Yubi HSM Integration with LineLogixPC
Ellis Driver, GlueLogix Inc.
info@GlueLogix.com
www.GlueLogix.com
USA 919.342.0201
Rev A, 10Jun2026

1. Overview

An HSM (Hardware Security Module) is a hardware device that securely stores cryptographic keys, encrypting and decrypting data without giving users direct access to the keys themselves. This document will cover how to set up and connect to a Yubi HSM with a PC, how to save sensitive information to the HSM, and how to use LineLogixPC to securely encode/encrypt data with the HSM.

The Yubi HSM is a secure and effective way to store sensitive data, though it lacks a user interface and setup requires manual commands (all of which are included in this document). The Yubi HSM can store up to 256 keys, or "objects," in memory, each with a unique Object ID. The Object ID value ranges from 0x0000 – 0xFFFF, but this document will restrict itself to Object IDs below 256 (0x00 to 0xFF).

YubiHSM is available direct from the manufacturer:

https://www.yubico.com/product/yubihsm-2-series/yubihsm-2/

Make sure you get the non-FIPS version, since the FIPS version does not support AES. GlueLogix does not gain anything from your purchase.

1.1. Revision History

Rev A, 10Jun2026: Original

2. Initial Setup

This section is for users starting with a brand-new Yubi HSM device. If your Yubi HSM is already set up, **skip ahead to the next section, "Encoding on LineLogixPC with Yubi HSM".**

The first step is downloading the Yubi HSM software. Go to https://docs.yubico.com/hardware/yubihsm-2/hsm-2-user-guide/hsm2-quick-start.html#set-up-the-yubihsm-2-environment and download the latest YubiHSM software onto your LineLogixPC laptop.

- Unzip the download file and install every .msi in the folder EXCEPT the Connector (*yubihsm-connector-3.0.6-windows-amd64.msi*).
- All programs will reside in folder **C:\Program Files\Yubico**

2.1. Connecting to the HSM

- Plug the Yubi HSM into your LineLogixPC Computer. **Be careful when plugging it in!** You can accidentally trigger a physical reset. Try to handle it by the sides and **not the metal rim/touch sensor on the back**.

If you touch the metal rim while inserting Yubi HSM into a USB port for 10 seconds, you will trigger an automatic factory reset and lose all stored data/keys!!!

- When connected, your Yubi HSM should be blinking with a steady green light.
- Open a command window and run the following command to open the HSM Shell Program with direct USB access:
  - `C:\Program Files\Yubico\YubiHSM Shell\bin>yubihsm-shell.exe -C yhusb://`
- Now run the `connect` command. Your feedback should look like this:
  - ```
    C:\Program Files\Yubico\YubiHSM Shell\bin>yubihsm-shell.exe -C yhusb://
    yubihsm> connect
    Session keepalive set up to run every 15 seconds
    ```
- If you cannot connect, open the **Start Menu/Windows Tools → Services** app and make sure that the YubiHSM Connector Service is **NOT** running. If it is running after being mistakenly installed, remove it in Control Panel and double check Services.

2.2. Reconfiguring Default Password

This section is for **deleting the insecure default password** and programming **GlueLogix-specific keys into the HSM**. If your HSM is already programmed, **skip ahead to the next section.**

- When you get a new Yubi HSM, it has one password ("password") with Object ID = 1 with restricted rights and permissions. You will need to add your own password with full permissions and delete the default one.
  - Follow these commands to open a session in Yubi Shell:
    - ```
      yubihsm> connect
      Session keepalive set up to run every 15 seconds
      yubihsm> session open 1 password
      Created session 0
      ```
  - Follow these commands to set Object ID 2 to password: "`gluelogix`." This password is used by LineLogixPC.
    - `yubihsm> put authkey 0 2 yourLabel 1,2,3 all all gluelogix`
    - Where `0` = session, `2` = Object ID, `yourLabel` = Object label, `1,2,3` = domain, `all all` = all security settings, and `gluelogix` = the password required to authenticate this Object for a new Session.
    - To confirm the PUT authkey command, run the command: `get objectinfo 0 2 authentication-key`, which will output: `id: 0x0002, type: authentication-`

`key, algorithm: aes128-yubico-authentication…` followed by a long list of all capabilities.

- **It is important to delete the insecure default key "password" saved as Object ID 1!** To delete the default password saved as Object 1, open a new session with our new password "`gluelogix`" :
  - `yubihsm> session open 2 gluelogix`
  - `Created session 1`
  - `yubihsm> delete 1 1 authentication-key`
    - Where the first `1` = session, and the second `1` = Object ID
- To confirm we deleted the Object 1 data, re-send the original `session open` command:
  - `yubihsm> session open 1 password`
  - `Failed to create session: Object not found`

Note: You will need the new password "`gluelogix`" to set an encryption key in Section 4, using the Demo Key 0011..EEFF as shown. Load the Demo Key to Object 0xFF to support a unit test built into LineLogixPC. When you are ready to set your secret key, proceed to the final section of this document and store it as Object ID 0x10, or **whatever Object ID indicated by GlueLogix for your purposes**.

2.3. Testing YubiHSM with LineLogixPC

With all the above steps performed and the YubiHSM in a USB jack, start LineLogixPC with any JOB. In menu item File, Debug (not *TCU*, Debug! See screenshot in section 4.3.2), enter `YubiDiversificationUnitTest()` (copy and paste from this document to avoid typos). You should see feedback as follows. The most important line is highlighted:

```
Running Debug Command: YubiDiversificationUnitTest()

YubiHSM Unit Test Passed

Debug Result: YubiDiversificationUnitTest()

{'ciphertext': '5\x1d\xb9\x89\xa4|
\xcad\x84\xcc\xe3F\xfdZ\xe7g\xa8\xddc\xa3\xb8\x9dT\xb3|\xa8\x02G?\xda\x91u',
'K0t': 'fde4fbae4a09e020eff722969f83832b', 'K1t':
'FBC9F75C9413C041DFEE452D3F0706D1', 'Db': "\x01\x04x.!\x80\x1d\x800B\xf5NXP
A\x95\xe6n\xb9('\x80\x83\xbf\xdc\x8aZ~\x0e\r%", 'K2t':
'F793EEB928278083BFDC8A5A7E0E0D25', 'divkey':
'a8dd63a3b89d54b37ca802473fda9175', 'error': ''}
```

3. Encoding on LineLogixPC with Yubi HSM

This section will cover how to set up and encode with LineLogixPC, using secret keys stored on a Yubi HSM.

3.1. JOB File

Formatfirst: If you are not 100% certain that your tags are all at factory settings, make sure the LineLogixPC JOB application string includes: `formatfirst` to restore each tag to defaults before encoding.

- **Note: This adds ~1 second to the encoding process.**

An HSM-compatible JOB File must include the following two lines in the first section (before the [buttons] section).

- `yubihsmpath=C:\Program Files\Yubico\YubiHSM Shell\bin`
  `yubihsmshell=yubihsm-shell.exe`

3.2. DAT File

Adjust static data in your DAT File. Keys in the DAT File will be disregarded by the Macro (next section). Make sure DAT File has the following line:

- `[aeskey.picckeysareencrypted False]`

3.3. LineLogixPC Macro

Because HSM encryption operations are too flexible to be supported in the main code of LineLogixPC, your YubiHSM access will be implemented in a Macro. That is a small file of Python commands executed for every encode when the JOB includes a Macro specification. To view your JOB's Macro, click the **Set Parameters** button. The "Use Macro?" box will be checked, and a .py file is listed. Click **Browse** to set a new Macro, or **Edit** to change the current one.

You will need to edit the Macro **if the Object ID of the secure key changes**. In the example Macro line below, the **Object ID of the Encryption/Secret key is 0x10**. If that ObjectID ever changes in the YubiHSM, the Macro will need to be edited to match. See the next section for programming the encryption key.

- `dkey = HsmYubiDiversifyKey(cookedId, appId, sysId, exePath, 0x10)`

A note on diversified keys:

In the above example, the LineLogixPC Macro is using the UID (`cookedId`), Application Identifier (`appId`), and System Identifier (`sysId`) to generate a Diversified Key according to NXP's AN10922 Diversification (https://www.nxp.com/docs/en/application-note/AN10922.pdf) methodology—a process by which each card derives its **own unique cryptographic key** from a single master key, ensuring that a data breach of one card will not endanger the entire system. This is the industry standard for Symmetric Key Diversification and not limited to NXP use cases.

4. Saving the Secret Key in Yubi HSM

This section is for the user who will be **setting the secret key in the Yubi HSM**. Once the key is saved, it will be unreadable by any user, but usable by software.

Sections 4.1, 4.2, and 4.3 use the Demo Key: `00112233445566778899AABBCCDDEEFF` as Object ID =**0xFF** (255). This is just a demonstration, and **not the Key or Object ID you will use for LineLogixPC encoding!**

When you understand how to ***store***, ***check***, and ***delete*** key objects in the Yubi HSM, proceed to the final section to save your **actual encryption key** as the desired Object ID for LineLogixPC encoding.

4.1. Storing a Demo Key

Connect to Yubi HSM via the Yubi Shell program (*yubihsm-shell.exe*). If you do not have the Shell program or do not know how to access it, or open a session, please return to the **Initial Setup** and **Connecting to the HSM** sections at the beginning of this document.

Below are the commands to open the Yubi Shell program (must run in a command window, cannot double-click to start) and connect to the Yubi HSM:

- ```
  C:\Program Files\Yubico\YubiHSM Shell\bin>yubihsm-shell.exe -C yhusb://
  yubihsm> connect
  Session keepalive set up to run every 15 seconds
  ```

Open a **session (session 0)** with the password "**gluelogix**" (Object ID = **2**). Note: All the following commands are in **Session 0.** If you open a new session, ***make sure your commands match your current session value!***

- ```
  yubihsm> session open 2 gluelogix
  Created session 0
  ```

Set **hex** format:

- ```
  yubihsm> set informat hex
  yubihsm> set outformat hex
  ```

Now send a new symmetric key to Object 0xFF using the **put symmetric** command (*More information on the **put symmetric** command: https://docs.yubico.com/hardware/yubihsm-2/hsm-2-user-guide/hsm2-cmd-reference.html#put-symmetric-key-command*):

- ```
  yubihsm> put symmetric 0 0xFF DemoKey 1,2,3 encrypt-ecb,decrypt-
  ecb,encrypt-cbc,decrypt-cbc aes128 00112233445566778899AABBCCDDEEFF
  ```

  - Where `0` = Current session number from the `session open` reply
  - `0xFF` = Key ID for Demo Encoding
  - `DemoKey` = Key Label
  - `1,2,3` = Yubico domain
  - `encrypt-ecb,decrypt-ecb,encrypt-cbc,decrypt-cbc` = Capabilities
  - `aes128` = Algorithm
  - `00112233445566778899AABBCCDDEEFF` = Demo Key.

Note that the explicit permissions `encrypt-ecb,decrypt-ecb,encrypt-cbc,decrypt-cbc` prevent anyone from using the YubiHSM Wrap feature to export your secret key.

If the command works, you will see a "`Stored symmetric key 0x00FF`" response:

- ```
  yubihsm> put symmetric 0 0xFF DemoKey 1,2,3 encrypt-ecb,decrypt-
  ecb,encrypt-cbc,decrypt-cbc aes128 00112233445566778899AABBCCDDEEFF
  Stored symmetric key 0x00FF
  ```

If the command does **not** work:

- `Invalid argument`: Make sure you are in **session 0**, or update the session value. Check for typos.
- `Failed to store symmetric key: An Object with that ID already exists:` There is already a key at Object ID 0xFF. Proceed to **Deleting Keys** (final section).
- If you are having connectivity issues, your **session may have timed out.** Restart the Yubi Shell software and re-open session 0 and try again.

4.2. Checking that a Key is Present

To confirm that you have saved a Symmetric Key Object at Object ID 0xFF, run the `get objectinfo` command:

- ```
  yubihsm> get objectinfo 0 0xFF symmetric-key
  id: 0x00FF, type: symmetric-key, algorithm: aes128, label: "DemoKey",
  length: 16, domains: 1:2:3, sequence: 6, origin: imported, capabilities:
  decrypt-cbc:decrypt-ecb:encrypt-cbc:encrypt-ecb
  ```

From the result, we can see that Object 0xFF is a symmetric-key object with the label "DemoKey," and our settings from the **put symmetric** command above.

4.3. Checking that the Key is Correct

Once your Key is saved in the HSM, you will want to confirm that the key is correct, and saved with the correct settings. Once the Key is stored in HSM memory, it is no longer viewable. However, you can send a **test encryption command** and see if your data is encrypted as expected.

Run an **encrypt aescbc** command (https://docs.yubico.com/hardware/yubihsm-2/hsm-2-user-guide/hsm2-cmd-reference.html#encrypt-aes-cbc-command) and receive back an encrypted value. In the example below, we are using 16-bytes of zeroes as our vector and data. Send the following 3 commands in the Yubi HSM Shell:

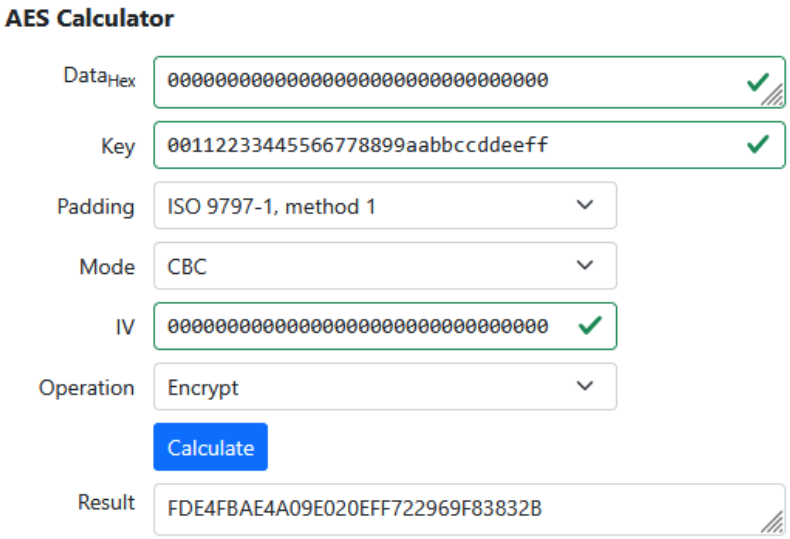
- ```
  yubihsm> set informat hex
  yubihsm> set outformat hex
  yubihsm> encrypt aescbc 0 0xFF 00000000000000000000000000000000
  00000000000000000000000000000000
  fde4fbae4a09e020eff722969f83832b
  ```

If you are not still in Session 0, correct the Session Number (the '0' between "aescbc" and "0xFF")
The returned value `fde4fbae4a09e020eff722969f83832b` is the result of an **AES CBC encryption** with an all-zero Vector, all-zero data, and our Demo Key `00112233445566778899AABBCCDDEEFF`. Now use an external AES calculator to run the same encryption, and confirm that you get the same result.
  ```

This means that your key is stored correctly. If your secret key came with a Key Check Value (KCV), that should match the first three bytes of the encryption result.

4.3.1. Checking Encrypted Values with an Online Calculator (Not Secure!)

You can use an online AES calculator to run the same encryption from the previous section and confirm your result. See example below (from http://aes.online-domain-tools.com/):



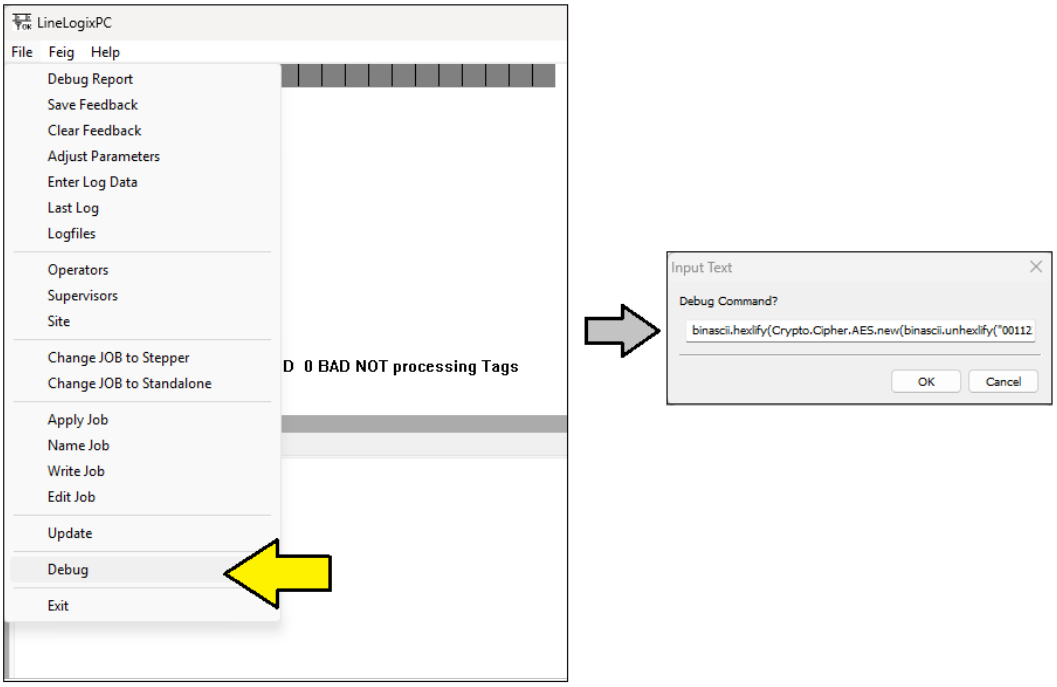
This is acceptable method to test a Demo Key. However, this method is **not secure** and exposes your key to the internet. Continue to the next section for running encryption commands offline.

4.3.2. Checking Encrypted Values with an Offline Calculator

We recommend against using online encryption calculators with <u>production keys</u>, so it is more secure to use an AES calculator hosted on your computer. Alternatively, you can send an offline encryption command with **LineLogixPC**, following the insructions below.

Running an Encryption Debug Command in LineLogixPC:

- Open any JOB File in LineLogix PC
- Go to **File → Debug**
- Paste the following **Debug Command** into the text box, replacing the **Demo Key** with your **Actual Key**.
  - `binascii.hexlify(Crypto.Cipher.AES.new(binascii.unhexlify("00112233445566778899aabbccddeeff"), Crypto.Cipher.AES.MODE_CBC, IV='\x00'*16).encrypt('\x00'*16))`



Sending a Debug Command in LineLogixPC

The command above uses the Demo Key and a 16-byte all-zero Initialization Vector (IV) to encrypt 16 bytes of all-zero Data. This is the **same command** we sent to the Yubi HSM Shell (`yubihsm> encrypt aescbc 0 0xFF 00000000000000000000000000000000 00000000000000000000000000000000`), and it should give the **same result** (`fde4fbae4a09e020eff722969f83832b`) if run with the Demo Key (`00112233445566778899aabbccddeeff`).

Debug Command result from the LineLogixPC Feedback Tab:

Debug Result:
binascii.hexlify(Crypto.Cipher.AES.new(binascii.unhexlify("00112233445566778899aabbccddeeff"), Crypto.Cipher.AES.MODE_CBC, IV='\x00'*16).encrypt('\x00'*16))
fde4fbae4a09e020eff722969f83832b

Both encryption commands output the value: "fde4fbae4a09e020eff722969f83832b," confirming that our Default Key in Object ID 0xFF is stored correctly. Repeat this process with your production key.

4.4. Deleting Keys

If there is already an Object stored with your desired Object ID, or you need to re-put your key, you must first **delete the HSM Object.** The deletion command is as follows. In the example, we are **deleting our Demo Key** from **Object 0xFF**:

- `yubihsm> session open 2 gluelogix`
  `Created session 0`
  `yubihsm> delete 0 0xFF symmetric-key`
  - Where `0` = Current session number from the `session open` reply
  - `0xFF` = Object ID
  - `symmetric-key` = Object Type (for **put symmetric** command, Object Type = `symmetric-key`)

If the object is deleted, there will be no response. A simple way to confirm deletion is to send the same delete command a second time:

- `yubihsm> delete 0 0xFF symmetric-key`
  `Failed to delete object: Object not found`

Or send the get objectinfo command and confirm that there is no Object 0xFF:

- `yubihsm> get objectinfo 0 0xFF symmetric-key`
  `Failed to get object info: Object not found`

Once Object ID 0xFF is deleted, you may put a new key.

4.5. Setting the Secret Key

When you are ready to send a production key, change the parameters of the **put symmetric** command to store your **secret key** into the desired HSM Object ID for LineLogixPC encoding. Below we are loading the Secret Key into **Object ID 0x10.**

Note: Confirm your **Object ID**! It may not be 0x10! Store your secret key as **whatever Object ID has been indicated by GlueLogix for your purposes.**

Double-check your PUT command before sending it! It is crucial that the key's value and settings are correct and addressed to the correct session number and Object ID.

- `yubihsm> put symmetric 0 0x10 YourKey 1,2,3 encrypt-ecb,decrypt-ecb,encrypt-cbc,decrypt-cbc aes128 ******************************`

  - Where `0` = Current session number from the `session open` reply
  - `0x10` = Key ID for LineLogixPC Encoding
  - `YourKey` = Key Label
  - `1,2,3` = Yubico domain
  - `encrypt-ecb,decrypt-ecb,encrypt-cbc,decrypt-cbc` = Capabilities
  - `aes128` = Algorithm
  - `******************************` = Input your **Actual Secret Key!**

*(More information on the **put symmetric** command:*
https://docs.yubico.com/hardware/yubihsm-2/hsm-2-user-guide/hsm2-cmd-reference.html#put-symmetric-key-command