ORACLE

Integration Guide

Applicable Devices:

KMES Series 3

TABLE OF CONTENTS

[1] DOCUMENT INFORMATION

[1.1] DOCUMENT OVERVIEW

The purpose of this document is to provide information regarding the configuration of the Futurex KMES Series 3 for integration with Oracle TDE (Transparent Data Encryption.) For additional questions related to the KMES Series 3, see the relevant user guide.

[1.1.1] ABOUT ORACLE-TDE

From the Oracle Documentation: “Transparent data encryption enables you to encrypt sensitive data, such as credit card numbers, stored in table columns. Encrypted data is transparently decrypted for a database user who has access to the data. Transparent data encryption helps protect data stored on media in the event that the storage media or data file gets stolen.”

[1.2] COPYRIGHT AND TRADEMARK NOTICES

[1.3] TERMS OF USE

[2] OUR STORY

For over 39 years, Futurex has been a globally recognized provider of secure, robust, and cost-effective data encryption solutions for device manufacturers, financial institutions, major retail organizations, and others worldwide. More than 15,000 customers have trusted Futurex's innovative technology to provide market-leading solutions for the secure encryption, storage, and transmission of sensitive data. Futurex maintains an unyielding commitment to offering advanced, standards-compliant data encryption solutions, including:

- Hardware security modules for secure, reliable data encryption, information management, and key generation
- Remote key management and injection platforms
- Certificate Authority issuance and management
- Secure, hand-held devices for configuration, management, and compliant key loading
- High availability solutions for load balancing, monitoring, and disaster recovery
- Secure storage and access of sensitive data

Throughout every facet of our organization, we maintain a focus on providing exceptional customer service, best-in-class technology, and cost-effective solutions for our customers. Our dedication to meeting the growing business needs of our global customers and partners is exhibited by the continuous expansion of our innovative products and services. Futurex has established technology partnerships with Fortune 500 organizations across wide-ranging industries, allowing us to provide value through integration with numerous data transaction networks. Through our results-oriented engineering culture, we have provided organizations worldwide with custom solutions supporting aggressive times to market.

Our products satisfy the most rigorous security requirements, and as we move forward, Futurex will continue to be a global leader in the data security and electronic transaction industries by maintaining high performance standards, providing quality service, and expanding our best-in-class product suite.

[3] INTEGRATION WITH ORACLE-TDE

Oracle TDE integration with the KMES Series 3 requires the use of the Futurex PKCS #11 (FXPKCS11) library. Futurex provides users with a number of files to set up and configure the PKCS #11 library to allow Oracle database users to take advantage of the KMES Series 3 and store the Master Encryption Key used for TDE. By storing this key in a FIPS 140-2 Level 3 validated HSM, an additional layer of protection for data at rest is added.

The Master Encryption Key is used to encrypt the Oracle Table Keys, which in turn are used to encrypt or decrypt columns or tablespaces locally in the database. Each table has its own table key.

From the client application perspective, the encryption and decryption process is transparent, meaning there is no need to make any changes to the existing application.

The connection between the PKCS #11 library and the KMES Series 3 must be a TLS connection. Therefore, TLS/SSL certificates must be created (using either *OpenSSL* or an external CA,) to provide certificates for the KMES Series 3 and the Oracle Server where the PKCS #11 library is running. This guide is intended to provide the user with the required information to be able to configure and execute basic tests of the Oracle TDE environment when integrated with the KMES Series 3, the process for which is summarized in the following image.

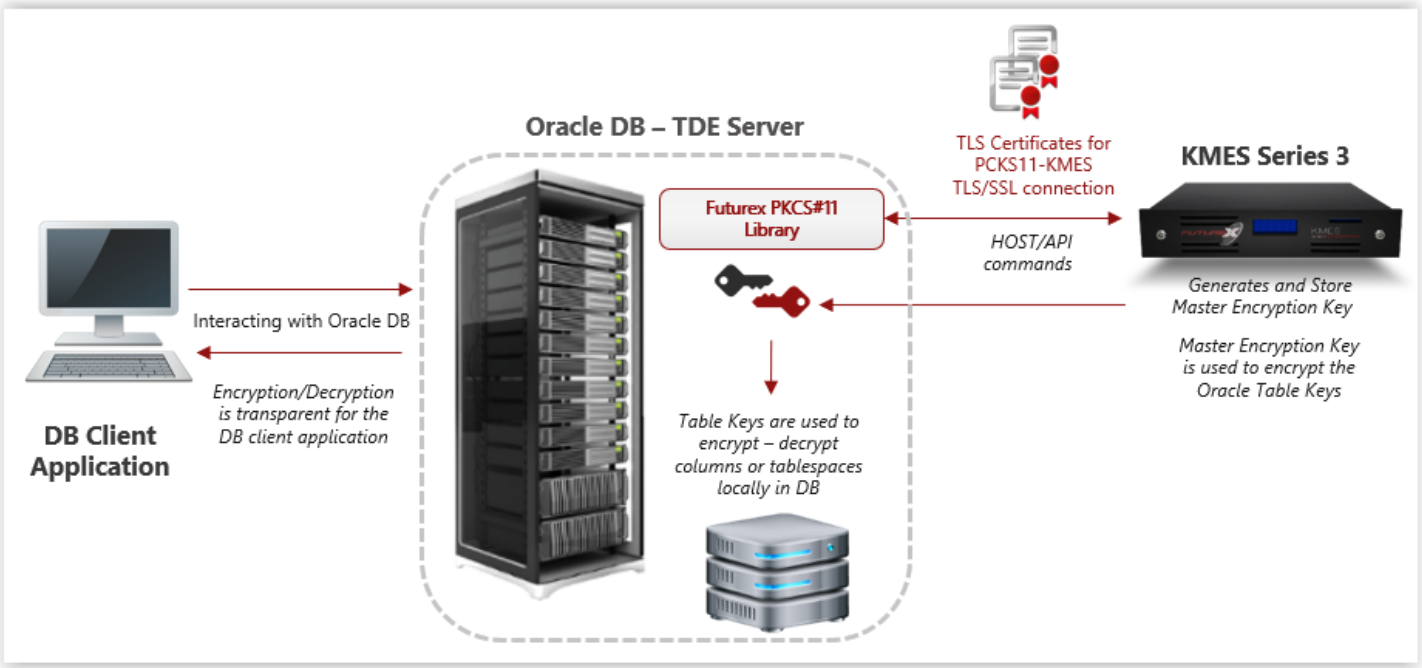


FIGURE: PROCESS OF ORACLE TDE INTEGRATION

[4] PRE-REQUISITES:

For this scenario to work, the following will be required:

1. Linux Server (CentOS 7 for this example – GUI not required): https://www.centos.org/download/
2. OpenSSL for Linux (optional)
3. Oracle Database 12c
4. Installation Guide for Oracle Database 12c: https://docs.oracle.com/en/database/oracle/oracle-database/12.2/ladbi/index.html
5. Oracle Guide to securing data with Oracle TDE (General concepts – how to enable TDE): https://docs.oracle.com/cd/B28359_01/network.111/b28530/asotrans.htm#g1011122
6. Oracle Guide to use HSM with TDE: https://docs.oracle.com/cd/E11882_01/network.112/e40393/asotrans.htm#ASOAG640
7. KMES Series 3 with minimum software version 6.1.2.4–ck6a85 and Futurex-signed TLS certificates preloaded
8. TLS client certificates to allow the PKCS #11 library to connect with the KMES Series 3. Refer to the section on the KMES Series 3 TLS certificates and PKCS certificates and the section on the PKCS #11 configuration file for additional details.
9. FXPKCS#11 library version 3.6-ck43f4

NOTE: For the purposes of this document, it is assumed that the Linux server is running and that the user has root permissions. It is also assumed that the Oracle database is already installed (https://docs.oracle.com/en/database/oracle/oracle-database/12.2/ladbi/index.html) and configured to support TDE: https://docs.oracle.com/cd/B28359_01/network.111/b28530/asotrans.htm#g1011122.

This guide is intended to show users how to configure the Futurex PKCS #11 library to be used as an interface for the Oracle TDE to connect to a KMES Series 3 HSM wallet, based on: https://docs.oracle.com/cd/E11882_01/network.112/e40393/asotrans.htm#ASOAG640.

[5] SETTING UP THE ORACLE ENVIRONMENT AND THE PKCS #11 LIBRARY

Based on: https://docs.oracle.com/cd/E11882_01/network.112/e40393/asotrans.htm#ASOAG640

[5.1] SET UP ORACLE DATABASE ENVIRONMENT

The "Oraenv" tool will set up the Oracle database environment for the current session and allow use of the sqlplus command. To set the Oracle environment, follow the next command sequence:

```
[root@localhost ~]# su oracle
[oracle@localhost root]$ cd
[oracle@localhost ~]$ . /usr/local/bin/oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /home/oracle/app/oracle
[oracle@localhost ~]$
```

[5.2] SET THE ENCRYPTION_WALLET_LOCATION

The ENCRYPTION_WALLET_LOCATION parameter specifies the location of the Oracle wallet. This parameter must be changed to reflect the fact that an HSM is to be used in place of the software wallet.

Use the following steps to set the ENCRYPTION_WALLET_LOCATION parameter:

- Open the sqlnet.ora file. This file is located in the *$ORACLE_HOME/network/admin* directory.

```
[oracle@localhost local]$ env | grep ORACLE_HOME
ORACLE_HOME=/home/oracle/app/oracle/product/12.2.0/dbhome_1
[oracle@localhost local]$
```

```
[oracle@localhost /]$ cd /home/oracle/app/oracle/product/12.2.0/dbhome_1/network
/admin/
[oracle@localhost admin]$ ls
listener.ora  samples  shrept.lst  sqlnet.ora  tnsnames.ora
[oracle@localhost admin]$
```

- Add the ENCRYPTION_WALLET_LOCATION parameter to the *sqlnet.ora* file, as follows:

ENCRYPTION_WALLET_LOCATION=(SOURCE=(METHOD=HSM))

If the ENCRYPTION_WALLET_LOCATION parameter is already present in the *sqlnet.ora* file, change the METHOD value to HSM:

```
ENCRYPTION_WALLET_LOCATION=
 (SOURCE=(METHOD=HSM)(METHOD_DATA=
 (DIRECTORY=/app/wallet)))
```

```
 GNU nano 2.3.1                File: sqlnet.ora

# sqlnet.ora Network Configuration File: /home/oracle/app/oracle/product/12.2.0$
# Generated by Oracle configuration tools.

NAMES.DIRECTORY_PATH= (TNSNAMES, EZCONNECT)
ENCRYPTION_WALLET_LOCATION=
   (SOURCE=(METHOD=HSM) (METHOD_DATA=
      (DIRECTORY=/app/wallet)))
```

Save and close the file.

NOTE: If a DIRECTORY value is present in the ENCRYPTION_WALLET_LOCATION parameter, make sure not to delete it. Although the KMES Series 3 does not require a DIRECTORY value, the value is used to locate the old software wallet when migrating to HSM-based Transparent Data Encryption. Also, the DIRECTORY value might be required by tools such as the Recovery Manager (RMAN) to locate the software wallet.

[5.3] COPY THE FUTUREX PKCS #11 LIBRARY TO THE CORRECT PATH

The Futurex PKCS #11 library file (libfxpkcs11.so) must be copied to the following path:

/opt/oracle/extapi/[32,64]/hsm/futurex/X.X.X/

Where X.X.X is the library version.

```
[oracle@localhost opt]$ cd /opt/oracle/extapi/64/hsm/futurex/3.6.0/
[oracle@localhost 3.6.0]$ ls
libfxpkcs11.so
[oracle@localhost 3.6.0]$
```

Copy the PKCS #11 Manager and configuration file (*fxpkcs11.cfg*) into the /etc directory

The TLS client certificates used for the TLS connection between the PKCS #11 library and the KMES Series 3 should be in a location accessible to the PKCS #11 library. The path is defined in the *fxpkcs11.cfg* file (as an example, *home/oracle/FXCerts*).

NOTE: Once the files are copied, use the *fxpkcs11.cfg* to set the configuration options. Refer to the sections on features required in the KMES Series 3 Internal HSM, the KMES Series 3 TLS certificates and PKCS certificates, and the PKCS #11 configuration file or the Futurex General-Purpose Technical Reference for more details about configuring PKCS #11.

NOTE: If a TLS/SSL connection is needed between the PKCS #11 library and the KMES Series 3, please refer to the section on the KMES Series 3 TLS certificates and PKCS certificates for more details.

[5.4] CONFIGURE THE KMES SERIES 3

Please refer to the final three sections for detailed instructions related to KMES Series 3 configuration, TLS certificate creation and configuration in the *fxpkcs11.cfg* file, and for a *fxpkcs11.cfg* example.

[5.5] TEST THE CONNECTION BETWEEN THE PKCS #11 LIBRARY AND THE KMES SERIES 3

Before testing the connection between the PKCS #11 library and the KMES Series 3, it is required to complete the KMES Series 3 configuration (including TLS certificates) described in the sections on features required in the KMES Series 3 Internal HSM, the KMES Series 3 TLS certificates and PKCS certificates, and the PKCS #11 configuration file.

To test the connection, complete the following steps:

Go to /etc and run: ./PKCS11Manager

```
[root@localhost etc]# ./PKCS11Manager
```

If everything is functioning properly, the following menu will appear:

```
[2019-07-10 00:16:08] | DEBUG | 7F77BCA8B880 | C_GetSlotList: Called.
[2019-07-10 00:16:08] | INFO  | 7F77BCA8B880 | C_GetSlotList: 1 slots found.
[2019-07-10 00:16:08] | DEBUG | 7F77BCA8B880 | C_OpenSessionRecursive: Called wi
th slot #0.
[2019-07-10 00:16:08] | INFO  | 7F77BCA8B880 | C_OpenSessionRecursive: Created s
ession 1 on slot 0.

Main Menu
    1. Print Library/Token Info

    2. Generate Key
    3. Input Clear Key

    4. Find Objects
    5. Modify Objects
    6. Delete Objects

    7. Generate Random Data

    8. Login
    9. Logout

    0. Exit
```

Log in to the KMES Series 3 using option 8. It will ask for the password. This is the password created in the KMES Series 3 for the Crypto1 user.

```
Password - Input Mode
    1. Text
    2. Hex Encoded
    3. File
    4. None

    0. Exit
```

Select “1” and type in the password. If everything has been set up correctly, there should be no errors. To validate, see the /tmp/fxpkcs11.log file.

[5.6] TEST THE KEY CREATION IN THE KMES SERIES 3 USING THE PKCS #11 MANAGER

Users can test to see if it is possible to create a Master Key in the key group defined for TDE in the KMES Series 3, in this case, OracleTDE.

To test this, complete the following steps:

- Run the PKCS #11 Manager, as shown in the section on configuring the KMES Series 3.
- Log in to the KMES Series 3, as shown in the section on configuring the KMES Series 3.
- On the menu after login, select option “2” (Generate Key).

Follow these options for Key Generation:

- Create Key or Key Pair: AES
- Bits: 256
- Token: Yes (NO is not allowed by the KMES Series 3.)
- Private: (Could be YES or NO)
- Sensitive: (Could be YES or NO)
- Modifiable: (Could YES or NO)
- Label: Select “1” -> Type the desired key label (in this example, TestKeyTDE1)
- Key Usage: ED
- For the rest of the parameters: none

If everything processed correctly, users can then proceed to KMES Series 3 -> *Keys* -> and see if the new key (*TestKeyTDE1*) was created under the key group previously created when copying the Futurex PKCS #11 library to the correct path (*OracleTDE* in this example).

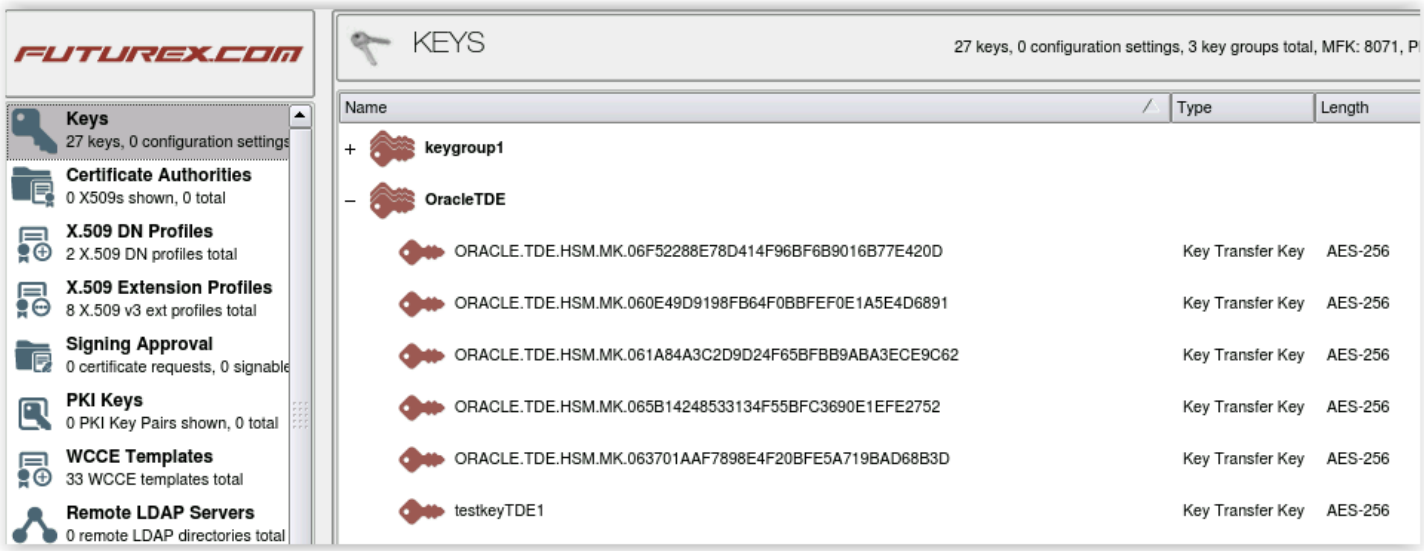


FIGURE: NEW KEY IN KEY GROUP IN KMES SERIES 3

[6] TESTING ORACLE TDE – KMES SERIES 3

[6.1] SETTING A MASTER ENCRYPTION KEY FOR HSM-BASED ENCRYPTION

To start using HSM-based encryption, it is necessary to create a Master Encryption Key that will be stored inside the HSM. The Master Encryption Key is used to encrypt or decrypt table keys inside the HSM.

Set Oracle environment and start the Oracle instance:

```
[root@localhost ~]# su oracle
[oracle@localhost root]$ cd
[oracle@localhost ~]$ . /usr/local/bin/oraenv
ORACLE_SID = [oracle] ? orcl
The Oracle base has been set to /home/oracle/app/oracle
[oracle@localhost ~]$ sqlplus / as sysdba

SQL*Plus: Release 12.2.0.1.0 Production on Fri Jul 19 10:08:43 2019

Copyright (c) 1982, 2016, Oracle.  All rights reserved.

Connected to an idle instance.

SQL> startup
ORACLE instance started.

Total System Global Area  771751936 bytes
Fixed Size                  8797536 bytes
Variable Size             608174752 bytes
Database Buffers          150994944 bytes
Redo Buffers                3784704 bytes
Database mounted.
Database opened.
SQL>
```

Use the following command to create the Master Encryption Key:

SQL> ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY "Cryptouser_Password"

```
SQL> ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY "safest";

System altered.

SQL>
```

NOTE: The *Cryptouser_Password* is the password set for the *cryptouser* created for the KMES Series 3 in the section on copying the Futurex PKCS #11 library to the correct path.

NOTE: If the user is already using transparent data encryption and not using an HSM, they must use the MIGRATE USING wallet_password clause in the preceding command (wallet_password is the password required to open an existing Oracle wallet on the file system.) This decrypts the existing table keys and re-

encrypts them with the newly created HSM-based Master Encryption Key.

SQL> ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY " CryptouserPassword " [MIGRATE USING "wallet_ password"]

NOTE: If the database contains columns encrypted with a public key, the columns are decrypted and re-encrypted with the Oracle table key, which is encrypted/decrypted with the AES symmetric key generated by HSM-based transparent data encryption.

Generated keys in the KMES Series 3 will look like the following image:

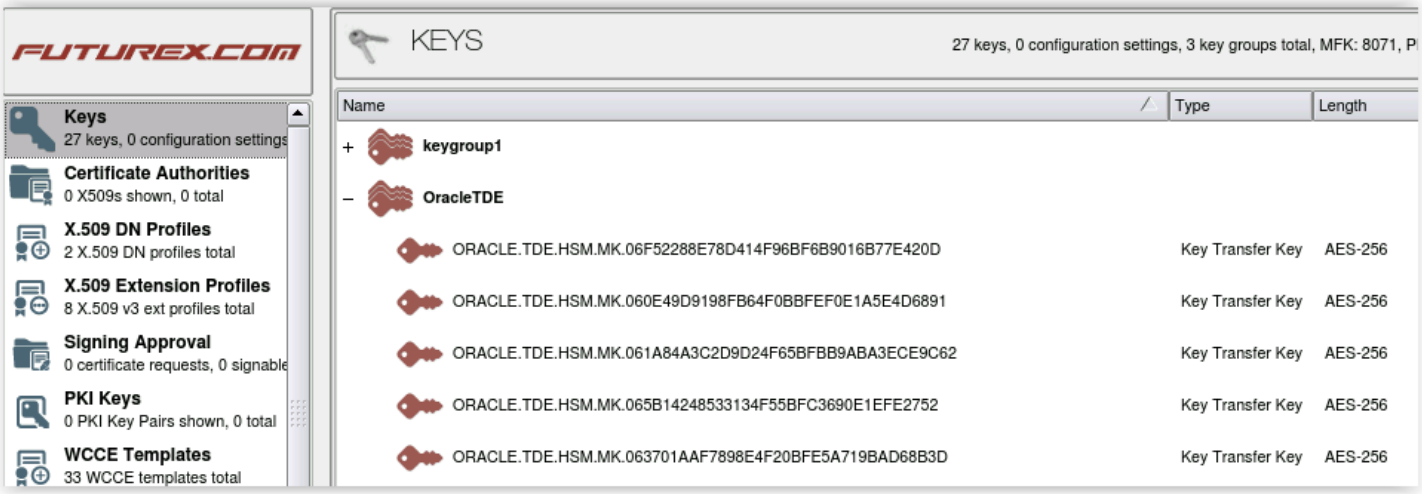


FIGURE: GENERATED KEYS IN THE KMES SERIES 3

[6.2] ENSURE THE KMES SERIES 3 IS AVAILABLE

The security administrator must make sure that the KMES Series 3 is accessible to the database before any encryption or decryption can be performed. This is analogous to opening the Oracle wallet. Use the following command to make the HSM accessible:

SQL> ALTER SYSTEM SET ENCRYPTION WALLET OPEN IDENTIFIED BY "user_Id:password"

NOTE: Access to the HSM needs to be reenabled every time the database instance is restarted. The security administrator can disable access to the HSM using the ALTER SYSTEM SET ENCRYPTION WALLET CLOSE IDENTIFIED BY "user_Id:password" command.

```
SQL> ALTER SYSTEM SET ENCRYPTION WALLET OPEN IDENTIFIED BY "safest";

System altered.

SQL>
```

```
SQL> ALTER SYSTEM SET ENCRYPTION WALLET CLOSE IDENTIFIED BY "safest";

System altered.

SQL>
```

[6.3] RESETTING THE MASTER ENCRYPTION KEY

The same command is used for the MEK generation [6.1].

SQL> ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY "Cryptouser_Password"

```
SQL> ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY "safest";

System altered.

SQL>
```

[7] FEATURES REQUIRED IN THE KMES SERIES 3 INTERNAL HSM

[7.1] CRYPTO OPERATOR GROUP AND USERS

Users must create a new group (as an example: “Crypto Operators” which will include a “crypto_operator” user and their credentials.) This functionality was released in the KMES Series 3 version 6.1.2.4 – ck6a85. Any user created within the “Crypto Operator” group can be used by the PKCS #11 library to access the KMES Series 3.

NOTE: This “crypto user” will be used later in the *fxpkcs11.cfg* file to allow the PKCS #11 library to connect to the KMES Series 3.

To define the “Crypto Operators” group and user(s), complete the following sequence:

- Log in to KMES Series 3 with a user with privileges to Add/Modify user groups.
- Go to *Users -> Admin Group*. Right-click “Add Group.”
- Set:
    - *Group name*: CryptoOperators
    - Number of users required to log in:1
    - *User type*: Normal Users
    - *Members can Authenticate to*: Client, Host API
- Set at least the following permissions:
    - Manage Keys -> All
    - Manage Encryption Device Groups -> All
    - Perform Cryptographic operations -> All
- Go to *Users -> CryptoOperator Group.* Right-click “Add User”
- Create the username: crypto1 (as an example.)
- Create the password for this user: safest (as an example.) This password will be required later.

The KMES Series 3 should now look something like this:

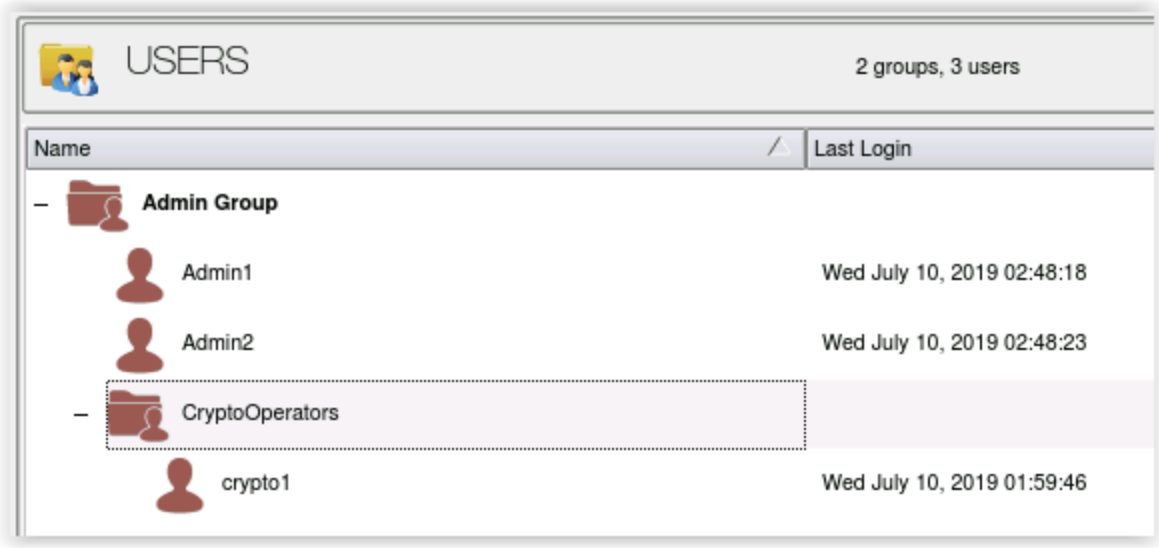


FIGURE: CRYPTOOPERATORS IN KMES SERIES 3

[7.2] CREATE THE KEY GROUP FOR ORACLE TDE KEYS

A key group should be created for Oracle TDE – KMES Series 3 integration. This key group will contain the created or renewed Master Keys for TDE.

NOTE: The name of this key group could be anything, but the same name will be used later in the *fxpkcs11.cfg* file.

To create the key group, complete the following steps:

- Log in to KMES Series 3 with a user with privileges to Add/Modify Key Groups.
- Go to *Keys*. Right-click "Add Key Group."
- Set the following details:
    - *Name:* OracleTDE, (as an example)
    - *OwnerGroup:* CryptoOperators (The name of the previously created user group)
    - Permissions:
    - AdminGroup: Add, implicit
    - CryptoOperator: none, implicit

The KMES Series 3 should now look like this:

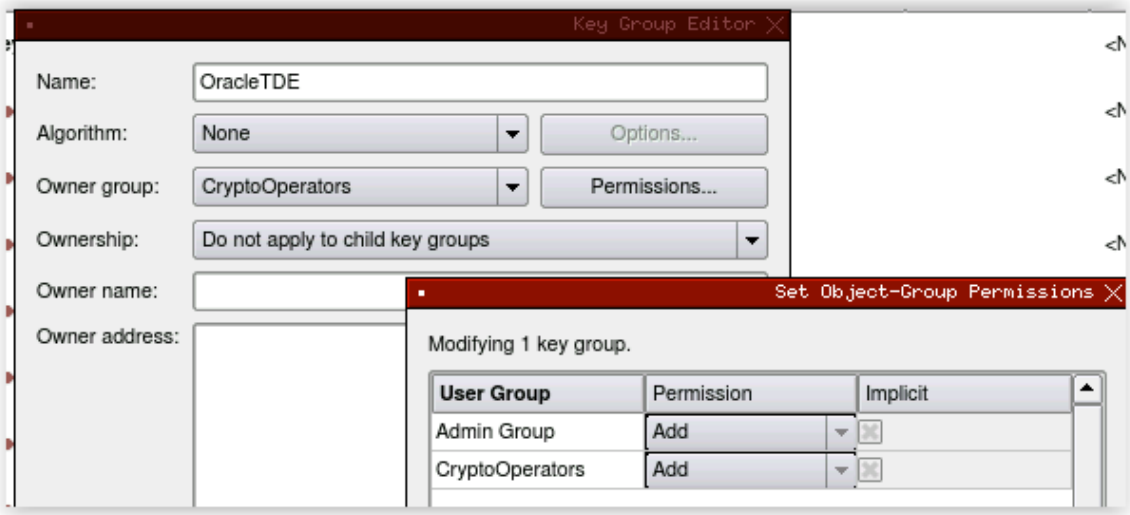


FIGURE: SETTING OBJECT-GROUP PERMISSIONS IN KMES SERIES 3

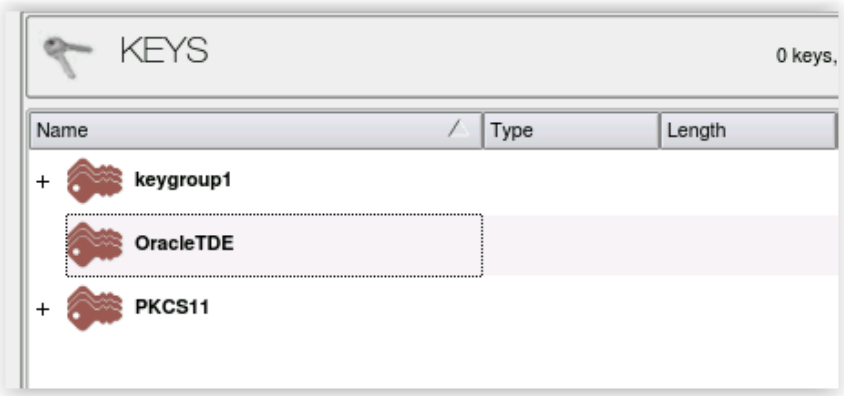


FIGURE: KEY GROUPS IN KMES SERIES 3

The crypto user ("crypto1" in this example,) as well as the key group name ("OracleTDE" in this example) will be required in the following fields in the *fxpkcs11.cfg* file:

```
# KMES in first slot
<KMS>
    # Which PKCS11 slot
    <SLOT>                  0                       </SLOT>

    # Login username
    <CRYPTO-OPR>            crypto1                 </CRYPTO-OPR>

    # Key group name
    <KEYGROUP-NAME>         OracleTDE               </KEYGROUP-NAME>
```

[7.3] ENABLE TLS CONNECTIONS IN THE KMES SERIES 3

To enable TLS connections in the KMES Series 3 for the PKCS #11 library, complete the following steps:

- Log in to the KMES Series 3 with privileges to modify network options.
- Go to *Configuration* -> *Network Options* -> TLS/SSL Settings tab.
- Select System/Host API in the dropdown combo.
- Set the following configuration:

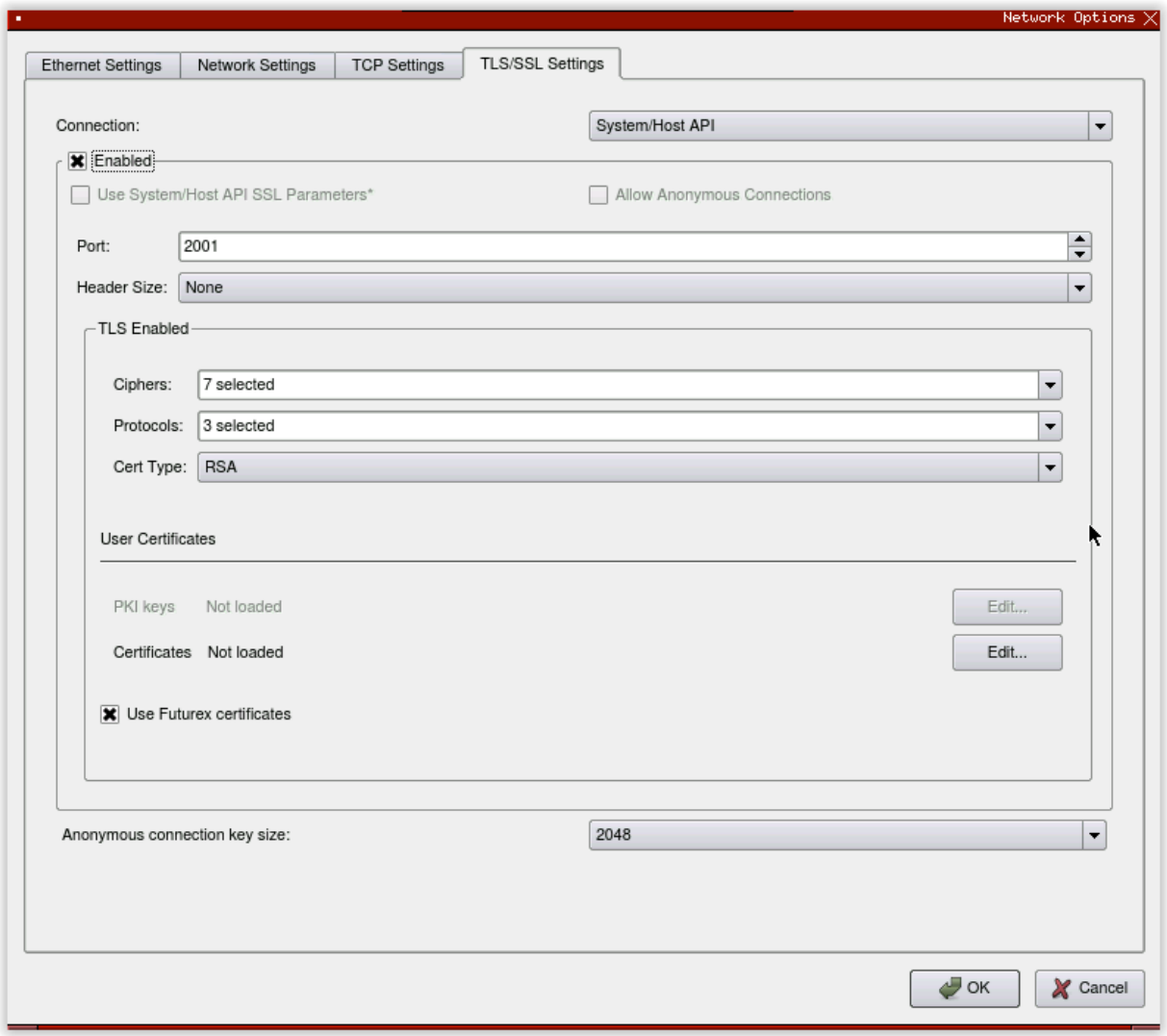


FIGURE: PROCESS TO ENABLE TLS/SSL SETTINGS

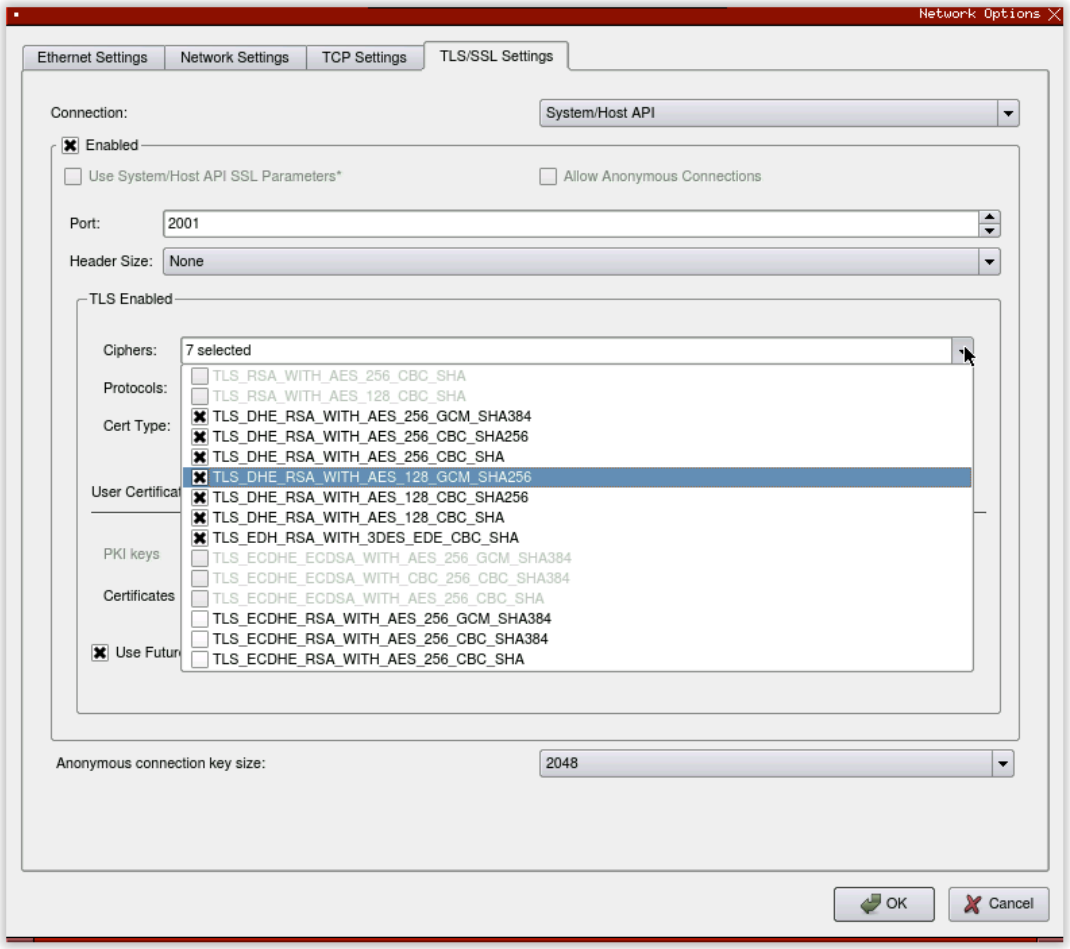


FIGURE: SELECT PICTURED TLS CONNECTION

NOTE: The port will be required in the *fxpkcs11.cfg* file.

```
# KMES in first slot
<KMS>
    # Which PKCS11 slot
    <SLOT>                  0                       </SLOT>

    # Login username
    <CRYPTO-OPR>            crypto1                 </CRYPTO-OPR>

    # Key group name
    <KEYGROUP-NAME>         OracleTDE               </KEYGROUP-NAME>

    # Connection information
    <ADDRESS>               10.0.5.134    </ADDRESS>
    <PROD-PORT>             2001                    </PROD-PORT>
    <PROD-TLS-ENABLED>      YES                      </PROD-TLS-ENABLED>
    <PROD-TLS-ANONYMOUS>    NO                       </PROD-TLS-ANONYMOUS>
```

FIGURE: TLS IS ENABLED

[7.4] ENABLE THE HOST API COMMANDS REQUIRED FOR THE ORACLE TDE OPERATION

Because the connection to the PKCS #11 library will use the Host API port, users must define which commands will be enabled for execution by the PKCS #11 library. To set the enabled commands, complete the following steps:

- Log in to KMES Series 3 with privileges to modify Host API configuration.
- Go to *configuration* -> *Host API Options* -> check the commands that are required to be enabled.

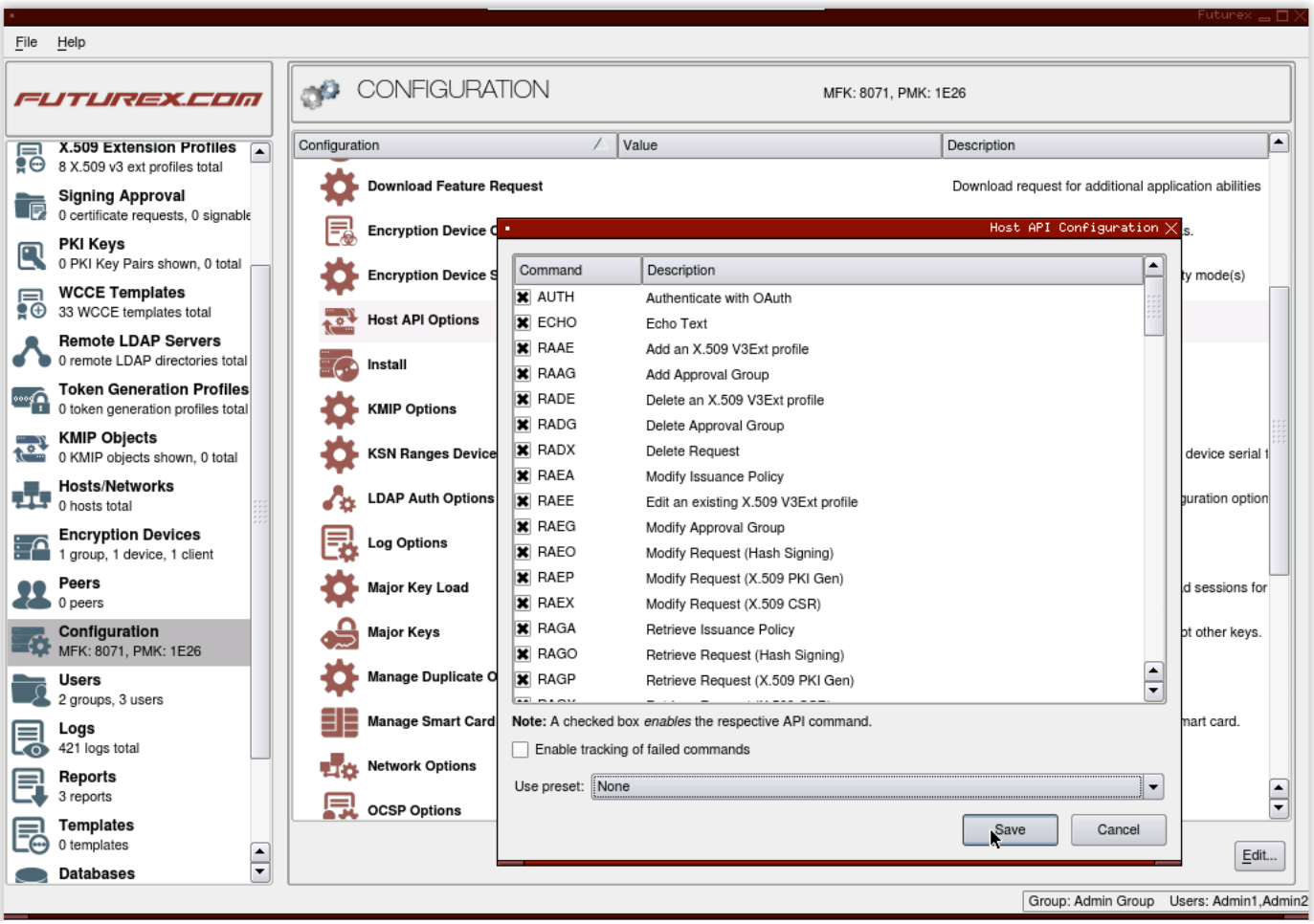


FIGURE: HOST API IS ENABLED

[8] KMES SERIES 3 (SERVER) TLS CERTIFICATES AND PKCS #11 (CLIENT) CERTIFICATES

When an application uses the PKCS #11 library to connect to the KMES Series 3, the application is considered the "client" and the KMES Series 3 is considered the "server". In a TLS/SSL connection, both peers (client and server) must have their own TLS certificates in order to trust each other. Each peer should generate its own private key - public key pair, as well as its own Certificate Signing Request (CSR). This signing request must be sent to a 3rd Party CA (Certificate Authority). The CA returns each peer a CA signed certificate. The CA also sends each peer the CA certificate tree or CA root public certificate used to sign the peer certificates. This CA root public certificate will be used by one peer to validate the authenticity of the signed certificate sent by the other peer at the connection time. Once this validation process is completed, the peers can share the secret key that will be used for data encryption.

In the simplest approach, to establish a TLS connection between a client with the PKCS #11 library and the KMES Series 3, consider using a Futurex certificate authority to sign the CSRs from both the KMES Series 3 and the client.

The KMES Series 3 server should be signed with the Futurex certificates. (This can be verified in *HSM web portal-> SSL/TLS*.)

Admin TLS port:

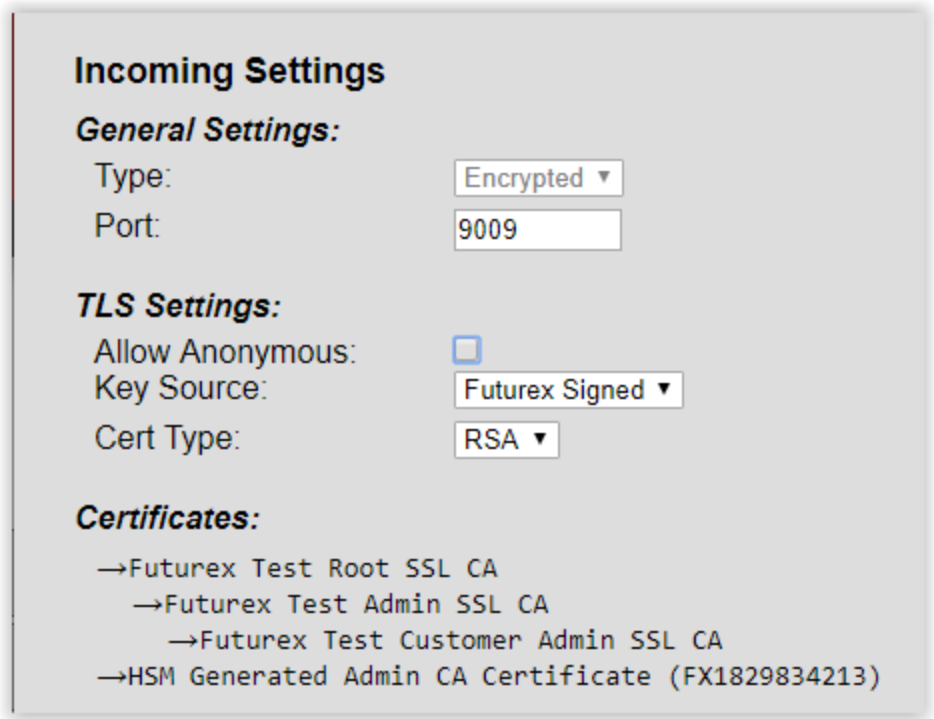


FIGURE: ADMIN TLS PORT

Excrypt (Production) TLS port:

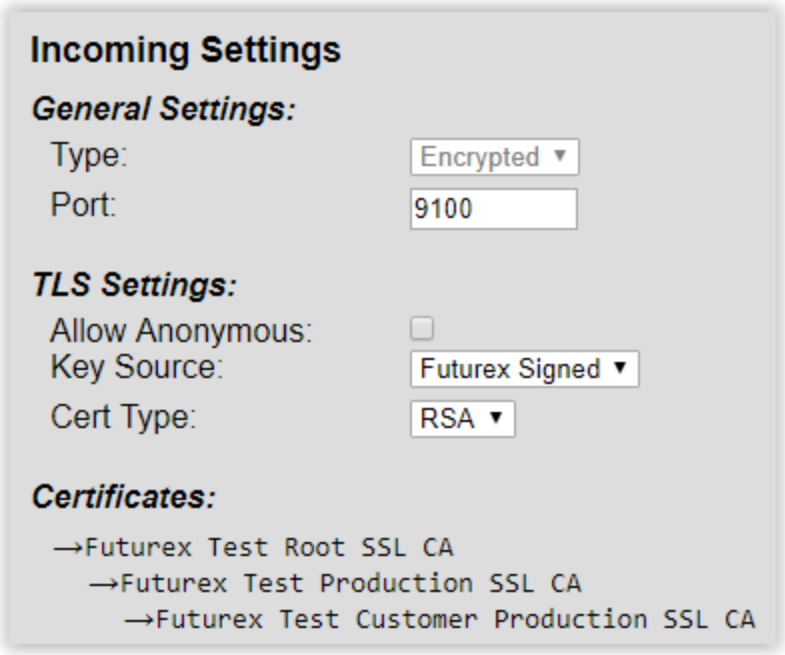


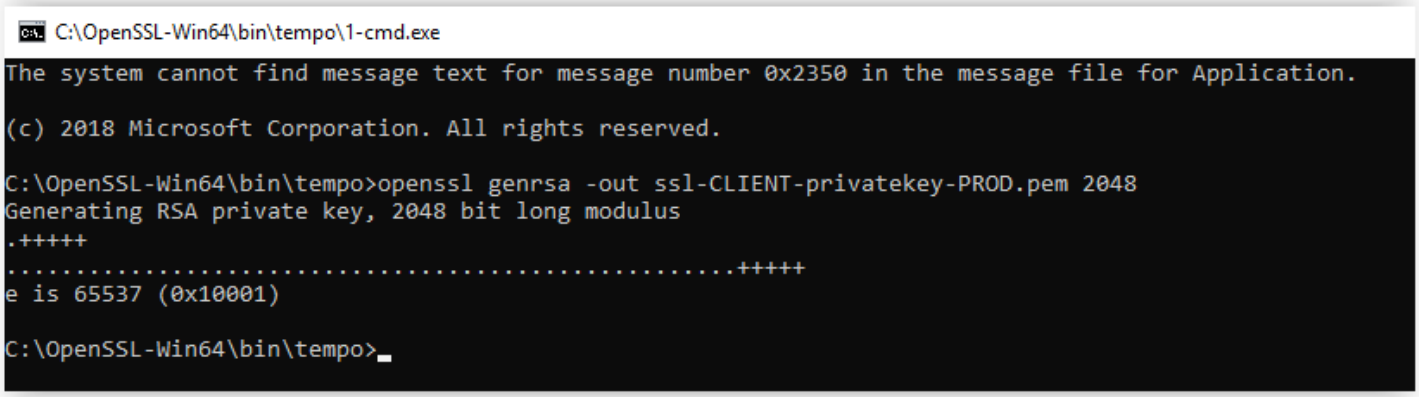
FIGURE: EXCRYPT (PRODUCTION) TLS PORT

Assuming the KMES Series 3 is already signed with Futurex certificates, the next step would be to obtain the client TLS certificates. In this case, the client would be the PKCS #11 library.

The following commands will describe the process to generate the Client Certificate Signing Request (CSR) using OpenSSL. These CSRs must be sent to support@futurex.com to be signed by the Futurex certificate authority. The returned signed certificate, as well as the tree of certificate used by the CA to sign the CSRs, are the certificates that will be referenced in the *fxpkcs11.cfg* configuration file.

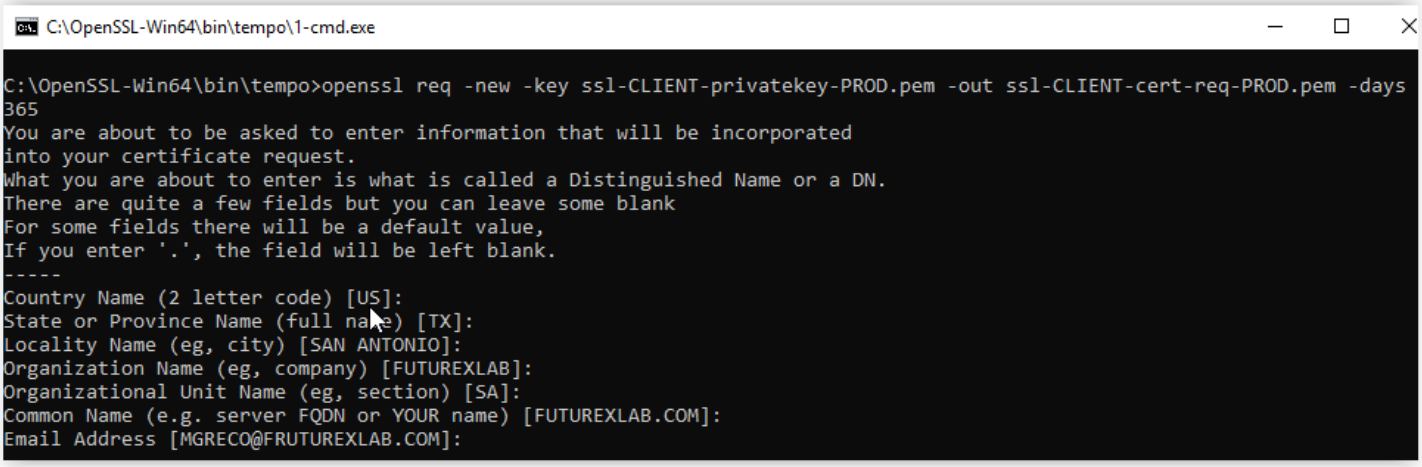
[8.1] CREATE A CLIENT PUBLIC-PRIVATE KEY PAIR

openssl genrsa -out ssl-CLIENT-privatekey-PROD.pem 2048



[8.2] CREATE A CLIENT CERTIFICATE REQUEST (CSR) REQUEST TO CA:

openssl req -new -key ssl-CLIENT-privatekey-PROD.pem -out ssl-CLIENT-cert-req-PROD.pem -days 365



[8.3] OUTPUT FILES

The output file, *ssl-CLIENT-cert-req-PROD.pem*, must be sent to support@futurex.com to get the corresponding signed certificate and the CA tree use to sign. Returned files should look like the following:

Signed cert: *PKCS11_PROD_Signed.PEM*

CA tree:

Fx1.pem (Futurex_Test_Root_SSL_CA.pem)

Fx2.pem (Futurex_Test_Production_SSL_CA.pem)

Fx3.pem (Futurex_Test_Customer_Production_SSL_CA.pem)

These files should be properly configured in the PKCS#11.cfg file to allow the TLS connection.

[8.3.1] PRODUCTION PORT:

Connection information

```
<ADDRESS>                              10.0.5.134                                  </ADDRESS>

<PROD-PORT>                            2001                                        </PROD-PORT>

<PROD-TLS-ENABLED>              YES                                                </PROD-TLS-ENABLED>

<PROD-TLS-ANONYMOUS>            NO                                                 </PROD-TLS-ANONYMOUS>

<PROD-TLS-CA>           /home/oracle/FXcerts/Fx1.pem                               </PROD-TLS-CA>

<PROD-TLS-CA>           /home/oracle/FXcerts/Fx2.pem                               </PROD-TLS-CA>

<PROD-TLS-CA>           /home/oracle/FXcerts/Fx3.pem                               </PROD-TLS-CA>

<PROD-TLS-CERT>         /home/oracle/FXcerts/PKCS11_PROD_Signed.pem                </PROD-TLS-CERT>

<PROD-TLS-KEY>         /home/oracle/FXcerts/ssl-CLIENT-privatekey-PROD.pem         </PROD-TLS-KEY>

#<PROD-TLS-KEY>         certs/HostPKI.p12                                          </PROD-TLS-KEY>

#<PROD-TLS-KEY-PASS>            safest                                             </PROD-TLS-KEY-PASS>
```

```
# KMES in first slot
<KMS>
    # Which PKCS11 slot
    <SLOT>                  0                       </SLOT>

    # Login username
    <CRYPTO-OPR>            crypto1                 </CRYPTO-OPR>

    # Key group name
    <KEYGROUP-NAME>         OracleTDE               </KEYGROUP-NAME>

    # Connection information
    <ADDRESS>               10.0.5.134    </ADDRESS>
    <PROD-PORT>             2001                   </PROD-PORT>
    <PROD-TLS-ENABLED>      YES                     </PROD-TLS-ENABLED>
    <PROD-TLS-ANONYMOUS>    NO                      </PROD-TLS-ANONYMOUS>
    <PROD-TLS-CA>           /home/oracle/FXcerts/Fx1.pem       </PROD-TLS-CA>
    <PROD-TLS-CA>           /home/oracle/FXcerts/Fx2.pem     </PROD-TLS-CA>
    <PROD-TLS-CA>           /home/oracle/FXcerts/Fx3.pem     </PROD-TLS-CA>
    <PROD-TLS-CERT>         /home/oracle/FXcerts/PKCS11_PROD_Signed.pem       </PROD-TLS-CERT>
    <PROD-TLS-KEY>          /home/oracle/FXcerts/ssl-CLIENT-privatekey-PROD.pem      </PROD-TLS-KEY>
    #<PROD-TLS-KEY>          certs/HostPKI.p12       </PROD-TLS-KEY>
    #<PROD-TLS-KEY-PASS>     safest                  </PROD-TLS-KEY-PASS>

    # YES = This is communicating through a Guardian
    <FX-LOAD-BALANCE>       NO                      </FX-LOAD-BALANCE>
</KMS>
```

FIGURE: PRODUCTION PORT

[9] THE PKCS #11 CONFIGURATION FILE

The PKCS#11.cfg allows the user to configure the PKCS #11 library to connect to the KMES Series 3. The following is a configuration example. For additional details, see the Futurex General-Purpose Technical Reference.

```
#
# Futurex PKCS11 Cryptoki Configuration File
#
# General configuration
<CONFIG>

# Log Configuration
<LOG-MODE>                      DEBUG3              </LOG-MODE> # NONE, ERROR, INFO, TRAFFIC, DEBUG,
DEBUG2, DEBUG3, ALL
<LOG-FILE>                      /tmp/fxpkcs11.log   </LOG-FILE>
<LOG-TRAFFIC>                     YES                 </LOG-TRAFFIC> # Debug binary only
<LOG-TEMPLATES>                YES                </LOG-TEMPLATES> # Debug binary only

# Ping every 60 seconds to maintain connection
<KEEPALIVE-EXCRYPT-ENABLED>     YES             </KEEPALIVE-EXCRYPT-ENABLED>
<KEEPALIVE-EXCRYPT-INTERVAL>    60              </KEEPALIVE-EXCRYPT-INTERVAL>

# Each session will have its own connection to the HSM
# Will require each session to C_Login separately
<UNIQUE-CONNECTIONS>        NO                  </UNIQUE-CONNECTIONS>

# Reload key information cache if C_FindObjectInit finds no results
<RELOAD-KEYS-ON-ERROR>      NO                  </RELOAD-KEYS-ON-ERROR>

# Milliseconds to wait before an Excrypt request times out
<EXCRYPT-TIMEOUT>           5000                </EXCRYPT-TIMEOUT>
<SANITIZE-LABELS>               NONE              </SANITIZE-LABELS>

# Milliseconds to wait before a TLS handshake times out
<TLS-HANDSHAKE-TIMEOUT>     20000               </TLS-HANDSHAKE-TIMEOUT>

# Default key usage when none is specified in template
<DEFAULT-SYMMETRIC-USAGE>   ENCRYPT | DECRYPT   </DEFAULT-SYMMETRIC-USAGE>
<DEFAULT-ASYMMETRIC-USAGE>  SIGN | VERIFY       </DEFAULT-ASYMMETRIC-USAGE>
</CONFIG>

# KMES in first slot
<KMS>

# Which PKCS11 slot
<SLOT>                  0                       </SLOT>

# Login username
<CRYPTO-OPR>            crypto1                 </CRYPTO-OPR>

# Key group name
<KEYGROUP-NAME>         OracleTDE               </KEYGROUP-NAME>
```

```
# Connection information
<ADDRESS>                                    10.0.5.134      </ADDRESS>
<PROD-PORT>                                  2001                    </PROD-PORT>
<PROD-TLS-ENABLED>                      YES                      </PROD-TLS-ENABLED>
<PROD-TLS-ANONYMOUS>            NO                               </PROD-TLS-ANONYMOUS>
<PROD-TLS-CA>           /home/oracle/FXcerts/Fx1.pem       </PROD-TLS-CA>
<PROD-TLS-CA>           /home/oracle/FXcerts/Fx2.pem     </PROD-TLS-CA>
<PROD-TLS-CA>           /home/oracle/FXcerts/Fx3.pem     </PROD-TLS-CA>
<PROD-TLS-CERT>         /home/oracle/FXcerts/PKCS11_PROD_Signed.pem       </PROD-TLS-CERT>
<PROD-TLS-KEY>          /home/oracle/FXcerts/ssl-CLIENT-privatekey-PROD.pem      </PROD-TLS-KEY>
#<PROD-TLS-KEY>                        certs/HostPKI.p12       </PROD-TLS-KEY>
#<PROD-TLS-KEY-PASS>                   safest                  </PROD-TLS-KEY-PASS>
# YES = This is communicating through a Guardian
<FX-LOAD-BALANCE>       NO                      </FX-LOAD-BALANCE>
</KMS>
```

NOTE: Once the fxpkcs11.cfg is edited, run the *PKCS #11 Manager* (/etc/ ./PKCS11Manager)to test the connection against the HSM and check the /tmp/fxpkcs11.log for errors and information. It will be also possible to execute some tasks from the PKCS #11 Manager Main Menu.

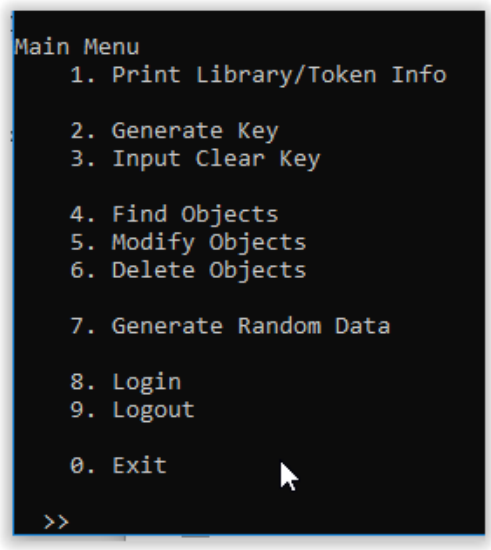


FIGURE: PKCS #11 MANAGER MAIN MENU

ENGINEERING CAMPUS

864 Old Boerne Road

Bulverde, Texas, USA 78163

Phone: +1 830-980-9782

+1 830-438-8782

E-mail: info@futurex.com

XCEPTIONAL SUPPORT

24x7x365

Toll-Free: 1-800-251-5112

E-mail: support@futurex.com

SOLUTIONS ARCHITECT

E-mail: solutions@futurex.com