YubiHSM 2 for EJBCA Deployment Guide

Securing PrimeKey EJBCA with YubiHSM 2
Copyright
© 2019 Yubico Inc. All rights reserved.

Trademarks
Yubico and YubiKey are registered trademarks of Yubico Inc. All other trademarks are the property of their respective owners.

Disclaimer
The contents of this document are subject to revision without notice due to continued progress in methodology, design, and manufacturing. Yubico shall have no liability for any error or damages of any kind resulting from the use of this document.

The Yubico Software referenced in this document is licensed to you under the terms and conditions accompanying the software or as otherwise agreed between you or the company that you are representing.

Contact Information
Yubico Inc
530 Lytton Ave, Suite 301
Palo Alto, CA 94301
USA
yubi.co/contact

Document Release Date
March 29, 2019
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>4</td>
</tr>
<tr>
<td>Prerequisites and Preparations</td>
<td>5</td>
</tr>
<tr>
<td>Setting up a Test Installation of EJBCA</td>
<td>5</td>
</tr>
<tr>
<td>Installing YubiHSM 2 Tools and Software</td>
<td>7</td>
</tr>
<tr>
<td>Default YubiHSM 2 Default Device Configuration</td>
<td>8</td>
</tr>
<tr>
<td>Understanding Key Splitting and Key Custodians</td>
<td>9</td>
</tr>
<tr>
<td>Configuring the Primary YubiHSM 2 Device</td>
<td>10</td>
</tr>
<tr>
<td>YubiHSM 2 Configuration Steps</td>
<td>10</td>
</tr>
<tr>
<td>Verifying the YubiHSM 2 Setup</td>
<td>16</td>
</tr>
<tr>
<td>Deploying YubiHSM 2 with EJBCA</td>
<td>19</td>
</tr>
<tr>
<td>Configuring the YubiHSM PKCS #11 Library</td>
<td>20</td>
</tr>
<tr>
<td>Verifying the YubiHSM PKCS #11 library with EJBCA Client Toolbox</td>
<td>21</td>
</tr>
<tr>
<td>Configuring the EJBCA CA with YubiHSM 2</td>
<td>22</td>
</tr>
<tr>
<td>Re-deploying EJBCA with YubiHSM 2 Configuration</td>
<td>23</td>
</tr>
<tr>
<td>Adding YubiHSM 2 as Crypto Token using the EJBCA AdminWeb</td>
<td>23</td>
</tr>
<tr>
<td>Creating CA with YubiHSM 2 using the EJBCA AdminWeb</td>
<td>25</td>
</tr>
<tr>
<td>Verifying the EJBCA CA Setup</td>
<td>27</td>
</tr>
<tr>
<td>Backing Up Key Material</td>
<td>29</td>
</tr>
<tr>
<td>Backing Up the YubiHSM 2</td>
<td>29</td>
</tr>
<tr>
<td>Confirming the Duplicated YubiHSM 2</td>
<td>32</td>
</tr>
<tr>
<td>Alternative Scenarios</td>
<td>34</td>
</tr>
<tr>
<td>Migrating an Existing EJBCA Root Key to YubiHSM 2</td>
<td>34</td>
</tr>
<tr>
<td>Changing an EJBCA Crypto Token to YubiHSM 2</td>
<td>37</td>
</tr>
<tr>
<td>Subordinate CAs</td>
<td>37</td>
</tr>
<tr>
<td>Alternative Backup and Restore Procedures</td>
<td>38</td>
</tr>
<tr>
<td>Getting Help</td>
<td>39</td>
</tr>
<tr>
<td>Terminology</td>
<td>40</td>
</tr>
</tbody>
</table>
Introduction

This guide is intended to help systems administrators deploy YubiHSM 2 with YubiHSM PKCS11 Library for use with PrimeKey EJBCA in a Linux server environment. The expected outcome is that the EJBCA Certification Authority (CA) root key is created securely on a YubiHSM 2 and that a hardware-based backup copy of key materials has been produced.

These guidelines for deployment cover basic topics, and the reference system is based on a downloadable EJBCA test environment, so the instructions should be modified as required for your specific environment. It is assumed that the installation is performed on a single server destined to become a Certificate Authority root. It is also assumed that you are familiar with the concepts and processes for working with PrimeKey EJBCA.

Furthermore, this guide focuses on how to deploy EJBCA with YubiHSM 2 in a Linux server environment. For information on how to install EJBCA on a Windows Server platform, see the EJBCA installation guidelines.

We recommend that you use this guide for installing and testing the EJBCA installation and setup of the YubiHSM 2 in a test or lab environment before deploying to production. For guidance on setting up a PKI using EJBCA in a production environment, see PrimeKey’s EJBCA documentation space.

Scenario: In an EJBCA PKI environment, the CA root key must be protected in hardware.

Benefits: YubiHSM 2 guards the CA root key and protects all signing and verification services using the root key.
Prerequisites and Preparations

The audience of this document is an experienced system administrator with a good understanding of EJBCA management. In addition, it is helpful to be familiar with the terminology, software and tools specific to YubiHSM 2. As a primer for these, refer to the Terminology Used chapter.

In order to follow the steps provided in this guide, be sure to meet the following prerequisites:

- Access to EJBCA v6 or higher in a secure computer network that is physically isolated from unsecured networks such as the Internet. The EJBCA should be installed on a Linux server running a Java EE application server, using a MySQL or MariaDB database backend. The system administrator must also have elevated system privileges.

- YubiHSM 2 software and tools for Linux Ubuntu 16.04 or higher have been downloaded from the Yubico YubiHSM 2 Release page and are available on the system to be used.

- Two (2) YubiHSM 2 devices, one for deployment and one for backup in hardware.

- Your organization policies may require key custodians to be available for the YubiHSM 2 deployment. For more information about key custodians and the associated ‘M of N’ key shares, see Understanding Key Splitting and Key Custodians.

Setting up a Test Installation of EJBCA

This section is intended for users who do not have access to an EJBCA installation but who nonetheless need a test installation in order to verify the YubiHSM 2 integration. Users who already have an EJBCA installation can ignore this section and skip ahead to Installing YubiHSM 2 Tools and Software.

This section describes how to set up a test installation of EJBCA on a VirtualBox virtual machine (VM). This EJBCA test installation consists of the following components:

- EJBCA Community v6.

- Linux Ubuntu 16.04 with Xfce 4.12 desktop.

- WildFly 10 and Java 8 used for the Java application server.

- MariaDB database backend.

Follow the steps below to install EJBCA on a VirtualBox VM:

1. Download the latest version of Oracle VirtualBox from https://www.virtualbox.org/wiki/Downloads.
2. Install Oracle VirtualBox on your host operating system by using the default settings in the installation wizard.


4. Unzip ejbcav6ce-vm.zip, which results in the unpackaged EJBCA_CE_6_10.ova file, and move this file to an appropriate location. This OVA-file contains a pre-packaged VM that features a full EJBCA Community v6 install, running on Ubuntu 16.04 with Xfce 4.12 desktop, with JBoss WildFly 10 and Java 8, using a MariaDB database backend.

5. Start VirtualBox and select File > Import Appliance. Select the EJBCA_CE_6_10.ova file as Appliance to import and import the OVA-file by selecting it with the default settings. When the EJBCA is successfully imported, it appears in VirtualBox as the VM labeled "EJBCA_CE_6".

6. Start the EJBCA_CE_6 VM in VirtualBox by right-clicking or double-clicking on the EJBCA_CE_6 VM and selecting Start > Normal start. Linux Ubuntu with Xfce desktop is launched in a VM.

7. Right-click on the Xfce desktop, select Applications > Web Browser. Firefox is launched. Select the bookmark EJBCA Public Web, which opens the EJBCA web GUI at https://127.0.0.1:8080/ejbca/.

8. In the EJBCA web GUI, click the link Administration, which opens the EJBCA Administration page. In order to login, select the SuperAdmin certificate when prompted for it in Firefox. The default password of the SuperAdmin PKCS12 certificate is "foo123". If the installation is successful, the EJBCA Administration web page should be displayed as shown in the screenshot below.

![Figure 1 – The EJBCA Administration web page.](image-url)
Installing YubiHSM 2 Tools and Software

Download and install the YubiHSM 2 tools and software in order to complete the procedures in this guide. The YubiHSM Connector should be installed on the base operating system (installed on the server hardware) to get stable USB connection to the YubiHSM 2 device. The YubiHSM library and PKCS #11 library should be installed on the same operating system that EJBCA is installed on; this operating system could either be the base operating system, a VM running on the base operating system, or a separate server.

TIP: A generic prompt, $, is used in command line examples in this document. Depending on your command line application, your prompt may be different.

1. Download the installation package for Linux Ubuntu from YubiHSM 2 Releases on the Yubico Developers website. (For the specific Linux Ubuntu test installation described in Setting up a test installation of EJBCA, download the yubihsmsdk-2.0.0-ubuntu1604-amd64.tar.gz package.) Once the package has been downloaded, move the contents to an appropriate directory.

2. On your Linux Ubuntu OS, start a Terminal. (This can be done, for example, by right-clicking the Xfce desktop and selecting the option Open Terminal Here.) Navigate to the directory where the YubiHSM 2 installation package tar-file is stored.

3. Extract the YubiHSM 2 installation package tar-file by executing the sudo tar -xzf command. For example:

   $ sudo tar -xzf yubihsmsdk-2.0.0-ubuntu1604-amd64.tar.gz

4. Navigate to the YubiHSM 2 directory where the extracted files are located, for example by running the command cd yubihsmsdk/.

5. Install the YubiHSM 2 *.deb files by executing the apt command in the following sequence:

   $ sudo apt install ./libyubihsmshttp1_2.0.0-1_amd64.deb
   $ sudo apt install ./libyubihsmsusb1_2.0.0-1_amd64.deb
   $ sudo apt install ./libyubihsms_2.0.0-1_amd64.deb
   $ sudo apt install ./libyubihsms-dev_2.0.0-1_amd64.deb
   $ sudo apt install ./yubihsms-connector_2.0.0-1_amd64.deb
   $ sudo apt install ./yubihsms-shell_2.0.0-1_amd64.deb
   $ sudo apt install ./yubihsms-setup_2.0.0-1_amd64.deb
   $ sudo apt install ./yubihsms-wrap_2.0.0-1_amd64.deb
   $ sudo apt install ./yubihsms-pkcs11_2.0.0-1_amd64.deb

All the components listed above should be installed since the libraries are dependent on each other. This guide references primarily the items listed in the following table. They are included in the archive file you downloaded from the Yubico Developers website.
<table>
<thead>
<tr>
<th>Software</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>YubiHSM Connector</strong></td>
<td>Enables communication between the YubiHSM 2 and applications that use it. However, it is recommended that the YubiHSM Connector run on the host operating system if the EJBCA is deployed to a VM.</td>
</tr>
<tr>
<td><strong>YubiHSM Shell</strong></td>
<td>The administrative command line tool used to interact with and configure the YubiHSM 2 device. If the YubiHSM Shell is installed on a VM, it will connect to the Connector over a networked connection.</td>
</tr>
<tr>
<td><strong>YubiHSM Setup</strong></td>
<td>Helps with setting up a device for specific use cases. Currently supports setting up for use with PKCS #11 Library.</td>
</tr>
<tr>
<td><strong>YubiHSM PKCS#11 Library</strong></td>
<td>The YubiHSM PKCS#11 Module is a native library for interacting with a YubiHSM 2 device using the PKCS#11 interface. This library works as a translation layer between libyubihsmp and EJBCA using PKCS#11.</td>
</tr>
</tbody>
</table>

**Default YubiHSM 2 Default Device Configuration**

The YubiHSM 2 device comes with a default, factory-installed authentication key that has the password "password". As part of the configuration in this guide, the default authentication key will be destroyed. The device can be reset to its default configuration, destroying any objects stored on the device that are not factory-installed. Reset instructions can be found here.

**To verify that the YubiHSM 2 is in default configuration**

1. If EJBCA is running on a local host, the YubiHSM Connector can be started with the command `yubihsmp-connector` without additional parameters. If EJBCA is running on a VM, start the YubiHSM Connector on the host operating system in networking mode. For example, if the host machine's IP address is 192.168.100.252, the Connector should be launched on the host OS with the following command:

   $$ yubihsmp-connector -l 192.168.100.252:12345 $$

   **TIP:** For testing or debugging the YubiHSM Connector, the flag `–d` can be set.

2. To gain shell access to the YubiHSM 2, launch the YubiHSM Shell program. To do this, launch your Terminal, and run the command `yubihsmp-shell`. If a networked Connector is used, set the parameter `--connect <connector URL>`. For example:

   $$ yubihsmp-shell --connector 192.168.100.252:12345 $$

   **TIP:** For testing or debugging the YubiHSM Shell, the flag `–d` can be set.

3. To connect to the YubiHSM 2, at the `yubihsmp` command line, type `connect`. A message verifying that you have a successful connection is displayed.
4. To open a session with the YubiHSM 2, type `session open 1` (where 1 is the ID of the default authentication key pre-installed on the device).

5. Type in the default password: `password`. You will receive a confirmation message that the session has been set up successfully.

6. You now have an administrative connection to the YubiHSM 2 and can list the objects available. To list the objects, type `list objects 0` and press Enter. Your results should be similar to the following:

   id: 0x0001, type: authentication-key, sequence: 0

7. To exit type `quit`.

**Understanding Key Splitting and Key Custodians**

The preferred method for setting up YubiHSM 2 to secure EJBCA involves key splitting and rejoining, often called setting up an 'M of N' scheme (Samir's Secret Sharing (SSS)). This process ensures no individual can export key material off the device, and provides a way to control the import of key material that has been exported under wrap from one device into other devices.

A key is split among a predetermined number \( n \) of key custodians (also known as key shareholders), giving each participant its own unique share. In order to use the key, a minimum number of shares \( m \) must be present so that the key can be rejoined. This minimum number of shareholders is denoted as the *privacy threshold*. If this threshold is not attained, the key cannot be used. The number, \( n \), should be bigger than 1. The exact number of key shares and the privacy threshold ultimately depends on the requirements of your organization.

![Diagram](https://via.placeholder.com/150)

The YubiHSM Setup Tool enables you to perform the key splitting and assign shares to key custodians. If your organization has policies in place that define how this procedure should be performed, be sure you know these policies before proceeding.

To complete the setup process, you need to know who will have the role of wrap key custodians. During setup, all key custodians must be physically present to record his or her share. You should also have a predetermined practice in place specifying how the key shares must be recorded (written on paper, photographed, locally printed, or by some other means) and how they must be stored between uses (for example, offsite archive, safety deposit box, sealed envelope).
Configuring the Primary YubiHSM 2 Device

The YubiHSM Setup program, which is part of the YubiHSM 2 toolset, is used to perform the initial configuration of the primary YubiHSM 2 device. Specifically, during the setup process the YubiHSM 2 is configured so that:

- The necessary key material is generated on the YubiHSM 2 device:
  - One asymmetric key pair (mandatory)
  - One self-signed X.509 certificate with the same object ID as the asymmetric key (mandatory)
  - One wrap key (recommended)
  - One application authentication key (recommended)
  - One audit key (recommended)

- The wrap key is split among a predetermined number of key custodians, and each share can be recorded by its custodian.

- The asymmetric key, the X509 certificate, the authentication key and the audit key are exported under wrap to files located in the current working directory to which the user has write access.

The asymmetric key pair and the self-signed X.509 certificate are the mandatory objects that must be generated on the YubiHSM 2 device by the setup script. However, the wrap key, application authentication key and audit key are strongly recommended for the setup.

To safeguard the integrity of the device, it is recommended that the YubiHSM 2 configuration be performed on a fresh system in an air-gapped environment. To carry out such air-gapped installation, the steps in YubiHSM 2 Configuration Steps and Verifying the YubiHSM 2 Setup should be performed on a stand-alone computer with Linux Ubuntu 16.04 or higher and the YubiHSM 2 software installed.

YubiHSM 2 Configuration Steps

After you have inserted the primary factory preset HSM2 device into the air-gapped system, the configuration steps are summarized as follows. These steps are described in detail in the following procedure.

1. Set up communication between the YubiHSM 2 tools and the device.
2. Start the configuration process.

3. Authenticate to the YubiHSM 2 device.

4. Enter the name of the domain in which you need the asymmetric key, application authentication key, audit key and asymmetric key and certificate to be available.

5. Create the wrap key. The wrap key is very important as it allows you to export and import objects from and to the device. For example, you would export and import objects for backup purposes, as described in the section [Backup Key Material](#).

6. Split the wrap key into shares and record the number of shares required to rejoin the wrap key.

7. Create the application authentication key, which is used to authenticate to the YubiHSM 2 device through the PKCS #11 library. The password to the authentication key must be at least eight (8) characters.

8. Create the audit key to access the internal audit log of the device and to reset the audit log. The device audit log holds information about the last 62 operations.

9. Select the appropriate asymmetric key algorithm, label and domain for the EJBCA asymmetric key and self-signed certificate.

10. Create the EJBCA asymmetric key and self-signed certificate.

---

**Figure 2** – Flowchart illustrating the YubiHSM 2 setup

**Preconditions:**
- Factory preset YubiHSM 2 device available to host
- YubiHSM 2 software installed on air-gapped computer

**Postconditions:**
- EJBCA asymmetric key created
- EJBCA self-signed certificate created
- Wrapkey created and split among custodians
- Auditekey created and saved to disk under wrap
- Application Authkey created and saved under wrap
- Default Authkey deleted

---
To configure your primary YubiHSM 2 device

1. Enable communication with the YubiHSM 2 device by running the YubiHSM Connector on the system where the device is inserted.

   If the YubiHSM Connector is running on a host machine to which the YubiHSM 2 is physically connected, the Connector should be started in networked mode. For example, if the host IP address is 192.168.100.252, the Connector should be started with the following command on the host machine:

   $$ yubihs\text{-}connector -l 192.168.100.252:12345 $$

   In this scenario, you can validate that the Connector is running properly by typing the following URL into your web browser:

   http://192.168.100.252:12345/connector/status

   The output in the web browser should be similar to:

   ```
   status=OK
   serial=* 
   version=1.0.0
   pid=* 
   address=192.168.100.252
   port=12345
   ```

2. In the Linux Terminal, navigate to a directory to which you have write access. Then run YubiHSM Setup with the argument EJBCA.

   $$ yubihs\text{-}setup ejbca $$

   If EJBCA is installed on a different machine than the YubiHSM Connector, the `connector` flag needs to specify the Connector URL. For example:

   $$ yubihs\text{-}setup --connector http://192.168.100.252:12345 ejbca $$

   **TIP 1:** The YubiHSM Setup tool has a `help` argument that you can call to learn more about its usage.

   **TIP 2:** For test purposes you can set the `yubihs\text{-}setup -d` flag to keep the default authentication-key with the administrative privileges; this will allow you to delete keys on the YubiHSM 2 for testing purposes only. For production purposes, however, the `yubihs\text{-}setup` command must be executed without the `-d` flag to ensure that the factory preset authentication key is properly deleted from the YubiHSM 2 device.

   **Note:** When replacing the default authentication key using this setup, the new
authentication key will only be operable in the same domain as the asymmetric key. This is due to the domain concept that is used to compartmentalise the YubiHSM 2 device. A possible scenario for storing different keys in different domains is to have the root CA key and the EJBCA database protection key stored on the same YubiHSM 2, but accessible on different domains and with different authentication keys, thus completely separating the two keys.

3. To start the YubiHSM Setup process, type the default authentication key password "password". A message is displayed, confirming that the default authentication key was used and that you are authenticated to the device:

Using authentication key 0x0001

Object IDs are displayed in the YubiHSM Setup Tool using hexadecimal numbers. In this case the default authentication key has ID 1 (or 0x0001 in hexadecimal format).

4. The next question to be answered is what domain(s) you need the keys to be available in.

Unless you have a requirement to assign more than one domain, enter a single number between 1 and 16. In this guide, we assume that domain 1 was entered.

5. You are prompted to create a wrap key and enter its ID. Do one of the following:

- To manually assign a wrap key ID, type the number. As object ID ‘1’ is already in use by the default application authentication key, it is recommended to assign ID ‘2’ to the wrap key.

- To allow the system to generate a wrap key ID automatically, type 0.

A confirmation message is displayed:

Stored wrap key with ID 0x0002 on the device

6. You are prompted to configure splitting the wrap key among a number of key custodians. For this example, we will assume that the wrap key is split into three shares, out of which at least two shares must be present in order to rejoin the key. If there are not two key custodians present, the wrap key cannot be rejoined afterwards.

Note: Each key custodian must make sure to record the shares independently and store them safely in order to be able to restore the wrap key for this YubiHSM 2 device in the future.

TIP: For test purposes, such as in a lab scenario, it is not necessary to specify that the wrap key should be shared between key custodians. Instead, you can use a single key. To do this, when configuring the device using YubiHSM Setup, indicate the number of shares to be 1 and the privacy threshold to also be 1.
When prompted, do the following:

a. Enter the number of shares. In this example, enter 3.

b. Enter the privacy threshold. In this example, enter 2.

When the relevant prompt is displayed, each of the three wrap key custodians should take their turn in front of the screen to record their respective share.

A warning notice appears advising you that the shares are not stored anywhere.

It is important that each custodian record the whole string presented, including the prefix (in this example) “2-1-” which indicates the number of shares required to rejoin the key (or the privacy threshold) and the number identifying the share itself, i.e., its location in the sequence of shares created.

c. To start having the custodians record the key shares, press Enter.

d. The first custodian records his share and leaves the screen. The next one enters and records the key share for the second share, and so on. Each custodian confirms by pressing ‘y’ that the share was recorded before handing over to the next. The screen buffer is cleared before each share is presented. Following is an example of a share presented on the screen:

```
2-1-WWmTQj5PHGJQ4H9Y2ouURm8m75QkDOeYzFzOX1VyMpAOeF3YKYZYA...
```

```
Have you recorded the key share? (y/n)
```

7. The setup configuration continues by asking if you want to create an application authentication key. This key can be used to authenticate to the device through the PKCS #11 library, allowing EJBCA to perform operations in YubiHSM 2. Since object ID 1 and 2 are already in use by the default authentication key and the wrap key respectively, the example in this guide assumes that the application authentication key to be created gets ID 3. Do one of the following:

- To manually assign an application authentication key ID, type 3.
- To allow the system to generate a wrap key ID automatically, type 0.

You also need to choose a password for the application authentication key. Be sure to store the password of the application authentication key that you will use so that it cannot be compromised. You will need this information later to configure the PKCS #11 library for use with EJBCA. The password should be at least eight (8) characters.

Enter the application authentication key password. A confirmation message appears:

```
Stored application authentication key with ID 0x0003 on the
```
Saved wrapped application authentication key to 0x0003.yhw

The wrapped application authentication key (0x0003.yhw) is saved to the current working directory. Although encrypted using the wrap key, we recommend that you do not store keys under wrap on network-accessible or otherwise potentially compromisable storage media.

Leave the file where it was saved for now as it will be used later to create a backup.

8. The next step in the YubiHSM 2 setup process is to decide whether to create an audit authentication key. The audit key is used to access the internal audit log of the device which holds information about the last 62 operations performed. It is also used to purge the log if needed. Depending on your local requirements, you may not need to create an audit key. If you are unsure of your requirements, we suggest you create an audit key. To log into the YubiHSM 2 with this authentication key, both the key ID and the password will be needed.

When prompted to create an audit key, type ’y’. You are then prompted to assign a key ID to the audit key. Be sure to make a note of the ID you enter (for example, key ID 4). You are also prompted to enter the audit key password. Be sure to store this password as well, so that it cannot be compromised. Finally, the audit key will be exported under wrap to the current working directory. Using our example of key ID 4, the file will be named 0x0004.yhw.

9. The next step in the YubiHSM 2 setup process for EJBCA is to generate the root asymmetric key and corresponding self-signed certificate. You are prompted with a list of supported asymmetric key algorithms. Select an appropriate asymmetric key algorithm, for example rsa4096.

$ Enter asymmetric key algorithm: rsa4096

10. Next you will be prompted to enter the key label. Enter an appropriate name, for example EJBCA YubiHSM2 root CA.

$ Enter key label: EJBCA YubiHSM2 root CA

11. You are prompted for the domain(s) you need the asymmetric key and certificate to be available in.

Unless you have a requirement to assign more than one domain, enter a single number between 1 and 16. In this guide, we assume that domain 1 was entered.

This command will result in generating an asymmetric key and a self-signed certificate in the YubiHSM 2 device. This self-signed certificate is however an attestation certificate, which only wraps the public key that will later be read and parsed by EJBCA when creating the actual root CA certificate in section Creating CA with YubiHSM 2 using the EJBCA AdminWeb.
The confirmation will look like the following:

Generated asymmetric keypair with ID 0x929e on the device

Stored selfsigned certificate with ID 0x929e on the device

12. The setup tool (in default mode) finishes by letting you know that the previous authentication key has been deleted.

Previous authentication key 0x0001 deleted

All done

Finally, the YubiHSM Setup application exits. At this stage, the YubiHSM 2 device is equipped with the asymmetric key-pair and a self-signed certificate that will be used for EJBCA root CA. The device is also setup with the symmetric keys for wrap, audit, and application authentication.

If you need to generate more asymmetric keys and self-signed certificates, without creating the auxiliary wrap, audit and authentication keys, run the YubiHSM setup script once more and use the flag –a.

**Verifying the YubiHSM 2 Setup**

You can verify the results of the YubiHSM Setup program by using the YubiHSM Shell program, and logging in using the application authentication key.

**To verify the YubiHSM 2 setup**

1. In your Terminal, run the following command:

   $ yubihsms-shell

   If the YubiHSM Connector is running on a host machine to which the YubiHSM 2 is physically connected, the YubiHSM Shell program should be started in networked mode. For example, if the host server IP-address is 192.168.100.252, the YubiHSM Shell program should be started with the following command at the VM:

   $ yubihsms-shell --connector http://192.168.100.252:12345

2. To connect to the YubiHSM 2, at the yubihsms prompt, type `connect`. A message verifying that you have a successful connection is displayed.

3. To open a session with the YubiHSM 2, type `session open 3`.

4. Type in the password for the application authentication key.

   You will receive a confirmation message that session 0 has been set up successfully.
5. You now have an administrative connection to the YubiHSM 2 and can list the objects available. To list the objects, type `list objects 0`. Your results should be similar to the following:

```
Found 5 object(s)
id: 0x0002, type: wrap-key, sequence: 0
id: 0x0003, type: authentication-key, sequence: 0
id: 0x0004, type: authentication-key, sequence: 0
id: 0x929e, type: asymmetric-key, sequence: 0
id: 0x929e, type: opaque, sequence: 1
```

As you can see by looking at their IDs, these objects correspond to the wrap key, the application authentication key and the audit key that were just created.

6. To obtain more information about any one of the objects, for example, the application authentication key (object ID 3), including its capabilities, type the following command:

```
yubihs> get objectinfo 0 3 authentication-key
```

The response you receive should look similar to the following:

```
```

This indicates that YubiHSM 2 has now been configured to:

- Generate asymmetric objects
- Compute signatures using RSA-PKCS1v1.5
- Compute signatures using RSA-PSS
- Compute signatures using ECDSA
- Sign attestation certificates
- Export other objects under wrap
- Import wrapped objects
- Mark an object as exportable under wrap
In addition, this object (the application authentication key, object ID 3) also has delegated capabilities. Delegated capabilities define the set of capabilities that can be set or "bestowed" onto other objects that it creates.

7. To exit, type quit.
Deploying YubiHSM 2 with EJBCA

With a YubiHSM 2 device now configured for use with EJBCA, the next set of steps covers configuration of the YubiHSM PKCS #11 library and the deployment in an existing EJBCA environment.

 Deploying YubiHSM 2 with EJBCA consists of seven steps as follows. These steps are described in detail in the following procedure.

1. Configuring the YubiHSM PKCS #11 library for the primary YubiHSM 2 device that was setup in Configuring the Primary YubiHSM 2 Device.
2. Verifying the YubiHSM PKCS #11 library with the EJBCA Client Toolbox.
3. Configuring the EJBCA Certification Authority (CA) for use with YubiHSM 2.
4. Re-deploying the EJBCA CA with the YubiHSM 2 configuration.
5. Adding YubiHSM 2 as Crypto Token using the EJBCA AdminWeb.
6. Creating CA with YubiHSM 2 as crypto token using the EJBCA AdminWeb.
7. Verifying the YubiHSM 2 deployment using the EJBCA registration authority.

Figure 3 - Flowchart illustrating YubiHSM 2 deployment with EJBCA
The EJBCA CA that these steps are performed on is assumed to be configured as a root CA in standalone operation mode.

These instructions include steps for a basic configuration and should be performed by an experienced system administrator.

**Configuring the YubiHSM PKCS #11 Library**

The YubiHSM PKCS #11 library is a software component that is used for integrating the YubiHSM 2 device with the EJBCA CA. Hence, it is important that the YubiHSM PKCS #11 library software has been installed on the operating system as described in section [Installing YubiHSM 2 Tools and Software](#) before proceeding.

In order to configure the YubiHSM PKCS #11 library, create the configuration file `yubihsm_pkcs11.conf` in the same directory as the YubiHSM PKCS #11 library. (The default location on a Linux Ubuntu operating system is `/usr/lib/x86_64-linux-gnu/pkcs11/`). The content of the `yubihsm_pkcs11.conf` file is described at the web page [PKCS#11 with YubiHSM 2](#).

If the YubiHSM 2 device is connected to a host machine, while the EJBCA and PKCS #11 library are running on a different (for example virtual) machine, the Connector parameter must point to the networked Connector running at the host machine.

Below is a sample of a `yubihsm_pkcs11.conf` configuration file:

```plaintext
# This is a sample configuration file for the YubiHSM PKCS11 module
# Uncomment the various options as needed

# URL of the connector to use. This can be a comma-separated list
connector = http://192.168.100.252:12345

# Enables general debug output in the module
# debug

# Enables function tracing debug output in the module
# dinout

# Enables libyubihsm debug output in the module
# libdebug

# Redirects the debug output to a specific file.
# The file is created if it does not exist.
# The content is appended.
```

---

*yubico*

YubiHSM 2 for EJBCA Deployment Guide © 2019 Yubico. All rights reserved. Page 20 of 40
Next, set the environment variables `YUBIHSM_PKCS11_CONF` to point out the location and name of the YubiHSM PKCS #11 configuration file (which is `/usr/lib/x86_64-linux-gnu/pkcs11/yubihsm_pkcs11.conf` on Linux Ubuntu) and `YUBIHSM_PKCS11_MODULE` to point out the location and name of the YubiHSM PKCS #11 library (which is `/usr/lib/x86_64-linux-gnu/pkcs11/yubihsm_pkcs11.so` on Linux Ubuntu). The environment variable must be set as a system-wide permanent variable. For example, on Linux Ubuntu, a system-wide permanent environment variable can be set by creating the file `yubihsm.sh` in the directory `/etc/profile.d/` with the following content:

```bash
export YUBIHSM_PKCS11_CONF=/usr/lib/x86_64-linux-gnu/pkcs11/yubihsm_pkcs11.conf
export YUBIHSM_PKCS11_MODULE=/usr/lib/x86_64-linux-gnu/pkcs11/yubihsm_pkcs11.so
```

Login and logout in order to for the environment variables to take effect.

**Verifying the YubiHSM PKCS #11 library with EJBCA Client Toolbox**

In order to verify that the YubiHSM PKCS #11 library can be accessed from the EBCJA Java environment, it is recommended to compile and run the [EJBCA ClientToolBox](https://www.yubico.com/products/ejbcacertificates-toolbox/).

1. Launch a Terminal and navigate to the `$EJBCA_HOME/ejbca/modules/` directory.

2. Execute the following command to compile the ClientToolBox:

   ```bash
   $ ant clientToolBox
   ```
3. The compiled clientToolBox is distributed to $EJBCA_HOME/dist/clientToolBox/. Use the Terminal to navigate to this directory.

4. Execute the `ejbcaclientToolBox.sh` command with the parameter `PKCS11HSMKeyTool` to load and test the YubiHSM PKCS #11 library. For example:

   ```
   $ ./ejbcaclientToolBox.sh PKCS11HSMKeyTool test
   /usr/lib/x86_64-linux-gnu/pkcs11/yubihsm_pkcs11.so 0
   ```

5. When prompted for the password, enter the PKCS #11 password in the form of `<AuthenticationKey-ID>|<Password>`. For example, if the AuthenticationKey-ID is 0x0003 and its password is `password`, the requested PKCS #11 password becomes `0003password`.

If the YubiHSM PKCS #11 library is successfully loaded, and its connection to the YubiHSM Connector is established, the EJBCA ClientToolBox will display the supported mechanisms, available slots and perform test signature operations.

**Configuring the EJBCA CA with YubiHSM 2**

The next step is to configure the existing EJBCA CA to be used with the YubiHSM PKCS #11 library. In this section, it is assumed that the test EJBCA installation described in section Setting up a test installation of EJBCA will be used as baseline. The steps below explain how to upgrade the test EJBCA installation configuration to be used with YubiHSM 2.

1. Launch the File Manager and navigate to the $EJBCA_HOME/conf/ directory. This directory will be used throughout this section.

2. Configure the EJBCA installation according to the instructions at Upgrading EJBCA.

   **TIP:** For the pre-packaged EJBCA installation described in Setting up a test installation of EJBCA, you can copy and rename the following configuration files:

   - Copy `install.properties.sample` and rename it `install.properties`
   - Copy `ejbca.properties.sample` and rename it `ejbca.properties`
   - Copy `cesecore.properties.sample` and rename it `cesecore.properties`
   - Copy `catoken.properties.sample` and rename it `catoken.properties`
   - Copy `web.properties.sample` and rename it `web.properties`

3. Open the configuration file `web.properties` in a text editor and add the following entries:
For example:

<table>
<thead>
<tr>
<th>cryptotoken.p11.lib.&lt;id&gt;.name=&lt;YubiHSM 2 name&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>cryptotoken.p11.lib.&lt;id&gt;.file=&lt;PKCS11 library path&gt;/yubihs_pkcs11.so</td>
</tr>
</tbody>
</table>

Save and close the modified `web.properties` file.

**Re-deploying EJBCA with YubiHSM 2 Configuration**

In order for the configuration changes to take effect, the existing EJBCA application needs to be re-deployed to the Java EE application server (for example, JBoss WildFly). The command `deployear` should be used when re-deploying an existing EJBCA installation with updated configuration files. The command `deployear` will build a new `ejbca.ear` file and deploy the package at the Java EE application server. It may take up to a minute for the changes to take effect, so make sure to wait until the Java EE application server is up and running again.

1. Launch a Terminal and navigate to the `$EJBCA_HOME/` directory.

2. Run the command:

   `$ ant clean deployear`

**Adding YubiHSM 2 as Crypto Token using the EJBCA AdminWeb**

Next, the YubiHSM PKCS #11 library should be added as a Crypto Token on the EJBCA AdminWeb.

1. For this particular test installation, launch a web browser at the EJBCA server and navigate to [https://127.0.0.1:8443/ejbca/adminweb/](https://127.0.0.1:8443/ejbca/adminweb/). To login, select the SuperAdmin certificate when prompted for it in Firefox. The default password of the SuperAdmin PKCS12 certificate is “foo123”.

   **Note:** For generic installations of EJBCA AdminWeb, however, most administrators will have their own PKCS #12 files with administrator credentials at a different client. In such a scenario, the specific PKCS #12 file should be used instead and the domain name of the EJBCA server should be set in the URL above.
2. In the EJBCA AdminWeb front page, select the option Crypto Tokens in the left-hand pane and click Create new....

3. In the EJBCA AdminWeb “New Crypto Token” web page, enter the following values.

   Note: “YubiHSM2” in the “PKCS#11: Library: YubiHSM2” drop-down list below is the same value as the `cryptotoken.p11.lib.255.name` in the `web.properties` file.

   Name: <Enter the name of the YubiHSM 2 device, for example “YubiHSM2”>

   Type: PKCS#11

   Authentication Code: <Enter the YubiHSM PKCS #11 password in the form of `<AuthenticationKey-ID>|<Password>`. For example, if the AuthenticationKey-ID is `0x0003` and its password is `password`, the requested PKCS #11 authentication code becomes `0003password`.

   Repeat authentication Code: <Enter the same authentication code as above.>

   Auto-activation: <Leave blank>

   PKCS#11: Library: <Select YubiHSM2 from the drop-down list>

   PKCS#11: Reference Type: Slot ID

   PKCS#11: Reference: 0

   PKCS#11: Attribute File: Default

   ![Figure 4 – EJBCA administration of new crypto token](image)
4. In the EJBCA AdminWeb, select the option **Crypto Tokens** in the left-hand pane. Deactivate and then reactivate the YubiHSM2 crypto token that was created in the previous step.

![Figure 5 - EJBCA administration of listed crypto tokens](image)

The CA root key and certificate on the YubiHSM 2 are now ready to be used.

**Creating CA with YubiHSM 2 using the EJBCA AdminWeb**

Now the EJBCA AdminWeb will be used to create a CA that will use the YubiHSM 2 as crypto token.

1. In the EJBCA AdminWeb front page, select the option **Certification Authorities** in the left-hand pane. Under the section Add CA, enter the name of the new CA (for example “YubiHSM CA”) and press the button **Create**.

2. In the EJBCA AdminWeb page Create CA, enter the following values at minimum.

   **Note:** “YubiHSM2” in the Crypto Token drop-down list below is the same value of the `cryptotoken.p11.lib.255.name` in the `web.properties` file. Furthermore, “EJBCA YubiHSM2 root CA” in the drop-down lists below originates from the YubiHSM 2 label of the root CA key-pair that was generated in section [Configuring the Primary YubiHSM 2 Device](#).

   For the other values the default/blank settings may be used for test purposes. When setting up a production CA, however, the values should correspond to the applicable certificate policy and certificate practice statement.

   **Crypto Token:** <Select YubiHSM2 from the drop-down list>

   **default key:** EJBCA YubiHSM2 root CA
certSignKey: EJBCA YubiHSM2 root CA

crlSignKey: Use same as Certificate Signing Key (certSign key)

keyEncryptKey: EJBCA YubiHSM2 root CA

hardTokenEncrypt: EJBCA YubiHSM2 root CA

testKey: EJBCA YubiHSM2 root CA

Subject DN: <Enter the subject DN, for example: “CN=YubiHSM CA, O=Yubico, C=SE”>

Validity: <Enter the validity period as described in EJBCA AdminWeb, for example “10Y”>

When all values have been provided, click **Create**.

---

**Figure 6 – EJBCA administration of new CA**

3. The configured CA will appear in the AdminWeb under Manage Certification Authorities.
The self-signed YubiHSM CA root certificate will be stored in the EJBCA database, and the corresponding private key resides in the YubiHSM 2 device. However, the self-signed attestation certificate (that was created as part of the YubiHSM 2 setup) will remain at the YubiHSM 2 device, although it is not used by the EJBCA when issuing end-entity certificates. However, the PKCS #11 Java wrapper implementation requires that you keep the self-signed certificate in the YubiHSM 2.

**Verifying the EJBCA CA Setup**

Finally, the EJBCA CA with the YubiHSM 2 can be verified by issuing a test certificate.

1. In the EJBCA AdminWeb front page, select the option **RA Web** in the left-hand pane.

2. In the EJBCA RA web page, select **Make A New Request**.

3. In the EJBCA RA Make Request web page, enter the following test values at minimum. For test purposes, default/blank settings may be used for the other values.

   Certificate Subtype: ENDUSER (default)

   CA: YubiHSM CA

   Key-pair generation: On server

   Key algorithm: RSA 2048 bits

   CN Common Name: Yubico User

   Enrollment code: 123456
Confirm enrollment code: 123456

![EJBCA RA for issuance of test certificate](image)

*Figure 8 – EJBCA RA for issuance of test certificate*

When all values have been provided, click **Download PEM**.

4. In the pop-up window, select **Open with: View file (default)**. The test user certificate issued by the YubiHSM CA is displayed.

![Test certificate](image)

*Figure 9 – Test certificate*

When a user certificate has been successfully issued using the YubiHSM CA, the EJBCA setup has been verified.
Backing Up Key Material

We strongly recommend making a backup copy of all production objects residing on your production devices, particularly once the CA root key has been generated on the YubiHSM 2. If there is a hardware failure of the production device, having a backup ensures that you can resume operations quickly. The backup process will result in two identical YubiHSM 2 devices with the same number of objects, keys, labels, etc.

Specific recommendations for governance of your critical key material is out of scope for this guide. Make sure to design and document these security procedures in accordance with the requirements of your organization.

Backing Up the YubiHSM 2

The backup of the primary YubiHSM 2 contains duplicates of all the objects stored on the primary device that are to be exported under wrap and that are available using the same application authentication key as the one used for the primary YubiHSM 2. For instance, when following this guide, the wrap key (previously created with ID 2), the application authentication key (ID 3), the audit key (ID 4) (if created previously), and the CA root key will be duplicated onto the secondary device. The factory-installed authentication key (ID 1) on the secondary device will be destroyed. You will need assistance from the wrap key custodians to provide their respective wrap key shares, if applicable. In the example we used in this guide, 2 out of the 3 shares must be available.

Restoring a device should be performed in an air-gapped environment in order to guarantee integrity.

The backup procedure consists of the following steps, which are described in detail in the following procedure.

1. Locate, and if necessary export, the wrapped key material that was previously exported in section Configuring the Primary YubiHSM 2 Device.

2. Set up communication between the YubiHSM 2 tools and the secondary (backup) YubiHSM 2 device.

3. Start the configuration process and authenticate to the secondary YubiHSM 2 device.

4. Restore the key material onto a secondary YubiHSM 2 device.
Preconditions:
- Configured primary YubiHSM device
- Pre-configured secondary YubiHSM device inserted
- YubiHSM 2 software installed on air-gapped computer
- Set of keys from primary YubiHSM exported to disk under wrap

![Flowchart illustrating backup and recovery of YubiHSM 2 keys](image)

Figure 10 - Flowchart illustrating backup and recovery of YubiHSM 2 keys

To restore keys on the secondary YubiHSM 2 device

1. Verify that all the keys that were previously exported from the primary YubiHSM 2 under wrap are located as files in a directory to which you have read access rights.

   The YubiHSM Setup tool looks for files with the .yhw file extension in the current working directory and attempts to read and import them into the YubiHSM 2 device. The wrap key will be imported as a result of providing the wrap key shares to the tool. Given the example object IDs in this guide, the following wrapped key files should be present:

   - 0x0003-AuthenticationKey.yhw (Application authentication key under wrap)
   - 0x0004-AuthenticationKey.yhw (Audit authentication key under wrap)
   - 0x427a-Opaque.yhw (Self-signed CA certificate under wrap)
   - 0x427a-AsymmetricKey.yhw (CA root key under wrap)

   If all the necessary keys are not yet available on disk, you can export the keys under wrap by running the following command:

   $ yubihsm-setup dump

   TIP: If the initial authentication key (by default available as ID 0x0001) has been deleted, the new authentication application key should be identified with the flag yubihsm-setup --authkey. For example:

   $ yubihsm-setup --authkey 0x0003 dump

2. To begin the process of restoring the data onto the secondary YubiHSM 2, if the primary YubiHSM 2 device is inserted into your computer, remove it and insert the secondary device.
3. In your Terminal, run YubiHSM Setup with the argument restore in the directory containing the backed up *.yhw files as follows:

$ yubihsms-setup restore

TIP: For test purposes you can set the yubihsms-setup -d flag to keep the default authentication-key with the administrative privileges; this will allow you to delete keys on the YubiHSM 2 for testing purposes only. For production purposes, however, the yubihsms-setup command must be executed without the -d flag to ensure that the factory preset authentication key is properly deleted at the YubiHSM 2 device.

4. To start the YubiHSM Setup process, type the default authentication key password password. A confirmation message is displayed that the default authentication key was used and that you are authenticated to the YubiHSM 2 device:

Using authentication key 0x0001

You will now start the restore procedure, which involves providing the number of wrap key shares required by the privacy threshold defined when setting up the primary device.

5. When prompted, type the number of shares required by the privacy threshold. In this guide, we have specified that 2 shares are required to be rejoined. These must be present in order to proceed.

6. When prompted for share number 1, the wrap key custodian holding the first share inputs this information. A message is displayed that the share is received:

Received share 2-1WWmTQj5PHGJQ4H9Y2ouURm8m75QkDOeYzFz0X1VyMpAO...

7. Continue to have each wrap key custodian enter the share information for each of the wrap key shares required to rejoin the key share. Once a sufficient number of wrap key shares have been inserted by the wrap key custodians, a final message is displayed:

Stored wrap key with ID 0x0002 on the device

8. Note that the ID of the wrap key on the secondary device is the same as that for the primary device. Once the wrap key has been stored on the secondary device, the YubiHSM Setup program reads the files containing the application authentication key, the CA root key, and, if applicable, the audit key that were saved to file under wrap during the configuration of the primary device.

reading ./0x0004.yhw
Successfully imported object Authkey, with ID 0x0004
reading ./0x0003.yhw
Successfully imported object Authkey, with ID 0x0003
reading ./0x427a-AsymmetricKey.yhw
Successfully imported object Asymmetric, with ID 0x427a
Successfuly imported object Opaque, with ID 0x427a

9. If there are files containing wrapped objects with the *.yhw file extension in this directory that were exported with a wrap key other than the one reconstituted by the shares here, the setup tool attempts to read those too, but will fail gracefully. The setup tool only restores the files it can decrypt.

10. The restore process finishes and the setup tool informs you that the default, factory-installed authentication key has been deleted.

Previous authentication key 0x0001 deleted

All done

Finally, the YubiHSM Setup application exits.

**Confirming the Duplicated YubiHSM 2**

You now have a duplicate of the device configured with the three key objects you created on the primary device earlier. These are identical to the primary device that was configured earlier.

**To confirm the duplicated YubiHSM 2**

1. In your Terminal, run YubiHSM Shell program:

   $ yubihs-m-shell

2. To connect to the YubiHSM 2, at the yubihsm prompt, type `connect` and press Enter. A message verifying that you have a successful connection is displayed.

3. To open a session with the YubiHSM 2, type `session open 3` (where 3 is the ID for your application authentication key) and press Enter.

4. Type in the password for the application authentication key. You will receive a confirmation message that the session has been set up successfully.

5. To list the objects, type `list objects 0` (or instead of 0 the session number that was given to you in step 4). Verify that the secondary device now contains all of the key material that you intended to restore.

Depending on the order in which the keys under wrap were imported, the order of the enumerated keys on the secondary device may be different than on the primary device when using the `list` command. This has no practical implication and the object IDs are identical between the devices.
If you have verified that the secondary device now contains all of the key material that you intended to restore, you should now properly remove the keys under wrap currently on file. If deemed necessary, the computer's hard drive can be erased too.
Alternative Scenarios

This guide only covers basic setup and use of the YubiHSM 2 with EJBCA. Some alternative scenarios include migrating an existing CA root key to YubiHSM 2, or leveraging the YubiHSM 2 and YubiHSM PKCS #11 library in larger PKI installations using multiple hosts to serve the CA, including subordinate CAs. Although requirements for these scenarios can vary a great deal from one organization to the next, the following contains some references that might be useful when deploying YubiHSM 2.

Migrating an Existing EJBCA Root Key to YubiHSM 2

When deploying YubiHSM 2 to secure EJBCA, a CA root key might already exist, either in software or secured by hardware such as another Hardware Security Module (HSM). It is normally possible to migrate the CA root key over to the YubiHSM 2; however, depending on the pre-existing setup, the steps to take may vary.

One typical scenario is that (only) the root CA certificate and key have been exported from a third-party HSM to a PKCS #12 file that contains the private key and the certificate. In order to import the PKCS #12 file into the YubiHSM 2 and EJBCA, a combination of OpenSSL, YubiHSM Shell and EJBCA ClientToolBox is needed. Ensure that the YubiHSM 2 and EJBCA installation are isolated from the Internet when performing the import operations.

1. OpenSSL is needed to parse the PKCS #12 file into PEM-format. Download and install OpenSSL at the Linux Ubuntu machine running EJBCA. Launch a Terminal and parse the PKCS #12 file (for example root-ca.p12) into a PEM-file by using the OpenSSL command openssl pkcs12. Example:

   ```
   $ openssl pkcs12 -in root-ca.p12 -out root-ca.pem -passin
   pass:abcd1234
   ```

2. The root CA certificate must be parsed from the root-ca.pem file and saved in both binary and base64 formats. This can be done for example by copying the base64 encoded certificate between the ------BEGIN CERTIFICATE------ and ------END CERTIFICATE------ lines in the root-ca.pem file and saving the base64 encoded root CA certificate in a file called root-ca-cert.pem. Then base64 decode the file root-ca-cert.pem and save the binary root CA certificate in a file called root-ca-cert.crt, for example.

3. Next, the root CA key must be parsed from the root-ca.pem file. This can be done for example by copying the base64 encoded certificate between the ------BEGIN ENCRYPTED KEY------ and ------END ENCRYPTED KEY------ lines in the root-ca.pem file. Save the base64 encoded root CA key in a file called root-ca-key.pem, for example.

4. Next, use the YubiHSM Shell command `put_asymmetric_key` to import the PEM-file with the PEM-encoded private root key into the YubiHSM 2. To allow for RSA signing certificates
and for ECC signing certificates, the capabilities must be set to `sign-pkcs:sign-pss` respective `sign-edcsa`. If the root key should be exportable under wrap, the capabilities should also be set to `exportable-under-wrap`. Example:

```
$ yubihs> put asymmetric 0 0 rsakey 1
exportable-under-wrap:sign-pkcs:sign-pss:sign-edcsa
root-ca-key.pem
```

5. Run the `list objects` command in the YubiHSM Shell to identify the ID of the imported asymmetric key. Example:

```
$ yubihs> list objects 0
```

An example of the output is shown below, where `0xfca5` is the ID of the imported asymmetric key:

```
Found 9 object(s)
id: 0x0001, type: authentication-key, sequence: 0
id: 0x0002, type: wrap-key, sequence: 0
id: 0x0003, type: authentication-key, sequence: 0
id: 0x0004, type: authentication-key, sequence: 0
id: 0x3343, type: asymmetric-key, sequence: 0
id: 0x3343, type: opaque, sequence: 1
id: 0x929e, type: asymmetric-key, sequence: 0
id: 0x929e, type: opaque, sequence: 1
id: 0xfca5, type: asymmetric-key, sequence: 0
```

6. Import the corresponding binary root CA certificate into the YubiHSM 2 by importing the certificate file with the YubiHSM Shell command `put opaque`. Be sure to set the ID to the same value that was used for the asymmetric root key. Example:

```
$ yubihs> put opaque 0 0xfca5 ImportedRootCert 1 none
opaque-x509-certificate root-ca-cert.crt
```

7. Use the Terminal to navigate to the directory containing the compiled EJBCA ClientToolBox program (which is by default located at `~/$EJBCA_HOME/ejbca/dist/clientToolBox/`). If the EJBCA ClientToolBox is not compiled, do this according to the instructions in [Verifying the YubiHSM PKCS #11 library with EJBCA Client Toolbox](#).

Run the EJBCA ClientToolBox to test the imported root CA certificate and key to make sure they are usable from the EJBCA environment. Example:

```
$ ./ejbcaClientToolBox.sh PKCS11HSMKeyTool test
/usr/lib/x86_64-linux-gnu/pkcs11/yubihs_pkcs11.so 0
```

8. Edit the `properties` file `~/$EJBCA_HOME/ejbca/conf/catoken.properties` to configure the YubiHSM 2 settings. Example:
# YubiHSM Crypto Token:

```bash
sharedLibrary /usr/lib/x86_64-linux-gnu/pkcs11/yubihsn_pkcs11.so
slotLabelType=SLOT_NUMBER
slotLabelValue=0
```

# CA key configuration

defaultKey ImportedRootCert
certSignKey ImportedRootCert
crlSignKey ImportedRootCert
testKey ImportedRootCert

**Note:** The slot number must be set to 0.

9. In order for the configuration changes to take effect, restart the application server. (This can be done for example by re-deploying EJBCA as described in [Re-deploying EJBCA with YubiHSM 2 configuration](#).)

10. Use the Terminal to navigate to the directory with the `ejbca.sh` command (which is by default `/$EJBCA_HOME/ejbca/bin/`). Run the `ejbca.sh` command with the flags `ca` and `importca` in order to import the CA from the YubiHSM 2. Example:

    ```bash
    $ ./ejbca.sh ca importca ImportedCA --hard --cp
    org.cesecore.keys.token.PKCS11CryptoToken --ctpassword
    0001password --prop /home/ejbca/ejbca/conf/catoken.properties
    --cert /home/ejbca/MyTestFiles/root-ca-cert.pem
    ```

11. In the EJBCA AdminWeb front page, select “Crypto Tokens” in the left-hand pane, and reactivate the YubiHSM2 Crypto Token.

12. Optionally, test issuing a user certificate with the new imported CA by following the steps in [Verifying the EJBCA Certification Authority Setup](#).

If the configuration is successful, the imported root CA will appear as an active CA in the EJBCA AdminWeb “CA Activation” page. The CA's Crypto Token, which in this case is the YubiHSM 2 accessed through the PKCS #11 library, will by default be labeled as “ImportedCryptoToken<ID>”.

---

YubiHSM 2 for EJBCA Deployment Guide © 2019 Yubico. All rights reserved. Page 36 of 40
Figure 11 – AdminWeb with an imported root CA residing at the YubiHSM 2

When the YubiHSM 2 and EJBCA are properly configured with the imported CA certificate and private key, make sure to delete the PKCS #12 file from all machines and storage devices.

Changing an EJBCA Crypto Token to YubiHSM 2

The EJBCA command line tool provides the option of changing the crypto token a specific CA is using. This feature can be used if a CA is configured with soft keys in EJBCA and will be migrated to the YubiHSM 2.

Essentially, the same procedure as in Migrating an Existing EJBCA Root Key to YubiHSM 2 should be carried out; the only difference occurs at step 10 where the `ejbca.sh ca` command should be executed with the flags `ca` and `changecatoken` in order to replace the soft root CA key with the root CA key that resides in the YubiHSM 2. Example:

```
$ ./ejbca.sh ca changecatoken --caname “Imported Root CA”
--cryptotoken YubiHSM2 --execute --tokenprop
/home.ejbca/ejbca/conf/catoken.properties
```

Subordinate CAs

In order to enhance the security and scalability of your CA, consider installing the root CA on a standalone (offline) server, and use a subordinate CA for all certificate signing.

For additional information about implementing advanced configurations, see the relevant EJBCA documentation on the subject (such as the EJBCA Guide Creating CAs).

In order to setup the YubiHSM 2 for an EJBCA subordinate CA, follow the YubiHSM Configuration Steps to generate the asymmetric key-pair. If the YubiHSM 2 is already configured for the EJBCA root CA, then the wrap key, audit key and application authentication key had already been created and
there is no need to create new ones. In order to create just the subordinate CA key-pair and attestation certificate, run the `yubihsrm-setup` with the flag `-a`. For example:

```bash
$ yubihsrm-setup -a ejbca
```

## Alternative Backup and Restore Procedures

In more advanced installations, such as when the YubiHSM Setup program was not used to set up YubiHSM 2 for EJBCA, or when moving the YubiHSM 2 device containing the root CA key from one instance of EJBCA to another, see the information on the Yubico developers’ website at [Backup and Restore](https://docs.yubico.com/docs/backup/).
Getting Help

Should you require assistance when using this guide to deploy YubiHSM 2 with EJBCA, start by referencing the product documentation and currently known issues. If you need additional help, contact Yubico:

- Yubico Developers website
- Yubico Support
- Product documentation
- Known issues and limitations
# Terminology

The following terminology as it relates to YubiHSM 2 is used throughout this guide.

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default authentication key</td>
<td>Factory-installed AES key used when initializing the device. Possesses all capabilities.</td>
</tr>
<tr>
<td>Application authentication key</td>
<td>AES key used to authenticate to the device. Performs operations according to its defined capabilities.</td>
</tr>
<tr>
<td>Audit key</td>
<td>AES authentication key with rights to access audit log.</td>
</tr>
<tr>
<td>Wrap key</td>
<td>AES key used to protect key material when exporting to file from device and when importing from file to device. Key material exported under wrap will be encrypted and can only be decrypted using the wrap key.</td>
</tr>
<tr>
<td>Capability</td>
<td>A description of what operations are allowed on or with an object such as a key.</td>
</tr>
<tr>
<td>Delegated capability</td>
<td>An operation that an object is allowed to perform by virtue of receiving those permissions from the authentication key or wrap key that was used to create it.</td>
</tr>
<tr>
<td>Domain</td>
<td>A logical “container” for objects that can be used to control access to objects on the device.</td>
</tr>
<tr>
<td>Object ID</td>
<td>Object IDs are unique identifiers for any kind of object stored on YubiHSM2. An ID can range between 1 and 65535; however, the device can hold a maximum of 256 unique objects.</td>
</tr>
<tr>
<td>M of n</td>
<td>Scheme where wrap key is split into a total number of shares (n) held by key custodians, where at least a minimum number of shares (m) (sometimes this is also called ‘quorum’) is needed to use the key.</td>
</tr>
<tr>
<td>Key custodian</td>
<td>Holder of a wrap key share.</td>
</tr>
</tbody>
</table>