YubiHSM 2 BYOK User Guide for Azure

Introduction

To enable organizations to own and manage their own encryption keys in cloud/hybrid environments YubiHSM 2 supports ‘Bring Your Own Key’ (BYOK) for Azure. This capability enables organizations to securely and cost-effectively store and transfer data in a cloud environment using an on-premises YubiHSM 2 for secure management of cryptographic credentials – enabling regulatory compliance, enhanced data security in a cloud or hybrid environment, and better control over data portability, at a fraction of cost of traditional on-premise HSMs.

This guide shows how to integrate the YubiHSM 2 with BYOK for Azure.

Integrating YubiHSM 2 With Azure BYOK

BYOK specifications for Azure can be found here.

For detailed information on YubiHSM 2 and the associated commands used in this guide, please refer to YubiHSM 2 documentation here.

Prerequisites

1. YubiHSM device with firmware 2.4.0 or later.
2. Download the latest YubiHSM 2 SDK from here.
3. Install the YubiHSM Connector and the YubiHSM Shell from the YubiHSM 2 SDK.
4. Generate the keys on YubiHSM 2 that need to be imported into Azure Key Vault.

   RSA4096 key example:

   ```
   $ yubihsm> generate asymmetric 0 0x100 "YubiHSM Signing Key
   Azure KV" 1 exportable-under-wrap,sign-pkcs,sign-pss rsa4096
   ```

   EC-P256 key example:

   ```
   $ yubihsm> generate asymmetric 0 0x100 "YubiHSM Signing Key
   Azure KV" 1 exportable-under-wrap,sign-ecdsa ecp256
   ```

   Refer to YubiHSM 2 documentation on details related to creating keys in the YubiHSM 2.

5. Ensure that you have a Microsoft Azure account and it has been activated. Go to https://azure.microsoft.com for more information.

For the examples in this guide:

1. The name of the Azure Resource Group (RG) in Azure is “kms-rg”.
2. The name of Azure Key Vault is “AzureKeyVault1”.
3. The name of the YubiHSM 2 BYOK tool script is “YubiHSM_BYOK_Tool.sh”.
4. The object ID for the YubiHSM asymmetric key you want to bring to the Azure Key Vault is 0x100.
5. The object ID for the Key Exchange Key (KEK) once imported to the YubiHSM is 0x101.
6. We assume the YubiHSM Connector is listening on http://127.0.0.1:12345. Make sure to change the IP and port according to your configuration.

Azure Steps

1. Open an Azure BASH shell session.
2. Run the command below to launch the Nano editor:

   ```
   nano ./YubiHSM_BYOK_Tool.sh
   ```

3. Copy and paste the script code below into the Nano document - DO NOT COPY THE "### BEGIN" and "### END" LINES:

   ```
   ### BEGIN ----------
   #!/bin/bash

   #
   # k    - Azure KEK key ID
   # f    - Input wrapped key file
   # Sample:
   # ./YubiHSM_BYOK_Tool.sh
   # -k
   https://kv1-west.vault.azure.net/keys/mskek2048/118275477bd2454
   fbad778bcf3490645
   # -f assymmetrickey.wrapped

   while getopts "":k:f:"" opt; do
     case $opt in
       k) k=""$OPTARG""
       ;;
       f) f=""$OPTARG""
       ;;
       \?) echo ""Invalid option -$OPTARG"" >&2
       ;;
   ```

```
  esac
done

printf ""\nAzure KEK Key ID:            %s\n"" ""$k""
printf   ""Input wrapped key file name: %s\n"" ""$f""
printf   ""BYOK file name:              %s\n"" ""$f.byok""

openssl base64 -A -in $f -out $f.base64
sed -i '1s/^/\""/' $f.base64
#echo ""\"""" >> $f.base64

cat > $f.tmp1 << EOF
{
""schema_version"": ""1.0.0"",
""header"":
{
""kid"": ""$k"",
""alg"": ""dir"",
""enc"": ""CKM_RSA_AES_KEY_WRAP""
},
""ciphertext"":
EOF

cat $f.tmp1 $f.base64 > $f.tmp2

cat > $f.tmp3 << EOF
"",
""generator"": ""YubiHSM BYOK Tool v1.0; YubiHSM 2.4.0""
}
EOF

cat $f.tmp2 $f.tmp3 > $f.byok

printf ""BYOK file contents:\n""
cat $f.byok

rm -f $f.base64
rm -f $f.tmp*
### END ----------
```

4. From the Azure BASH shell, run the command below to make the script file executable:

   ```
   chmod a+x YubiHSM_BYOK_Tool.sh
   ```

5. Create an Azure Resource Group for the Key Vault (eg. kms-rg).

6. Create a Key Vault (eg. AzureKeyVault1) in the new Azure Resource Group you just created.
7. Create an RSA Key pair in the Key Vault. The public key of this key pair is used to wrap the key from YubiHSM 2 for import into the Azure Key Vault. This key pair is called the Key Exchange Key or KEK.

   Note: KEKs can only be RSA keys (4096-bit, 3072-bit or 2048-bit keys).

   ```
   az keyvault key create --kty RSA-HSM --size <Key_Size> --name
   <KEK_Name> --ops import --vault-name <Key_Vault_Name>
   ```

   Example:

   ```
   az keyvault key create --kty RSA-HSM --size 4096 --name
   kek0x101 --ops import --vault-name AzureKeyVault1
   ```

8. Record the key identifier information, which is encoded as the value for the “kid " parameter in the command output in step 7 above. This will be needed in later steps.

   Example:

   ```
   "kid":
   "https://AzureKeyVault1.vault.azure.net/keys/kek0x101/83957cf8d
   03b4d81af1c1d420d68a534"
   ```

9. Retrieve the public key of the KEK.

   ```
   az keyvault key download --name <KEK_Name> --vault-name
   <Key_Vault_Name> --file <KEK_Public_Key_PEM_File>
   ```

   Example:

   ```
   az keyvault key download --name kek0x101 --vault-name
   AzureKeyVault1 --file kek0x101.publickey.pem
   ```

10. Download the .pem file with the KEK’s public key to the machine with the YubiHSM 2.

YubiHSM 2 Steps on the Local Machine

1. Start the YubiHSM Connector:

   ```
   $ yubihsm-connector -d
   ```

2. Check the status of your connector and device by using a browser to visit http://127.0.0.1:12345/connector/status.

3. Start the YubiHSM Shell:

   ```
   $ yubihsm-shell
   ```

4. Connect to YubiHSM 2:

   ```
   $ yubihsm> connect
   ```

5. Open a session with YubiHSM 2.

   ```
   $ yubihsm> session open <Authentication_Key_Object_ID>
   <password>
   ```

   Example:

   ```
   session open 1 password
   ```

6. Import the KEK’s Public Key into the YubiHSM 2.

   If the key you want to bring to the Key Vault is an RSA key, use the command below:

   ```
   $ yubihsm> put pub_wrapkey <Session_Id> <KEK_Pub_Key_Object_Id>
   "<KEK_Pub_Key_Label>" <Domain> export-wrapped
   exportable-under-wrap,sign-pkcs,sign-pss,
   decrypt-pkcs,decrypt-oaep <KEK_Public_Key_PEM_File>
   ```

   Example:

   ```
   $ yubihsm> put pub_wrapkey 0 0x101 "Azure KEK" 1 export-wrapped
   exportable-under-wrap,sign-pkcs,sign-pss,
   decrypt-pkcs,decrypt-oaep kek0x101.publickey.pem
   ```

   If the key you want to bring to the Key Vault is an EC key, use the command below:

   ```
   $ yubihsm> put pub_wrapkey <Session_ID> <KEK_Pub_Key_Object_Id>
   "<KEK_Pub_Key_Label>" <Domain> export-wrapped
   exportable-under-wrap,sign-ecdsa,sign-eddsa, derive-ecdh
   <KEK_Public_Key_PEM_File>
   ```

   Example:

   ```
   $ yubihsm> put pub_wrapkey 0 0x101 "Azure KEK" 1 export-wrapped
   exportable-under-wrap,sign-ecdsa,sign-eddsa, derive-ecdh
   kek0x101.publickey.pem
   ```

7. Export the asymmetric key object from YubiHSM 2 that needs to be imported into Azure Key Vault. This is the target key used in Azure to perform cryptographic operations. The asymmetric key will be exported under wrap (encrypted) using the KEK:

   ```
   $ yubihsm> get rsa_wrapped_key <Session_ID>
   <KEK_Pub_Key_Object_Id> asymmetric-key <Asym_BYOK_Key_Object_Id>
   aes256 rsa-oaep-sha1 mgf1-sha1 <Asym_BYOK_Key_File_Name>
   ```

   Example:

   ```
   $ yubihsm> get rsa_wrapped_key 0 0x101 asymmetric-key 0x100
   aes256 rsa-oaep-sha1 mgf1-sha1 key0x100_kek0x101.wrapped
   ```

8. Upload the wrapped key file to Azure.

Additional Azure Steps

1. From the Azure BASH shell, run the command below. Replace the <*kid*> with the value identified previously. The script will create a file with the same name as the file created in the previous step, adding a “.byok” extension.

   ```
   ./YubiHSM_BYOK_Tool.sh  -k <kid> -f <Asym_BYOK_Key_File_Name>
   ```

   Using the previous example “kid” and example file name:

   ```
   ./YubiHSM_BYOK_Tool.sh  -k
   https://AzureKeyVault1.vault.azure.net/keys/kek0x101/83957cf8d0
   3b4d81af1c1d420d68a534 -f key0x100_kek0x101.wrapped
   ```

2. Import the wrapped asymmetric key into the Azure Key Vault. From the Azure BASH shell, run the command below:

   ```
   az keyvault key import --vault-name <Key_Vault_Name> --name
   <Asym_BYOK_Key_Name> --byok-file <Asym_BYOK_Key_File_Name>.byok
   --kty RSA --ops sign verify
   ```

   Example:

   ```
   az keyvault key import --vault-name AzureKeyVault1 --name
   key0x100 --byok-file key0x100_kek0x101.wrapped.byok --kty RSA
   --ops sign verify
   ```