EJBCA

Integration Guide

Applicable Devices:
Vectera Plus

TABLE OF CONTENTS

[1] DOCUMENT INFORMATION

[1.1] DOCUMENT OVERVIEW

The purpose of this document is to provide information regarding the configuration of Futurex HSMs with EJBCA using PKCS #11 libraries. For additional questions related to your HSM, see the relevant administrator's guide.

[1.2] APPLICATION DESCRIPTION

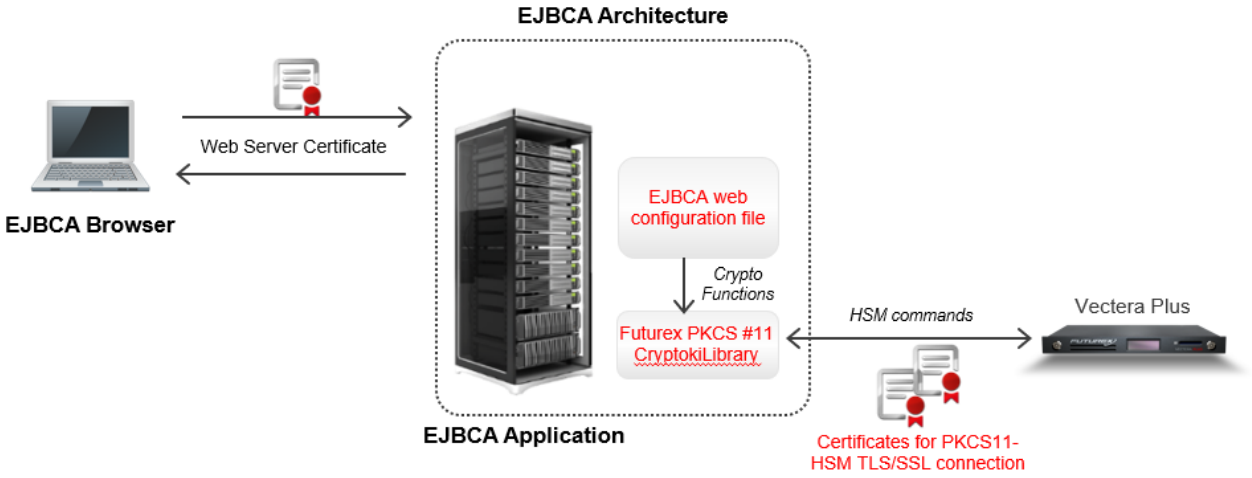
[1.2.1] About EJBCA

From the EJBCA website: "EJBCA is one of the longest running CA software projects, providing time-proven robustness and reliability. EJBCA is platform independent, and can easily be scaled out to match the needs of your PKI requirements, whether you're setting up a national eID, securing your industrial IOT platform or managing your own internal PKI."

[1.2.2] EJBCA Architecture and Integration with the Vectera Plus

The diagram below shows the architecture of a EJBCA environment that is integrated with a Vectera Plus HSM.

In this scenario, EJBCA is used to generate a PKI (RSA or ECC) on the HSM using the Futurex PKCS #11 library. EJBCA uses private keys to generate signatures and public keys to verify. All of these activities are initiated through the web browser, which helps the user to administrate the key usage and PKI pairs.



[1.3] GUARDIAN INTEGRATION

The Guardian Series 3 introduces mission-critical viability to core cryptographic infrastructure, including:

- Centralize device management
- Eliminates points of failure
- Distribute transaction loads
- Group-specific function blocking
- User-defined grouping systems

Please see applicable guide for configuring HSMs with the Guardian Series 3.

[2] PREREQUISITES

Supported Hardware:

- Vectera Plus, 6.7.0.10 and above

Supported Operating Systems:

- Linux Ubuntu 20.04 LTS

Other:

- OpenSSL
- Java 7, 8, 9
- EJBCA (**NOTE:** This document does not include the EJBCA installation process. EJBCA should already be installed on the server.)
- HSM with the following features:
    - PKCS 11 -> Enabled
    - Command Primary Mode -> General Purpose (GP)
    - Command Extension Mode -> GP with Financial
    - HSM Crypto User support -> This user will be used for the PKCS #11 library to be able to access the HSM.
    - Multi-usage combination for Asymmetric keys
- HSM with Futurex TLS certificates preloaded, and TLS client certificates to allow the PKCS #11 library to connect with the HSM.
- Futurex PKCS #11 (FXPKCS11) Library -> 4.36 and above

[3] INSTALL FUTUREX PKCS #11 (FXPKCS11)

In a Linux environment, to install the Futurex PKCS #11 (FXPKCS11) module you must download a tarball of the FXPKCS11 binaries from the Futurex Portal and then extract the tar file locally where you want the application to be installed on your system. Step-by-step installation instructions are provided below.

Note: Install FXPKCS11 on the same computer as the application integrating with the Vectera Plus HSM.

[3.1] INSTALLING THE FXPKCS11 MODULE IN LINUX

Extract the tarball file for your Linux distrubution in the desired working directory.

Note: To make the Futurex PKCS #11 module accessible system-wide, move it to the /usr/local/bin directory as an administrative user. If only the current user needs to use the module, then install it in $HOME/bin.

The extracted content of the tar file is a single fxpkcs11 directory. Inside the fxpkcs11 directory is the following files and directories:

- **fxpkcs11.cfg**: FXPKCS11 configuration file
- **x86/**: This folder contains the module files for 32-bit architecture
- **x64/**: This folder contains the module files for 64-bit architecture

The x86 and x64 directories each contain two subdirectories, OpenSSL-1.0.x and OpenSSL-1.1.x. These OpenSSL directories contain the following FXPKCS11 module files built with the respective OpenSSL versions:

- **configTest**: Program to test configuration and connection to the HSM
- **libfxpkcs11.so**: FXPKCS11 Library File
- **libfxpkcs11-Debug.so**: FXPKCS11 Debug Library File
- **PKCS11Manager**: Program to test connection and manage the HSM through the FXPKCS11 library

By default, the FXPKCS11 module looks for the FXPKCS11 configuration file (i.e., fxpkcs11.cfg) in the /etc directory. Alternatively, a system environment variable can be defined for the location of the FXPKCS11 configuration file. To do so permanently, open the /etc/profile file in a text editor as an administrative user, add the following line at the bottom, and save the file.

```
export FXPKCS11_CFG=/usr/local/bin/fxpkcs11/fxpkcs11.cfg
```

Note: The file location specified above must be specific to where the FXPKCS11 configuration file is saved on your system.

[4] INSTALL EXCRYPT MANAGER (IF USING WINDOWS)

Sections 4 and 5 of this integration guide cover the installation of Excrypt Manager and FXCLI. Excrypt Manager is a Windows application that provides a GUI-based method for configuring the HSM, while FXCLI provides a command-line-based method for configuring the HSM and can be installed on all platforms.

Note: If you will be configuring the Vectera Plus from a Linux computer, you can skip this section. If you will be configuring the Vectera Plus from a Windows computer, installing FXCLI in the next section is still required because FXCLI is the only method that can be used to configure TLS certificates in section 6.7.

Note: Install Excrypt Manager on the workstation you will use to configure the HSM.

Note: If you plan to use a Virtual HSM for the integration, all configurations will need to be performed using either FXCLI, the Excrypt Touch, or the Guardian Series 3.

Note: The Excrypt Manager version must be from the 4.4.x branch or later to be compatible with the HSM firmware, which must be 6.7.x.x or later.

To install Excrypt Manager, run the Excrypt Manager installer as an administrator and follow the prompts in the setup wizard to complete the installation.

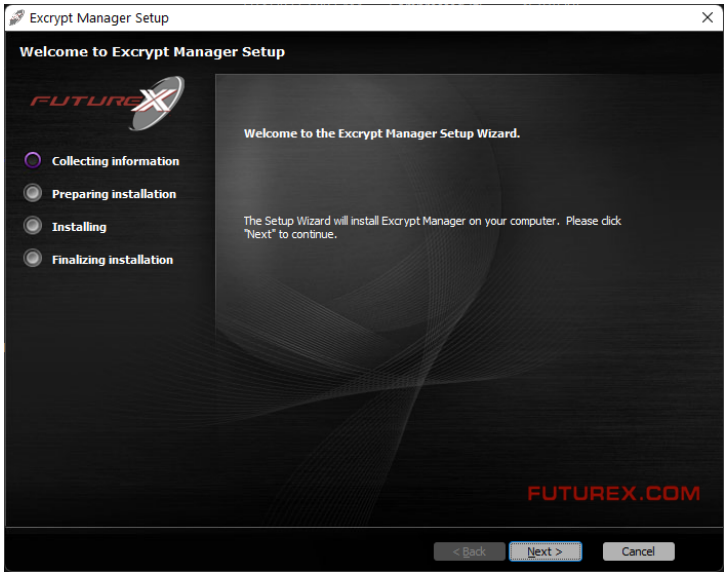


FIGURE: EXCRYPT MANAGER SETUP WIZARD

The installation wizard prompts you to specify where you want to install Excrypt Manager. The default location is C:\Program Files\Futurex\Excrypt Manager\. After choosing a location, select **[Install]**.

[5] INSTALL FUTUREX COMMAND LINE INTERFACE (FXCLI)

Note: Install FXCLI on the workstation you will use to configure the HSM.

[5.1] INSTALLING FXCLI IN WINDOWS

As mentioned in section 3, the FXTools installation package includes Futurex Client Tools (FXCLI). Similar to the Futurex PKCS #11 (FXPKCS11) module, the easiest way to install FXCLI on Windows is by installing FXTools. You can download FXTools from the Futurex Portal.

To install FXCLI, run the Futurex Tools installer as an administrator and follow the prompts in the setup wizard to complete the installation.

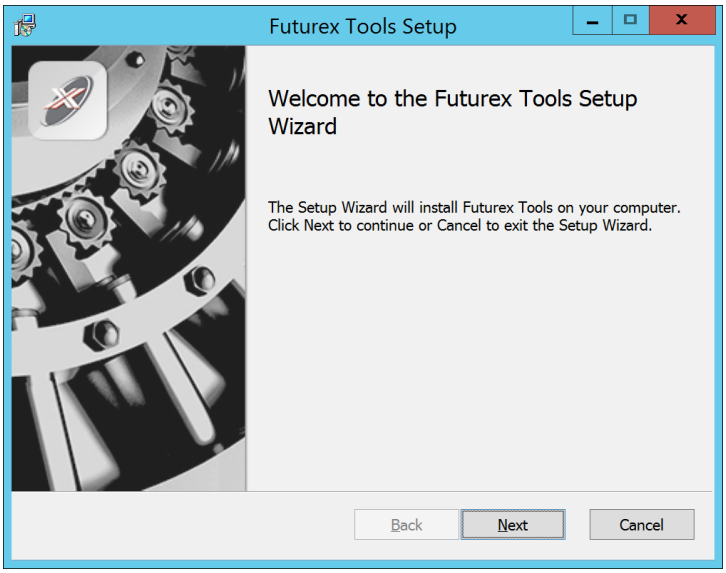


FIGURE: FUTUREX TOOLS SETUP WIZARD

The setup wizard installs all tools on the system by default. You can override the defaults and choose not to install certain modules. The installation provides the following services:

- **Futurex Client Tools**:Command Line Interface (CLI) and associated SDK for both Java and C.
- **Futurex CNG Module**:The Microsoft Next Generation Cryptographic Library.
- **Futurex Cryptographic Service Provider (CSP)**:The legacy Microsoft cryptographic library.
- **Futurex EKM Module**:The Microsoft Enterprise Key Management library.
- **Futurex PKCS #11 Module**:The Futurex PKCS #11 library and associated tools.
- **Futurex Secure Access Client**:A client used to connect a Futurex Excrypt Touch to a local laptop through USB, which can then connect to a remote Futurex device.

[5.2] INSTALLING FXCLI IN LINUX

Download FXCLI

You can download the appropriate FXCLI package files for your system from the Futurex Portal.

If the system is **64-bit**, select from the files marked **amd64**. If the system is **32-bit**, select from the files marked **i386**.

If running an OpenSSL version in the **1.0.x** branch, select from the files marked **ssl1.0**. If running an OpenSSL version in the **1.1.x** branch, select from the files marked **ssl1.1**.

Futurex offers the following features for FXCLI:

- Java Software Development Kit (**java**)
- HSM command line interface (**cli-hsm**)
- KMES command line interface (**cli-kmes**)
- Software Development Kit headers (**devel**)
- YAML parser used to parse bash output (**cli-fxparse**)

Install FXCLI

To install an rpm package, run the following command in a terminal:

```
$ sudo rpm -ivh [fxcl-xxxx.rpm]
```

To install a deb package, run the following command in a terminal:

```
$ sudo dpkg -i [fxcl-xxxx.deb]
```

Running FXCLI

To enter the HSM FXCLI prompt, run the following command in a terminal:

```
$ fxcli-hsm
```

After entering the FXCLI prompt, you can run **help** to list all of the available FXCLI commands.

[6] FUTUREX HSM CONFIGURATION

In order to establish a connection between the PKCS #11 library and the Futurex HSM, a few configuration items need to first be performed, which are the following:

NOTE: All of the steps in this section can be completed through either Excrypt Manager or FXCLI (if using a physical HSM rather than a virtual HSM). Optionally, steps 4 through 6 can be completed through the Guardian Series 3, which will be covered in Appendix A.

1. Connect to the HSM via the front USB port (NOTE: If you are using a virtual HSM for the integration you will have to connect to it over the network either via FXCLI, the Excrypt Touch, or the Guardian Series 3)
    a. Connecting via Excrypt Manager
    b. Connecting via FXCLI
2. Validate the correct features are enabled on the HSM
3. Setup the network configuration
4. Enable the DUS and EWV multi-usage combinations for asymmetric keys
5. Load the Futurex FTK
6. Configure a Transaction Processing connection and create a new Application Partition
7. Create a new Identity that has access to the Application Partition created in the previous step
8. Configure TLS Authentication. There are two options for this:
    a. Enabling server-side authentication
    b. Creating client certificates for mutual authentication

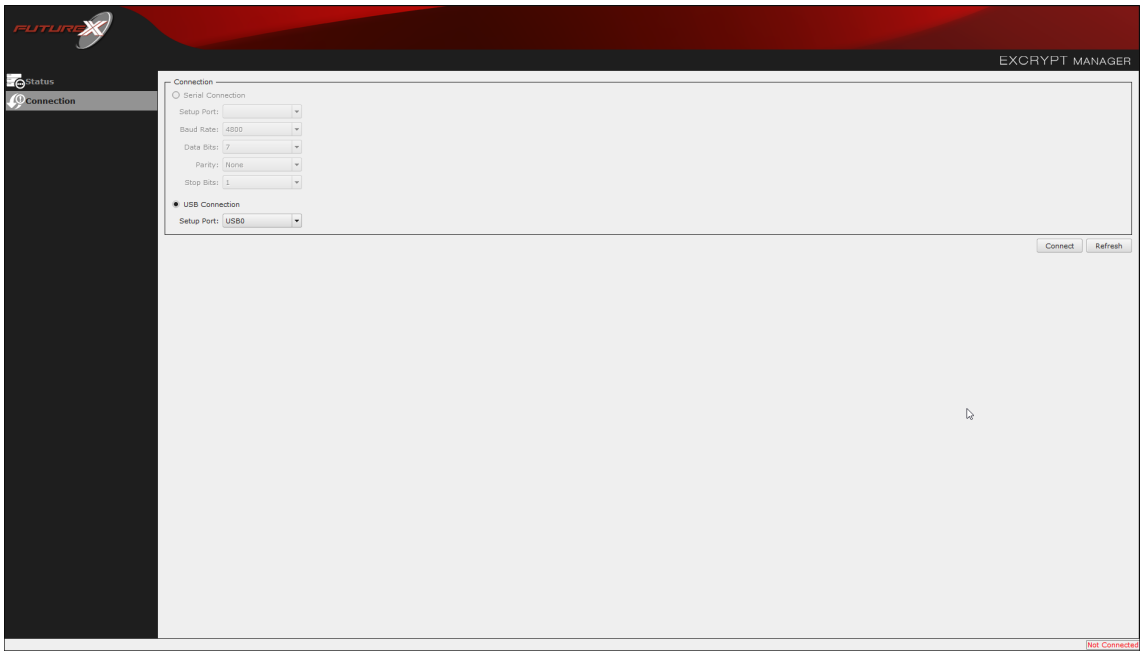
Each of these action items is detailed in the following subsections.

[6.1] CONNECT TO THE HSM VIA THE FRONT USB PORT

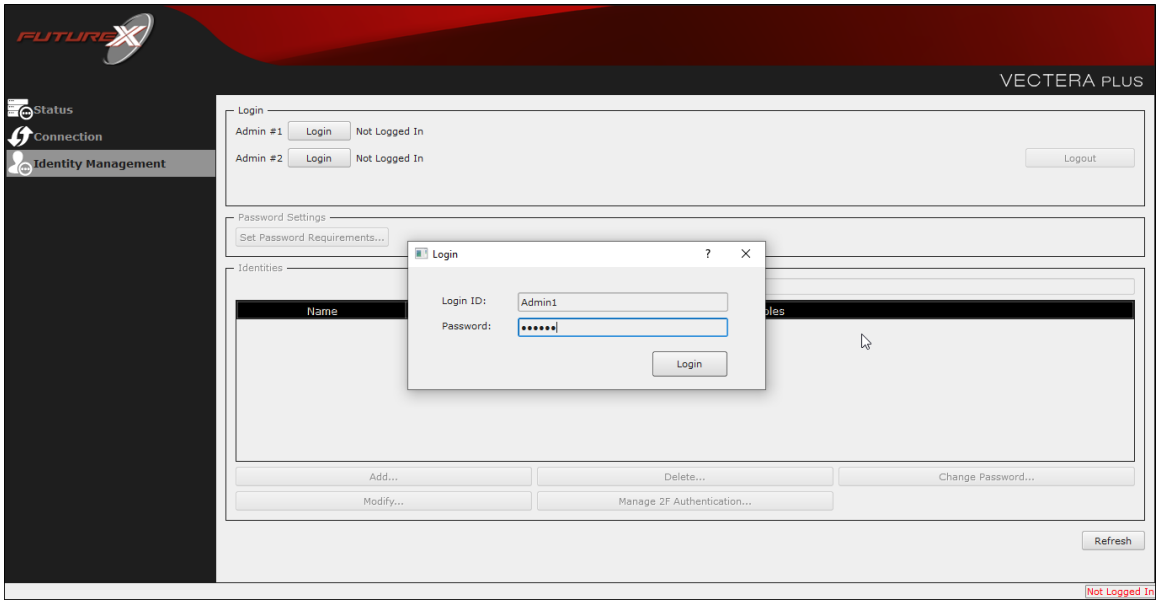
For both Excrypt Manager and FXCLI you need to connect your laptop to the front USB port on the HSM.

Connecting via Excrypt Manager

Open Excrypt Manager, click “Refresh” in the lower right-hand side of the Connection menu. Then select “USB Connection” and click “Connect”.

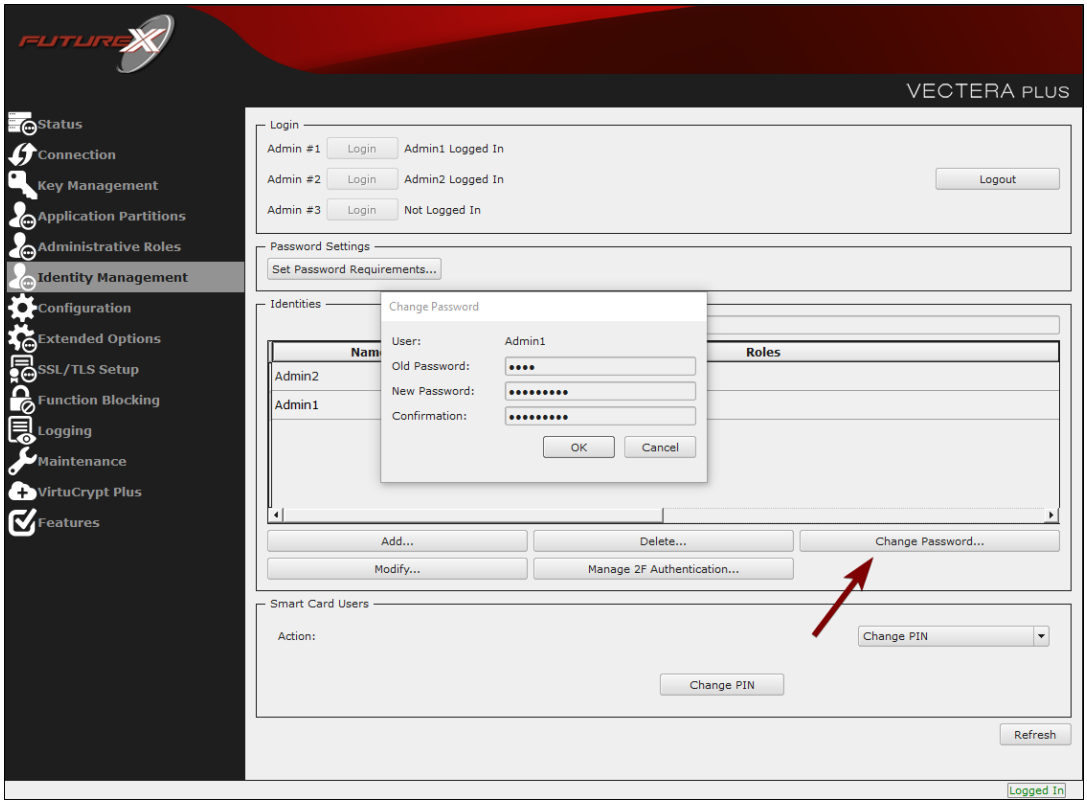


Login with both default Admin identities.



The default Admin passwords (i.e. “safe”) must be changed for both of your default Admin Identities (e.g. “Admin1” and “Admin2”) in order to load the major keys onto the HSM.

To do so via Excrypt Manager navigate to the Identity Management menu, select the first default Admin identity (e.g. “Admin1”), then click the “Change Password…” button. Enter the old password, then enter the new password twice, and click “OK”. Perform the same steps as above for the second default Admin identity (e.g. “Admin2”).



Connecting via FXCLI

Open the FXCLI application and run the following commands:

```
$ connect usb
$ login user
```

NOTE: The **"login"** command will prompt for the username and password. You will need to run it twice because you must login with both default Admin identities.

The default Admin passwords (i.e. “safe”) must be changed for both of your default Admin Identities (e.g. “Admin1” and “Admin2”) in order to load the major keys onto the HSM.

The following FXCLI commands can be used to change the passwords for each default Admin Identity.

```
$ user change-password -u Admin1
$ user change-password -u Admin2
```

NOTE: The user change-password commands above will prompt you to enter the old and new passwords. It is necessary to run the command twice (as shown above) because the default password must be changed for both default Admin identities.

[6.2] FEATURES REQUIRED IN HSM

In order to establish a connection between the PKCS #11 Library and the Futurex HSM, the HSM must be configured with the following features:

- **PKCS #11** -> Enabled
- **Command Primary Mode** -> General Purpose (GP)

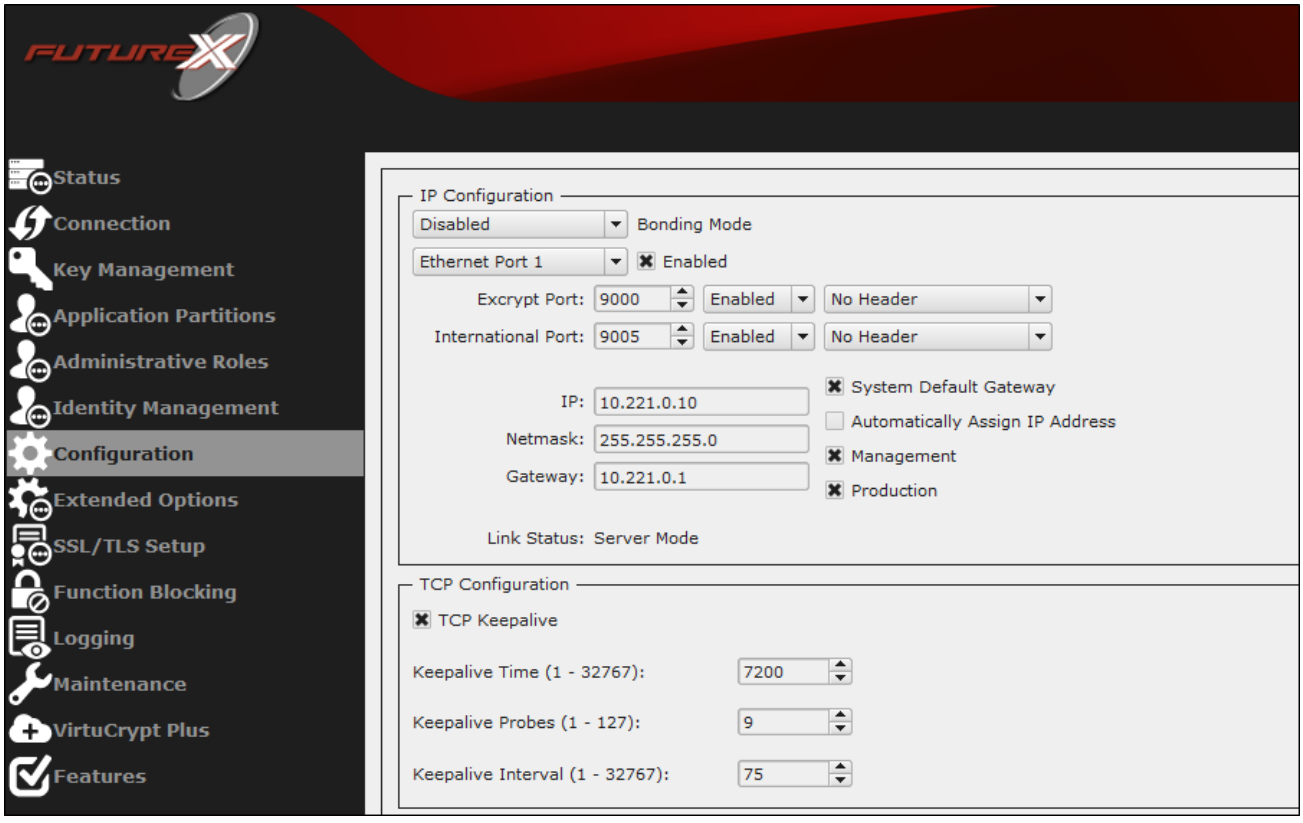
NOTE: For additional information about how to update features on your HSM, please refer to your HSM Administrator's Guide, section **"Download Feature Request File"**.

NOTE: Command Primary Mode = General Purpose, will enable the option to create the FTK major key in the HSM. This key will be required to be able to use the PKCS #11 library to communicate with the HSM. For detailed information about how to load major keys in HSMs please refer to your HSM Administrator's Guide.

[6.3] NETWORK CONFIGURATION (HOW TO SET THE IP OF THE HSM)

*For this step you will need to be logged in with an identity that has a role with permissions **Communication:Network Settings**. The default Administrator role and Admin identities can be used.*

Navigate to the *Configuration* page. There you will see the option to modify the IP configuration, as shown below:



Alternatively, the following **FXCLI** command can be used to set the IP for the HSM:

```
$ network interface modify --interface Ethernet1 --ip 10.221.0.10 --netmask 255.255.255.0 --gateway
10.221.0.1
```

NOTE: The following should be considered at this point:

- All of the remaining HSM configurations in this section can be completed using the Guardian Series 3 (please refer to Appendix A for instructions on how to do so), with the exception of the final subsection that covers how to create connection certificates for mutual authentication.
- If you are performing the configuration on the HSM directly now, but plan to add the HSM to a Guardian later, it may be necessary to synchronize the HSM after it is added to a Device Group on the Guardian.
- If configuration through a CLI is required for your use-case, then you should manage the HSMs directly.

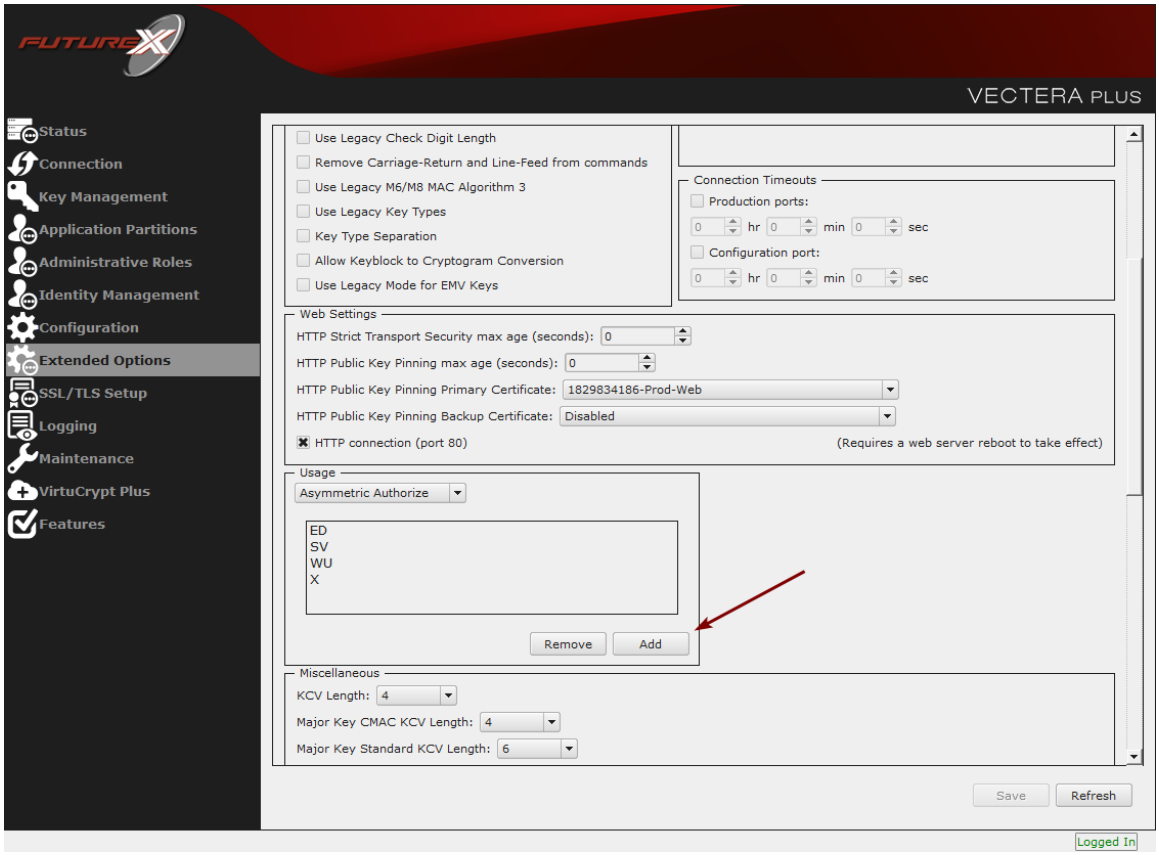
[6.4] ENABLE THE DUS AND EWV MULTI-USAGE COMBINATIONS FOR ASYMMETRIC KEYS

For this step you will need to be logged in with an identity that has a role with permissions ***Security:Key Settings****. The default Administrator role and Admin identities can be used.*

The EJBCA application requires asymmetric keys with multiple usages, which can be configured, but is not enabled by default on the Vectera Plus.

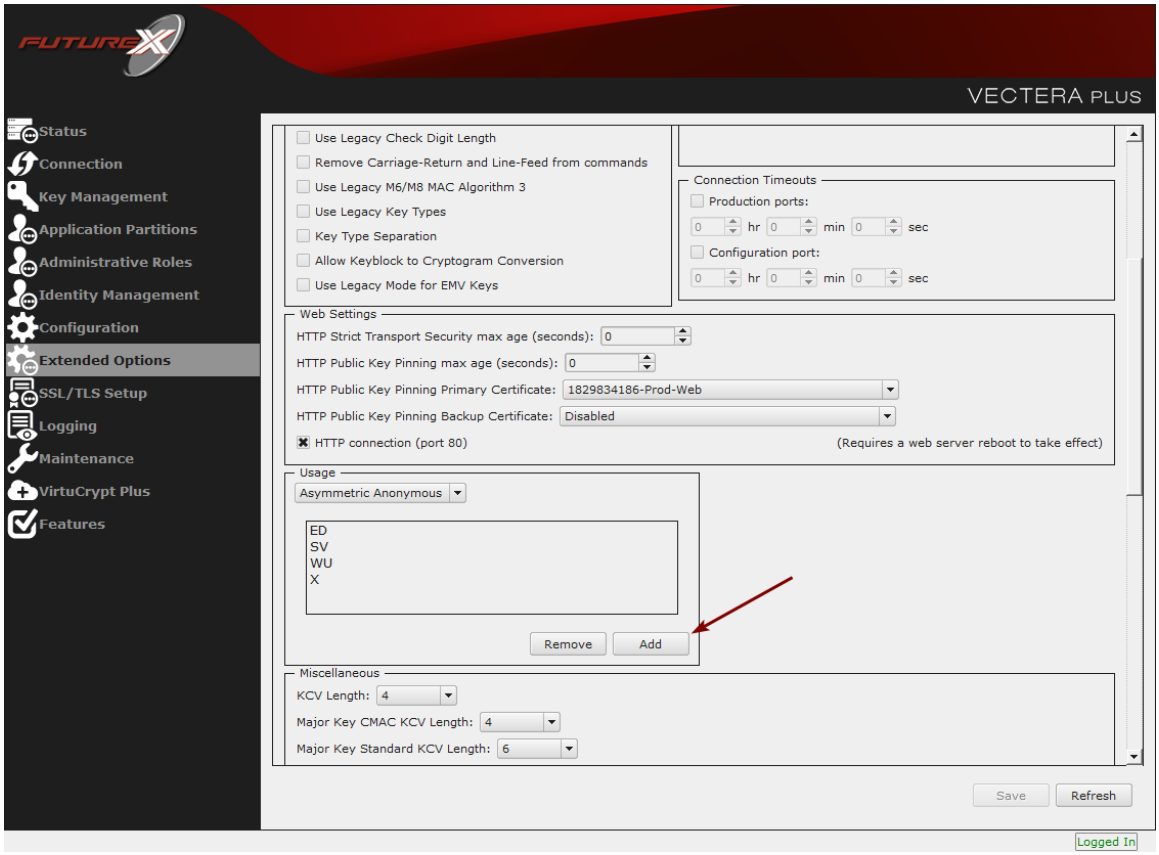
The specific multi-usage combinations that EJBCA requires is **DUS (i.e., Decrypt, Unwrap, Sign)** and **EWV (i.e., Encrypt, Wrap, Verify)**. It is necessary to enable this multi-usage combination for all users, including anonymous users, due to how the EJBCA application creates the keys on the HSM.

To configure this via Excrypt Manager, navigate to the *Extended Options* menu. In the "Usage" section, there is the option to add a new usage combination. With **Asymmetric Authorize** selected in the Usage dropdown, click the **Add** button.



Select the DUS usage combination and click "Ok". Repeat the same steps for adding the EWV multi-usage combination for Asymmetric Authorize.

Now, select **Asymmetric Anonymous** in the Usage dropdown and enable both the DUS and EWV multi-usage combinations.



Click the "Save" button on the bottom-right-hand side of the window to save the changes.

Alternatively, the following **FXCLI** commands can be used to add the DUS and EWV multi-usage combinations for asymmetric keys for all users:

```
$ multi-usage add --asymmetric --auth -d -u -s
```

```
$ multi-usage add --asymmetric --auth -e -w -v
```

```
$ multi-usage add --asymmetric --anon -d -u -s
```

```
$ multi-usage add --asymmetric --anon -e -w -v
```

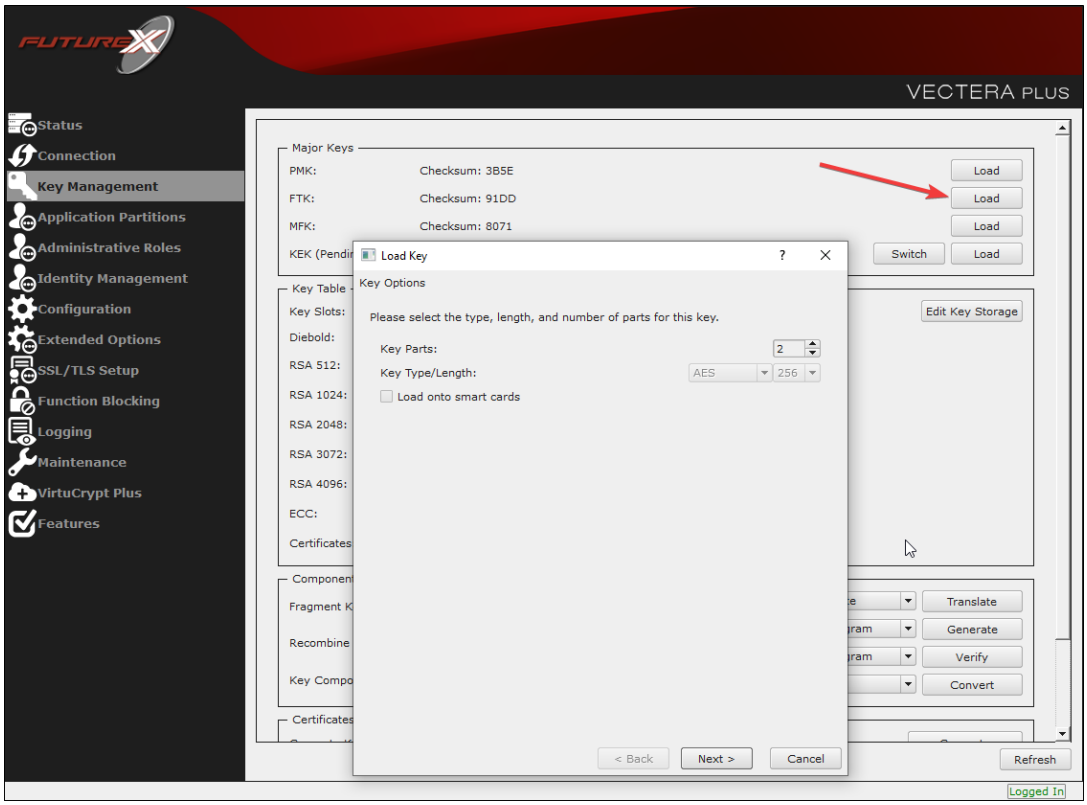
[6.5] LOAD FUTUREX KEY (FTK)

*For this step you will need to be logged in with an identity that has a role with permissions **Major Keys:Load**. The default Administrator role and Admin identities can be used.*

The FTK is used to wrap all keys stored on the HSM used with PKCS #11. If using multiple HSMs in a cluster, the same FTK can be used for syncing HSMs. Before an HSM can be used with PKCS #11, it must have an FTK.

NOTE: This process can also be completed using FXCLI, the Excrypt Touch, or the Guardian Series 3. For more information about how to load the FTK into an HSM using these tools/devices, please see the relevant Administrative Guide.

After logging in, select *Key Management*, then "Load" under FTK. Keys can be loaded as components that are XOR'd together, M-of-N fragments, or generated. If this is the first HSM in a cluster, it is recommended to generate the key and save to smart cards as M-of-N fragments.



Alternatively, the following **FXCLI** commands can be used to load an FTK onto an HSM.

If this is the first HSM you are setting up you will need to generate a random FTK. Optionally, you can also load it onto smart cards simultaneously with the -m and -n flags.

```
$ majorkey random --ftk -m [number_from_2_to_9] -n [number_from_2_to_9]
```

If it's a second HSM that you're setting up in a cluster then you will load the FTK from smart cards with the following command:

```
$ majorkey recombine --key ftk
```

[6.6] CONFIGURE A TRANSACTION PROCESSING CONNECTION AND CREATE AN APPLICATION PARTITION

For this step you will need to be logged in with an identity that has a role with permissions ***Role:Add, Role:Assign All Permissions, Role:Modify, Keys:All Slots****, and* ***Command Settings:Excrypt****. The default Administrator role and Admin identities can be used.*

NOTE: For the purposes of this integration guide you can consider the terms "Application Partition" and "Role" to be synonymous. For more information regarding Application Partitions, Roles, and Identities, please refer to the relevant Administrator's guide.

Configure a Transaction Processing Connection

Before an application logs in to the HSM with an authenticated user, it first connects via a “Transaction Processing” connection to the **Transaction Processing** Application Partition. For this reason, it is necessary to take steps to harden this Application Partition. The following three things need to be configured for the Transaction Processing partition:

1. It should not have access to the “All Slots” permissions
2. It should not have access to any key slots
3. Only the PKCS #11 communication commands should be enabled

Go to *Application Partitions*, select the Transaction Processing Application Partition, and click Modify.

Under the "Permissions" tab, leave the top-level **Keys** permission checked, but uncheck the **All Slots** sub permission.

Under the "Key Slots" tab you need to ensure that there are no key ranges specified. By default, the Transaction Processing Application Partition has access to the entire range of key slots on the HSM.

Lastly, under the “Commands” tab make sure that only the following **PKCS #11 Communication commands** are enabled:

- **ECHO**: Communication Test/Retrieve Version
- **PRMD**: Retrieve HSM restrictions
- **RAND**: Generate random data
- **HASH**: Retrieve device serial
- **GPKM**: Retrieve key table information
- **GPKS**: General purpose key settings get/change
- **GPKR**: General purpose key settings get (read-only)

Alternatively, the following **FXCLI** commands can be used to remove all permissions and key ranges that are currently assigned to the **Transaction Processing** role and enable only the PKCS #11 Communication commands:

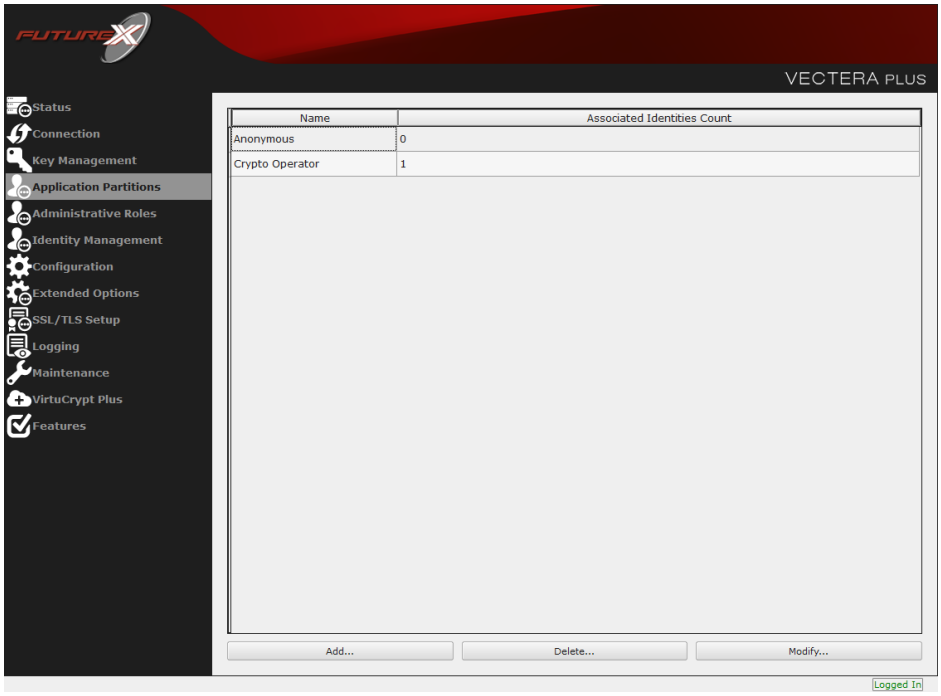
```
$ role modify --name Anonymous --clear-perms --clear-key-ranges
```

```
$ role modify --name Anonymous --add-perm "Keys" --add-perm Excrypt:ECHO --add-perm Excrypt:PRMD --
add-perm Excrypt:RAND --add-perm Excrypt:HASH --add-perm Excrypt:GPKM --add-perm Excrypt:GPKS --
add-perm Excrypt:GPKR
```

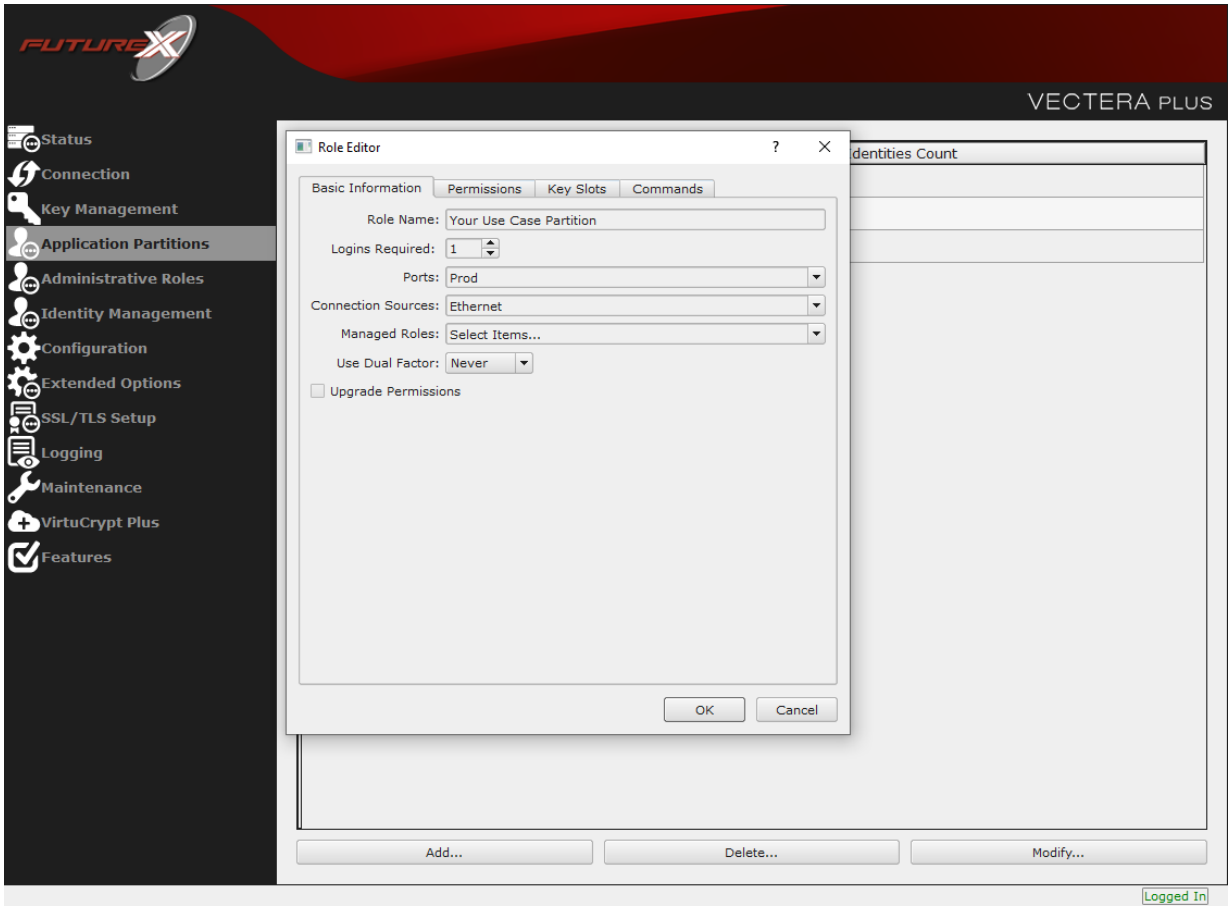
Create an Application Partition

In order for application segregation to occur on the HSM, an Application Partition must be created specifically for your use case. Application partitions are used to segment the permissions and keys on an HSM between applications. The process for configuring a new application partition is outlined in the following steps:

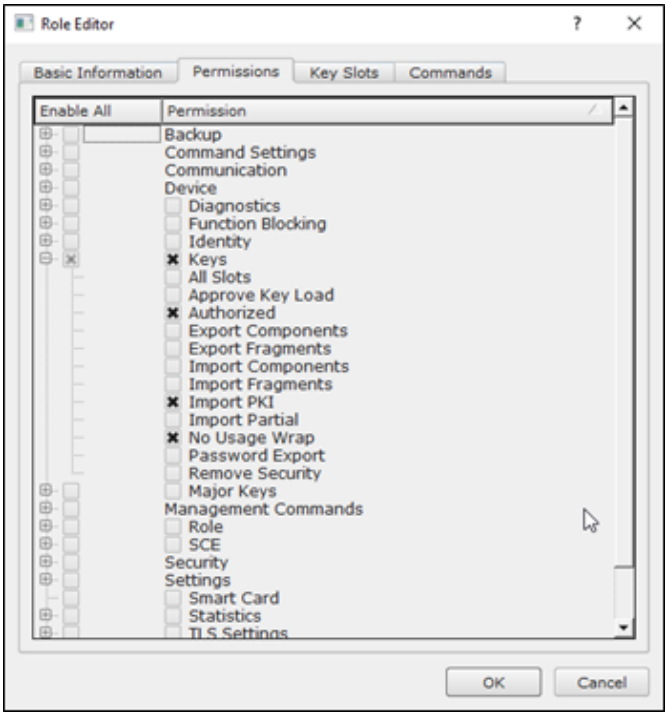
Navigate to the *Application Partitions* page and click the "Add" button at the bottom.



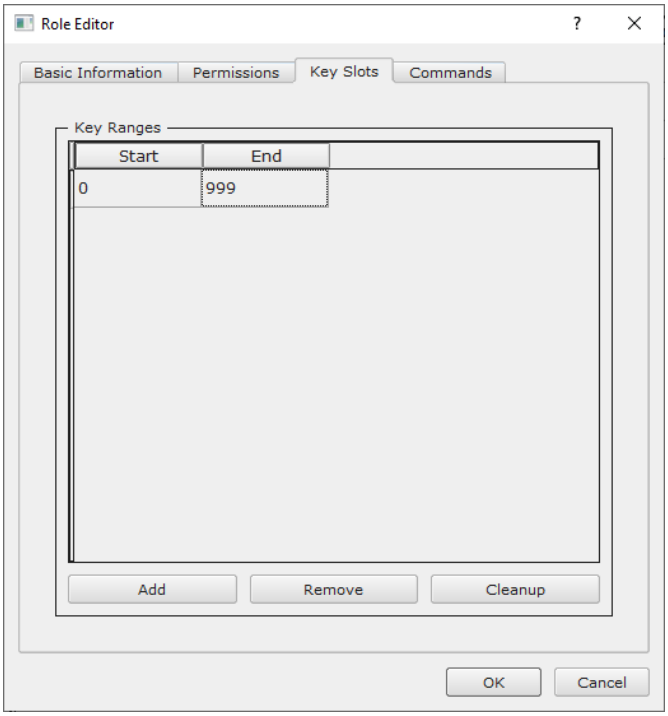
Fill in all of the fields in the *Basic Information* tab exactly how you see below (except for the *Role Name* field). In the *Role Name* field, specify any name that you would like for this new Application Partition. *Logins Required* should be set to "1". *Ports* should be set to "Prod". *Connection Sources* should be configured to "Ethernet". The *Managed Roles* field should be left blank because we'll be specifying the exact Permissions, Key Slots, and Commands that we want this Application Partition/Role to have access to. Lastly, the *Use Dual Factor* field should be set to "Never".



Under the “Permissions” tab, select the key permissions shown in the screenshot below. The **Authorized** permission allows for keys that require login. The **Import PKI** permission allows trusting an external PKI, which is used by some applications to allow for PKI symmetric key wrapping (It is not recommended to enable unless using this use case). The **No Usage Wrap** permission allows for interoperable key wrapping without defining key usage as part of the wrapped key (This is only recommended if exchanging keys with external entities or using the HSM to wrap externally used keys).



Under key slots, it is recommended that you create a range of 1000 total keys (here we've specified the key range 0-999), which do not overlap with another Application Partition. Within this range, there must be ranges for both symmetric and asymmetric keys. If more keys are required by the application, configure accordingly.



Based on application requirements there are particular functions that need to be enabled on the Application Partition in order to utilize the HSMs functionality. The most often used commands are included below. These can be enabled under the "Commands" tab.

PKCS #11 Communication Commands

- ECHO: Communication Test/Retrieve Version
- PRMD: Retrieve HSM restrictions
- RAND: Generate random data
- HASH: Retrieve device serial
- GPKM: Retrieve key table information
- GPKS: General purpose key settings get/change
- GPKR: General purpose key settings get (read-only)

Key Operations Commands

- APFP: Generate PKI Public Key from Private Key
- ASYL: Load asymmetric key into key table
- GECC: Generate an ECC Key Pair
- GPCA: General purpose add certificate to key table
- GPGS: General purpose generate symmetric key
- GPKA: General purpose key add
- GPKD: General purpose key slot delete/clear
- GRSA: Generate RSA Private and Public Key
- LRSA: Load key into RSA Key Table
- RPFP: Get public components from RSA private key

Interoperable Key Wrapping

- GPKU: General purpose key unwrap (unrestricted)
- GPUK: General purpose key unwrap (preserves key usage)
- GPKW: General purpose key wrap (unrestricted)
- GPWK: General purpose key wrap (preserves key usage)

Data Encryption Commands

- ADPK: PKI Decrypt Trusted Public Key
- GHSH: Generate a Hash (Message Digest)
  *Starting in firmware version 7.x, this function is enabled by default and does not need to be specified.
- GPED: General purpose data encrypt and decrypt
- GPGC: General purpose generate cryptogram from key slot
- GPMC: General purpose MAC (Message Authentication Code)
- GPSR: General purpose RSA encrypt/decrypt or sign/verify with recovery
- HMAC: Generate a hash-based message authentication code
- RDPK: Get Clear Public Key from Cryptogram

Signing Commands

- ASYS: Generate a Signature Using a Private Key
- ASYV: Verify a Signature Using a Public Key
- GPSV: General purpose data sign and verify
- RSAS: Generate a Signature Using a Private Key

Alternatively, the following **FXCLI** commands can be used to create the new Application Partition and enable all of the functions that are needed:

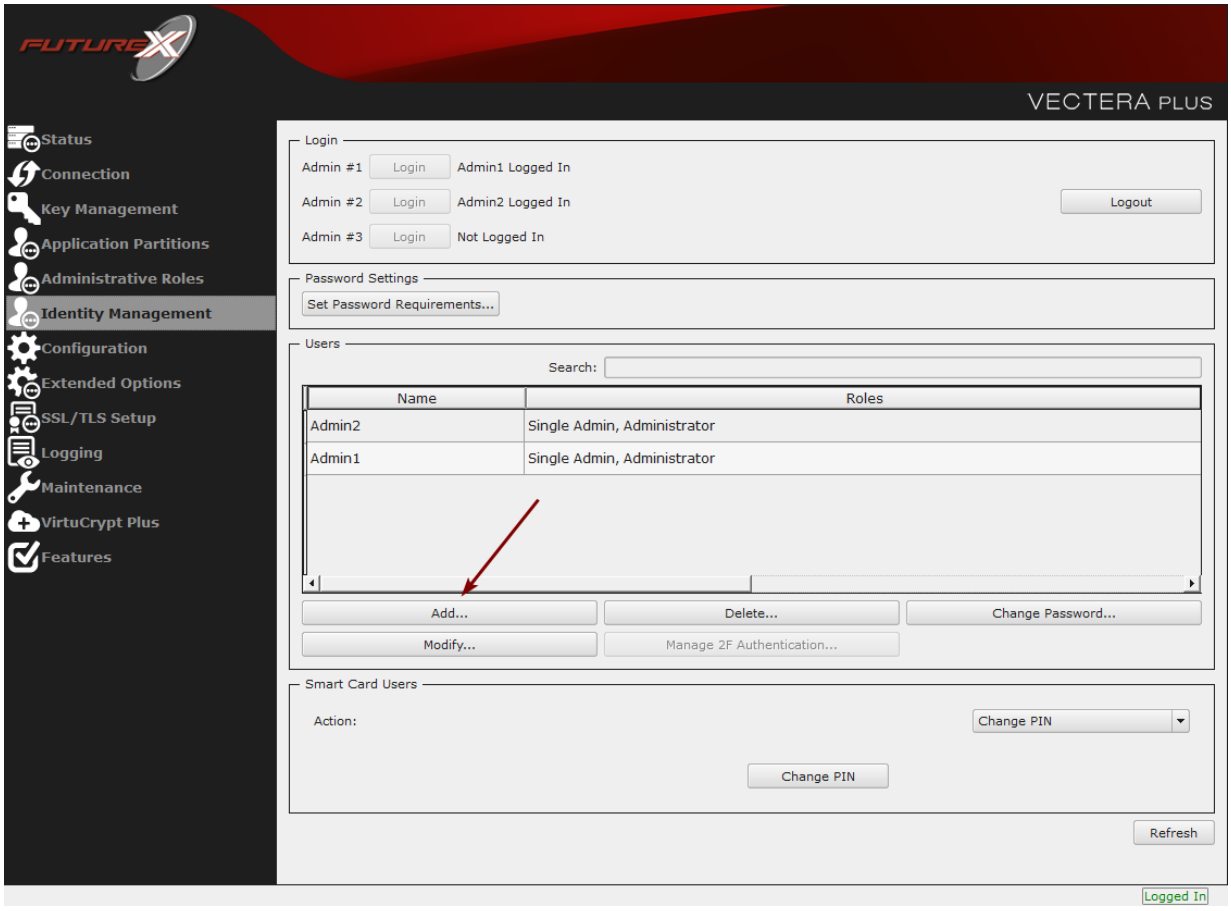
```
$ role add --name Role_Name --application --key-range (0,999) --perm "Keys:Authorized" --perm "Key-
s:Import PKI" --perm "Keys:No Usage Wrap"
```

```
$ role modify --name [role_name] --clear-perms --add-perm Excrypt:ECHO --add-perm Excrypt:PRMD --
add-perm Excrypt:RAND --add-perm Excrypt:HASH --add-perm Excrypt:GPKM --add-perm Excrypt:GPKS --
add-perm Excrypt:GPKR --add-perm Excrypt:APFP --add-perm Excrypt:ASYL --add-perm Excrypt:GECC --
add-perm Excrypt:GPCA --add-perm Excrypt:GPGS --add-perm Excrypt:GPKA --add-perm Excrypt:GPKD --
add-perm Excrypt:GRSA --add-perm Excrypt:LRSA --add-perm Excrypt:RPFP --add-perm Excrypt:GPKU --
add-perm Excrypt:GPUK --add-perm Excrypt:GPKW --add-perm Excrypt:GPWK --add-perm Excrypt:ADPK --
add-perm Excrypt:GHSH --add-perm Excrypt:GPED --add-perm Excrypt:GPGC --add-perm Excrypt:GPMC --
add-perm Excrypt:GPSR --add-perm Excrypt:HMAC --add-perm Excrypt:RDPK --add-perm Excrypt:ASYS --
add-perm Excrypt:ASYV --add-perm Excrypt:GPSV --add-perm Excrypt:RSAS
```

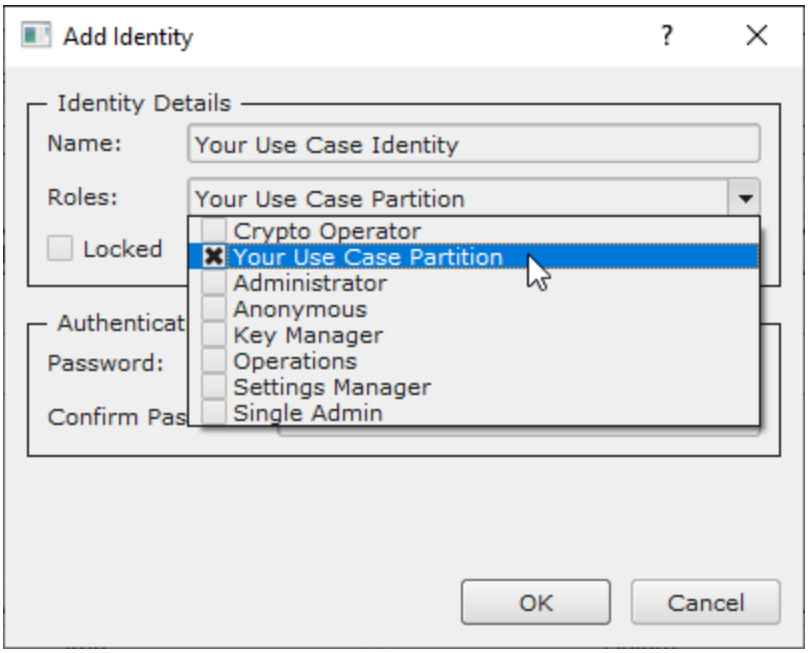
[6.7] CREATE NEW IDENTITY AND ASSOCIATE IT WITH THE NEWLY CREATED APPLICATION PARTITION

*For this step you will need to be logged in with an identity that has a role with permissions **Identity:Add**. The default Administrator role and Admin identities can be used.*

A new identity must be created, which will need to be associated with the Application Partition created in the previous step. To create this new identity, go to *Identity Management*, and click "Add".



Specify a name for the new identity, and in the Roles dropdown select the name of the Application Partition created in the previous step. This will associate the new Identity with the Application Partition that you created.



Alternatively, the following FXCLI command can be used to create a new Identity and associate it with the role that was created:

```
$ identity add --name Identity_Name --role Role_Name --password safest
```

This new identity must be set in fxpkcs11.cfg file, in the following section:

```
#HSM crypto operator identity name
<CRYPTO-OPR>     [insert name of identity that you created]     </CRYPTO-OPR>

# Production connection
<PROD-ENABLED>    YES        </PROD-ENABLED>
<PROD-PORT>        9100         </PROD-PORT>
```

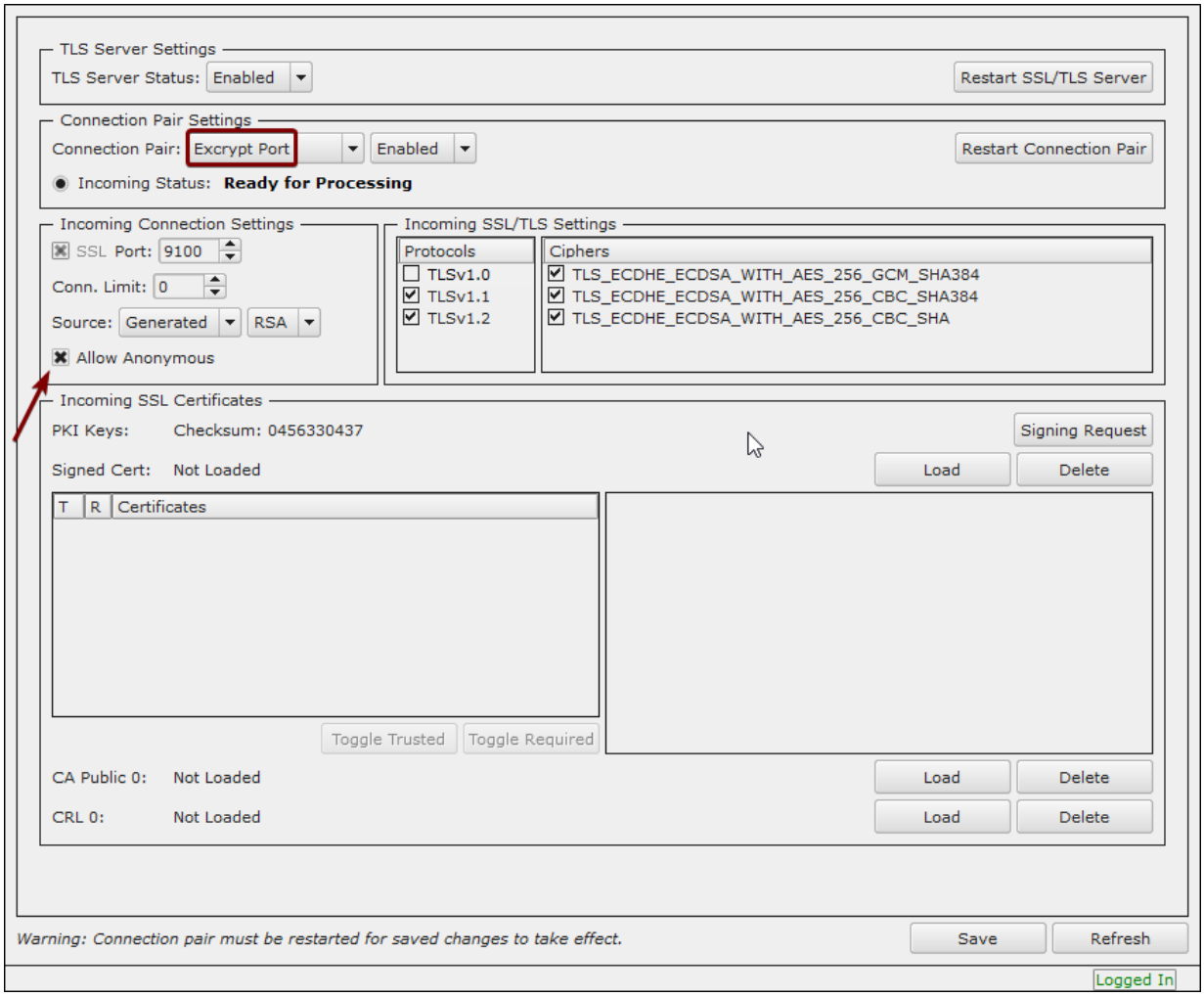
NOTE: Crypto Operator in the fxpkcs11.cfg file must match exactly the name of the identity created in the HSM.

[6.8] CONFIGURE TLS AUTHENTICATION

For this step you will need to be logged in with an identity that has a role with permissions ***Keys:All Slots, Management Commands:Certificates, Management Commands:Keys, Security:TLS Sign****, and* ***TLS Settings:Upload Key****. The default Administrator role and Admin identities can be used.*

Enable Server-Side Authentication (Option 1)

Mutually authenticating to the HSM using client certificates is recommended, but server-side authentication is also supported. To enable server-side authentication go to *SSL/TLS Setup*, then select the Excrypt Port and enable the "Allow Anonymous" setting.



Alternatively, the following **FXCLI** command can be used to enable server-side authentication with the "Allow Anonymous" SSL/TLS setting:

```
$ tls-ports set -p "Excrypt Port" --anon
```

Create Connection Certificates for Mutual Authentication (Option 2)

Mutually authenticating to the HSM using client certificates is recommended, and enforced by default. In the example below, FXCLI is utilized to generate a CA that then signs the HSM server certificate and a client certificate. The client keys and CSR are generated in Windows PowerShell with OpenSSL. For other options for managing certificates required for mutual authentication with the HSM, please review the relevant Administrator's guide.

Find the **FXCLI** program that was installed with FXTools, and run it as an administrator.

Things to note:

- For this example, the computer running FXCLI is connected to the front port of the HSM. Remote management is possible however, using the HSMs Web Portal, or the Excrypt Touch.
- For commands that create an output file, if you do not specify a file path (as is the case here) it will save the file to the directory from which the FXCLI program is executed.
- Using user-generated certificates requires a PMK to be loaded on the HSM.
- If you run **help** by itself it will show a full list of available commands. You can see all of the available options for any given command by running the command name followed by **help**.

```
# Connect your laptop to the HSM via the USB port on the front, then run this command.
$ connect usb
```

```
# Log in with both default Admin identities. This command will prompt for the username and pass-
word. You will need to run this command twice.
$ login user
```

```
# Generate TLS CA and store it in an available key slot on the HSM
$ generate --algo RSA --bits 2048 --usage mak --name TlsCaKeyPair --slot next
```

```
# Create root certificate
$ x509 sign \
    --private-slot TlsCaKeyPair \
    --key-usage DigitalSignature --key-usage KeyCertSign \
    --ca true --pathlen 0 \
    --dn 'O=Futurex\CN=Root' \
    --out TlsCa.pem
```

```
# Generate the server keys for the HSM
$ tls-ports request --pair "Excrypt Port" --file production.csr --pki-algo RSA
```

```
# Sign the server CSR with the newly created TLS CA
$ x509 sign \
    --private-slot TlsCaKeyPair \
    --issuer TlsCa.pem \
    --csr production.csr \
    --eku Server --key-usage DigitalSignature --key-usage KeyAgreement \
    --ca false \
    --dn 'O=Futurex\CN=Production' \
    --out TlsProduction.pem
```

```
# Push the signed server PKI to the production port on the HSM
$ tls-ports set --pair "Excrypt Port" \
    --enable \
    --pki-source Generated \
    --clear-pki \
    --ca TlsCa.pem \
```

```
    --cert TlsProduction.pem \
    --no-anon
```

***NOTE**: The following OpenSSL commands will need to be run from Windows PowerShell, rather than from the FXCLI program.*

```
# Generate the client keys
$ openssl genrsa -out privatekey.pem 2048
```

```
# Generate client CSR
$ openssl req -new -key privatekey.pem -out ClientPki.csr -days 365
```

Using FXCLI, sign the CSR that was just generated using OpenSSL.

```
# Sign the client CSR under the root certificate that was created
$ x509 sign  \
 --private-slot TlsCaKeyPair \
 --issuer TlsCa.pem \
 --csr ClientPki.csr \
 --eku Client --key-usage DigitalSignature --key-usage KeyAgreement \
 --dn 'O=Futurex\CN=Client' \
 --out SignedPki.pem
```

Switch back to Windows PowerShell for the remaining commands.

```
# Use OpenSSL to create a PKCS#12 file that can be used to authenticate, as a client, using our
PKCS #11 library
$ openssl pkcs12 -export -inkey privatekey.pem -in SignedPki.pem -certfile TlsCa.pem -out PKI.p12
```

[7] FUTUREX PKCS #11 CONFIGURATION AND TEST

In this section, the FXPKCS11 configuration will be described, as well as the connection test against the HSM using PKCS11Manager and EJBCA test.

[7.1] FXPKCS11 CONFIGURATION

The *fxpkcs11.cfg* file allows the user to set the PKCS #11 library to connect to the HSM. To edit, run a text editor as an Administrator and edit the configuration file accordingly. Most notably, the fields shown below must be set inside the **<HSM>** section (note that the full *fxpkcs11.cfg* file is not included).

NOTE: Our PKCS #11 library expects the PKCS #11 config file to be in a certain location (*/etc/fxpkcs11.cfg*), but that location can be overwritten using an environment variable (FXPKCS11_CFG).

Section definition for configuration:

- Log configuration includes the following four (4) options:
  - <LOG-MODE>: Sets the detail of the information to be saved in the log file for example NONE, ERROR, INFO, TRAFFIC, DEBUG, DEBUG2, DEBUG3, and ALL.
  - <LOG-FILE>: Sets the path where the log file will be stored.
  - <LOG-TRAFFIC>: Only for debug mode. This option displays information regarding the commands sent and received between the FXPKCS11 library and the HSM.
  - <LOG-TEMPLATES>: Only for debug mode. This option displays the CK_ATTRIBUTE mention in the traffic to the HSM and the application.
- Ping configuration is to maintain connection or to renew JWTs. This includes the following two (2) options:
  - <KEEPALIVE-EXCRYPT-ENABLE>: Enable with YES and Disable with NO.
  - <KEEPALIVE-EXCRYPT-INTERVAL>: Time set in seconds to have a reconnection or to renew the JWT.
- Unique connection could generate a new session to log in and will be separated from the original. (**NOTE:** This must be set to NO always.)
  - <UNIQUE-CONNECTIONS>: In order to work with EJBCA this must be set to NO.
- Excrypt Timeout section will allow us to set an alert when the connection against the HSM or Guardian is broken for a specified amount of time.
  - <EXCRYPT-TIMEOUT>: The recommended time is 2000 (which is in milliseconds).
- TLS Handshake Timeout section is for configuring the amount of time the TLS connection should be allowed to complete before assuming a timeout.
  - <TLS-HANDSHAKE-TIMEOUT>: The recommended time is 20000 milliseconds.
- Default key usage will cause symmetric and asymmetric keys that have no usages specified in their template to be set to this usage configuration. There are individual options for symmetric and for

asymmetric.

- <DEFAULT-SYMMETRIC-USAGE>: Must be set to Encrypt and Decrypt, with a pipe symbol in the middle as shown in the following example:
  - ENCRYPT | DECRYPT
- <DEFAULT-ASYMMETRIC-USAGE>: Must be set to Sign and Verify, with a pipe symbol in the middle as shown in the following example:
  - SIGN | VERIFY

Example:

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

  # Ping every 20 seconds to maintain connection and renew JWTs
  <KEEPALIVE-EXCRYPT-ENABLED>   YES        </KEEPALIVE-EXCRYPT-ENABLED>
  <KEEPALIVE-EXCRYPT-INTERVAL>  60         </KEEPALIVE-EXCRYPT-INTERVAL>

  # Object handles will be the same between instantiations of the library (HSM only)
  <PERSISTENT-OBJECT-IDS>   YES            </PERSISTENT-OBJECT-IDS>

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

  # Milliseconds to wait before an Excrypt request times out
  <EXCRYPT-TIMEOUT>         2000           </EXCRYPT-TIMEOUT>

  # 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>

  # Generate keys with login requirement even if the application does not explicitly mark them as "private"
  <KEY-REQUIRE-LOGIN>       NO             </KEY-REQUIRE-LOGIN>
</CONFIG>
```

Section definition for HSM:

- Slot configuration is where you indicate the PKCS #11 slot to work with.
  - <SLOT>: This can be left as the default value of zero (0).
- Crypto operator is the name of the Crypto Operator identity created in the HSM (see Appendix A for details). This user should be exclusive to this environment.
  - <CRYPTO-OPR>: Inside of this tag, specify the name of the identity that was created on the HSM. (**NOTE:** This field is case sensitive.)
- Connection information includes all of the following options:
  - <ADDRESS>: The IP or URL of the HSM or Guardian to connect to. In the case where a Guardian is used, the <FX-LOAD-BALANCE> option must be set to YES.
  - <PROD-PORT>: The port on the HSM or Guardian to connect to.
  - <PROD-TLS-ENABLE>: This must be set to YES in order to have a secure TLS connection. Even when connecting anonymously, this field must be set to YES.
  - <PROD-TLS-ANONYMOUS>: This option enables/disables anonymous TLS connection to the HSM. It is recommended to have this disabled in a production environment. Options that can be specified are YES or NO.
  - <PROD-TLS-CA>: Path and name of the Certificate Authority certificate. If the CA tree has multiple levels, this option should be added for each sub CA certificate.
  - <PROD-TLS-CERT>: Path and name of the signed client certificate for TLS connection.
  - <PROD-TLS-KEY>: This field defines the path of the client private key. Supported formats include PKCS #1 clear private keys, PKCS #8 encrypted private keys, or a PKCS #12 file that contains the private key and certificates encrypted under a password which can be defined in the <PROD-TLS-KEY-PASS> field.
  - <PROD-TLS-KEY-PASS>: This field defines the password for the PKCS #12 file specified in the <PROD-TLS-KEY> field (if applicable).
- Guardian balancing includes the one option below:
  - <FX-LOAD-BALANCE>: This option is used when the connection is not direct to an HSM, but rather, traffic to the HSM is being load balanced by a Guardian. The values that can be specified are YES or NO. This option can also be ignored by adding the "#" symbol at the beginning of the line.

Example:

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

  # HSM crypto operator username
  <CRYPTO-OPR>        unipagos1          </CRYPTO-OPR>

  # Connection information
  <ADDRESS>          us01hsm01test.virtucrypt.com    </ADDRESS>
  <PROD-PORT>        5216            </PROD-PORT>
  <PROD-TLS-ENABLED>    YES            </PROD-TLS-ENABLED>
  <PROD-TLS-ANONYMOUS>  NO              </PROD-TLS-ANONYMOUS>
  <PROD-TLS-CA>       /futurex/certs/L1.pem      </PROD-TLS-CA>
  <PROD-TLS-CA>       /futurex/certs/L2.pem    </PROD-TLS-CA>
  <PROD-TLS-CA>       /futurex/certs/L3.pem    </PROD-TLS-CA>
  <PROD-TLS-CA>       /futurex/certs/L4ap.pem    </PROD-TLS-CA>
  <PROD-TLS-CERT>      /futurex/certs/FX-UNIPAGOS-VCTest-signed.pem     </PROD-TLS-CERT>
  <PROD-TLS-KEY>       /futurex/certs/FX-UNIPAGOS-VCTest-privatekey.pem    </PROD-TLS-KEY>
  #<PROD-TLS-KEY-PASS>   p12pass            </PROD-TLS-KEY-PASS>

  # YES = This is communicating through a Guardian
  <FX-LOAD-BALANCE>     YES            </FX-LOAD-BALANCE>
</HSM>
```

[7.2] TEST FXPKCS11 - HSM CONNECTION

The executable file called "PKCS11Manager" is helpful for testing the connection against the HSM after configuring the *fxpkcs11.cfg* file. To run the commands below you need to first navigate in a terminal to the same directory as the PKCS11Manager application.

- To display the libraries, use `ls -lt`

```
/futurex# ls -lt
```

- Information displayed:

```
PKCS11Manager
certs
fxpkcs11.log
libfxpkcs11-Debug.so
libfxpkcs11.so
```

Once we validate the information, we run the following command:

- To run the application, we use `./PKCS11Manager`
- Main information after we execute the program.

```
| INFO  | 7F05CA3EA740 | C_Initialize: Called (Revision: aedf).
| DEBUG | 7F05CA3EA740 | C_Initialize: OS specific mutexing selected.
| DEBUG | 7F05CA3EA740 | initSlots: Called.
| INFO  | 7F05CA3EA740 | runThreads: Starting Surveyor.
| DEBUG | 7F05CA3EA740 | createThread: Called with 0x56284af93f63((nil)).
| INFO  | 7F05CA3EA740 | createThread: Created thread with ID 3393120261.
| INFO  | 7F05CA3E9700 | surveyor: Surveyor started (7F05CA3E9700).
| INFO  | 7F05CA3E9700 | getConfigFilePath: Configuration file: '/etc/fxpkcs11.cfg'.
```

- The main menu to work with the application:

```
Main Menu
     1. Print Library/Token Info

     2. Generate Key

     3. Find Objects
     4. Modify Objects
     5. Delete Objects

     6. Generate Random Data

     7. Login
     8. Logout

     0. Exit

   >>
```

With these options it is possible to login and execute various operations using the HSM, such as generating new keys of different types and querying information on the HSM. If you are able to login successfully, this confirms a successful connection between the FXPKCS11 library and the HSM.

[8] EJBCA SERVER CONFIGURATION

Connecting the EJBCA environment to the HSM via the FXPKCS11 library requires the configuration of various files.

Locate the *web.properties* file in the EJBCA path (e.g., /opt/ejbca-install/ejbca_ce_6_15_2_6/conf/web.properties) and open it in a text editor.

Add the 2 new lines shown below:

- cryptotoken.p11.lib.115.name=Futurex
- cryptotoken.p11.lib.115.file=/futurex/libfxpkcs11.so

NOTE: The number will depend on the previous libraries installed.

```
#cryptotoken.p11.lib.113.name=Cavium Nitrox III
#cryptotoken.p11.lib.113.file=/home/liquidsec_bin/lib/libliquidsec_pkcs11.so
#cryptotoken.p11.lib.114.name=AWS CloudHSM
#cryptotoken.p11.lib.114.file=/opt/PrimeKey/cloudhsm/lib/libliquidsec_pkcs11.so
cryptotoken.p11.lib.115.name=Futurex
cryptotoken.p11.lib.115.file=/futurex/libfxpkcs11.so
```

Save the file.

In order to apply these changes, rebuild the EJBCA environment using the following commands:

- Locate the *ejbca-install/* folder using the command "cd" as follows:

```
cd /opt/ejbca-install/
```

- Run the command:

```
ant -f ejbca_ce_6_15_2_6/build.xml -q clean deployear
```

- Locate the *ejbca-toolbox/* folder.

```
cd /opt/ejbca-install/ejbca_ce_6_15_2_6/
```

- Run the command:

```
ant clientToolBox
```

- Start web service with the following command:

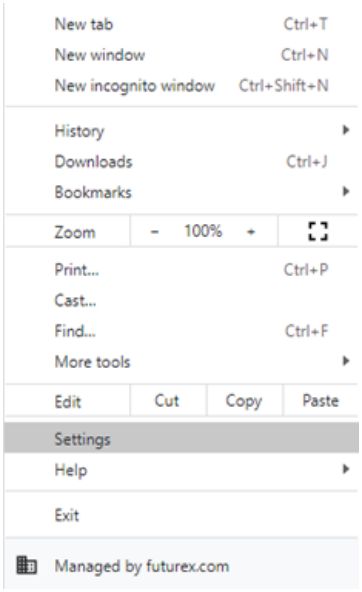
```
java -D[Standalone] -server -Xmx2048m -
Dorg.jboss.boot.log.file=/opt/wildfly/standalone/log/server.log -
Dlogging.configuration=file:/opt/wildfly/standalone/configuration/logging.properties -jar
/opt/wildfly/jboss-modules.jar -mp /opt/wildfly/modules org.jboss.as.standalone -
Djboss.home.dir=/opt/wildfly -Djboss.server.base.dir=/opt/wildfly/standalone -c standalone.xml
-b 0.0.0.0 &
```

[9] EJBCA SERVER TEST

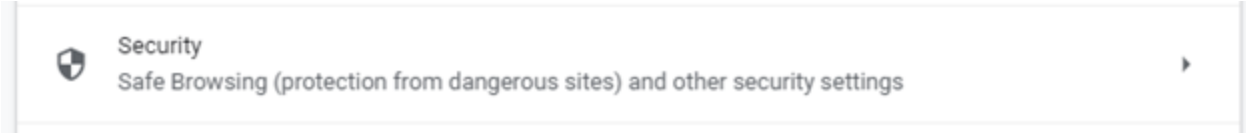
To test the web service we need to first import a certificate (generated by the EJBCA Administrator) into the web browser. The procedure to add the certificate is explained in the following steps:

NOTE: The example below was performed using Chrome.

- Open the menu and select **Settings**.

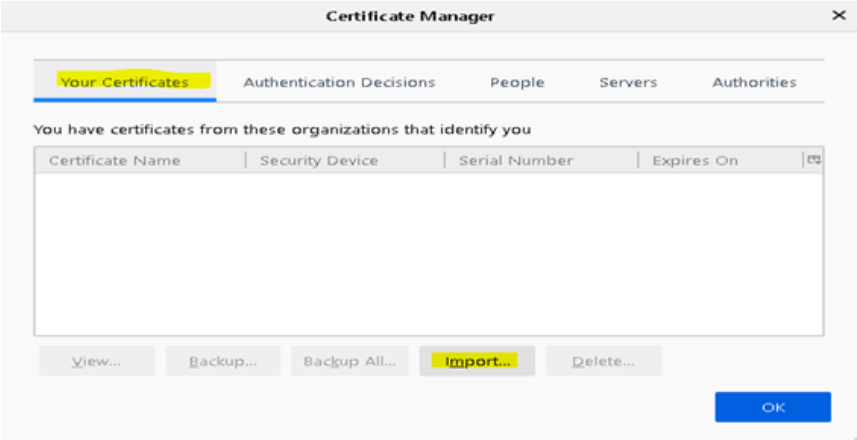


- Go to *Privacy and security -> Security*.

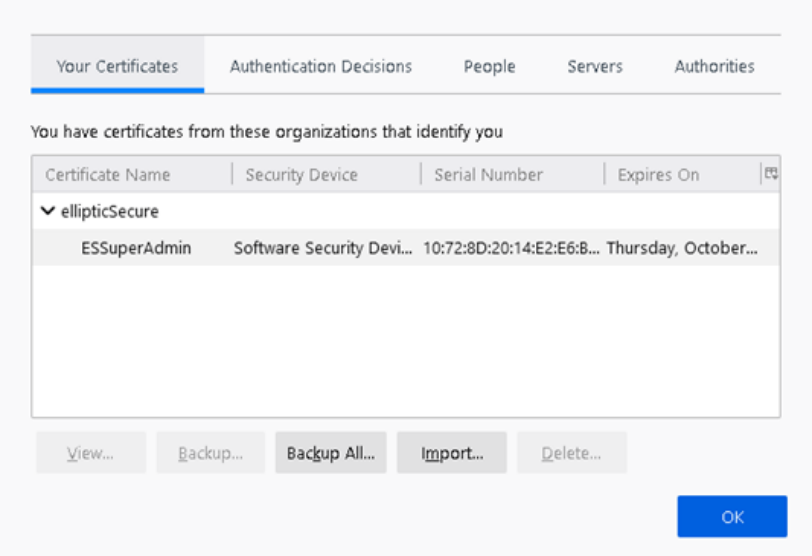


- Open the certificate settings to add the certificate and enable the communication.

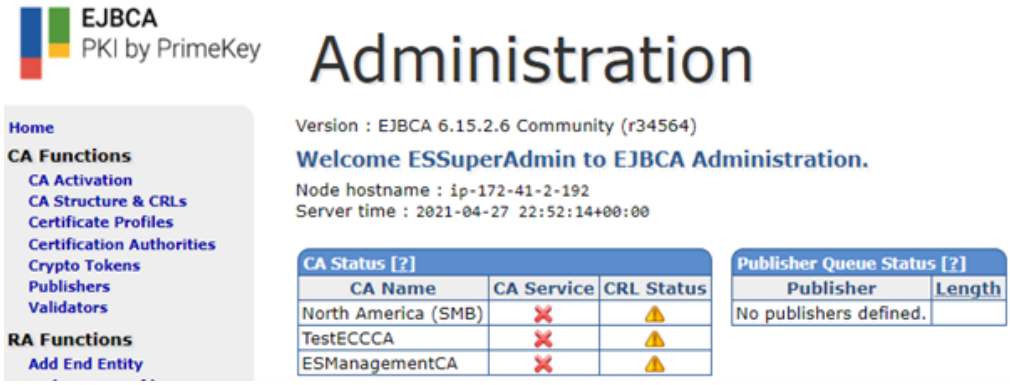
- Click on the **Import** button to add a new certificate.



- Validate the certificate generation to log in to the web service.



- Log in to the EJBCA web service at the following link: https://3.128.170.168:8443/ejbca/adminweb/

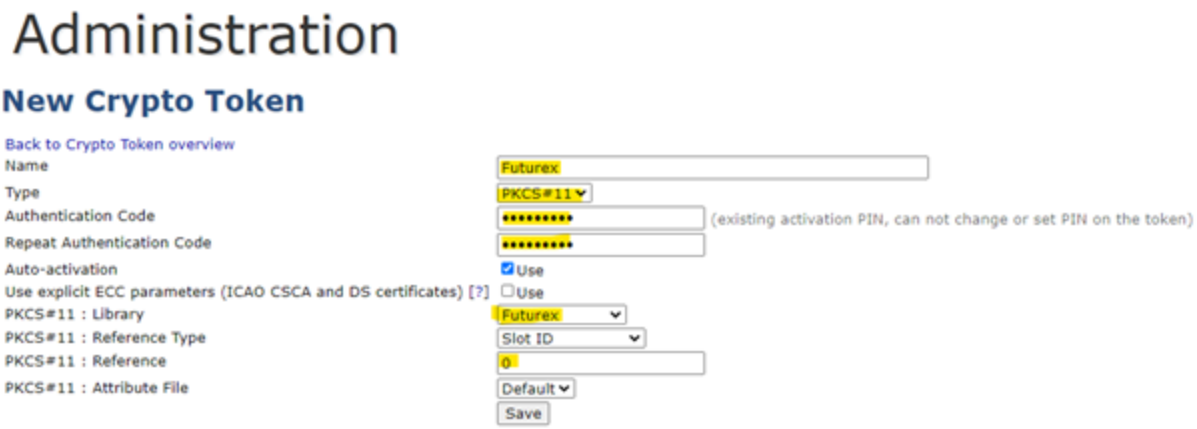


From the lefthand menu on the Administration site, certificates and Crypto Tokens can be generated to work with the HSM.

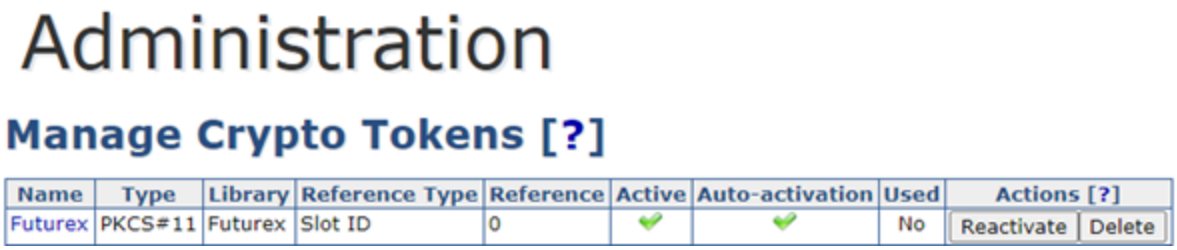
- Select the **Crypto Tokens** option in the lefthand menu:

- Click the **Create New** option. This will take bring up the following screen:

- Fill in the information as follows to create a new Crypto Token that uses the FXPKCS11 library, then click **Save**.



- Select the manager from the next menu to generate new keys:



- To generate a new key we need to specify a name and select the key algorithm, as shown below. Then click the **Generate new key pair** button.

- Click the **Test** button to verify that key pair generation is working properly.

APPENDIX A: XCEPTIONAL SUPPORT

In today's high-paced environment, we know you are looking for timely and effective resolutions for your mission-critical needs. That is why our Xceptional Support Team does whatever it takes to ensure you have the best experience and support possible. Every time. Guaranteed.

- 24x7x365 mission critical support
- Level 1 to level 3 support
- Extremely knowledgeable subject matter experts

At Futurex, we strive to supply you with the latest data encryption innovations as well as our best-in-class support services. Our Xceptional Support Team goes above and beyond to meet your needs and provide you with exclusive services that you cannot find anywhere else in the industry.

- Technical Services
- Onsite Training
- Virtual Training
- Customized Consulting
- Customized Software Solutions
- Secure Key Generation, Printing, and Mailing
- Remote Key Injection
- Certificate Authority Services

Toll-Free: 1-800-251-5112

E-mail: support@futurex.com

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