

ORACLE

Integration Guide

Applicable Devices: KMES Series 3

THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION PROPRIETARY TO FUTUREX, LP. ANY UNAUTHORIZED USE, DISCLOSURE, OR DUPLICATION OF THIS DOCUMENT OR ANY OF ITS CONTENTS IS EXPRESSLY PROHIBITED.



TABLE OF CONTENTS

[1] DOCUMENT INFORMATION	. 3
[1.1] DOCUMENT OVERVIEW	. 3
[1.2] COPYRIGHT AND TRADEMARK NOTICES	. 3
[1.3] TERMS OF USE	. 3
[2] OUR STORY	. 4
[3] INTEGRATION WITH ORACLE-TDE	. 5
[4] PRE-REQUISITES:	. 6
[5] SETTING UP THE ORACLE ENVIRONMENT AND THE PKCS #11 LIBRARY	. 7
[5.1] SET UP ORACLE DATABASE ENVIRONMENT	. 7
[5.2] SET THE ENCRYPTION_WALLET_LOCATION	. 7
[5.3] COPY THE FUTUREX PKCS #11 LIBRARY TO THE CORRECT PATH	. 8
[5.4] CONFIGURE THE KMES SERIES 3	. 9
[5.5] TEST THE CONNECTION BETWEEN THE PKCS #11 LIBRARY AND THE KMES SERIES 3	. 9
[5.6] TEST THE KEY CREATION IN THE KMES SERIES 3 USING THE PKCS #11 MANAGER	. 10
[6] TESTING ORACLE TDE – KMES SERIES 3	.11
[6.1] SETTING A MASTER ENCRYPTION KEY FOR HSM-BASED ENCRYPTION	. 11
[6.2] ENSURE THE KMES SERIES 3 IS AVAILABLE	. 12
[6.3] RESETTING THE MASTER ENCRYPTION KEY	. 13
[7] FEATURES REQUIRED IN THE KMES SERIES 3 INTERNAL HSM	. 14
[7.1] CRYPTO OPERATOR GROUP AND USERS	. 14
[7.2] CREATE THE KEY GROUP FOR ORACLE TDE KEYS	. 15
[7.3] ENABLE TLS CONNECTIONS IN THE KMES SERIES 3	. 17
[7.4] ENABLE THE HOST API COMMANDS REQUIRED FOR THE ORACLE TDE OPERATION	. 19
[8] KMES SERIES 3 (SERVER) TLS CERTIFICATES AND PKCS #11 (CLIENT) CERTIFICATES	.20
[8.1] CREATE A CLIENT PUBLIC-PRIVATE KEY PAIR	. 21
[8.2] CREATE A CLIENT CERTIFICATE REQUEST (CSR) REQUEST TO CA:	. 22
[8.3] OUTPUT FILES	. 22
[9] THE PKCS #11 CONFIGURATION FILE	.24



[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

Neither the whole nor any part of the information contained in this document may be adapted or reproduced in any material or electronic form without the prior written consent of the copyright holder.

Information in this document is subject to change without notice.

Futurex makes no warranty of any kind with regard to this information, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Futurex shall not be liable for errors contained herein or for incidental or consequential damages concerned with the furnishing, performance, or use of this material.

[1.3] TERMS OF USE

This integration guide, as well as the software and/or products described in it, are furnished under agreement with Futurex and may be used only in accordance with the terms of such agreement. Except as permitted by such agreement, no part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without prior written permission of Futurex.



[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 marketleading 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.

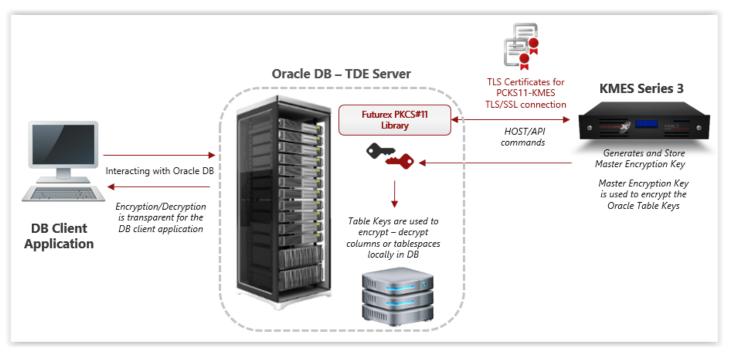


FIGURE: PROCESS OF ORACLE TDE INTEGRATION



[4] PRE-REQUISITES:

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

- 1. Linux Server (CentOS 7 for this example GUI not required): <u>https://www.centos.org/download/</u>
- 2. OpenSSL for Linux (optional)
- 3. Oracle Database 12c
- 4. Installation Guide for Oracle Database 12c: <u>https://docs.oracle.com/en/database/oracle/oracle-database/12.2/ladbi/index.html</u>
- 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: <u>https://docs.oracle.com/cd/E11882_</u> 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 (<u>https://docs.oracle.com/en/database/oracle/oracle-database/12.2/ladbi/index.html</u>) 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: <u>https://docs.oracle.com/cd/E11882_01/network.112/e40393/asotrans.htm#ASOAG640</u>.



[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 <u>samples</u> 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:

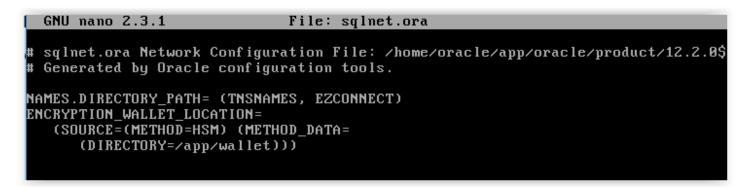
ENCRYPTION_WALLET_LOCATION=

(SOURCE= (METHOD=HSM) (METHOD_DATA=

(DIRECTORY=/app/wallet)))







Save and close the file.

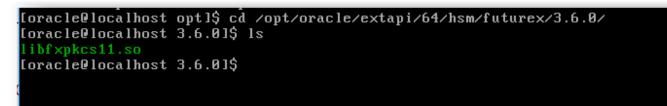
NOTE: If a DIRECTORY value is present in the ENCRYPTION_WALLET_LOCATION parameter, make sure not to delete it. Although the KMES Series 3 does not require a DIRECTORY value, the value is used to locate the old software wallet when migrating to HSM-based Transparent Data Encryption. Also, the DIRECTORY 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.



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

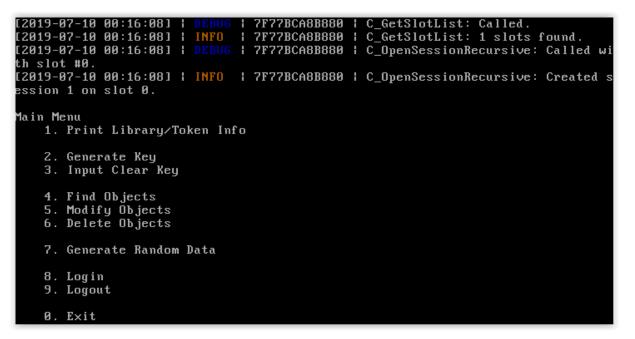
Before testing the connection between the PKCS #11 library and the KMES Series 3, it is required to complete the KMES Series 3 configuration (including TLS certificates) described in the sections on features required in the KMES Series 3 Internal HSM, the KMES Series 3 TLS certificates and PKCS certificates,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



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.





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).

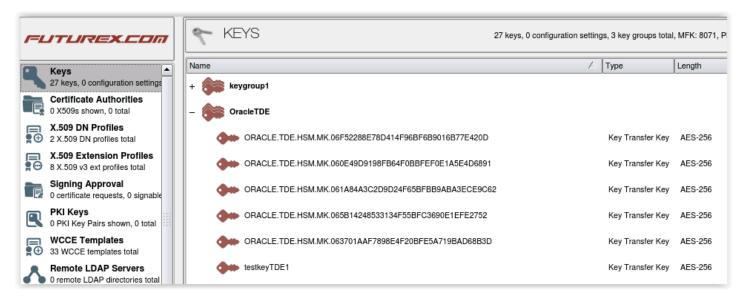


FIGURE: NEW KEY IN KEY GROUP IN KMES SERIES 3

FUTUR



[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"



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 reencrypted 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:

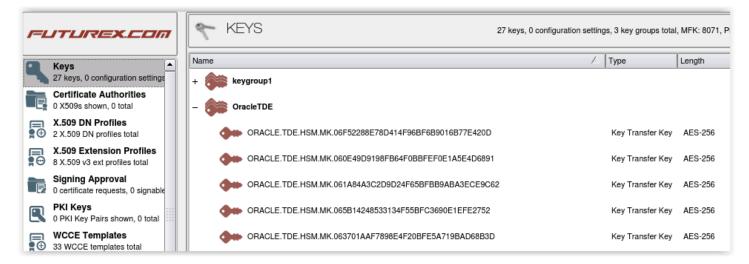


FIGURE: GENERATED KEYS IN THE KMES SERIES 3

[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:

USERS	2 groups, 3 users
Name	🛆 Last Login
– 📷 Admin Group	
Admin1	Wed July 10, 2019 02:48:18
Admin2	Wed July 10, 2019 02:48:23
- CryptoOperators	
Crypto1	Wed July 10, 2019 01:59:46

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



<١ OracleTDE Name: <١> None Algorithm: Ŧ Options.. <١ CryptoOperators -Permissions... Owner group: Ownership: Do not apply to child key groups Ŧ <١ Owner name: Set Object-Group Permissions imesOwner address: Modifying 1 key group. User Group Permission Implicit ۰ Add Admin Group **-** X CryptoOperators Add **-** X

FIGURE: SETTING OBJECT-GROUP PERMISSIONS IN KMES SERIES 3

< KEYS		0 keys
Name	🛆 Туре	Length
+ 💏 keygroup1		
+ PKCS11		

The KMES Series 3 should now look like this:



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:

<pre># KMES in first slot <kms> # Thick PEGG11 slot</kms></pre>		
# Which PKCS11 slot <slot></slot>	0	
<pre># Login username <crypto-opr></crypto-opr></pre>	crypto1	
# Key group name <keygroup-name></keygroup-name>	OracleTDE	





[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:

		Network Optio
thernet Settings	Network Settings TCP Settings	TLS/SSL Settings
Connection:		System/Host API
Enabled		
	/Host API SSL Parameters*	Allow Anonymous Connections
Port:	2001	A V
Header Size:	None	▼
-TLS Enable	d	
Ciphers:	7 selected	•
Protocols	3 selected	_
Cert Type	RSA	▼
PKI keys	Not loaded	Edit
Certificate	es Not loaded	Edit
🕱 Use Fut	urex certificates	
Anonymous conr	ection key size:	2048
		GK X Cance

FIGURE: PROCESS TO ENABLE TLS/SSL SETTINGS



Connection:	System/Host API	•
Enabled		
Use System	m/Host API SSL Parameters*	
Port:	2001	
Header Size:	None	-
-TLS Enable	led	
120 Enable		
Ciphers:	7 selected	•
Protocols	S: TLS_RSA_WITH_AES_256_CBC_SHA	
	TLS_RSA_WITH_AES_128_CBC_SHA	
Cert Type	e: XITLS_DHE_RSA_WITH_AES_256_CBC_SHA256	
	TLS DHE RSA WITH AES 256 CBC SHA	
	X TLS DHE RSA WITH AES 128 GCM SHA256	
User Certifi	ificat ILS_DHE_RSA_WITH_AES_128_CBC_SHA256	
	TLS_DHE_RSA_WITH_AES_128_CBC_SHA	
DIG	TLS_EDH_RSA_WITH_3DES_EDE_CBC_SHA	
PKI keys		
Certificate	TLS_ECDHE_ECDSA_WITH_CBC_256_CBC_SHA384	
Certificate	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA	
	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	
🗶 Use Fut	utur TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	
		-
Anonymous conn	nnection key size: 2048	•
, anonymoue com		

FIGURE: SELECT PICTURED TLS CONNECTION

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

# KI <km< th=""><th>MES in first slot S></th><th></th><th></th></km<>	MES in first slot S>		
	# Which PKCS11 slot <slot></slot>	0	
	<pre># Login username <crypto-opr></crypto-opr></pre>	crypto1	
	<pre># Key group name <keygroup-name></keygroup-name></pre>	OracleTDE	
	# Connection informatio	n	
	<address></address>	10.0.5.134	
	<pre><pre>PROD-PORT></pre></pre>	2001	
	<pre><pre>PROD-TLS-ENABLED></pre></pre>	YES	
	<pre><pre>PROD-TLS-ANONYMOUS></pre></pre>	NO	

FIGURE: TLS IS ENABLED



FUTUR

[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.

÷						Futurex 👝 🗖 🗙
File	Help					
<i>1=1</i>	JTUREX.COM	CONFIGURA	TION	MFK: 8071, PMK:	1E26	
	X.509 Extension Profiles	Configuration	∧	/alue	Description	
₽ ⊖	8 X.509 v3 ext profiles total	Download Feature R	aquaet		Download request for additional appl	ication abilities
	Signing Approval 0 certificate requests, 0 signable	W Download realdre H	equest		Download request for additional appr	cation abilities
-Le		Encryption Device C	•		Host API Configuration >	s.
	O PKI Key Pairs shown, 0 total		Commond.	Description		
	WCCE Templates	Encryption Device S		Description		ty mode(s)
	33 WCCE templates total	Host API Options	ECHO	Authenticate with OAuth		
	Remote LDAP Servers		X RAAE	Echo Text Add an X.509 V3Ext profile		
1	0 remote LDAP directories total	install	X RAAG	Add an X.509 VSEXt profile		
٣Ą	Token Generation Profiles		X RADE	Delete an X.509 V3Ext profile		
-0		KMIP Options	X RADG	Delete Approval Group		
	0 KMIP Objects 0 KMIP objects shown, 0 total	KSN Ranges Device	X RADX	Delete Request		device serial t
_		*	X RAEA	Modify Issuance Policy		
	0 hosts total	LDAP Auth Options	X RAEE	Edit an existing X.509 V3Ext profile		guration option
	Encryption Devices		X RAEG	Modify Approval Group		
Ē	1 group, 1 device, 1 client	Log Options	X RAEO	Modify Request (Hash Signing)		
	Peers	Major Key Load	X RAEP	Modify Request (X.509 PKI Gen)		d sessions for
	0 peers		X RAEX	Modify Request (X.509 CSR)		
	Configuration	Major Keys	X RAGA	Retrieve Issuance Policy		ot other keys.
-*	MFK: 8071, PMK: 1E26		X RAGO	Retrieve Request (Hash Signing)		
2	2 groups, 3 users	Manage Duplicate O	X RAGP	Retrieve Request (X.509 PKI Gen)		
	Logs	Manage Smart Card	Note: A checked bo	ox enables the respective API command.		mart card.
B	421 logs total		Enable tracking	of failed commands		
	Reports	Network Options				
C.	3 reports	OCSP Options	Use preset: None		▼	
	Templates				Save Cancel	
LΘ	0 templates					<u>E</u> dit
	Databases 💌					
					Group: Admin Group	Jsers: Admin1,Admin2

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:

Incoming Settings	
General Settings:	
Type:	Encrypted T
Port:	9009
TLS Settings:	
Allow Anonymous: Key Source:	□ Futurex Signed ▼
Cert Type:	RSA V
Certificates:	
\rightarrow Futurex Test Root \rightarrow Futurex Test Ac	
	Customer Admin SSL CA in CA Certificate (FX1829834213)

FIGURE: ADMIN TLS PORT



Excrypt (Production) TLS port:

Incoming Settings				
General Settings:				
Туре:	Encrypted T			
Port:	9100			
TLS Settings:				
Allow Anonymous:				
Key Source:	Futurex Signed ▼			
Cert Type:	RSA 🔻			
Certificates:				
\rightarrow Futurex Test Root SSL CA				
\rightarrow Futurex Test Production SSL CA				
→Futurex Test Cus	tomer Production SSL CA			

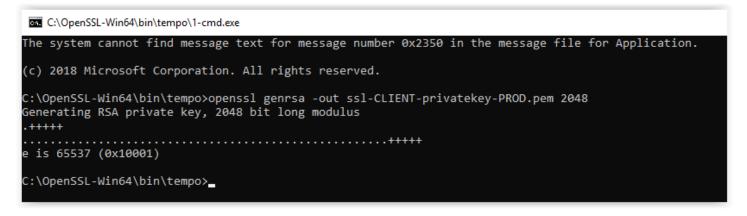
FIGURE: EXCRYPT (PRODUCTION) TLS PORT

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 <u>support@futurex.com</u> 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

C:\OpenSSL-Win64\bin\tempo\1-cmt.exe — — — X C:\OpenSSL-Win64\bin\tempo>openssl req -new -key ssl-CLIENT-privatekey-PROD.pem -out ssl-CLIENT-cert-req-PROD.pem -days 365 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) [US]: State or Province Name (full na e) [TX]: Locality Name (eg, city) [SAN ANTONIO]: Organization Name (eg, company) [FUTUREXLAB]: Organizational Unit Name (eg, section) [SA]: Common Name (e.g. server FQDN or YOUR name) [FUTUREXLAB.COM]: Email Address [MGRECO@FRUTUREXLAB.COM]:

[8.3] OUTPUT FILES

The output file, *ssl-CLIENT-cert-req-PROD.pem*, must be sent to <u>support@futurex.com</u> 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></address>	10.0.5.134	
<prod-port></prod-port>	2001	
<prod-tls-enabled></prod-tls-enabled>	YES	
<prod-tls-anonymous></prod-tls-anonymous>	NO	
<prod-tls-ca></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-cert></prod-tls-cert>	/home/oracle/FXcerts/PKCS11_PROD_Signed.pem	
<prod-tls-key></prod-tls-key>	/home/oracle/FXcerts/ssl-CLIENT-privatekey-PROD.pem	
# <prod-tls-key></prod-tls-key>	certs/HostPKI.p12	
# <prod-tls-key-pass></prod-tls-key-pass>	safest	
<pre># KMES in first slot <kms></kms></pre>		
# Which PKCS11 slo <slot></slot>	0	

<CRYPTO-OPR> crypto1 # Key group name <KEYGROUP-NAME> OracleTDE

Login username

</crypto-opr>

Connection information <ADDRESS> 10.0.5.134 </ADDRESS> <PROD-PORT> 2001 </prod-port> <PROD-TLS-ANONYMOUS> NO
<PROD-TLS-CA> </prod-tls-enabled> </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-CA>
 /home/oracle/FXcerts/Fx3.pem
 </PROD-TLS-CA>

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

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

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

 </prod-tls-cert> </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> # Debug binary only
<LOG-TRAFFIC>
                                 YES
<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></address>		10.0.5.134			
<prod-port></prod-port>		2001	<td>PORT></td>	PORT>	
<prod-tls-enabled></prod-tls-enabled>	YES		<td>BLED></td>	BLED>	
<prod-tls-anonymous></prod-tls-anonymous>	NO		<td>MOUS></td>	MOUS>	
<prod-tls-ca></prod-tls-ca>	/home/oracle/FXcerts/Fx	1.pem <td>OD-TLS-CA></td> <td></td>	OD-TLS-CA>		
<prod-tls-ca></prod-tls-ca>	/home/oracle/FXcerts/Fx	2.pem <td>-TLS-CA></td> <td></td>	-TLS-CA>		
<prod-tls-ca></prod-tls-ca>	/home/oracle/FXcerts/Fx	3.pem <td>-TLS-CA></td> <td></td>	-TLS-CA>		
<prod-tls-cert></prod-tls-cert>	/home/oracle/FXcerts/PK	CS11_PROD_Signed	.pem <td>)-TLS-CERT></td>)-TLS-CERT>	
<prod-tls-key></prod-tls-key>	/home/oracle/FXcerts/ss	l-CLIENT-private	key-PROD.pem		
# <prod-tls-key></prod-tls-key>	certs/H	ostPKI.p12			
# <prod-tls-key-pass></prod-tls-key-pass>	safest		<td>PASS></td>	PASS>	
# YES = This is communicating through a Guardian					
<fx-load-balance></fx-load-balance>	NO	<td>ICE></td> <td></td>	ICE>		

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.

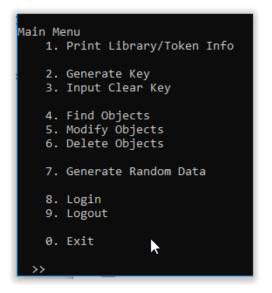


FIGURE: PKCS #11 MANAGER MAIN MENU



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