizing OCS cards can be used for all database instances.
- The number and quorum of Administrator Cards in the Administrator Card Set (ACS), and a policy for managing these cards.

  If OCS cards are to be used, you need to decide the number of Operator Cards in the OCS card set. K/N functionality is not currently supported. This means that you must create 1/N OCS card sets. The number of OCS cards in a card set must at least match the number of HSMs that will be in your configuration, and with more to spare in case of a card loss or failure.

- Entrust recommends that you create a policy for managing SQL scripts that allow use of credentials for the Oracle database. These SQL scripts should only be available to authorized users.
- Entrust recommends that you create a policy for managing the passphrases for your:

    - ACS
    - Softcard protection
    - OCS Card Set for FIPS Authorization

  For information on passphrases, see About the HSM credential.

- Entrust recommends that you create a policy for managing the physical security of your smartcards as used for ACS and OCS, and their deployment to authorized users.

As part of your preparation, Entrust recommends that you read Security Worlds key protection and failure recovery.

This guide assumes that Oracle database software, and (at least) one Oracle database, is already installed on your system. With Oracle database software already installed, ensure that any required patches have been added.

To integrate an Oracle database with Entrust WSOP, the following steps are required:

1. Environment configuration.
2. Setup and Install the Entrust WSOP server with the HSM, Security World software and WSOP software.
3. Configure Oracle database software to use the Entrust WSOP server.

Details of your installation and configuration will depend on:

- Whether you want to migrate encryption keys from an existing Oracle software keystore to an Entrust WSOP server, or start directly encrypting keys using an Entrust WSOP server.

The default oracle host server user is `oracle` unless stated otherwise.

For more information on how to configure your Entrust WSOP Server, see the *User Guide* for WSOP.

For more information on how to configure your Oracle environment, see the Oracle documentation.

For more detail or suggestions on how you may set up your system, see the following Appendixes:

- Security Worlds key protection and failure recovery.
- About the HSM credential.

Chapter 3. Install and configure the WSOP server

This section describes the steps needed to setup a WSOP server.

3.1. Create the WSOP server

This guide used Virtual Machine running in VMware vSphere. The VM had the following spec:

- 4 CPUs
- 16 GB RAM
- 100 GB Disk Space
- Red Hat 8 installed.

Once the VM is up and running, login as root to the server.

Open up port 18001 on the firewall. This port is used so clients can talk to the WSOP server.

```
% sudo firewall-cmd --permanent --add-port=18001/tcp
% sudo firewall-cmd --reload
```

3.2. Install Security World Software

1. Install the Entrust Security World software in accordance with its accompanying documentation.
2. Create or edit the `cknfastrc` file located in the `NFAST_HOME` directory, and set the following PKCS#11 environment variables:
   - Including soft card key protection, and HSM load sharing:

     ```
     CKNFAST_LOADSHARING=1
     ```

     For more information, study the PKCS#11 library environment variables in the *User Guide* for your HSM.
3. Create or load the Security World, or nShield Connect (if being used).

   If you are using RA for the ACS cards, you must do so through a registered client. Ensure the Security World data is copied to the `NFAST_KMDATA/local`

folder, and is loaded onto each nShield Connect used in the configuration.

4. Check the Security World on your various components as follows:

   The Security World must be shown as **Initialized** and **Usable**.

   a. Use the Entrust `nfkminfo` utility to check the Security World and configuration. In each case, the Security World must be shown as **Initialized** and **Usable**.
   b. nShield Connect: Front panel: MENU > Security World mgmt. > Display World Info.

      For further details, see the *User Guide* for your HSM.

5. Prepare protection method.

   If your Security World does not already contain the required protection method, then proceed as follows:

   a. The integration requires the use of Softcard protection.

      This will be created in the Oracle database server, when the WSOP client software is installed and configured.

   b. If you are using a FIPS 140 Level 3 world environment, then you also need an OCS card set (1/N) to provide FIPS authorization. If a suitable OCS card set is not already available in the Security World, then create an OCS card set for this purpose.

3.3. Install MongoDB

The WSOP server uses MongoDB as its database. The instructions below are for a RedHat Linux 8 server.

1. Create an `/etc/yum.repos.d/mongodb-enterprise-7.0.repo` file.

   The file should contain the following content to install MongoDB enterprise directly using yum.

```
[mongodb-enterprise-7.0]
name=MongoDB Enterprise Repository
baseurl=https://repo.mongodb.com/yum/redhat/$releasever/mongodb-enterprise/7.0/$basearch/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-7.0.asc
```

2. Install the MongoDB Enterprise packages

```
% sudo yum install -y mongodb-enterprise
```

3.4. Install the Entrust nShield WSOP Software

1. Download the Entrust nShield WSOP tar file and transfer to the server.
2. Place the file in a folder
3. In a terminal, create a temporary directory to unpack the WSOP tar file.

```
% sudo mkdir ~/wsop_install
```

4. Extract the WSOP tar to the temporary directory created above.

```
% cd ~/wsop_install
% sudo tar -xzf ~ /Downloads/wsop-p11-3.3.0-697-4bfd3ef.tar.gz
```

5. Change to the root directory. Then extract the following files from the unpacked WSOP tar.

```
% cd /
% sudo tar zxvf ~/wsop_install/corecrypto.tar.gz
% sudo tar zxvf ~/wsop_install/wsop-common.tar.gz
% sudo tar zxvf ~/wsop_install/dbmt.tar.gz
```

6. Install the Entrust nShield WSOP management tool.

   Execute the following to install the WSOP management tool:

```
% cd /opt/nfast/webservices/dbmt-2.2.0/
% sudo ./install.sh
```

3.5. Import or Create certificates.

Certificates are used to authenticate clients with Entrust nShield WSOP. In addition, Entrust nShield WSOP uses certificates to authenticate with the MongoDB database. Customer of Entrust nShield work out their own security requirements. For the purpose of this integration a Tiny certificate authority will be created.

The following certificates and keys are needed:

- CA Key.
- Root CA.
- Certificate for the MongoDB database server.
- Certificate for the Entrust nShield WSOP server.
- Certificate for Entrust nShield WSOP client.

3.5.1. Become a Tiny certificate authority.

1. Create the certificate directory.

```
% sudo mkdir -p /opt/nfast/webservices/corecrypto/tls/db
% cd /opt/nfast/webservices/corecrypto/tls/db
```

2. Generate the private key.

```
% sudo openssl genrsa -aes256 -out myCA.key 2048

Generating RSA private key, 2048 bit long modulus (2 primes)
........+++++
..............+++++
e is 65537 (0x010001)
Enter pass phrase for myCA.key:
Verifying - Enter pass phrase for myCA.key:
```

3. Generate the root certificate.

```
% sudo openssl req -x509 -new -nodes -key myCA.key -sha256 -days 1825 -out myCA.pem

Enter pass phrase for myCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:US
State or Province Name (full name) []:FL
Locality Name (eg, city) [Default City]:Sunrise
Organization Name (eg, company) [Default Company Ltd]:Entrust
Organizational Unit Name (eg, section) []:nShield
Common Name (eg, your name or your server's hostname) []:myca
Email Address []:test@myca.com
```

Now you should have two files:

- The CA Key: `myCA.key`
- The CA root certificate: `myCA.pem`

3.5.2. Create the certificate for the MongoDB database server.

1. Go to the certificate directory

```
% cd /opt/nfast/webservices/corecrypto/tls/db
```

2. Create an extension file for signing the certs.

   Create `/opt/nfast/webservices/corecrypto/tls/db/sign_db_server.ext`.

   IP is the IP address of the server running the MongoDB database server.

```
basicConstraints=CA:FALSE
keyUsage = digitalSignature
subjectAltName = @alt_names

[alt_names]
IP = xxx.xxx.xxx.xxx
DNS.1 = localhost
DNS.2 = wsopserver-redhat-8
```

3. Create a key for the MongoDB database.

```
% sudo openssl genrsa -out db_server.key 2048

Generating RSA private key, 2048 bit long modulus (2 primes)
.............+++++
....+++++
e is 65537 (0x010001)
```

4. Create a certificate request for the MongoDB database.

```
% sudo openssl req -new -key db_server.key -out db_server.csr

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:US
State or Province Name (full name) []:FL
Locality Name (eg, city) [Default City]:Sunrise
Organization Name (eg, company) [Default Company Ltd]:Entrust
Organizational Unit Name (eg, section) []:nShield
Common Name (eg, your name or your server's hostname) []:wsopserver-redhat-8
Email Address []:test@entrust.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:ncipher
An optional company name []:Entrust
```

5. Sign the certificate request for the MongoDB database server.

```
% sudo openssl x509 -req -in db_server.csr -CA myCA.pem -CAkey myCA.key -CAcreateserial -out db_server.crt
-days 825 -sha256 -extfile sign_db_server.ext

Signature ok
subject=C = US, ST = FL, L = Sunrise, O = Entrust, OU = nShield, CN = wsopserver-redhat-8, emailAddress =
test@entrust.com
Getting CA Private Key
Enter pass phrase for myCA.key:
```

6. Create a pem file containing the key and certificate for the MongoDB database server.

```
% sudo cat db_server.key > /tmp/db_server.pem
% sudo cat db_server.crt >> /tmp/db_server.pem
% mv /tmp/db_server.pem .
```

7. Copy the database root certificate and .pem file to a location accessible by the MongoDB database.

```
% sudo mkdir /etc/ssl/mongodb/
% sudo cp myCA.pem db_server.pem /etc/ssl/mongodb/
% sudo chown -R mongod:mongod /etc/ssl/mongodb
```

3.5.3. Create the certificate for the Entrust nShield WSOP server.

The Entrust nShield WSOP server is a client of the MongoDB database.

1. Go to the certificate directory.

```
% cd /opt/nfast/webservices/corecrypto/tls/db
```

2. Create an extension file for signing the certs.

   Create `/opt/nfast/webservices/corecrypto/tls/db/sign_wsop_server.ext`.

   IP is the IP address of the server running the Entrust nShield WSOP server.

```
basicConstraints=CA:FALSE
keyUsage = digitalSignature
extendedKeyUsage = clientAuth, serverAuth
subjectAltName = @alt_names

[alt_names]
IP = xxx.xxx.xxx.xxx
DNS.1 = localhost
DNS.2 = wsopserver-redhat-8
```

3. Create a key for Entrust nShield WSOP server.

```
% sudo openssl genrsa -out wsop_server.key 2048

Generating RSA private key, 2048 bit long modulus (2 primes)
.....................+++++
.....+++++
e is 65537 (0x010001)
```

4. Create a certificate request for the Entrust nShield WSOP server.

```
% sudo openssl req -new -key wsop_server.key -out wsop_server.csr

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:US
State or Province Name (full name) []:FL
Locality Name (eg, city) [Default City]:Sunrise
Organization Name (eg, company) [Default Company Ltd]:Entrust
Organizational Unit Name (eg, section) []:nShield
Common Name (eg, your name or your server's hostname) []:wsopserver-redhat-8
Email Address []:test@entrust.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:ncipher
An optional company name []:Entrust
```

5. Sign the certificate request for the Entrust nShield WSOP server.

```
% sudo openssl x509 -req -in wsop_server.csr -CA myCA.pem -CAkey myCA.key -CAcreateserial -out
wsop_server.crt -days 825 -sha256 -extfile sign_wsop_server.ext

Signature ok
subject=C = US, ST = FL, L = Sunrise, O = Entrust, OU = nShield, CN = wsopserver-redhat-7\088, emailAddress
= test@entrust.com
Getting CA Private Key
Enter pass phrase for myCA.key:
```

6. Create a pem file containing the key and certificate for the Entrust nShield WSOP server.

```
% sudo cat wsop_server.key > /tmp/wsop_server.pem
% sudo cat wsop_server.crt >> /tmp/wsop_server.pem
% sudo mv /tmp/wsop_server.pem .
```

3.5.4. Create the certificate for Entrust nShield WSOP client

1. Create the client certificate directory.

```
% sudo mkdir -p /opt/nfast/webservices/corecrypto/tls/external
```

2. Change directory to the certificate directory.

```
% cd /opt/nfast/webservices/corecrypto/tls/external
```

3. Create an extension file `/opt/nfast/webservices/corecrypto/tls/external/sign_wsop_client.ext` for signing the certs.

   IP is the IP address of the server running the Entrust nShield WSOP server.

```
basicConstraints = CA:FALSE
keyUsage = digitalSignature
extendedKeyUsage = clientAuth
subjectAltName = @alt_names

[alt_names]
IP.1 = xxx.xxx.xxx.xxx
DNS.1 = localhost
DNS.2 = wsopserver-redhat-8
```

4. Create a key for the client of Entrust nShield WSOP server.

```
% sudo openssl genrsa -out wsop_client.key 2048

Generating RSA private key, 2048 bit long modulus (2 primes)
.............................................+++++
.....................................................................................+++++
e is 65537 (0x010001)
```

5. Create a certificate request for the Entrust nShield WSOP client.

   Set the hostname to the name of the client machine connecting to the Entrust nShield WSOP server.

```
% sudo openssl req -new -key wsop_client.key -out wsop_client.csr

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:US
State or Province Name (full name) []:FL
Locality Name (eg, city) [Default City]:Sunrise
Organization Name (eg, company) [Default Company Ltd]:Oracle
Organizational Unit Name (eg, section) []:TDE
```

```
Common Name (eg, your name or your server's hostname) []:otde-wsop-redhat-9
Email Address []:test@oracle.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:ncipher
An optional company name []:Oracle
```

6. Sign the certificate request for the Entrust nShield WSOP client.

```
% sudo openssl x509 -req -in wsop_client.csr -CA /opt/nfast/webservices/corecrypto/tls/db/myCA.pem \
       -CAkey /opt/nfast/webservices/corecrypto/tls/db/myCA.key \
       -CAcreateserial -sha256 -days 1826 -out wsop_client.crt \
       -extfile sign_wsop_client.ext

Signature ok
subject=C = US, ST = FL, L = Sunrise, O = Oracle, OU = TDE, CN = otde-wsop-redhat-8\089, emailAddress =
test@oracle.com
Getting CA Private Key
Enter pass phrase for /opt/nfast/webservices/corecrypto/tls/db/myCA.key:
```

7. Create a pem file containing the key and certificate for the Entrust nShield WSOP client.

```
% sudo cat wsop_client.key > /tmp/wsop_client.pem
% sudo cat wsop_client.crt >> /tmp/wsop_client.pem
% sudo mv /tmp/wsop_client.pem .
```

3.6. Configure MongoDB

Now let's configure MongoDB so it can be used by WSOP.

1. Save the default `/etc/mongod.conf` by creating a copy.

```
% sudo cp /etc/mongod.conf /etc/mongod.conf.example
```

2. Edit `/etc/mongod.conf` with the applicable parameters.

   The resulting /etc/mongod.conf is:

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /var/log/mongodb/mongod.log

# Where and how to store data.
```

```
storage:
  dbPath: /var/lib/mongo

# how the process runs
processManagement:
  fork: true  # fork and run in background
  pidFilePath: /var/run/mongodb/mongod.pid  # location of pidfile
  timeZoneInfo: /usr/share/zoneinfo

# network interfaces
net:
  port: 27017
  bindIp: 0.0.0.0  # Enter 0.0.0.0,:: to bind to all IPv4 and IPv6 addresses or, alternatively, use the
net.bindIpAll setting.
  tls:
    mode: requireTLS
    certificateKeyFile: /etc/ssl/mongodb/db_server.pem
    CAFile: /etc/ssl/mongodb/myCA.pem

security:
  clusterAuthMode: x509

#operationProfiling:

replication:
  replSetName: rs1

#sharding:

# Enterprise-Only Options

#auditLog:
```

3. Open a separate terminal window and initiate the replication set.

   Open a new terminal window in the WSOP server and do the following:

   a. Kill the mongod process:

   ```
   % sudo pkill mongod
   ```

   b. Start MongoDB in the following manner:

   ```
   % sudo mongod --replSet rs1 --dbpath /var/lib/mongo --bind_ip 0.0.0.0
   ```

4. Back in the previous window, launch the MongoDB shell.

```
% mongosh

Current Mongosh Log ID: 664cf54837141997c3a26a12
Connecting to:
mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.2.6
Using MongoDB:          7.0.9
Using Mongosh:          2.2.6

For mongosh info see: https://docs.mongodb.com/mongodb-shell/
```

```
To help improve our products, anonymous usage data is collected and sent to MongoDB periodically
(https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2024-05-21T15:25:01.616-04:00: Access control is not enabled for the database. Read and write access to
data and configuration is unrestricted
   2024-05-21T15:25:01.616-04:00: You are running this process as the root user, which is not recommended
   2024-05-21T15:25:01.617-04:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'. We suggest
setting it to 'never' in this binary version
   2024-05-21T15:25:01.617-04:00: Soft rlimits for open file descriptors too low

Enterprise test>
```

5. Run rs.initiate().

```
Enterprise test> rs.initiate()

{
  info2: 'no configuration specified. Using a default configuration for the set',
  me: 'wsopserver-redhat-8:27017',
  ok: 1
}
Enterprise rs1 [direct: other] test>
```

6. Switch to admin.

```
Enterprise rs1 [direct: other] test> use admin

switched to db admin
```

7. Copy-paste the following to create users as needed and exit.

```
db.createUser(
    {
    user: "mAdmin",
    pwd: "admin",
    roles: [ { role: "userAdminAnyDatabase", db: "admin" },
             { role: "dbAdminAnyDatabase", db: "admin" },
         { role: "readWriteAnyDatabase", db: "admin" } ]
    }
)
```

   The output is as following:

```
db.createUser(
...     {
...         user: "mAdmin",
...         pwd: "admin",
...         roles: [ { role: "userAdminAnyDatabase", db: "admin" },
...                  { role: "dbAdminAnyDatabase", db: "admin" },
...                  { role: "readWriteAnyDatabase", db: "admin" } ]
...     }
... )
{
  ok: 1,
```

```
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1716319876, i: 4 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1716319876, i: 4 })
}
Enterprise rs1 [direct: primary] admin> exit
```

8. Kill the mongod process.
   a. Check for the mongod processes that are running.

```
% ps -aux | grep mongod

root      260503  0.0  0.2 138432  8300 pts/2    S+   15:25   0:00 sudo mongod --replSet rs1 --dbpath
/var/lib/mongo --bind_ip 0.0.0.0
root      260505  1.2  5.3 3067544 202004 pts/2  SLl+ 15:25   0:06 mongod --replSet rs1 --dbpath
/var/lib/mongo --bind_ip 0.0.0.0
root      262202  0.0  0.0  12144  1192 pts/1    S+   15:33   0:00 grep --color=auto mongod
```

   b. Now kill mongod

```
% sudo pkill mongod
```

9. Afterwards, close the separate terminal window opened above.
10. Change ownership of the following folder and files as follows.

```
% sudo chown -R mongod:mongod /var/lib/mongo
% sudo chown -R mongod:mongod /var/lib/mongo/WiredTiger.turtle
% sudo chown -R mongod:mongod /var/lib/mongo/journal/
```

11. Create a /data/db/ directory.

```
% sudo mkdir -p /data/db/
% sudo chown -R mongod:mongod /data/db/
```

12. Start the mongod process.

```
% sudo service mongod start
```

13. Enable mongod so it starts after a reboot.

```
% sudo systemctl enable mongod
```

14. Check the status of the mongod service.

```
% sudo service mongod status

Redirecting to /bin/systemctl status mongod.service
● mongod.service - MongoDB Database Server
   Loaded: loaded (/usr/lib/systemd/system/mongod.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2024-05-21 15:39:03 EDT; 36s ago
     Docs: https://docs.mongodb.org/manual
 Main PID: 263321 (mongod)
   Memory: 191.6M
   CGroup: /system.slice/mongod.service
           └─263321 /usr/bin/mongod -f /etc/mongod.conf

May 21 15:39:03 wsopserver-redhat-8 systemd[1]: Started MongoDB Database Server.
May 21 15:39:03 wsopserver-redhat-8 mongod[263321]: {"t":{"$date":"2024-05-21T19:39:03.390Z"},"s":"I",
"c":"CONTROL",  "id":7484500, "ctx":"main","msg":"Environment variable MONGOD···k\" to false"}
Hint: Some lines were ellipsized, use -l to show in full.
```

3.7. Configure the Entrust nShield WSOP server

1. Create the configuration file.

   Copy the example file into `/opt/nfast/webservices/corecrypto/conf/config.yaml`.

   ```
   % sudo cp /opt/nfast/webservices/corecrypto/conf/config.yaml.example
   /opt/nfast/webservices/corecrypto/conf/config.yaml
   ```

2. Run the following command to display the hknso of the HSM installed. Copy it to the clipboard.

   ```
   % nfkminfo | grep hknso

   hknso 0adead5baac6c31d69dd964e00309829601fcd05
   ```

3. Edit `/opt/nfast/webservices/corecrypto/conf/config.yaml`.

   Change the applicable parameters. These include the hknso above. The parameter `allow_unauthenticated_clients` was set to true for the purpose of the integration.

   The resulting `/opt/nfast/webservices/corecrypto/conf/config.yaml` is:

   ```
   # Server Authentication TLS (corecrypto server) Options
   external_tls:
       # Host for corecrypto to listen on
       host: 0.0.0.0

       # Port for corecrypto to listen on
       port: 18001

       # TLS Certificate
       cert_file: /opt/nfast/webservices/corecrypto/tls/db/wsop_server.pem
   ```

```
    # Appname:Ident of the key used to protect the TLS private key
    # key_appname_ident: appname:ident

    # Exclude TLS private key from queries
    # exclude_tls_key: true

    # TLS private key file - note only required if not using HSM protection
    key_file: /opt/nfast/webservices/corecrypto/tls/db/wsop_server.key

    # TLS CA Certificate for Mutual Authentication
    ca_certificate_file: /opt/nfast/webservices/corecrypto/tls/db/myCA.pem

    # Enable TLS Client Authentication
    # Warning: we suggest not disabling client_auth_enabled as this will allow unauthenticated client
requests
    client_auth_enabled: true

    # API Gateway
    # This flag is set to true when webservices are expected to work behind an API Gateway
    api_gateway: false

    # Headers
    # This option is only used when webservices work behind an API Gateway
    # By default webservices will look into Authorization headers passed by the gateway to identify its
clients by
    # issuer and subject.
    # If Authorization headers are not provided, then webservices will identify a clients' issuer and
subject values
    # from the request headers defined in issuer_header and subject_header config fields
    # headers:
    #     issuer_header:
    #     subject_header:

    # Directory where to look for the CRL files, these need to have extension .crl or.pem to be loaded
    crl_directory: /opt/nfast/webservices/corecrypto/tls/external/crls

    # Interval at which to poll the CRL directory for changes to CRL files
    crl_poll_interval: 1m

    # Preferred Cipher Suites for external clients.
    # The full list of supported cipher suites can be found in the nShield Web Services Option Pack User
Guide
    # The default provided in this file is a list of recommended cipher suites.
    cipher_suites:
        - ECDHE-ECDSA-AES128-GCM-SHA256
        - ECDHE-RSA-AES128-GCM-SHA256
        - ECDHE-ECDSA-AES256-GCM-SHA384
        - ECDHE-RSA-AES256-GCM-SHA384
        - ECDHE-ECDSA-CHACHA20-POLY1305
        - ECDHE-RSA-CHACHA20-POLY1305

logging:
    # The loglevel
    # Valid values: Trace, Debug, Info, Warning, Error
    loglevel: Warning

    # Logging to console (stdout or stderr)
    console:
        # The console level output
        # Valid values: stderr, stdout, discard or empty
        output: discard

    # Logging to a file
    file:
        # Enable logging to a file
        enabled: true
```

```
        # The absolute path of the file that logs should be written to
        filepath: /opt/nfast/log/corecrypto.log

    # Logging to syslog
    syslog:
        # Enable logging to a configured syslog server
        enabled: false

        # The networking protocol to be used to send logs to syslog
        # Optional - if syslog is enabled but network is not configured, the local syslog daemon will be
used
        # Valid values: udp
        network: udp

        # The IP/Hostname of the machine hosting syslog and the port to access syslog on in the format
hostname:port
        # Optional - if syslog is enabled but host is not configured, the local syslog daemon will be used.
        # Valid values: localhost, and IPv4 addresses in the format x.x.x.x where x is a value between 0
and 255
        host: localhost:514

# Health Check Options
health:
    # Interval in seconds between estate health check
    # Estate means the modules which belong of the security world
    estate_check_interval: 5s

    # Period in seconds after which the estate health check will timeout
    estate_check_timeout: 4s

    # Interval in seconds between each security world check
    # Its value is a multiple of the estate_check_interval value
    # Note: each time a security world check takes place, an estate
    # health check also takes place
    sworld_check_interval: 300s

    # Interval in seconds between each database health check
    database_check_interval: 5s

    # Period in seconds after which a database health check will timeout
    database_check_timeout: 30s

    # Allow unauthenticated clients to probe the health check endpoint.
    # Only applicable when tls.client_auth_enabled is true
    allow_unauthenticated_clients: true

    # Extend health check status to include whether the service can acquire FIPS-authentication
    include_fips_ready_check: false

# World
world:
    # Hash of the security world Officer
    # It needs to match the world in the database and needs to be set before starting corecrypto.
    hknso: 0adead5baac6c31d69dd964e00309829601fcd05

# Concurrent Request Limiter Options
# Too many concurrent requests can lead to a denial of service
concurrent_request_limiter:
    # Maximum number of concurrent requests that the server supports
    outstanding_requests_limit: 500
    # Maximum number of concurrent requests for generating or importing RSA keys
    rsa_key_requests_limit: 10
    # Maximum number of concurrent requests for generating or importing ECDSA keys
    ecdsa_key_requests_limit: 30

# Caching Options
cache:
```

```
    # Period of inactivity in minutes after which a key manager will be closed
    # Only applicable when tls.client_auth_enabled is true
    # Setting of 0 is used to disable closing an inactive key manager.
    key_manager_inactivity: 1440m

    # Capacity of the key cache. When the number of keys in the cache reaches this
    # capacity, then the keys are evicted based on a least recently used (LRU) policy.
    # Minimum value 100
    key_cache_capacity: 30000

    # Period of time for which a key stays in the cache before it is evicted
    # Minimum value 1m
    key_TTL_period: 60m

    # Capacity of the group cache: when the number of groups in the cache reaches the
    # capacity, then the groups are evicted based on least recently used (LRU) policy
    # Minimum value 100
    group_cache_capacity: 20000

    # Time to live value for a group in the cache
    # Minimum value 1m
    group_TTL_period: 120m

    # Maximum number of protection domains which are stored in the corecrypto cache
    # Minimum value 100
    max_number_of_active_protection_domains: 10000

    # Time to live value for protection domains in the cache
    # Minimum value 1m
    domain_TTL_period: 720m

# Database Options
database:
    # List of database hosts
    hosts:
        - localhost:27017

    # The loglevel of the database driver component
    # Valid values: Trace, Debug, Info, Warning, Error
    loglevel: Info

    # Time before a database request should fail
    timeout: 5s

    # Maximum returned keys when listing, large queries can hurt the service.
    # This limit affects the list keys endpoint.
    # If you wish to retrieve more than the maximum limit then you may make multiple API
    # requests and combine the results within your application using the offset and limit.
    # listkeys_max_limit: 300000

    # Name of the database
    db_name: nshield-corecrypto

    # Webservices Corecrypto Segregation (WCS) Options
    # The segregation database must be defined in order
    # to enable WCS. When enabled, corecrypto objects
    # will be segregated based on the mappings defined in the collection.
    # If the segregation database is not defined,
    # WCS is disabled.
    #
    # segregations_db_name: segregation_db
    # segregations_collection_name: segregations

    # Authentication method with database. Valid values: [none, pwd, tls]
    # none - no authentication
    # pwd  - username and password authentication using mongodb SCRAM
    # tls  - x509 authentication
```

```
    auth_type: tls

    # If 'auth_type' is pwd, 'auth_username_file' and 'auth_password_file'
    # options define the location of a secure file containing the username
    # and passphrase to use for authentication.
    # 'auth_source' is a database-type specific identifier for what to authenticate
    # against. For mongodb this is the name of the authentication database.
    #
    # auth_username_file: /opt/nfast/webservices/corecrypto/pwd-auth/config-username-auth
    # auth_password_file: /opt/nfast/webservices/corecrypto/pwd-auth/config-password-auth
    # auth_source: userdb

    # Transport Layer Security. Default is false.
    disable_tls: false

    # Path to the mongoDB TLS certificate
    db_ca_file: /opt/nfast/webservices/corecrypto/tls/db/myCA.pem

    # Path to the corecrypto client certificate (used when Mutual Authentication is enabled)
    db_cert_file: /opt/nfast/webservices/corecrypto/tls/db/wsop_server.pem

    # Path to the corecrypto client private key (used when Mutual Authentication is enabled)
    db_key_file: /opt/nfast/webservices/corecrypto/tls/db/wsop_server.key

    # Type of database. Supported values: mongodb
    db: mongodb

    # Disable this option when WSOP is running with keySafe5 instance.
    is_WSOP_standalone: true

    mongodb:
        # Name of the Replication Set
        replica_set: rs1

        # Timeout for connection to the database server
        connect_timeout: 5s
        # Timeout for selecting a connection from the pool
        selection_timeout: 5s
        # Timeout waiting for read/write in the socket
        socket_timeout: 5s

        # Minimum and maximum connections to use in mongodb's connection pool
        min_pool_size: 1
        max_pool_size: 100

#group options
group:
    # Allow all group deletions.
    # Disable this option to prevent the deletion of the reserved groups belonging
    # to either the Module Protection or Well-Known Key Protection domains.
    # A reserved group is defined as any public segregated group which has the same name as
    # the protection domain it belongs to.
    allow_all_group_deletions: true
```

4. Configure the Entrust nShield WSOP management tool.
    a. Create the configuration file.

    Copy the example file into `/opt/nfast/webservices/dbmt-2.2.0/config.yaml`.

    ```
    % sudo cp /opt/nfast/webservices/dbmt-2.2.0/config_example.yaml /opt/nfast/webservices/dbmt-
    2.2.0/config.yaml
    ```

b. Edit `/opt/nfast/webservices/dbmt-2.2.0/config.yaml`.

Use the applicable parameters.

The resulting /opt/nfast/webservices/dbmt-2.2.0/config.yaml is:

```
# Database hostname (ip address)
db_host : localhost
# Database port number
db_port : 27017

# Name of the database
db_name: nshield-corecrypto

# Database Management System. Valid values: [mongodb]
db: mongodb

mongodb:
  # Transport Layer Security. Default is enabled
  # disable_tls: false

  #
  # Authentication method with database. Valid values: [none, pwd, tls]
  auth_type: tls

  #
  # If 'auth_type' is pwd, 'auth_pwd_file' defines the location of a secure file
  # containing the username and passphrase to use for authentication.
  # 'auth_source' is a database-type specific identifier for what to authenticate
  # against. For mongodb this is the name of the authentication database.
  #
  # auth_username_file: config-username-auth
  # auth_password_file: config-password-auth
  # auth_source: userdb

  #
  # Certificate Authority files to use for TLS
  db_ca_file: /opt/nfast/webservices/corecrypto/tls/db/myCA.pem
  db_cert_file: /opt/nfast/webservices/corecrypto/tls/db/wsop_server.pem
  db_key_file: /opt/nfast/webservices/corecrypto/tls/db/wsop_server.key

  #
  # Webservices Corecrypto Segregation (WCS) Options
  # The segregation database and collection must be defined in order
  # to enable WCS. When enabled, corecrypto objects
  # will be segregated based on the mappings defined in the collection.
  # If the segregation database and collection are not defined,
  # WCS is disabled.
  #
  # segregations_db_name: segregation_db
  # segregations_collection_name: segregations
```

3.8. Start the Entrust nShield WSOP service

1. Initialize the database.

```
% sudo /opt/nfast/python3/bin/dbmt db-init --config /opt/nfast/webservices/dbmt-2.2.0/config.yaml

Starting initialisation of the database...
```

```
Establishing connection to hardserver
TLS enabled
X509 Authentication enabled
Establishing mongo connection to: localhost:27017
Creating database with name: nshield-corecrypto
Setting indices on collections: nshield-corecrypto
Adding to database: Security World: Security World identifier: 57f6ab61-5547-4ed3-a249-4a9c1710c167
Migrating the module certificates
Migrating "module_BD10-03E0-D947" module file
Adding to database: module certificates: esn: BD10-03E0-D947, hkml:
1dd6a3890c4ec65010466ee3ba5eaef3d6a61777
Migrating cardsets
Migrating cards
getting protection domain by uuid
Adding to database: domain: name: Module Protection type: Module id: 5ee26b95-7e90-53c3-ae75-2e5ddea011bc
Adding to database: group_id: 5ee26b95-7e90-53c3-ae75-2e5ddea011bc name: Module Protection domain_id:
5ee26b95-7e90-53c3-ae75-2e5ddea011bc domain_type Module
getting protection domain by uuid
Adding to database: domain: name: Well-Known Key Protection type: WellKnown id: 2bd40730-85b1-5deb-8417-
fb78a7735743
Adding to database: group_id: 2bd40730-85b1-5deb-8417-fb78a7735743 name: Well-Known Key Protection
domain_id: 2bd40730-85b1-5deb-8417-fb78a7735743 domain_type WellKnown
Initialisation of the database completed.
Finished
```

This command can also be used to load new content from /opt/nfast/kmdata/local into the database.

2. Install the corecrypto service.

```
% sudo /opt/nfast/webservices/sbin/install

-- Running install fragment corecrypto
Creating wsopd group.
Checking for user 'wsopd'
Creating wsopd user.
useradd: warning: the home directory already exists.
Not copying any file from skel directory into it.
Checking user 'wsopd' is in correct group 'wsopd'
users created correctly
Installing startup scripts for 'corecrypto'.
Enabling the systemd service unit
Adding and enabling a systemd unit
Created symlink /etc/systemd/system/multi-user.target.wants/nc_corecrypto.service →
/etc/systemd/system/nc_corecrypto.service.
Note: Forwarding request to 'systemctl enable nc_corecrypto.service'.
Starting nCipher 'corecrypto' server process.
Job for nc_corecrypto.service failed because the control process exited with error code.
See "systemctl status nc_corecrypto.service" and "journalctl -xe" for details.
● nc_corecrypto.service - nFast corecrypto
   Loaded: loaded (/etc/systemd/system/nc_corecrypto.service; enabled; vendor preset: disabled)
   Active: failed (Result: exit-code) since Tue 2024-05-21 16:02:03 EDT; 30ms ago
     Docs: https://nshielddocs.entrust.com/
  Process: 295512 ExecStopPost=/bin/chgrp $WSOPD_GROUP corecrypto.log (code=exited, status=0/SUCCESS)
  Process: 295510 ExecStopPost=/bin/chown $WSOPD_USER corecrypto.log (code=exited, status=0/SUCCESS)
  Process: 295508 ExecStopPost=/bin/bash -c echo corecrypto quit results: $SERVICE_RESULT $EXIT_CODE
$EXIT_STATUS >> corecrypto.log (code=exited, status=0/SUCCESS)
  Process: 295506 ExecStopPost=/bin/bash -c echo "'corecrypto' shut down" >>corecrypto.log (code=exited,
status=0/SUCCESS)
  Process: 295503 ExecStopPost=/bin/bash -c date >>corecrypto.log (code=exited, status=0/SUCCESS)
  Process: 295501 ExecStopPost=/bin/rm -f corecrypto.pid (code=exited, status=0/SUCCESS)
  Process: 295352 ExecStartPost=/bin/bash -c if [ -f "/opt/nfast/scripts/startup/wait-for-corecrypto" ];
then exec "/opt/nfast/scripts/startup/wait-for-corecrypto"; fi (code=exited, status=1/FAILURE)
```

```
  Process: 295351 ExecStart=/bin/bash -c if [ -f /etc/nfast.conf ]; then . /etc/nfast.conf; fi; exec
"/opt/nfast/sbin/crypto"  >>corecrypto.log 2>>corecrypto.log (code=exited, status=1/FAILURE)
  Process: 295349 ExecStartPre=/bin/bash -c if [ -f /etc/nfast.conf ]; then owrite=$(stat -c%A
/etc/nfast.conf | awk '{print substr($0,length-1,1)}'); if [ "$owrite" != "-" ]; then echo "/etc/nfast.conf
can be written to by non-root users" >> corecrypto.log; exi\
t 1; fi; fi (code=exited, status=0/SUCCESS)
  Process: 295347 ExecStartPre=/bin/bash -c if [ -f /etc/nfast.conf ]; then owner=$(stat -c%u:%g
/etc/nfast.conf); if [ "$owner" != "0:0" ]; then echo "/etc/nfast.conf is not fully owned by root" >>
corecrypto.log; exit 1; fi; fi (code=exited, status=0/SUCCESS)
  Process: 295345 ExecStartPre=/bin/chgrp $WSOPD_GROUP corecrypto.log (code=exited, status=0/SUCCESS)
  Process: 295343 ExecStartPre=/bin/chown $WSOPD_USER corecrypto.log (code=exited, status=0/SUCCESS)
  Process: 295341 ExecStartPre=/bin/touch corecrypto.log (code=exited, status=0/SUCCESS)
  Process: 295339 ExecStartPre=/bin/chgrp $WSOPD_GROUP corecrypto.pid (code=exited, status=0/SUCCESS)
  Process: 295337 ExecStartPre=/bin/chown $WSOPD_USER corecrypto.pid (code=exited, status=0/SUCCESS)
  Process: 295335 ExecStartPre=/bin/touch corecrypto.pid (code=exited, status=0/SUCCESS)
 Main PID: 295351 (code=exited, status=1/FAILURE)

May 21 16:01:58 wsopserver-redhat-8 systemd[1]: Starting nFast corecrypto...
May 21 16:01:58 wsopserver-redhat-8 systemd[1]: nc_corecrypto.service: Main process exited, code=exited,
status=1/FAILURE
May 21 16:02:03 wsopserver-redhat-8 bash[295352]: waiting for 'corecrypto'
May 21 16:02:03 wsopserver-redhat-8 bash[295352]: 'corecrypto' did not start; see
/opt/nfast/log/corecrypto.log.
May 21 16:02:03 wsopserver-redhat-8 systemd[1]: nc_corecrypto.service: Control process exited, code=exited
status=1
May 21 16:02:03 wsopserver-redhat-8 systemd[1]: nc_corecrypto.service: Failed with result 'exit-code'.
May 21 16:02:03 wsopserver-redhat-8 systemd[1]: Failed to start nFast corecrypto.
2024-05-21 16:01:58.416 [INFO] [DBADAPTER] [295351] TLS CA key loaded
2024-05-21 16:01:58.416 [ERROR] [DBADAPTER] [295351] TLS configuration failed: open
/opt/nfast/webservices/corecrypto/tls/db/wsop_server.key: permission denied
2024-05-21 16:01:58.416 [FATAL] [WSOP] [295351] [server] NewWSOPServer: cannot initialise database invalid
adapter configuration
Tue May 21 16:02:03 EDT 2024
'corecrypto' shut down
corecrypto quit results: exit-code exited
```

3. Change the ownership of the following folders as follows:

```
% sudo chown -R root:wsopd /opt/nfast/webservices/corecrypto/tls
% sudo chmod -R 750 /opt/nfast/webservices/corecrypto/tls
```

4. Make sure hostname is visible via DNS or by using the /etc/hosts file.

   Edit /etc/hosts and add the hostname to it.

```
xxx.xxx.xxx.xxx wsopserver-redhat-8
```

5. Restart corecrypto to take in the changes.
   a. Stop the service.

```
% sudo /opt/nfast/scripts/init.d/corecrypto stop
```

   b. Start the service.

```
% sudo /opt/nfast/scripts/init.d/corecrypto start
```

6. Verify connection to the Entrust nShield WSOP service.

```
% curl -k 'https://xxx.xxx.xxx.xxx:18001/health' | jq


  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    52  100    52    0     0   4333      0 --:--:-- --:--:-- --:--:--  4333
{
  "releaseId": "1.5.0",
  "status": "pass",
  "version": "1"
}
```

7. Verify the secure connection.

```
% sudo curl -X GET \
  --cacert /opt/nfast/webservices/corecrypto/tls/db/myCA.pem \
  --cert /opt/nfast/webservices/corecrypto/tls/external/wsop_client.pem \
  --key /opt/nfast/webservices/corecrypto/tls/external/wsop_client.key \
  --header 'Content-Type: application/json' \
  --header 'Accept: application/json' 'https://xxx.xxx.xxx.xxx:18001/health' | jq

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    52  100    52    0     0   5200      0 --:--:-- --:--:-- --:--:--  5200
{
  "releaseId": "1.5.0",
  "status": "pass",
  "version": "1"
}
```

8. Check the version

```
% /opt/nfast/sbin/crypto --version

crypto, 2.3.0-714-5306bc4
```

Chapter 4. Install the Entrust WSOP Client software on each client.

Each Oracle server needs to have the WSOP client software installed so it can use the WSOP PKCS#11 API library. Make sure you are logged into the Oracle database server as you follow these instructions. To install and configure the WSOP PKCS#11 library, you will have to:

4.1. Download and install software.

1. Untar the WSOP tar file so you can access the Web Services PKCS #11 library

```
% mkdir wsop
% cd wsop
% tar zxvf /path/to/wsop-p11-3.3.0-714-5306bc4.tar.gz
```

2. Sign in as a user with root privileges and extract the PKCS#11 tar to the root directory.

   This installs all the files required by PKCS#11 to `/opt/nfast/webservices/pkcs11`.

```
% sudo su - root
% cd /
% tar zxvf /path/to/wsop/nShield-WebServicesClient-Linux-1.3.0-431-bb1eab8.tar.gz
```

4.2. Configure WSOP configuration parameters.

1. Create the configuration file.

   Go to the `/opt/nfast/webservices/pkcs11/conf` folder and copy the `example_pkcs11webservices.cfg` file to `pkcs11webservices.cfg`.

```
% cd /opt/nfast/webservices/pkcs11/conf
% cp example_pkcs11webservices.cfg pkcs11webservices.cfg
```

2. If required, change the log file destination or grant permission to the appropriate user to write to the default log folder:

```
/opt/nfast/webservices/pkcs11/log
```

3. Configure the Web Services PKCS#11 library.

The pkcs11webservices.cfg file contains an example PKCS #11 configuration. To use this file, you need to alter it to point to the correct certificate locations.

The PKCS #11 library is installed with a default configuration. Entrust recommends reviewing and updating the initial configuration before the library is called to ensure that all configuration settings are appropriate for the deployment environment. Pay special attention to the TLS connection and logging to ensure that the system is used securely. Ensure that the configuration file and TLS authentication files have restrictive access control, so that only the application using the PKCS #11 client library has access to these files.

4. Configuration parameters.

   The following configuration options are available:

   a. Web Services server hostname

      ```
      HOST <host name of server>
      ```

      Set this to the Web Services server host name.

   b. Log level

      ```
      LOGLEVEL 4
      ```

      The LOGLEVEL field in the configuration file controls the PKCS #11 library logging. The available log levels are:

      - 0(None)
      - 1(Fatal)
      - 2(Error)
      - 3(Warning)
      - 4(Debug 1)
      - 5(Debug 3)
      - 6(Debug 9)

        By default, the PKCS #11 logging level is set at 4 (Debug 1).

        Higher log levels include all logs from lower levels. If the logging level is set to 4(Debug 1), you only get log events with Debug 1, Warning, Error, and Fatal. If the logging level is set to 6(Debug 9) it enables all the log levels. To turn off logging set LOGLEVEL to 0(None).

c. Enable logging to console.

   By default, PKCS#11 has console logging disabled. In order to enable this, set LOGSTDOUT to 1 in the configuration file.

   ```
   LOGSTDOUT 1
   ```

d. Retry Web Services communication.

   Since the PKCS #11 library operates over a network connection, it is possible that network fluctuations could cause an internal request to the Web Services server to fail. To ensure the reliability and robustness of PKCS #11 applications, the request can be retried.

   You can configure the maximum number of retries for each request and the delay (in seconds) between each retry. For example:

   ```
   MAXRETRIES 5
   RETRYDELAY 10
   ```

   If you don`t define either of these entries, the PKCS#11 library uses the defaults.

   For MAXRETRIES, the default is 5 and the maximum number is 256.

   To disable retries, set MAXRETRIES to 0.

   For RETRYDELAY, the default is 10 seconds. There is no maximum value but higher values will reduce performance on unstable systems. Only integer values are supported.

   To disable the retry delay, set RETRYDELAY to 0. This will cause retries to occur consecutively without delay.

e. TLS certificate.

   ```
   CERT <path to TLS certificate>
   ```

   Set this to the file path of the client TLS certificate.

f. TLS private key.

   ```
   KEY <path to private key>
   ```

   Set this to the file path of the client TLS private key.

g. TLS client authentication.

```
AUTH <path to TLS CA Certificate for Mutual Authentication>
```

Set this to the server root certificate, which forms the trust anchor for authenticating the server`s certificates received during TLS handshake.

h. Log file path on Linux.

By default, PKCS #11 outputs the logs to /opt/nfast/webservices/pkcs11/log/pkcs11webservices.log. To change the default path set LOGFILEPATH to any valid existing path.

```
LOGFILEPATH /opt/nfast/webservices/pkcs11/log/pkcs11webservices.log
```

To enable logging for a user, change the path to a Linux user path so that the user has write permission to the folder.

```
LOGFILEPATH /home/<username>/log/pkcs11webservices.log
```

i. Enable logging to syslog.

You can direct PKCS #11 logs to a syslog server. By default this configuration is disabled. In order to enable this, set LOGSYS to 1 in the configuration file.

```
LOGSYS 1
```

4.3. Server / client authentication with the Web Services PKCS #11 library.

The Web Services PKCS #11 Library can only communicate securely with a WSOP Server if the following certificates are installed:

- The WSOP Server`s CA certificate.
- An appropriate client certificate (with each PKCS #11 client using its own client certificate).
- Any intermediate CA certificates that are to be used to form a complete chain to verify the client certificate on the WSOP Server.

The following guidance should be followed when installing the PKCS #11 Library's

certificates in a Linux platform:

- If intermediate certificates are used in the PKCS #11 library`s client certificate hierarchy, they should be included in the same file as the client certificate. The root client certificate should not be included in this file.
- The recommended format for the client certificate file is PEM.
- The recommended format for the client certificate key file is PEM.

Proceed with the following Setup:

1. Add the WSOP utilities to the PATH.

```
% export PATH=$PATH:/opt/nfast/webservices/pkcs11/bin
```

2. Import client certificates.

   These steps copy the client certificates created in the Entrust nShield WSOP server to the application server:

   a. Make a directory where the client certificates will reside.

```
% sudo mkdir -p /opt/ssl/wsop_client
```

   b. Copy the client certificates created in the Entrust nShield WSOP server.

   ...Copy the CA certificate.

```
% sudo scp root@WSOP_SERVER_IP_ADDRESS:/opt/nfast/webservices/corecrypto/tls/db/myCA.pem
/opt/ssl/wsop_client/.
```

   i. Copy the Certificate Key.

```
% sudo scp
root@WSOP_SERVER_IP_ADDRESS:/opt/nfast/webservices/corecrypto/tls/external/wsop_client.key
/opt/ssl/wsop_client/.
```

   ii. Copy the Client Certificate.

```
% sudo scp
root@WSOP_SERVER_IP_ADDRESS:/opt/nfast/webservices/corecrypto/tls/external/wsop_client.pem
/opt/ssl/wsop_client/.
```

   iii. Set the permissions to the certificates so the oracle user owns it.

```
% sudo chown oracle /etc/ssl/wsop_client/*
```

3. Edit web services option pack configuration file so it can communicate with the WSOP server.

   Go to the /opt/nfast/webservices/pkcs11/conf folder and edit the pkcs11webservices.cfg file.

   - Set HOST to WSOP_SERVER_IP_ADDRESS
   - Set CERT to `/opt/ssl/wsop_client/wsop_client.pem`
   - Set KEY to `/opt/ssl/wsop_client/wsop_client.key`
   - Set AUTH to `/opt/ssl/wsop_client/myCA.pem`

4. Open up the permissions to the log file so the application can write to it.

```
% sudo touch /opt/nfast/webservices/pkcs11/log/pkcs11webservices.log
% sudo chmod 777 /opt/nfast/webservices/pkcs11/log/pkcs11webservices.log
```

4.4. Create a softcard

Because PKCS#11 does not directly support Softcard generation, a command line tool is provided. The Softcard tool uses the same configuration file as the PKCS#11 library for the Web Services server secure connection. It does not support any logging.

To generate a new Softcard run the following command:

```
% /opt/nfast/webservices/pkcs11/bin/softcardtool -g --name=testSC
```

When prompted, enter a new passphrase for the Softcard.

If using FIPS Level 3 world make sure an OCS card is available in the WSOP Server to provide FIPS authorization, otherwise the following message will show:

```
% /opt/nfast/webservices/pkcs11/bin/softcardtool -g --name=mysoftcard
New softcard generation
Enter softcard pass phrase:
Verifying - Enter softcard pass phrase:

Failed to generate softcard errorcode WSOP_ERROR_HTTP_RESPONSE
Error message: BadRequest: fips auth required for operation but not available: card not in slot
```

4.5. Check WSOP Web Server connection.

Become the oracle user and test the connection. To verify the Web Services server connection, run the tool with the verbose and list options:

```
% /opt/nfast/webservices/pkcs11/bin/softcardtool -vl

Connecting to xxx.xxx.xxx.xxx port 18001 with:
Client Certificate File /opt/ssl/wsop_client/wsop_client.pem
Client Private Key File /opt/ssl/wsop_client/wsop_client.key
Certificate Authority file /opt/ssl/wsop_client/myCA.pem
Server connection health check passed

Softcard list
ID                                  "Name"
```

You can also use the following curl command to check the connection:

```
% curl -X GET --cacert /opt/ssl/wsop_client/myCA.pem \
        --cert /opt/ssl/wsop_client/wsop_client.pem \
        --key /opt/ssl/wsop_client/wsop_client.key \
        --header 'Content-Type: application/json' \
        --header 'Accept: application/json' 'https://WSOP_SERVER_IP_ADDRESS:18001/km/v1/groups' | jq

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   496  100   496    0     0   4769      0 --:--:-- --:--:-- --:--:--  4769
{
  "groups": [
    {
      "groupid": "ad319421-031f-5261-a1e6-6f04849be23e",
      "keys": "/km/v1/groups/ad319421-031f-5261-a1e6-6f04849be23e/keys",
      "name": "Module Protection",
      "protection": "/km/v1/protectiondomains/ad319421-031f-5261-a1e6-6f04849be23e",
      "type": "Module"
    },
    {
      "groupid": "2bd40730-85b1-5deb-8417-fb78a7735743",
      "keys": "/km/v1/groups/2bd40730-85b1-5deb-8417-fb78a7735743/keys",
      "name": "Well-Known Key Protection",
      "protection": "/km/v1/protectiondomains/2bd40730-85b1-5deb-8417-fb78a7735743",
      "type": "WellKnown"
    }
  ]
}
```

4.6. Deleting a softcard and its keys.

If you need to delete a Softcard, and remove all keys associated with the Softcard and use the following command:

```
% /opt/nfast/webservices/pkcs11/bin/softcardtool -d -i <deleted-softcard-ID>
```

Chapter 5. Oracle Configuration

5.1. Configure the Oracle PKCS#11 library folder

This will configure Oracle to use the Entrust WSOP PKCS#11 API.

After creating the Oracle database, you will have to:

1. Create the following directory path for the Entrust API library as the `oracle` user.

   Make ownership and permissions on the directory as:

   - owner=oracle
   - group=oinstall
   - permissions=775.

   ```
   % mkdir -p $ORACLE_BASE/extapi/64/hsm/nCipher/13.4.5
   % chown oracle $ORACLE_BASE/extapi/64/hsm/nCipher/13.4.5
   % chgrp oinstall $ORACLE_BASE/extapi/64/hsm/nCipher/13.4.5
   % chmod 775 $ORACLE_BASE/extapi/64/hsm/nCipher/13.4.5
   ```

2. Copy the WOP PKCS#11 library into the directory as the `oracle` user.

   ```
   % cp /opt/nfast/webservices/pkcs11/lib/libpkcs11webservices.so $ORACLE_BASE/extapi/64/hsm/nCipher/13.4.5
   ```

   

   The Entrust WSOP PKCS#11 API library is the only means by which the Oracle database system can communicate with the Entrust WSOP system. If this interface is not set up correctly, you will not be able to get these two systems to operate together.

3. Configure Oracle keystore folder.

   You only need to provide a keystore-folder if encryption key migration is required. Do this as the oracle user. If a keystore-folder is required, then create a folder as follows:

   ```
   % mkdir -p $ORACLE_BASE/admin/FREE/keystore-folder
   ```

   Make sure ownership/permissions on the 'keystore-folder' allow the 'oracle' OS user to use it.

4. Make sure that the environment variable `ORACLE_SID` is set to `FREE`.

   Edit the oracle user `.bashrc` file and change the variable. Log out and log back in as oracle.

   `FREE` is the database that automatically comes installed for you when you install Oracle 23ai. Set `ORACLE_SID` according to the database used in your environment.

5.2. Configure Oracle database software to use the Entrust WSOP.

Before proceeding, it is assumed that:

- You have followed the set up and configuration instructions in this guide. That is:
  - The Oracle database software is installed with at least one database instance. This guide uses the FREE database already provided by Oracle 23ai.
  - The Entrust WSOP Server, Security World software and HSM are installed and configured.
  - THE Entrust WSOP Client software has been installed and configured in each Oracle Database Server.
  - Your protection method has been prepared. (softcard created)
  - OCS Cards have been created to provide FIPS authorization if a FIPS Level 3 world file is going to be used.
- The target container database (CDB) is open, and all PDBs are open. The target container database in this guide is the FREE database that comes with Oracle 23ai installation.

You can use the following instructions to configure your Oracle database software to function using the Entrust WSOP server in one of the following scenarios:

- Migration from keystore to WSOP: One or more database instances are already using TDE encryption, each instance with its own software keystore, and you want to continue using TDE encryption after migrating the TDE master keys from at least one keystore to the Entrust WSOP server.
- Create keys directly in the WSOP server: One or more database instances are not using TDE encryption, and you want to start using TDE encryption for at least one database, using the Entrust WSOP server.

Before attempting key migration, see Key migration and legacy keys.

The SQL commands that will be used later in this document might:

- Require more than one user with suitable database privileges to make the specific database connections, and run the SQL commands in the sequences as shown. Respect the connections shown in order to satisfactorily run SQL on your target. See Database connections. Your system administrator should have sufficient knowledge to create users and associated privileges according to your organization's security policies.
- Need to be run as a certain user. If you are instructed in this guide to make a connection as a particular user, continue with that connection until instructed otherwise.
- Use `<credential>` to denote your chosen protection method. When a protection method has been invoked, you must continue with the same protection method unless you decide to alter it as described in About the HSM credential.

Oracle documentation uses the `<credential-name>|<credential-passphrase>` order. However, tests showed that the ordering `<credential-passphrase>|<credential-name>` works. In SQL, the credential used to open a keystore must match the credential used to create an encryption key.

5.3. Opening and closing a keystore or HSM

Oracle has a control system that gates access to a software keystore or HSM:

- If a keystore or HSM is open, then you can access its contents.
- If a keystore or HSM is closed, then you cannot access its contents.

5.3.1. You can open or close a software keystore or HSM with the following SQL statements.

This section assumes the respective CDB and PDB databases are open:

- To open/close keystore for the container (CDB) only.

  CONNECT C##TESTER@FREE

```
ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "<credential>";
```

```
ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY "<credential>";
```

- To open/close keystore for the container (CDB) and all PDBs it holds.

  CONNECT C##TESTER@FREE

  ```
  ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "<credential>" CONTAINER=ALL;
  ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY "<credential>" CONTAINER=ALL;
  ```

  If you want to close all keystores, use the following SQL:

  ```
  ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE CONTAINER=ALL;
  ```

- To open/close keystore for a single PDB, you must use same credential as used by the containing CDB.

  CONNECT PDB<k>TESTER@FREEPDB<k>

  ```
  ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "<credential>";
  ADMINISTER KEY MANAGEMENT SET KEYSTORE CLOSE IDENTIFIED BY "<credential>";
  ```

5.3.2. Issues closing keystores

During migration from Software Wallet to HSM Keystore, you may experience issues closing the keystore. To resolve this, disable the auto-login keystore to close all keystores. See How To Disable Auto-Login Keystore for full details.

```
% sudo -u oracle mv <path-to-keystorefolder>/<keystore-folder>/tde/cwallet.sso <path-to-
keystorefolder>/<keystore-folder>/tde/cwallet.sso.backup
```

5.4. Active credentials

The first time you open a keystore or HSM using a credential for a particular database instance, it activates the credential you are referencing. You should then be able to create master encryption keys, or use (any) existing master encryption keys, that are protected by that credential. You cannot have more than one active credential at the same time for the same instance. You must close the keystore or HSM to deactivate the credential.

You can simultaneously use different credentials for different database instances on the same host server. For a container database only its CDB is a real instance. All PDBs within the same CDB must use the same active credential.

See About the HSM credential if you want to change a credential.

5.5. Migrating from software keystore to HSM

The following procedure applies when you are already using a software wallet with TDE encryption.

Repeat the following procedure for each software keystore from which you want to migrate.

See About the HSM credential if you want to change a credential.

Use the `WALLET_ROOT` and `TDE_CONFIGURATION` parameters.

1. Back up your software keystore before attempting key migration to the HSM:

   CONNECT sysdba@FREE

   ```
   ADMINISTER KEY MANAGEMENT BACKUP KEYSTORE USING '<PreMigrationBackupString>' IDENTIFIED BY
   "<keystorepassphrase>";
   ```

2. Prepare for key migration by running the following SQL script:

   CONNECT sysdba@FREEROOT

   ```
   ALTER SYSTEM SET TDE_CONFIGURATION = "KEYSTORE_CONFIGURATION=HSM|FILE" SCOPE=BOTH SID='*';
   ```

3. Create an auto-login keystore where `<credential>` is the HSM credential you want to use:

   CONNECT sysdba@FREEROOT

   ```
   ALTER PLUGGABLE DATABASE ALL OPEN;
   ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY <keystore-passphrase> CONTAINER = ALL;
   ADMINISTER KEY MANAGEMENT ADD SECRET "<credential>" FOR CLIENT 'HSM_PASSWORD' IDENTIFIED BY <keystore-
   passphrase> WITH BACKUP;
   ADMINISTER KEY MANAGEMENT CREATE AUTO_LOGIN KEYSTORE FROM KEYSTORE <path-to-keystorefolder>/<keystore-
   folder>/tde' IDENTIFIED BY KeystorePassword1;
   ```

4. Migrate from the keystore to HSM:

   CONNECT sysdba@FREEROOT

   ```
   ADMINISTER KEY MANAGEMENT SET ENCRYPTION KEY IDENTIFIED BY "<credential>" MIGRATE USING <keystore-
   passphrase> WITH BACKUP;
   ```

Use the Entrust WSOP `cklist-dynamic` utility to check that your encryption keys have been created in the WSOP server before proceeding.

```
% sudo -u oracle /opt/nfast/webservices/pkcs11/bin/cklist-dynamic -p SOFTCARD_PASSPHASE --library
/opt/nfast/webservices/pkcs11/lib/libpkcs11webservices.so | grep CKA_LABEL
```

5.6. Create master keys directly in an HSM keystore

The following procedure applies when there is no preexisting software keystore.

Repeat the following procedure for each database in which you want to create keys.

See About the HSM credential if you want to change a credential.

You must create the container (CDB) master key first. After the CDB master key has been created you have a choice of creating master keys for all the PDBs it contains in one operation, or else for each PDB individually.

The PDB(s) must use the same protection credential as the CDB.

5.6.1. Use the WALLET_ROOT and TDE_CONFIGURATION parameters

1. Set up the `WALLET_ROOT` and `TDE_CONFIGURATION` parameters as follows. You must set up the `WALLET_ROOT` parameter even if you do not use a keystore.

   CONNECT sysdba@FREEROOT

   ```
   ALTER SYSTEM SET WALLET_ROOT = '<path-to-keystore>' scope=SPFILE;
   ```

2. Bounce the database after setting up the `WALLET_ROOT` parameter.
3. Run the following command:

   ```
   ALTER SYSTEM SET TDE_CONFIGURATION = "KEYSTORE_CONFIGURATION=HSM" SCOPE=BOTH SID='*';
   ```

4. Bounce the database after setting up the `TDE_CONFIGURATION` parameter.

5.6.2. Create the CDB and then all PDB master keys in one operation

1. Select the protection method you require below, and run the SQL:

   CONNECT C##TESTER@FREE

   ```
   ALTER PLUGGABLE DATABASE ALL OPEN;

   --This will activate the credential
   ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "<credential>" CONTAINER=ALL;
   ```

2. Activate master keys for the CDB and all the PDBs in one operation:

   ```
   ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "<credential>" WITH BACKUP CONTAINER=ALL;
   ```

   

   Use the Entrust WSOP `cklist-dynamic` utility to check that your encryption keys have been created in the WSOP server before proceeding.

   ```
   % sudo -u oracle /opt/nfast/webservices/pkcs11/bin/cklist-dynamic -p SOFTCARD_PASSPHASE --library
   /opt/nfast/webservices/pkcs11/lib/libpkcs11webservices.so | grep CKA_LABEL
   ```

   Encrypt your database using tablespace encryption, column encryption, or both.

5.6.3. Create the CDB master key and a single PDB master key

1. Create the CDB master key:

   CONNECT C##TESTER@FREE

   a. Select the protection method you require below, and run the SQL:

      ```
      --This will activate the credential if it isn't already
      ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "<credential>";
      ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "<credential>" WITH BACKUP;
      ```

   b. Once you have created the CDB master key, you can repeat the following commands for creating a single PDB master key, for any PDB you select.

2. Create a single PDB master key:

   CONNECT PDB<k>TESTER@FREEPDB<k>

You must use the same protection method (credential) as the containing CDB. Run the SQL.

```
--If the PDB is already open, you don't need to do this.
ALTER PLUGGABLE DATABASE FREEPDB<k>> OPEN READ WRITE;

--If the keystore is already open, you don't need to do this.
ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "<credential>";

--Make the master key for the PDB you should be currently connected to.
ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "<credential>" WITH BACKUP;
```

Use the Entrust WSOP `cklist-dynamic` utility to check that your encryption keys have been created in the WSOP server before proceeding.

```
% sudo -u oracle /opt/nfast/webservices/pkcs11/bin/cklist-dynamic -p SOFTCARD_PASSPHASE --library
/opt/nfast/webservices/pkcs11/lib/libpkcs11webservices.so | grep CKA_LABEL
```

Encrypt your database using tablespace encryption, column encryption, or both.

5.7. Rekeying or key rotation

After you have established your HSM as the primary protector for your master encryption keys, for security reasons you may want to periodically replace the keys, or rekey. For your particular system, you can do this by following the instructions below.

The following subsections show how to perform a rekey in Oracle. After rekey, the new encryption keys should be immediately available and usable by the client that initiated the rekey.

5.7.1. Rekey when sharing keys between clients

Reconnect all users/applications on the client that are using the database encryption facilities.

Test your rekey arrangements in a safe environment before committing to a production environment. Transactions restricted to unencrypted data will not be affected by rekey operations.

5.7.2. Rekey with CDB and all the PDBs in one operation

CONNECT TESTER@FREE

The following instructions begin by assuming the required CDB has started, and required PDBs and HSM (keystore) to be already open.

```
ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "<credential>" WITH BACKUP CONTAINER=ALL;
```

5.7.3. Rekey with CDB only

The following instructions begin by assuming the required CDB has started and HSM (keystore) to be already open.

CONNECT TESTER@FREE

```
ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "<credential>" WITH BACKUP;
```

5.7.4. Rekey for a single PDB only

The following instructions begin by assuming the required CDB has started, the required PDB and HSM (keystore) to be already open.

CONNECT PDB<k>TESTER@FREEPDB<k>

```
--Make the master key for the PDB you should be currently connected to
ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "<credential>" WITH BACKUP;
```

Chapter 6. Troubleshooting

Oracle error messages may sometimes show error symptoms rather than the root cause. If you see an error you have not encountered before, search for further information online before attempting to resolve the error. If you remain unable to resolve the error, contact Oracle support.

If you edit an Oracle configuration file, use a simple text editor running on the host. Do not cut and paste the file contents from another file using a formatting editor, as it may insert hidden characters that are difficult to detect and which can stop the file from working. Entrust also suggests you avoid copying files onto a UNIX host via a Windows intermediary (this includes library files).

6.1. An SQL command is run, and there is no output, or an unexpected output or error occurs

1. Try reconnecting to the database.
2. If that does not work, try bouncing the database.

6.2. After a change to a configuration file, no resultant change in the database behavior is observed

1. Try reconnecting to the database.
2. If that does not work, try bouncing the database.

6.3. ORA-28367: wallet does not exist

1. Check that you have correctly installed and configured the Entrust WSOP PKCS#11 library.
2. Try reconnecting to the database.
3. Try bouncing the database.

6.4. ORA-28367: cannot find PKCS11 library

1. Ensure that you have correct permissions to use the `/opt/oracle/extapi/⋯` directory.

2. Check that you are using a library for the correct local architecture (32/64).
3. Check that you are using the appropriate Java version (32/64).
4. Refer to advice given above about editing Oracle files, or copying them.
5. Try reconnecting to the database.
6. Recopy the `libpkcs11webservices.so` library file to `/opt/oracle/extapi/`.

6.5. ORA-28353: failed to open wallet

1. Ensure that the HSM wallet pass phrase is correct.
2. Ensure that the softcard, the name and passphrase are correct and are separated by a **|** or a **:**.
3. If you have migrated from an Oracle wallet to an HSM wallet, you must update the passphrase.

6.6. ORA-28407: Hardware Security Module failed with PKCS#11 error CKR_FUNCTION_FAILED (%d)

1. This may be caused by Oracle defect 23528412. Contact Oracle support in order to obtain a patch for this defect.
2. Ensure that if a FIPS 140 Level 3 Security World is in use, an OCS card is inserted in the HSM slot. Make sure the WSOP server can see the OCS card.
3. Check that you are using the correct passphrase/credential to access the HSM.
4. In the WSOP server, If you are using an nShield Connect, use its front panel to check the Security World is loaded on to the HSM itself and is both **Initialized** and **Usable**.
5. Try restarting the Entrust hardserver in the WSOP server.

6.7. Encryption keys do not migrate correctly from a software keystore to an HSM (or vice-versa)

1. This may be caused by Oracle defect 17409174. Contact Oracle support in order to obtain a patch for this defect.

6.8. ORA-00600: internal error code

6.8.1. arguments: [kzthsmgmk: C_GenerateKey], [6], [],[], [], [], [], []

1. Ensure that you have added the `oracle` user to the `nfast` group. In some cases, you may have to re-login with the `oracle` user for this to take effect.
2. Ensure that if a FIPS 140 Level 3 Security World is in use, an OCS card is inserted in the HSM slot and visible in the WSOP server.

6.8.2. arguments: [ksqgel:null_parent], [], [],[], [], [], [], []

1. Sometimes occurs using encrypted tablespaces.
2. This may be caused by Oracle defect 21080143. Contact Oracle support in order to obtain a patch for this defect.

6.9. ORA-28374: Typed master key not found in wallet

1. Oracle software thinks there is a mismatch between encrypted object(s) and available master key(s). There is more than one possible cause for this and it is usually quite difficult to resolve. Contact Oracle support, or search for a solution online.
2. If all else fails, try and restore your system from backups.

6.10. ORA-12162: TNS: net service name is incorrectly specified

1. Check that you have correctly set the value for ORACLE_SID in your local environment.

Chapter 7. Appendix

7.1. Security worlds, key protection, and failure recovery

This section highlights some considerations when choosing security world and key protection options for use with the Entrust nShield Security World software. It focuses on recovery of security world authorization where a system has temporarily failed (for instance after a power outage) and is then returned to operation. This does not apply to other failure recovery functions. These considerations are applicable to security worlds, key protection and failure recovery for both standalone systems and database clusters. For more information on security worlds and key protection, see the *User Guide* for your HSM.

In the event of a temporary failure of the security world, there may be a consequent loss of:

- Credential authorization.
- Authorization if you are using a FIPS 140 Level 3 security world.

A credential authorization can be granted using either a Softcard or an OCS card, with passphrase. In the case of an OCS, a card must be always available in a valid HSM card reader in order to grant reauthorization after a failure, and permit automatic recovery.

Where FIPS authorization is required, this can be granted either by using an OCS card specifically for this purpose, or through an OCS card that is also used for credential authorization. A card from the OCS must be always available in a valid HSM card reader in order to grant reauthorization after a failure, and permit automatic recovery.

If you are using OCS cards through a RA secure channel, then if the secure channel is lost it must be reestablished before recovery using the OCS cards can begin. There is no automatic mechanism to reestablish the secure channel, which would have to be re-established manually, or through a user-defined script. For this reason, Entrust recommends that RA is not used for systems requiring automatic recovery.

Oracle auto-login facilities need to be set up to implement automatic recovery in the event of a temporary failure.

Never use ACS cards for FIPS authorization, because they do not

> support automatic recovery. Softcards or OCS must be members of the same security world.

The following table describes the authorization recovery behavior of the security world after a temporary outage.

security world type	Protection/ Credential	Stand-alone system	Database cluster
FIPS level 2	Module	Recovers automatically	Recovers automatically
	Softcard	Recovers automatically	Recovers automatically
	OCS	Use OCS for credential authorization: (1) Use 1/N quorum. Same passphrase for all cards (2) Leave an OCS card in HSM slot. Recovers automatically.	Use OCS for credential authorization: (1) Use 1/N quorum. Same passphrase for all cards (2) Leave an OCS card in slot of every HSM in cluster. Recovers automatically.
FIPS level 3	Module	Use OCS for FIPS authorization (only): Leave an OCS card in HSM slot. Recovers automatically	Use OCS for FIPS authorization (only): Leave an OCS card in slot of every HSM in cluster. Recovers automatically
	Softcard	Use OCS for FIPS authorization (only): Leave an OCS card in HSM slot. Recovers automatically	Use OCS for FIPS authorization (only): Leave an OCS card in slot of every HSM in cluster. Recovers automatically
	OCS	Use OCS for both credential and FIPS authorization: (1) Use 1/N quorum. Same passphrase for all cards. (2) Leave an OCS card in HSM slot. Recovers automatically.	Use OCS for both credential and FIPS authorization: (1) Use 1/N quorum. Same passphrase for all cards. (2) Leave an OCS card in slot of every HSM in cluster. Recovers automatically.

If you are using an OCS to facilitate automatic recovery of the security world:

- If you are using the OCS for credential authorization, all must be members of the same card set for the same credential, and the same passphrase must be assigned to every card in the set.
- If you are using the OCS for FIPS authorization purposes only, the quorum automatically defaults to 1/N, and (any) passphrase is ignored.

Authorization acquired through a persistent operator card does not automatically reinstate itself after loss due to a temporary failure.

7.2. About the HSM credential

The protection methods available with the Entrust HSM are, in order of enhanced authentication:

- Module: Encryption keys are protected by an security world protecting key in the HSM. This type of protection is not currently supported by WSOP PKCS#11 mechanism.
- Softcard: Encryption keys are protected by a named Softcard (software based) token key, a passphrase, and security world protecting key in the HSM. You can have many different Softcards, but each is singular and works on its own.
- OCS: Encryption keys are protected by the presence of a named physical token (OCS smartcard), an OCS token key, a passphrase, and security world protecting key in the HSM. OCS cards are usually part of a set of several OCS cards, or card set, and any member of the same card set protects the same encryption keys. You can have many different OCS card sets where each card set may protect different encryption keys. This type of protection is not currently supported by WSOP PKCS#11 mechanism.

The Softcard protection method must be set up within the Entrust HSM before they can be used by an Oracle database. See your HSM *User Guide* for details. Setting up the Softcard or OCS includes creating and naming the token(s), with a passphrase (see the *User Guide* for your HSM).

Within SQL scripts as used by Oracle, identify the protection method using a `<credential>`. WSOP only supports softcard protection:

Protection Type	Credential or <credential>
Softcard protection	<softcard-passphrase>|<softcard-name>

Oracle documentation gives the ordering <credential-name>|<credential-passphrase>. However, tests showed that the ordering <credential-passphrase>|<credential-name>` works.

Oracle SQL uses the separator symbol **|** or **:** to divide the <credential-passphrase> and <credentialname>. Hence the total Oracle SQL string for a credential comprises:

- Softcard protection: <credential-passphrase> + <separator> + <credential-name>.

In Entrust recommends the following restrictions on token names, or credentialname:

- Maximum length of 254 characters.
- ASCII 7-bit characters only, restricted to:

  A-Z, a-z, 0-9, $ - _ (no white space).

Entrust places the following restrictions on passphrases, or credential-passphrases:

- Maximum length of 254 characters.
- ASCII 7-bit characters only:

  A-Z, a-z, 0-9, ! @ # $ % ^ & * - _ + = [ ] { } | \ : ' , . ? / ` ~ " < > ( ) ; (no white space).

However, the Oracle SQL interface imposes further restrictions on top of the restrictions for what can comprise the string <credential-passphrase> + <separator> + <credential-name>, as follows:

- The total string length, including separator, can be no more than 30 characters. This leaves 29 characters for the <credential-passphrase> + <credential-name>.
- The symbols **|**, **:**, **"** and **'** cannot be used within the <credential-passphrase> or <credential-name>.

From the Oracle side, if:

- N is the length of the credential name.
- P is the length of the credential passphrase, then 2 ⇐ (N+P) ⇐ 29, where 1 ⇐

N ⇐ 28, and 1 ⇐ P ⇐ 28, assuming a minimum of one character for passphrase and name.

Permitted symbols are:

- `<credential-passphrase>`:

  A-Z, a-z, 0-9, ! @ # $ % ^ & * - _ + = [ ] { } \ , . ? / ~ < > ( ) ; (no white space)

- `<credential-name>`:

  A-Z, a-z, 0-9, $ - _ (no white space).

Use a passphrase of sufficient length to meet your current security requirements.

Oracle (wallet manager) states “Passwords must have a minimum length of eight characters and contain alphabetic characters combined with numbers or special characters".

When you are using the Softcard, an SQL script that uses the credential must get the `<credential-passphrase>` and `<credential-name>` exactly correct. After creating encryption keys or rekeying, then immediately use the Use the Entrust WSOP `cklist-dynamic` utility to check that your encryption keys have been created in the WSOP server before proceeding.

In the examples shown in this guide, credentials may be given descriptive names to make it clear what they are used for, such as `<keystore-credential>`. In practice, replace the descriptive names with the actual credential passphrases and names you are using.

Chapter 8. Additional resources and related products

8.1. Entrust digital security solutions

8.2. nShield product documentation