Microsoft

Azure Key Vault BYOK

2.87.0

Integration Guide

u.trust GP HSM

6.5.0

utimaco®

Imprint

Copyright 2026	Utimaco IS GmbH Krefelder Straße 220 52070 Aachen Germany
Phone	AMERICAS +1-844-UTIMACO (+1 844-884-6226) EMEA +49 800-627-3081 APAC +81 800-919-1301
Internet	https://support.hsm.utimaco.com/
e-mail	support@utimaco.com
Document Version	1.0.0
Date	2026-07-01
Status	PUBLISHED
Document No.	IG-2026-0062

Table of Contents

1 Introduction

This guide is part of the information and support provided by Utimaco. Additional documentation produced to support your Utimaco SecurityServer product can be found in the document directory of the Utimaco SecurityServer product bundle.

All Utimaco SecurityServer product documentation is available from Utimaco's website at *https://utimaco.com/*.

1.1 About This Guide

This guide provides instructions for integrating Utimaco u.trust General Purpose Hardware Security Module (GP HSM) with Microsoft Azure Key Vault using the Bring Your Own Key (BYOK) approach. It describes the process of securely generating and transferring cryptographic keys from the Utimaco HSM into Azure Key Vault while ensuring compliance with security and key management best practices.

1.2 Target Audience

This guide is intended for Azure administrators and HSM administrators.

1.3 Purpose of the Integration

The purpose of this integration is to enable secure generation and management of cryptographic keys within the Utimaco u.trust GP HSM, while allowing their controlled usage in Azure Key Vault through the Bring Your Own Key (BYOK) model, ensuring enhanced security, compliance, and ownership of key material.

1.4 Abbreviations

The following abbreviations are used in this guide:

Abbreviation	Meaning
API	Application Programming Interface

Abbreviation	Meaning
BYOK	Bring Your Own Key
CD	Compact Disc
CLI	Command line interface
CNG	Cryptography API Next Generation
CXI	Cryptographic eXtended Services Interface
GUI	Graphical User Interface
HSM	Hardware Security Module
JCE	Java Cryptography Extension
KEK	Key Exchange Key
LAN	Local Area Network
MBK	Master Backup Key
PCIe	PCI Express Interface
PKCS#11	Public-Key Cryptography Standard #11
RSA	Rivest-Shamir-Adleman
SKU	Stock-Keeping-Unit

Abbreviation	Meaning
ECC	Elliptic Curve Cryptography
RHEL	Red Hat Enterprise Linux
NIST	National Institute of Standards and Technology
CentOS	Community ENTerprise Operating System

Table 1: List of abbreviations

1.5 Document Conventions

The following conventions are used in this guide:

Convention	Use	Example
Bold	Items of the Graphical User Interface (GUI), e.g., menu options	Select **Details** and click on **Properties** button
`Monospaced`	Code that is given for explanation or as an example, file paths	You will find the file `example.conf` in the `/exmp/demo/ directory`.
Italic	References and important terms	*https://utimaco.com/*

Table 2: Document conventions

Special icons are used to highlight the most important notes and information.

Here you will find important safety information that should be followed.

Here you will find additional notes or supplementary information.

This message marks the result expected after the successful execution of an instruction.

2 Product Overview

2.1 Azure Key Vault

Azure Key Vault is a cloud service for securely storing and accessing "secrets". A secret is any information that you want to carefully control access to. Such examples can be API keys, passwords, certificates, or cryptographic keys. The Key Vault service supports two types of containers: vaults and managed HSM pools. Vaults support storing software and HSM backed keys, secrets, and certificates. Managed HSM pools only support HSM-backed keys which need an HSM. See *Azure Key Vault REST API* overview for complete details.

2.2 Utimaco u.trust GP HSM

The u.trust GP HSM is a hardware security module developed by Utimaco IS GmbH. The u.trust GP HSM is a physically protected, specialized computer unit designed to perform sensitive cryptographic tasks and to securely manage, as well as store, cryptographic keys and data. It can be used as a universal, independent security component for heterogeneous computer systems.

2.3 Joint Value Proposition

This integration combines the secure key management capabilities of Utimaco u.trust GP HSM with the scalability and flexibility of Azure Key Vault, enabling customers to maintain full control over their cryptographic keys while leveraging cloud-based services for application security and data protection.

3 Integration Requirements and Prerequisites

Ensure that the system environment you will be using meets the following hardware and software requirements. This guide assumes that you already have a working Azure subscription and users have been created on the portal to be able to configure the Azure Key Vault.

3.1 Tested Versions

The Azure Key Vault BYOK integrations that have been successfully tested with the Utimaco HSM.

Operating System	Software Requirements	Utimaco Security Server Version	Utimaco HSM
Windows 10 Enterprise Rocky Linux 9.6	BYOK tool 1.0.0	SecurityServer V6.5.0	u.trust GP HSM CSe-Series/Se-Series

Table 3: List of tested versions

This integration is not supported with external keystore.

3.2 Software Requirements

Software	Software Requirements
Operating system	Windows 64 bit, Linux 64bit
BYOK tool	byoktool 1.0.0 developed by Utimaco
Java	Version 8, Update 271 or higher

Software	Software Requirements
P11tool2	PKCS#11 administration tool developed by Utimaco. For EC keys, release 4.40 or higher is required.
Microsoft Azure CLI	Version 2.87.0 or higher. For EC keys, version 2.19 or higher is required

Table 4: List of software requirements

3.3 Hardware Requirements

Hardware	Hardware Requirements
Utimaco LAN HSM	u.trust GP HSM CSe-Series/Se-Series LAN with firmware SecurityServer 6.5.0 or higher
Utimaco PCI-e HSM	u.trust GP HSM CSe-Series/Se-Series PCI-e with firmware SecurityServer 6.5.0 or higher

Table 5: List of hardware requirements

3.4 Prerequisites

Before you begin, ensure that you have following prerequisites completed:

- Set up and configure the Utimaco HSM. Refer to the u.trust GP HSM documentation to set up the HSM.
- Create and store the MBK onto each HSM. Refer to the u.trust GP HSM documentation to set up the MBK.
- Replace the u.trust GP HSM Default Admin with a new admin user.
- Install and set up the operating system listed in *Tested Versions*.
- Install and set up the u.trust GP HSM as listed in *Tested Versions*.

- Set up an admin user, as this is required for installing software.

4 Installing and Configuring Utimaco SecurityServer Software

This section describes the process of configuring the Utimaco PKCS#11 Provider for Azure BYOK.

4.1 On Linux

4.1.1 Download and Install Utimaco Software

If you have not already done so, please create and request an Utimaco Support Portal Account. This will allow you to download the software components needed for this installation.

1. Copy the downloaded software at the appropriate location on the Server.
2. Create an `utimaco` folder under the `/opt` directory and further create 2 directories:
    a. `/opt/utimaco/bin`
    b. `/opt/utimaco/lib`

`›_ Console`

```
# mkdir -p /opt/utimaco/bin # mkdir -p /opt/utimaco/lib
```

3. Copy pkcs11 library file `libcs_pkcs11_R3.so` from Utimaco CryptoServer software to the `/opt/utimaco/lib` directory.

`›_ Console`

```
# cp ~/path_to_application_folder/lib/libcs_pkcs11_R3.so /opt/utimaco/lib
```

4. Copy the `csadm` and `p11tool2` files from Utimaco CryptoServer software to `/opt/utimaco/bin` directory and make both files executable.

```
›_ Console
```

```
# cd ~/path_to_application_folder

# cp csadm p11tool2 /opt/utimaco/bin

# chmod +x /opt/utimaco/bin/csadm /opt/utimaco/bin/p11tool2
```

4.1.2 u.trust GP HSM PKCS#11 Configuration

1. Create the directory `/etc/utimaco`. Locate the Utimaco PKCS#11 configuration file in your SecurityServer directory, `Linux/x86-64/Crypto_APIS/PKCS11_R3/sample`. Copy the Utimaco PKCS#11 configuration file `cs_pkcs11_R3.cfg` into the `/etc/utimaco` directory.

```
›_ Console
```

```
# mkdir /etc/utimaco

# cd <install directory>/Software/Linux/x86-64/Crypto_APIs/PKCS11_R3/sample
# cp cs_pkcs11_R3.cfg /etc/utimaco

# cd /etc/utimaco
```

2. Edit the `cs_pkcs11_R3.cfg` file and make the appropriate changes to the file.

`cs_pkcs11_R3.cfg`

```
[Global]
# For unix:
Logpath = /tmp
# Loglevel (0 = NONE; 1 = ERROR; 2 = WARNING; 3 = INFO; 4 = TRACE)
Logging = 1 Keepalive = true
# Set the Device to connect with [CryptoServer]
# Device specifier Device = <HSM_IP>
```

This integration is not supported with external keystore.

For more information regarding the commands and command parameters, please check the Utimaco CryptoServer documentation. The device may be a CryptoServer (PCIe or LAN) device. The device line will follow one of these patterns, based on the HSM form-factor:

Device = 288@<HSM IP address> Hardware (LAN) HSM

or

Device = /dev/cs2.0 Hardware (PCIe) HSM

To make your testing easier, it would be good to enable the PKCS#11 log file. That can be enabled by editing the Logging Loglevel. Set the **LogPath** and Logging **Loglevel** to **1**. For testing you may want to increase it to **4**.

The added **LogPath** points to a writable directory, not to a file.

If you encounter problems, check the log file named `cs_pkcs11_R3.log` in the **LogPath** defined directory. When you are done testing, you should change **Logging** to **1** or **2**. This will limit the logging to only critical and important messages.

4.1.3 Create SO, User and Initialize a Slot

You must initialize a slot with a custom label using `p11tool2`.

First using `p11tool2` create, the SO or Security Officer and then using the `p11tool2` command initialize the Slot that you want to use, as well as the slot user, as shown below.

`›_ Console`

```
# ./p11tool2 slot=<slot_no> Label=<token_label> Login=ADMIN,ADMIN.key
InitToken=<SO_PIN>

# ./p11tool2 slot=<slot_no> LoginSO=<SO_PIN> InitPin=<CryptoUser_PIN>
```

4.2 On Windows

On Windows, as part of CryptoServer software installation, `cs_pkcs11_R3.cfg` will be automatically created and will be available under the `C:\ProgramData\Utimaco\PKCS11_R3` folder.

4.2.1 Update cs_pkcs11_R3.cfg

Example Values

`cs_pkcs11_R3.cfg`

```
[Global]

# For windows:

Logpath = C:/ProgramData/Utimaco/PKCS11_R3

# Loglevel (0 = NONE; 1 = ERROR; 2 = WARNING; 3 = INFO; 4 = TRACE)

Logging = 1

# Prevents expiring session after inactivity of 15 minutes KeepAlive = true

# Set the Device to connect with [CryptoServer]

# Device specifier Device = <HSM_IP>
```

This integration is not supported with external keystore.

For more information regarding the commands and command parameters, please check the Utimaco CryptoServer documentation. The device may be a CryptoServer (PCIe or LAN) device. The device line will follow one of these patterns, based on the HSM form-factor:

Device = 288@<HSM IP address> Hardware (LAN) HSM

OR

Device = /dev/cs2.0 Hardware (PCIe) HSM

To make your testing easier, it would be good to enable the PKCS#11 log file. That can be enabled by editing the Logging Loglevel. Set the **LogPath** and Logging **Loglevel** to **1**. For testing you may want to increase it to **4**.

The added **LogPath** points to a writable directory, not to a file.

If you encounter problems, check the log file named `cs_pkcs11_R3.log` in the **LogPath** defined directory. When you are done testing, you should change **Logging** to **1** or **2**. This will limit the logging to only critical and important messages.

5 Implementing BYOK

Please, make sure that you change the labels in brackets to what suits your use case the best.

Example: Change <resource_group> to AzureKeyVaultGroup

5.1 Azure CLI Installation

The version of the Azure CLI used in this guide is 2.87.0. For more information regarding the most recent release, please see the *Azure CLI Release Notes* from Microsoft.

5.1.1 Azure CLI Installation on Linux

1. Import the Microsoft repository key.

```
›_ Console
```

```
> rpm --import https://packages.microsoft.com/keys/microsoft.asc
```

2. For RHEL 9, CentOS Stream 9, Rocky Linux 9, or other RHEL-compatible distributions, add the `packages-microsoft-com-prod` repository.

```
›_ Console
```

```
> dnf install -y https://packages.microsoft.com/config/rhel/9.0/packages-microsoft-prod.rpm
```

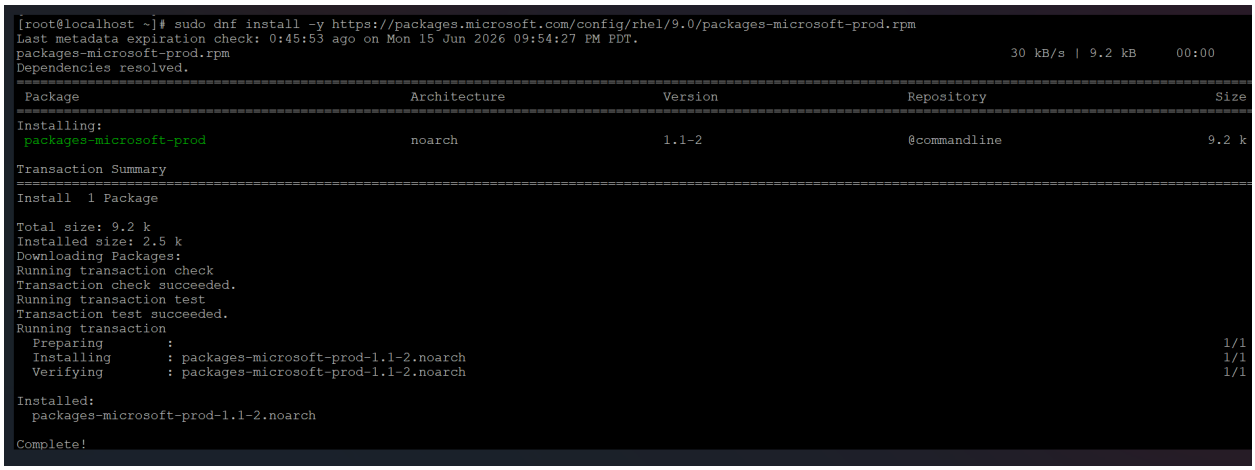

Figure 1 : Microsoft packages

3. Install `azure-cli` with the `dnf` install command.

`›_ Console`

```
> dnf install azure-cli
```

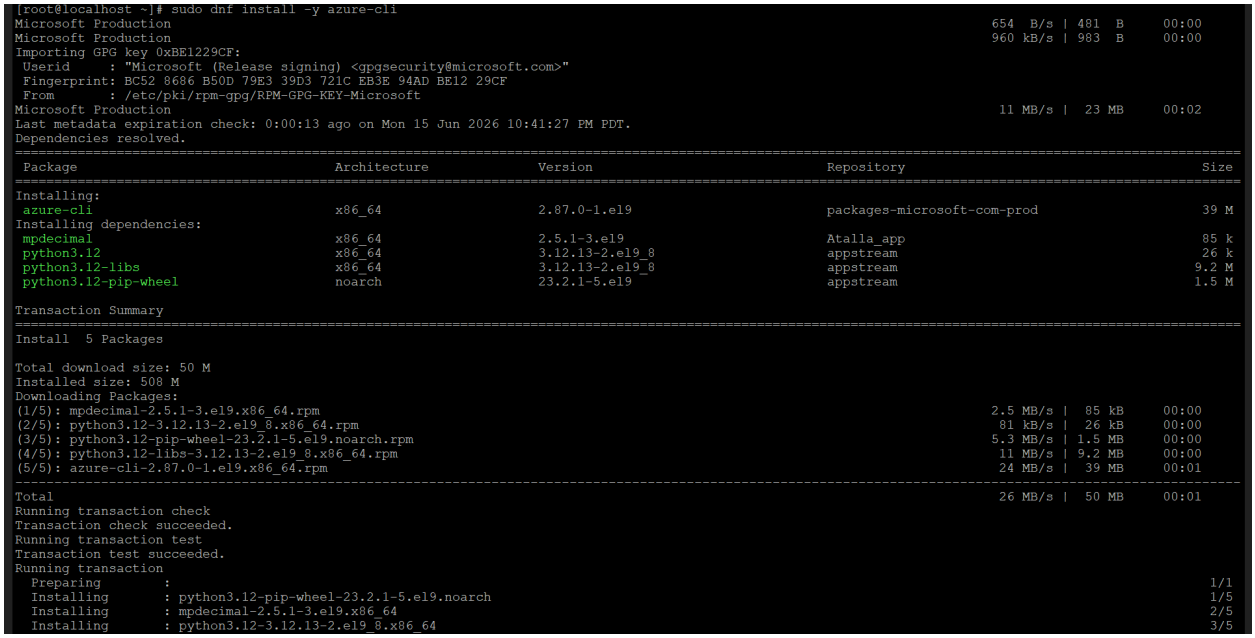

Figure 2 : Installing Azure CLI packages

5.1.2 Azure CLI Installation on Windows

1. Download the Azure CLI latest version from the above link and click **Install**.

Figure 3 : Installing Azure CLI

2. Click on the **Finish** button to complete the setup.

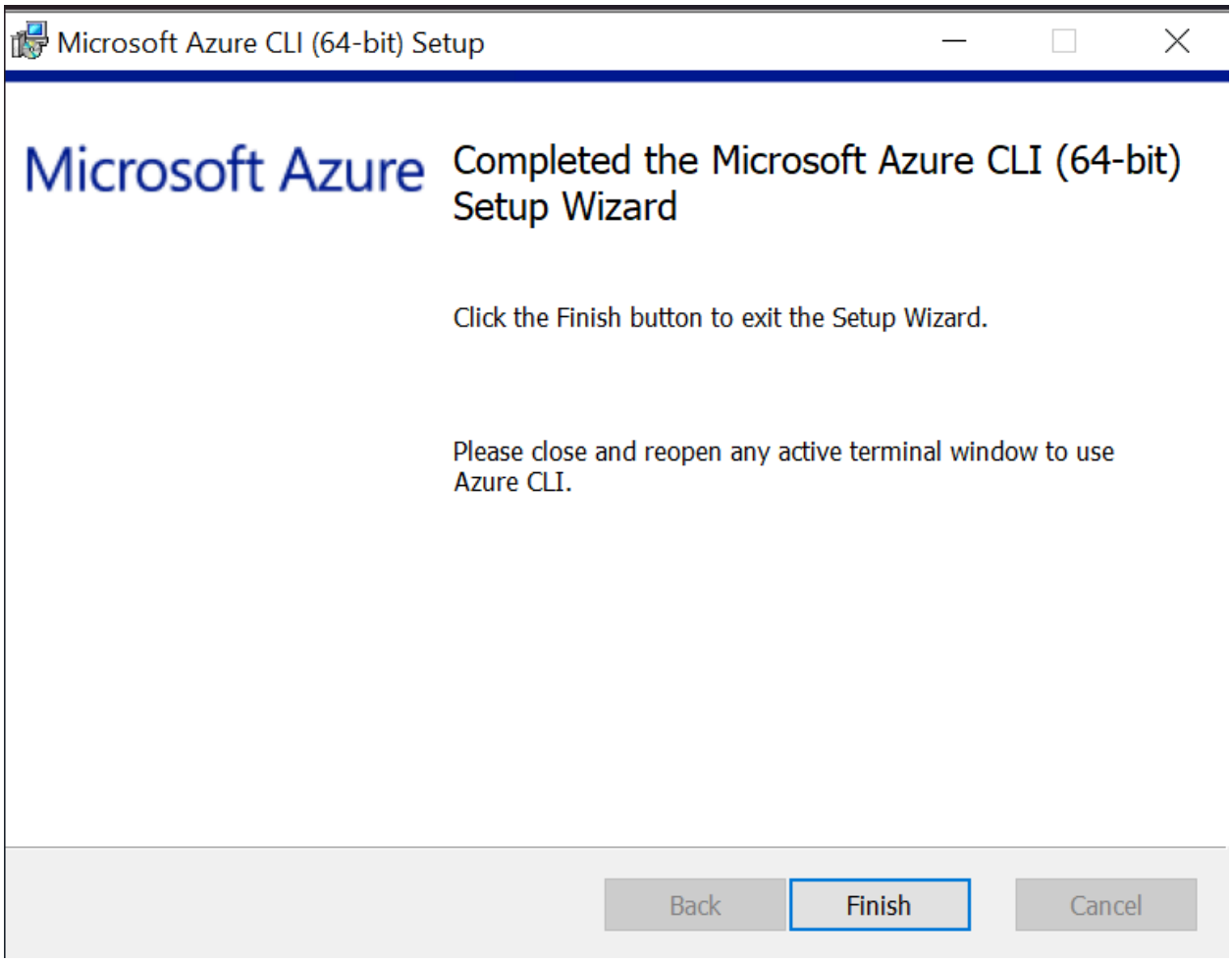

Figure 4 : Azure CLI setup

5.2 Azure Key Vault Premium Subscription

An active Azure subscription with a Premium tier key vault is required to be able to create HSM protected keys.

5.3 Download Utimaco byoktool

To simplify the key export and import process of tenant keys, Utimaco has created a Bring Your Own Key (BYOK) tool. Please reach out to Utimaco so this tool can be provided to you. The byok tool supports all key types (PKCS#11, CNG, JCE, CXI). The storage of keys is still restricted to the internal storage on the Utimaco CryptoServer HSM. The BYOK tool does not support key

creation, only migration. That is why it is important that the settings of the keys' attributes, that you would like to migrate, are set to extractable.

5.4 Creating a Key Vault in Azure

5.4.1 Creating Azure Key Vault with Azure Portal

1. Navigate to portal.azure.com and log in to your Azure Account.
2. Locate the **Key vaults** Azure service, click on it to get redirected to your key vaults.

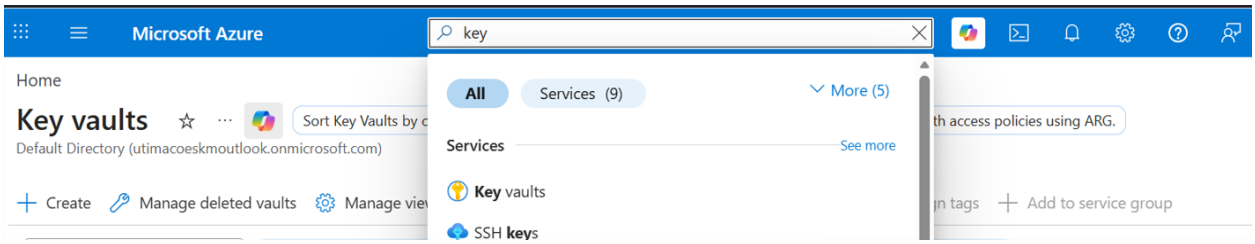

Figure 5 : Azure services

3. Click on **+ Create** to start the process of creating a key vault.

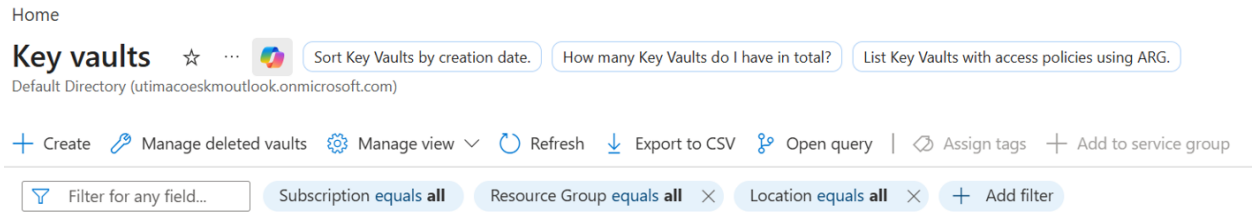

Figure 6 : Home page of Azure Key Vaults

4. Select your subscription and the resource group. Enter the **Key vault name** and **Region** for the key vault you will be using. Make sure to select the **Premium** pricing tier to include support for the HSM backed keys. Set the **Recovery options** according to your company's policies.

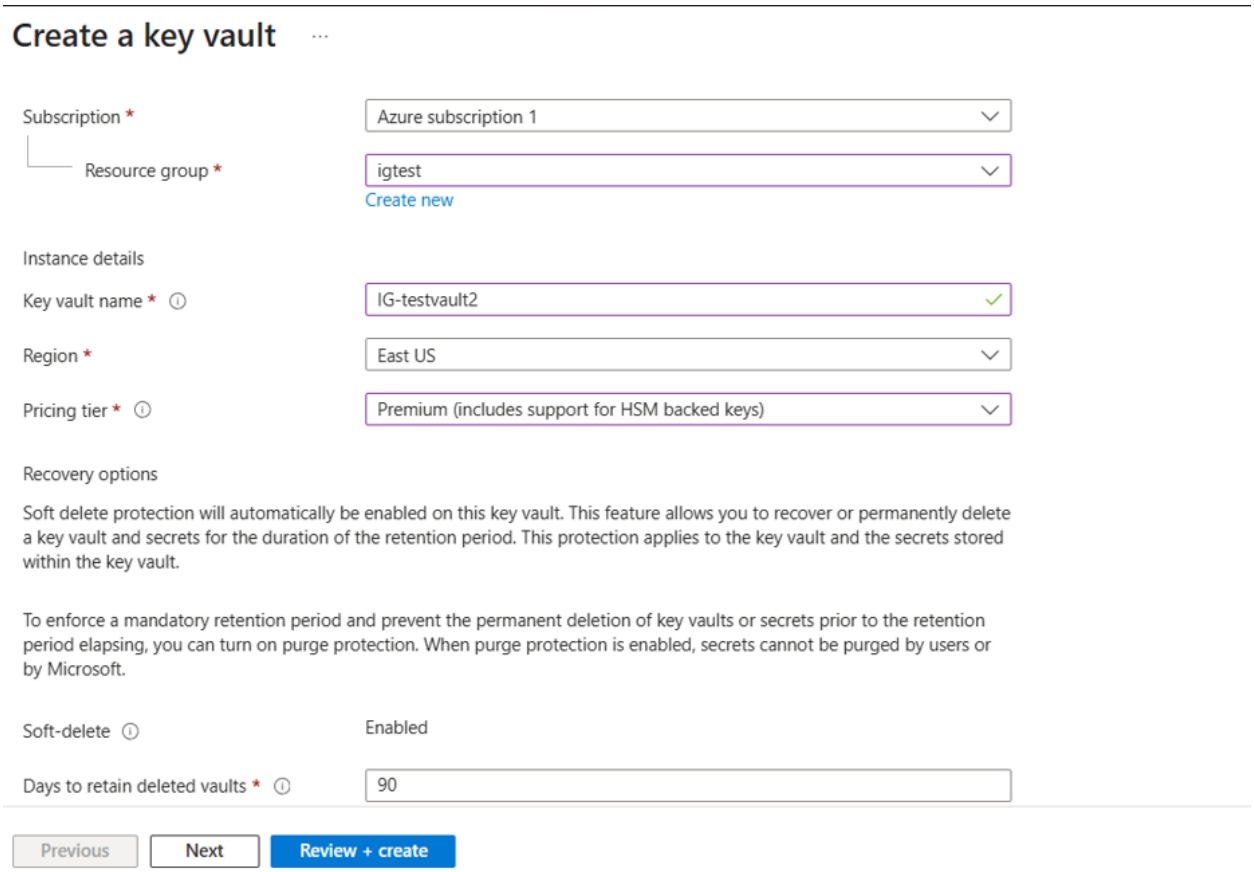

Figure 7 : Create key vault page

5. Enable access to your selected users and administer the policy which is the most suitable for your company.

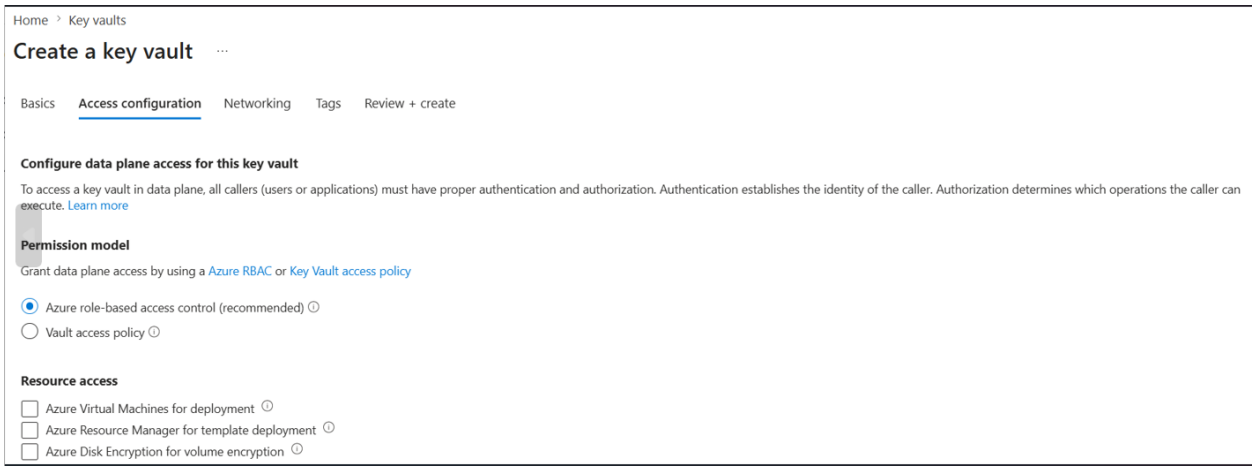

Figure 8 : Create key vault access policy page

6. Select your connectivity method.

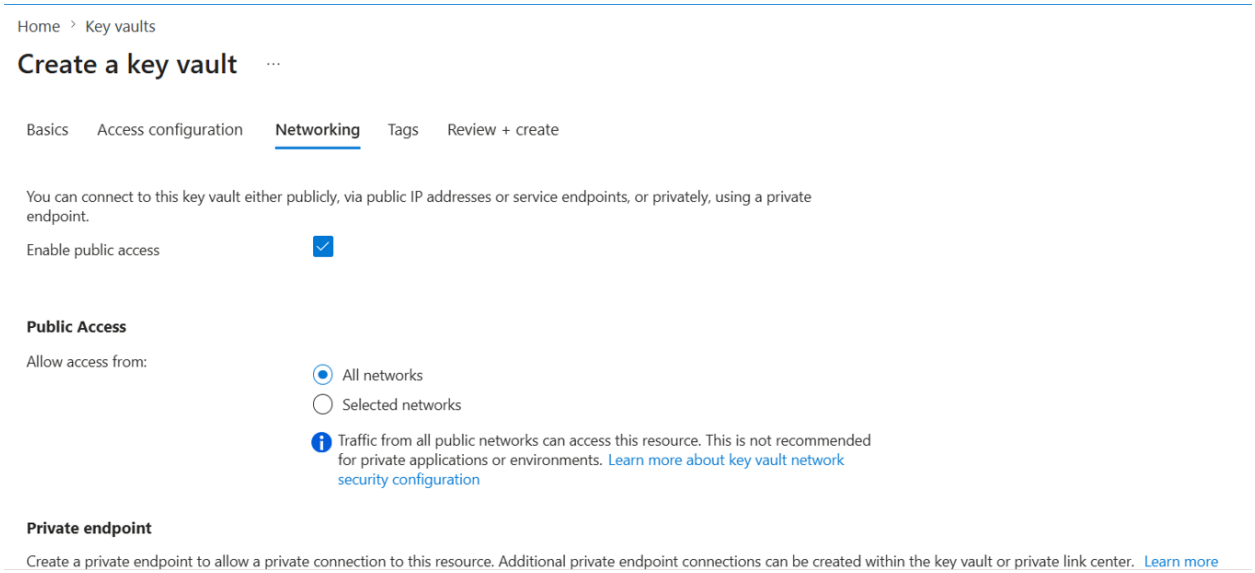

Figure 9 : Create key vault networking page

7. If needed, add tags to categorize resources and view consolidated billing.

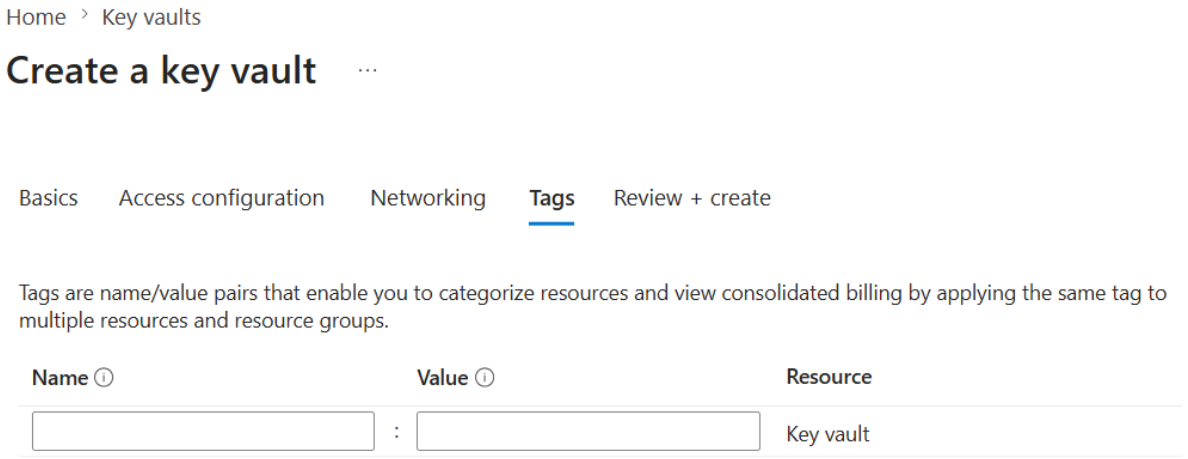

Figure 10 : Create key vault tags page

8. Review the creation of the key vault. After reviewing, click on **Create** to finish the creation of the key vault.

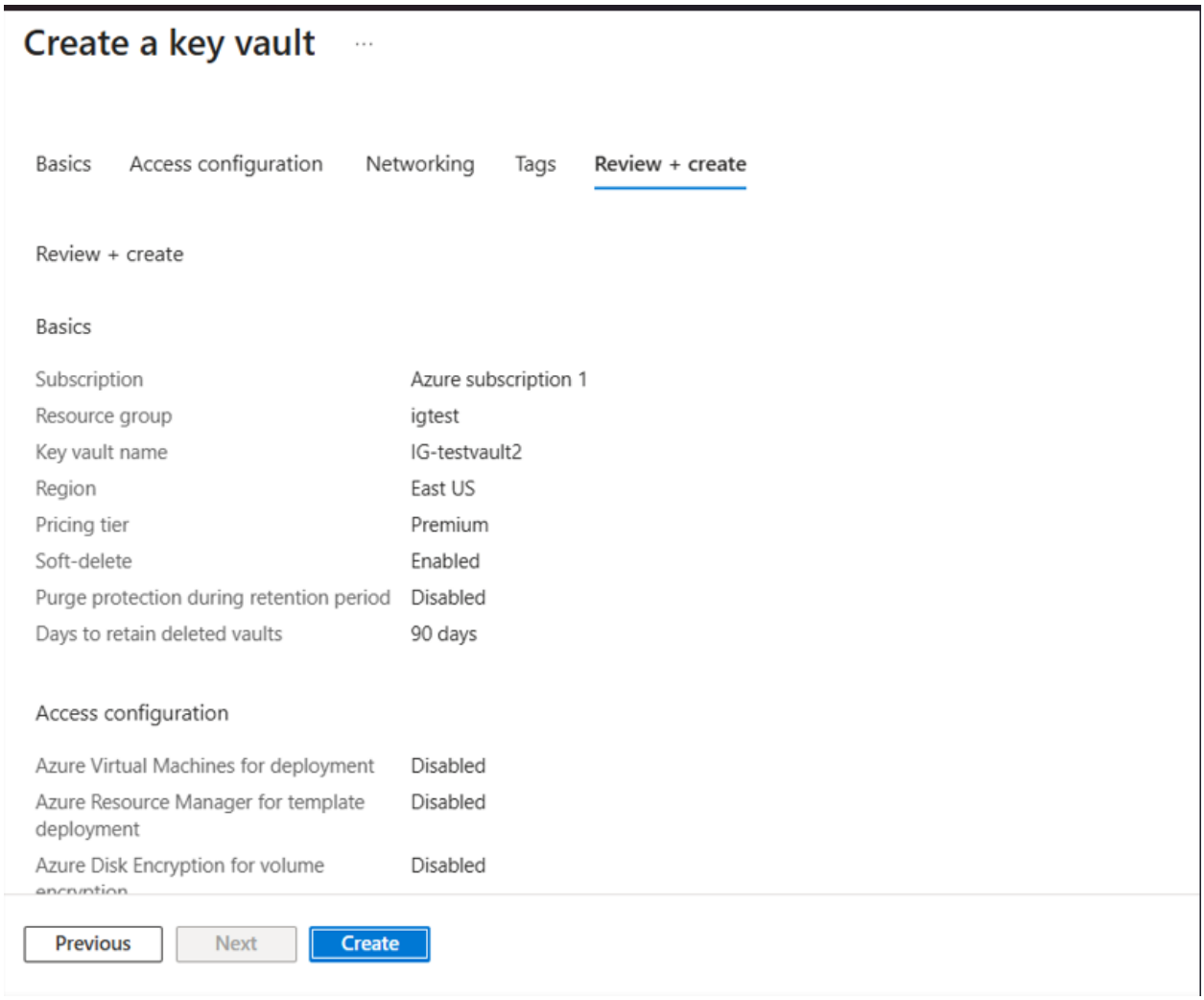

Figure 11 : Create key vault review and create page

5.4.2 Creating Azure Key Vault on Linux

1. Open the command line interpreter and log in to Azure with the following command:

```
›_ Console
```

```
# az login
```

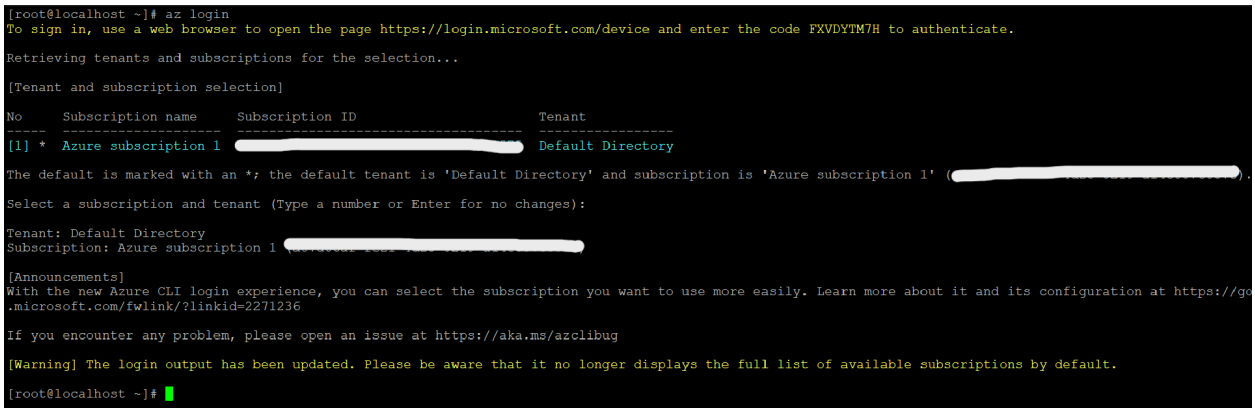

Figure 12 : Azure login

2. To create the Azure Key Vault, you will need to create a resource group. Use the following command to create your own resource group.

```
›_ Console
```

```
# az group create --location "<location>" --name "<resource_group>"
```

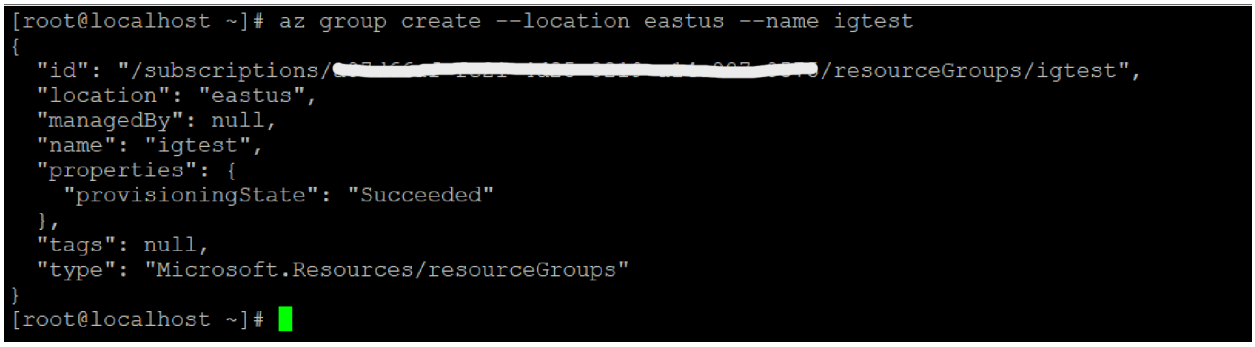

Figure 13 : Creating resource group

For more information regarding the commands and command parameters, please check the Microsoft Azure CLI documentation.

3. Create an Azure Key Vault with premium SKU.

```
›_ Console
```

```
# az keyvault create --location "<location>" --name "<keyvault>"
--resource-group "<resource_group>" --sku premium
```

Figure 14 : Creating Azure Key Vault

5.4.3 Creating Azure Key Vault on Windows

1. Open the command line interpreter and log in with the following command:

```
›_ Console
```

```
> az login
```

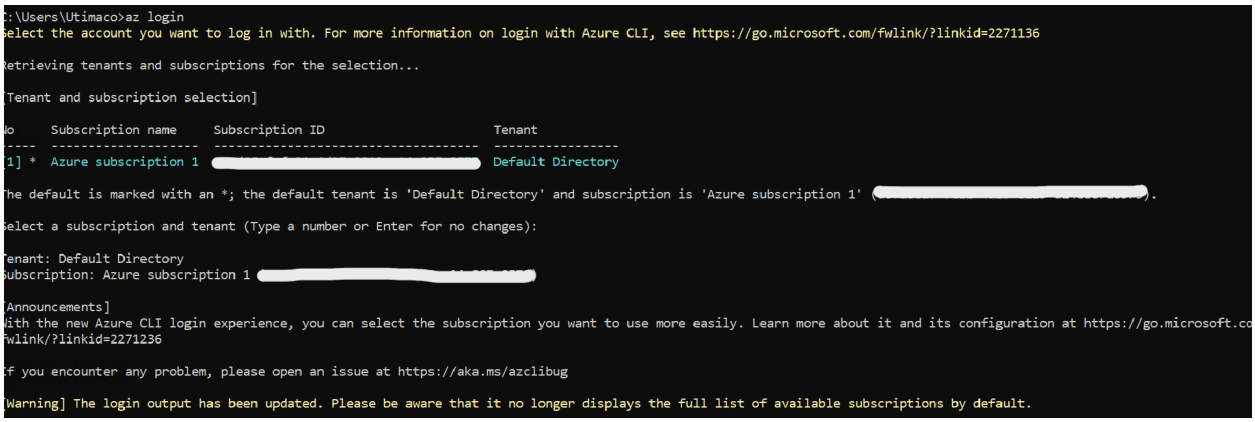

Figure 15 : Azure login

2. To create an Azure Key Vault, you will need to create a resource group. Use the following command to create your own resource group.

```
›_ Console
```

```
> az group create --location "<location>" --name "<resource_group>"
```

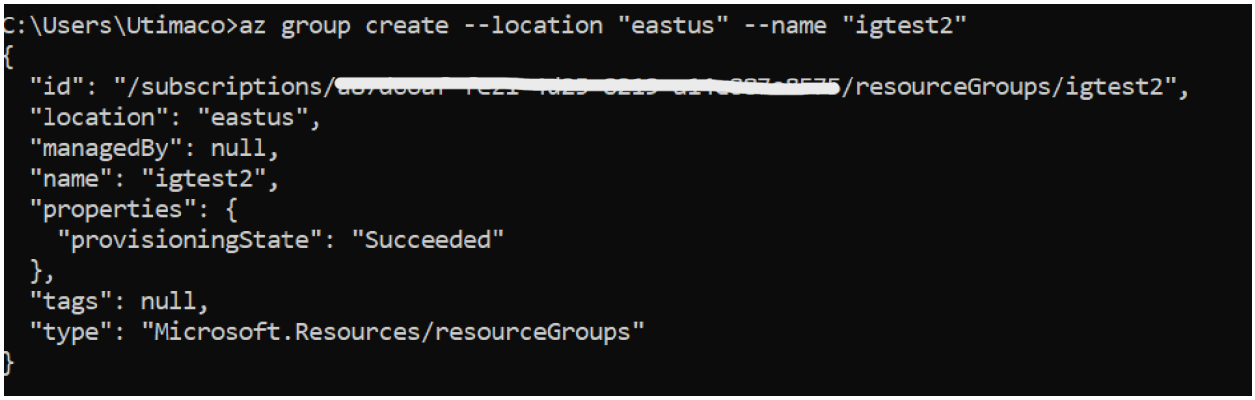

Figure 16 : Creating resource group

For more information regarding the commands and command parameters, please check the Microsoft Azure CLI documentation.

3. Create an Azure Key Vault with premium SKU.

```
›_ Console
```

```
> az keyvault create --location "<location>" --name "<keyvault>"
--resource-group "<resource_group>" --sku premium
```

Figure 17 : Creating Azure Key Vault

5.5 Generating Key Exchange Key (KEK)

The KEK (Key Exchange Key) is an RSA key, generated in the Key Vault. The KEK must be:

1. An RSA-HSM key (2048-bit or 3072-bit or 4096-bit).
2. Generated in the same key vault where you intend to import the tenant key to.
3. Created with the allowed key operations set to import.

5.5.1 Creating a KEK with Azure Portal

To create a key within your recently created key vault, click on the name of your Azure **Key vault** and follow the next steps.

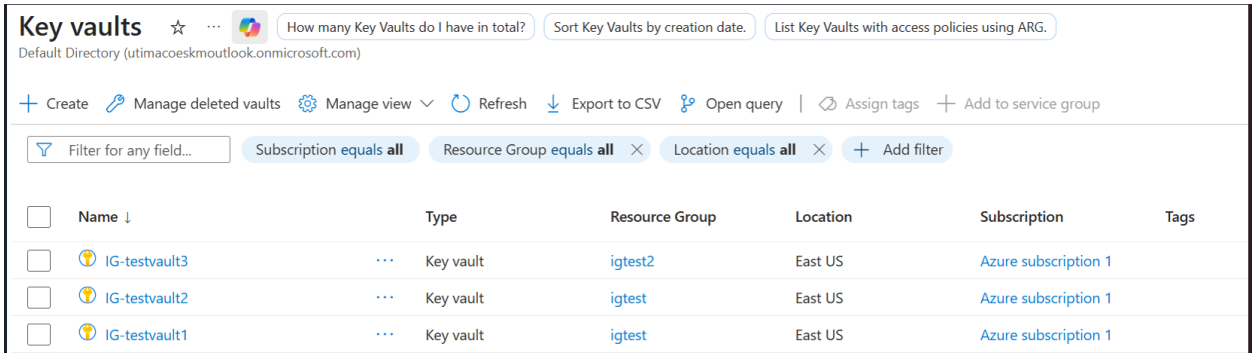

Figure 18 : Key vaults home page

1. Under the **Settings** menu select the **Keys** setting.

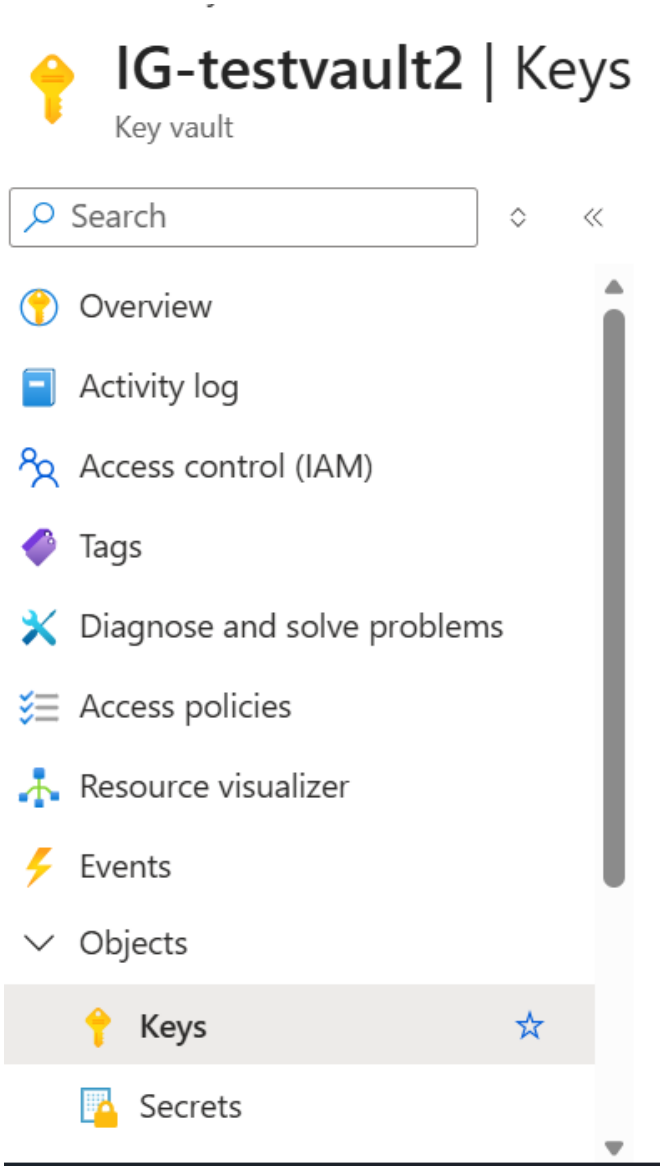

Figure 19 : Key vault menu

2. Click on **Generate/Import**.

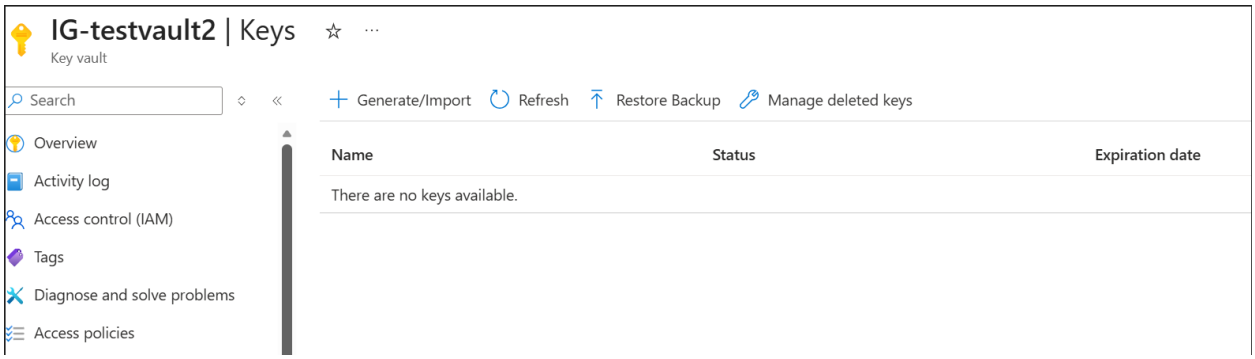

Figure 20 : Default key vault keys setting page

3. In the **Options** drop-down menu select **Generate Key Encryption Key for importing HSM-protected Keys**, add a name to the key and select the RSA key size. If needed, set the activation and expiration date for the key.

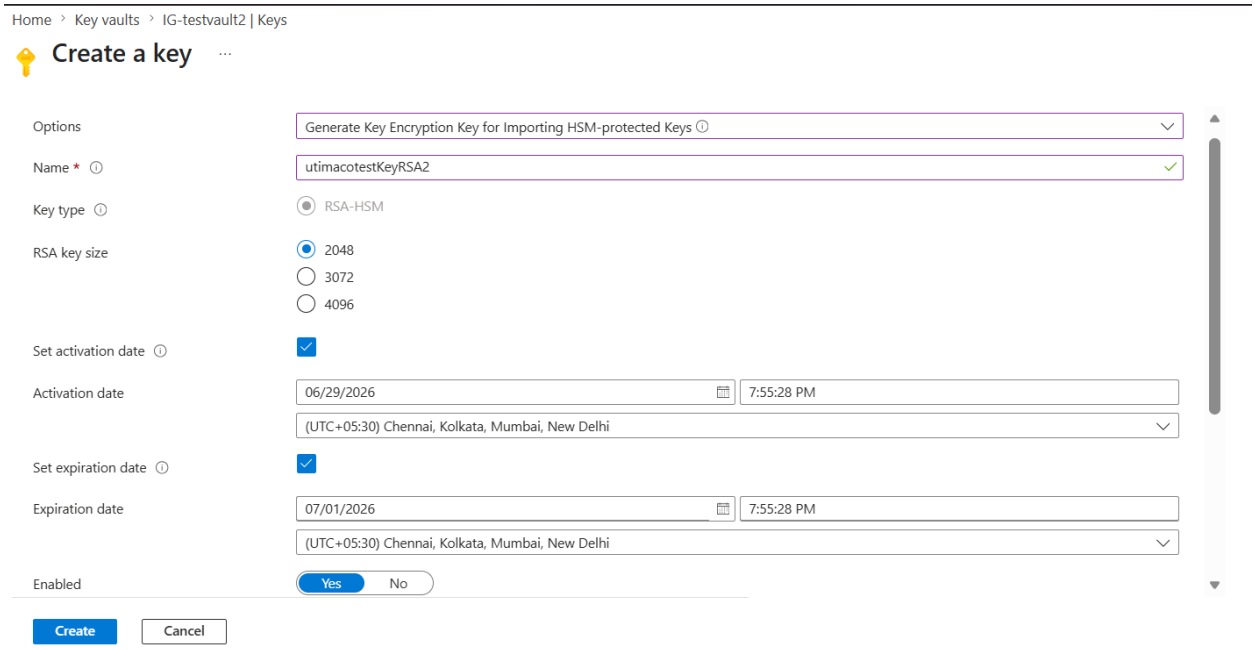

Figure 21 : Example of create a key page

4. After the key is created it will be display under the key vault created and being used. Navigate to the following path to see the newly generated key.

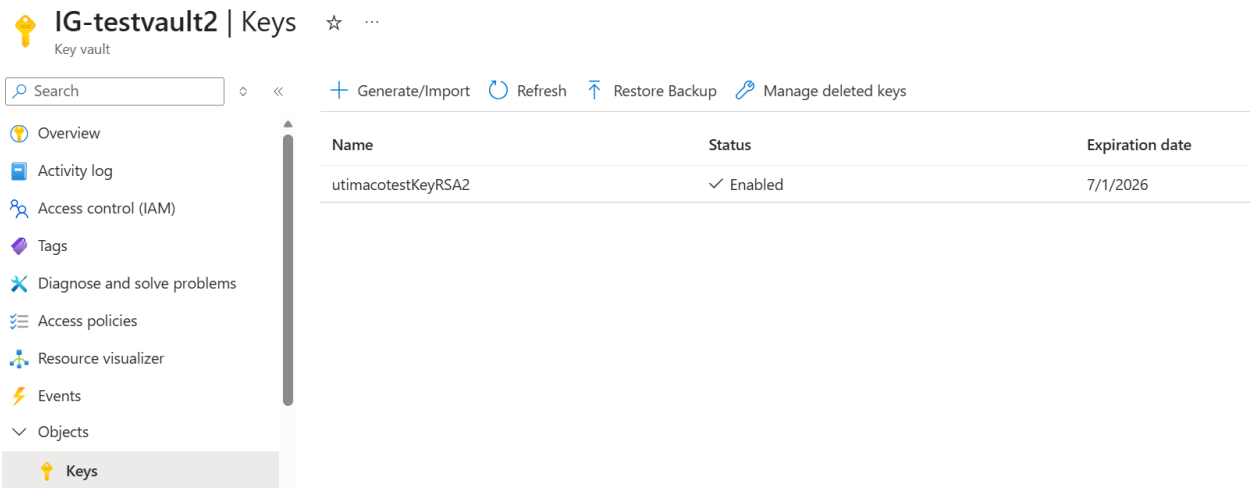

Figure 22 : Generated keys

5. The key we just created has an identifier which will be needed in the next steps. To find this **Key Identifier**, click on the newly created key in your **Key Vault** to display its properties. The **Key Identifier** will be needed in the steps for *Generating and Preparing your Tenant Key*.

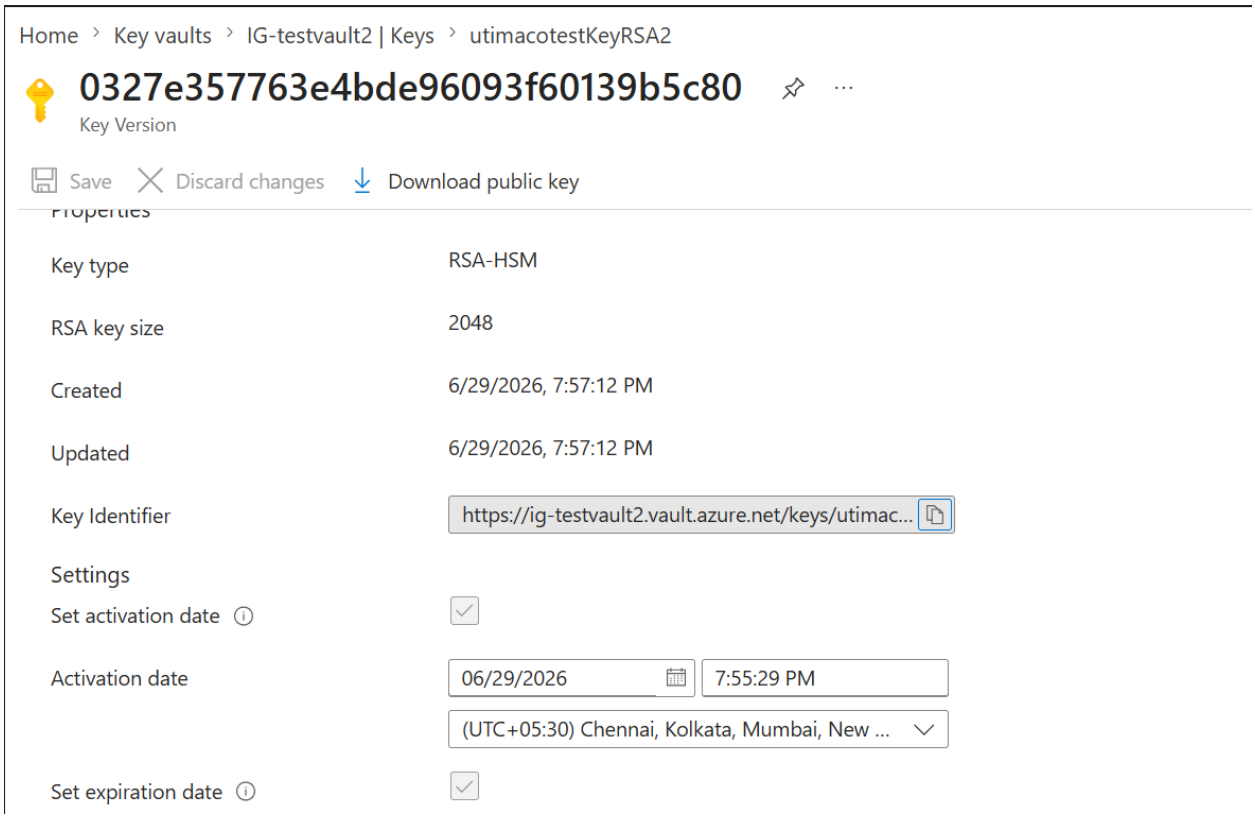

Figure 23 : Key properties

5.5.2 Creating a KEK on Linux

1. Create a KEK with the key operations set to import. The KEK can be an RSA key of different sizes such as: 2048-bit, 3072-bit or 4096-bit. It is advisable to create a key with the length suitable for your use case.

```
›_ Console
```

```
# az keyvault key create --name "<keyvault_key>" --vault-name "<keyvault>"
-- kty RSA-HSM --size 2048 --ops import
```

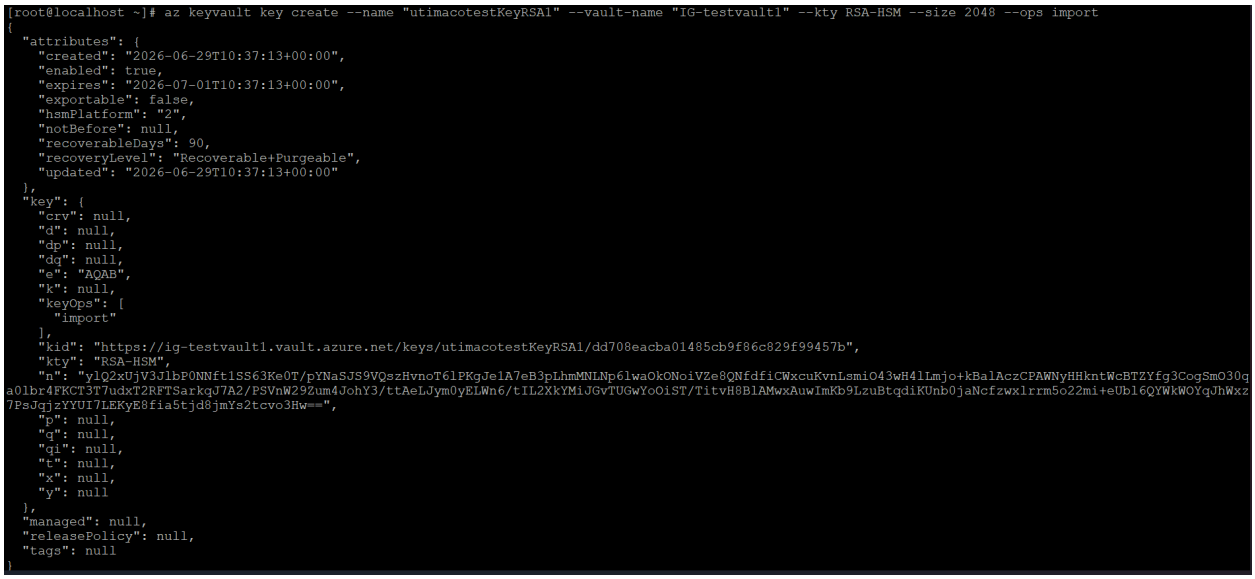

Figure 24 : Creating KEK

After the command has been successfully executed, please make sure to note down the key identifier in the command printout as it will be used when wrapping your tenant key.

5.5.3 Creating a KEK on Windows

1. Create a KEK with the key operations set to import. The KEK can be an RSA key of different sizes such as: 2048-bit, 3072-bit or 4096-bit. It is advisable to create a key with the length suitable for your use case.

`›_ Console`

```
> az keyvault key create --name "<keyvault_key>" --vault-name "<keyvault>"
-- kty RSA-HSM --size 2048 --ops import
```

Figure 25 : Creating KEK on Windows

After the command has been successfully executed, please make sure to note down the key identifier in the command printout as it will be used when wrapping your tenant key.

5.6 Downloading KEK Public Key

You will need to download the Public Key of the Key Exchange Key generated on the Azure Key Vault. The Public Key of the KEK will be used to encrypt your tenant key.

5.6.1 Downloading KEK Public Key with Azure Portal

To download the KEK public key, navigate to your Azure Portal. Go to your **Key vaults** and select the key you would like to download. Click on **Download public key**.

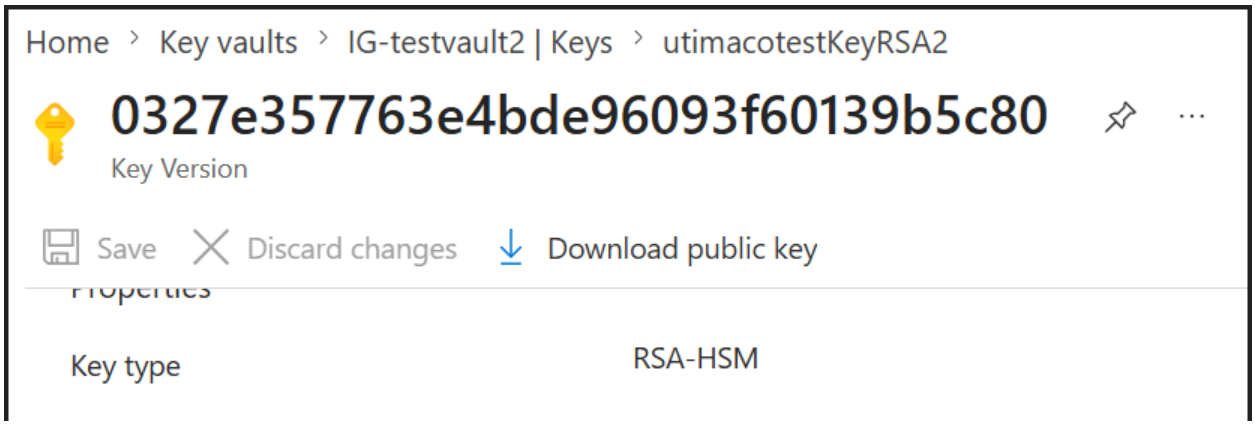

Figure 26 : Key properties

5.6.2 Downloading KEK Public Key on Linux

Execute the following command to download the KEK public key in the `*.pem` format.

`›_ Console`

```
# az keyvault key download --name "<keyvault_key>" --vault-name "<keyvault>" --file <keyvault_key>.publickey.pem
```

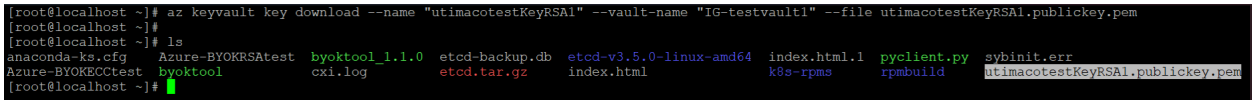

Figure 27 : Download key

5.6.3 Downloading KEK Public Key on Windows

Execute the following command to download the KEK public key in the `*.pem` format.

```
›_ Console
```

```
# az keyvault key download --name "<keyvault_key>" --vault-name
"<keyvault>" --file <keyvault_key>.publickey.pem
```

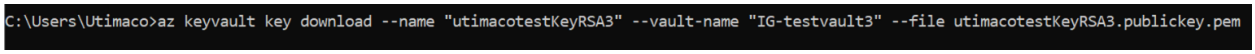

Figure 28 : Download key

5.7 Generating and Preparing your Tenant Key

Ensure that you have created an HSM user that can manage crypto operations (CryptoUser). The byoktool supports all key types (PKCS#11, CNG, JCE, CXI). In this guide we will create a PKCS#11 key, see [CSPKCS11RM] for more information. For other types, refer to the documentation provided to you on the Utimaco Product CD.

5.7.1 Generating Tenant Key on Linux

1. Use the following command to generate an RSA tenant key:

```
›_ Console
```

```
# p11tool2 slot=<slot_no.> loginuser=<user_password>
PubKeyAttr=CKA_LABEL="<tenant_public_key>",CKA_MODULUS_BITS=<keysize>
PrvKeyAttr=CKA_LABEL="<tenant_private_key>",CKA_EXTRACTABLE=CK_TRUE
GenerateKeyPair=RSA
```

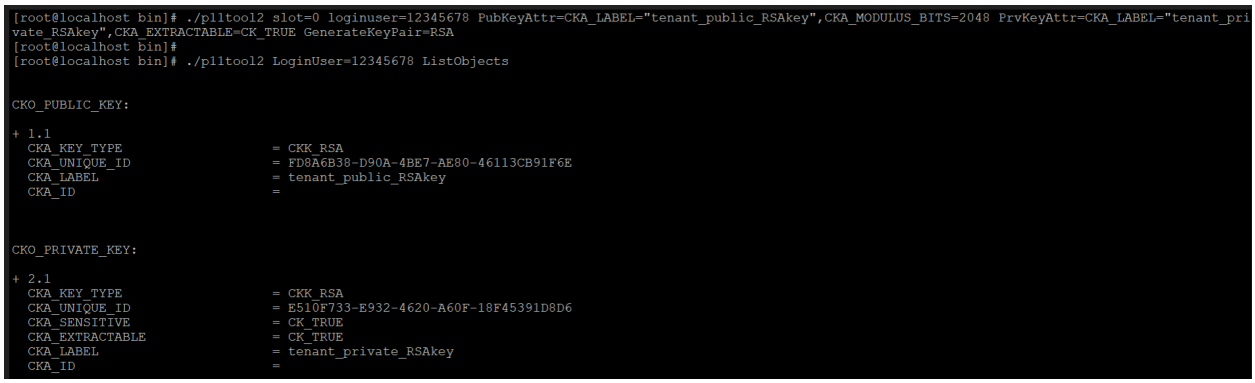

Figure 29 : List newly generated RSA keys

`<keysize>` is the RSA key size in bits. Azure supports 2048, 3072 and 4096. Use the following command to generate an EC tenant key:

`›_ Console`

```
# p11tool2 slot=<slot_no.> loginuser=<user_password>
PubKeyAttr=CKA_LABEL="<tenant_public_key>",CKA_EC_PARAMS=oid:<curvename>
PrvKeyAttr=CKA_LABEL="<tenant_private_key>",CKA_EXTRACTABLE=CK_TRUE
GenerateKeyPair=ECC
```

`<curvename>` is the name of the EC curve. For Azure, this needs to be NIST-P256, NIST- P384 or NIST-P521.

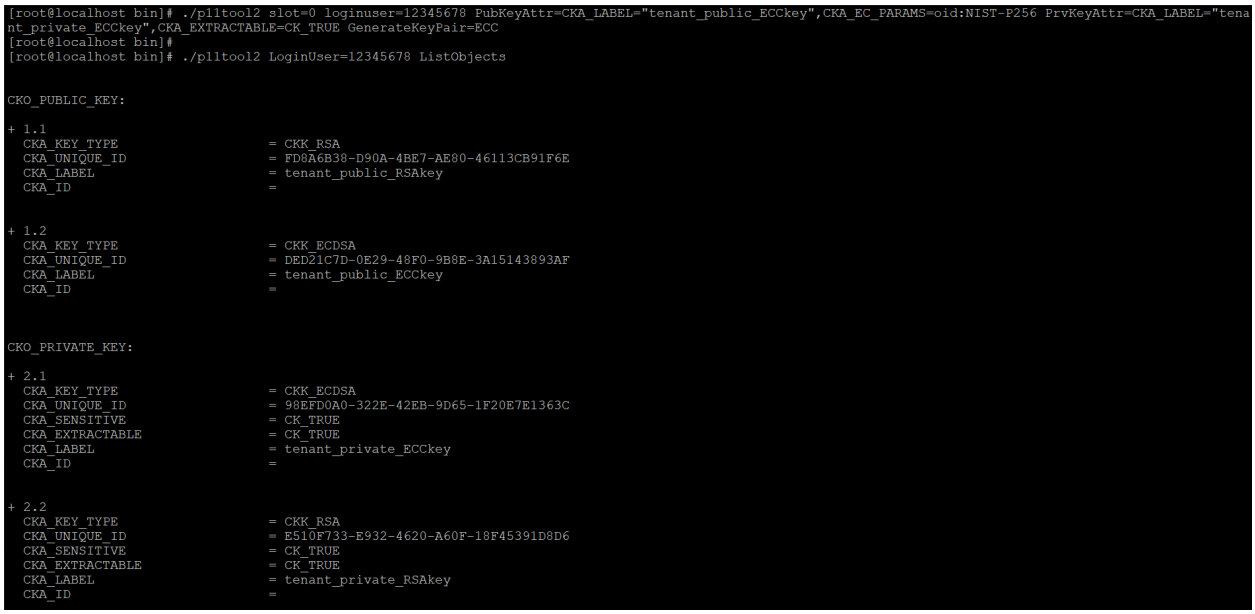

Figure 30 : List newly generated ECC keys

The "oid:<curvename>" syntax is supported starting with SecurityServer 4.40.

The attribute CKA_EXTRACTABLE must be set to CK_TRUE.

2. Navigate to the folder where you have the `byoktool` saved. Execute the following command to wrap the tenant key by using the `KeyVaultKey`, downloaded from the Azure Key Vault.

`›_ Console`

```
# byoktool Dev=<IP_of_UTIMACO_HSM> LogonPass=<user>,<user_password>
Label="<tenant_private_key>" CSP=azure
PublicKey="<keyvaultkey>.publickey.pem" KID="<kid>"

WrappedKey="<wrappedkey>"
```

For the RSA Key

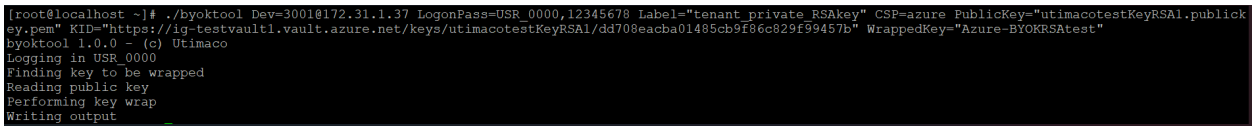

Figure 31 : Wrap the tenant key for RSA

For the ECC Key

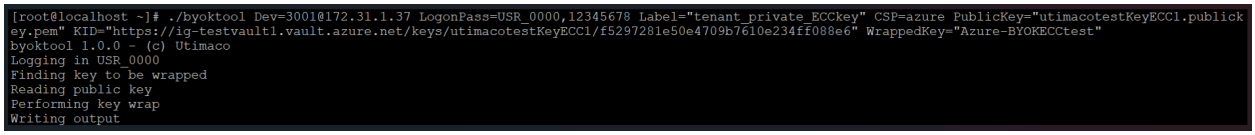

Figure 32 : Wrap the tenant key for ECC

Command Parameters:

- `<user>` is "USR_0013" for PKCS#11 slot 13. Any other user/password combination with access to the tenant_private_key is possible as well.
- `<keyvaultkey>` is the filename of the public key downloaded from Azure key vault.
- `<kid>` is the key identifier of the KEK in Key Vault (e.g., https://KeyVaultUtimacoHSM.vault.azure.net/keys/mykek/eba63d27es214e028839s77fc905621).
- `<wrappedkey>` is the filename of the wrapped tenant key.

5.7.2 Generating Tenant Key on Windows

1. Use the following command to generate an RSA tenant key:

`›_ Console`

```
> p11tool2 slot=<slot_no.> loginuser=<user_password>
PubKeyAttr=CKA_LABEL="<tenant_public_key>",CKA_MODULUS_BITS=<keysize>
PrvKeyAttr=CKA_LABEL="<tenant_private_key>",CKA_EXTRACTABLE=CK_TRUE
GenerateKeyPair=RSA
```

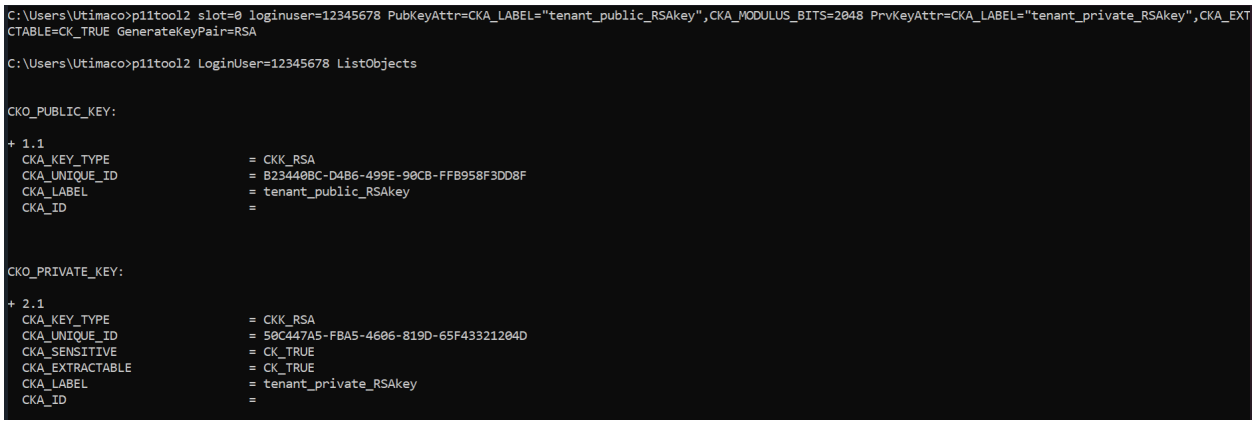

Figure 33 : List newly generated RSA keys

`<keysize>` is the RSA key size in bits. Azure supports 2048, 3072 and 4096. Use the following command to generate an EC tenant key:

`›_ Console`

```
> p11tool2 slot=<slot_no.> loginuser=<user_password>
PubKeyAttr=CKA_LABEL="<tenant_public_key>",CKA_EC_PARAMS=oid:<curvename>
PrvKeyAttr=CKA_LABEL="<tenant_private_key>",CKA_EXTRACTABLE=CK_TRUE
GenerateKeyPair=ECC
```

`<curvename>` is the name of the EC curve. For Azure, this needs to be NIST-P256, NIST- P384 or NIST-P521.

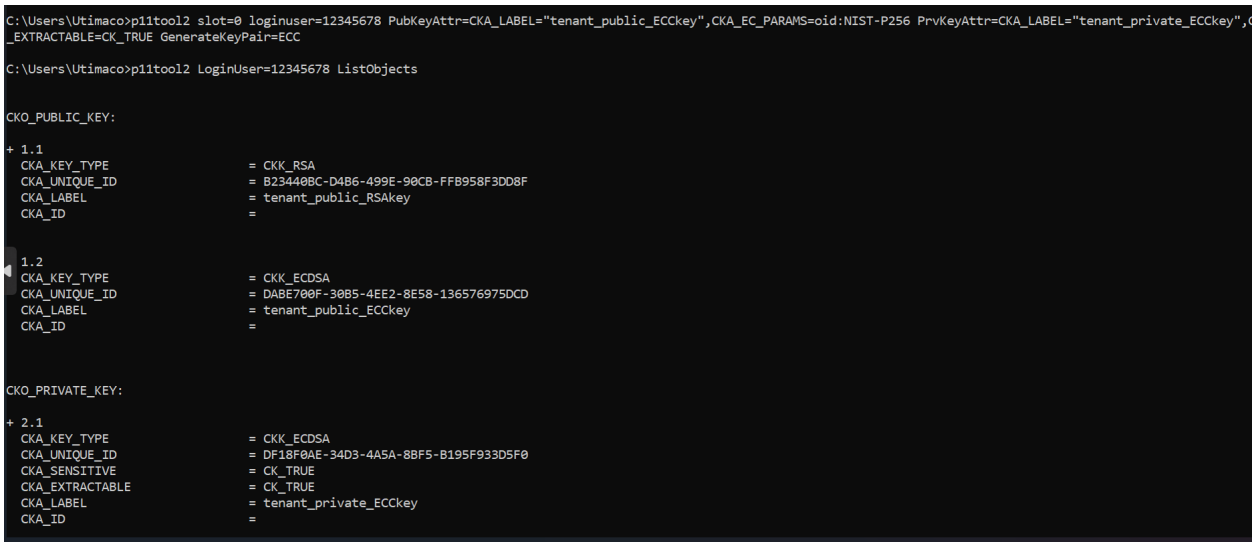

Figure 34 : List newly generated ECC keys

The "oid:<curvename>" syntax is supported starting with SecurityServer 6.5.0.

The attribute CKA_EXTRACTABLE must be set to CK_TRUE

2. Navigate to the folder where you have the `byoktool` saved. Execute the following command to wrap the tenant key by using the `KeyVaultKey`, downloaded from the Azure Key Vault.

`›_ Console`

```
> byoktool Dev=<IP_of_UTIMACO_HSM> LogonPass=<user>,<user_password>
Label="<tenant_private_key>" CSP=azure
PublicKey="<keyvaultkey>.publickey.pem" KID="<kid>"

WrappedKey="<wrappedkey>"
```

For the RSA Key

Figure 35 : Wrap the tenant key for RSA

For the ECC Key

Figure 36 : Wrap the tenant key for ECC

Command Parameters:

- `<user>` is "USR_0020" for PKCS#11 slot 20. Any other user/password combination with access to the tenant_private_key is possible as well.
- `<keyvaultkey>` is the filename of the public key downloaded from Azure key vault.
- `<kid>` is the key identifier of the KEK in Key Vault (e.g., https://KeyVaultUtimacoHSM.vault.azure.net/keys/mykek/eba63d27es214e028839s77fc905621).
- `<wrappedkey>` is the filename of the wrapped tenant key.

5.8 Importing Tenant Key to Azure Key Vault

5.8.1 Importing on Linux

Use the Azure CLI to import the wrapped key to your Azure key vault, generated in the *previous steps*.

1. Execute the following command to import an RSA tenant key:

›_ Console

```
# az keyvault key import --vault-name <keyvault> --name <WrappedKeyName> --byok-file "wrappedkey>"
```

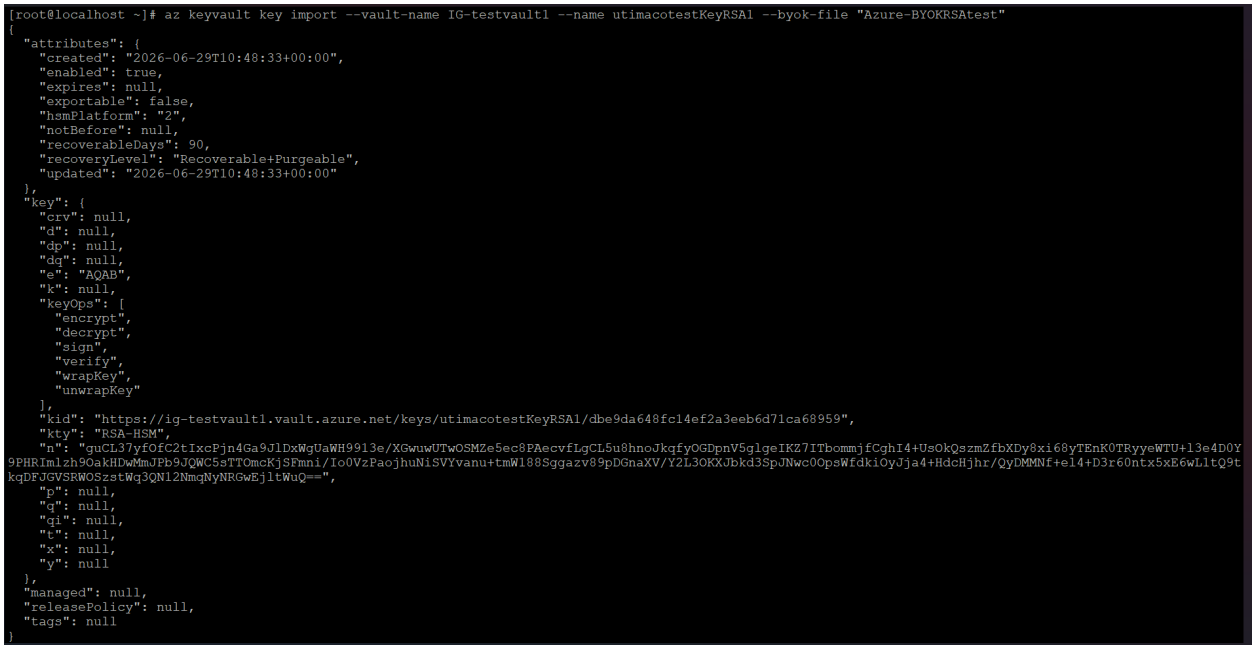

Figure 37 : List importing tenant key

Execute the following command to import an EC tenant key:

```
›_ Console
```

```
# az keyvault key import --vault-name <keyvault> --name <WrappedKeyName> --byok-file "wrappedkey>" --kty EC --curve <curvename>
```

For `<curvename>` Azure supports P-256, P-384 and P-521. The curve name must match the actual key.

Figure 38 : List importing tenant key

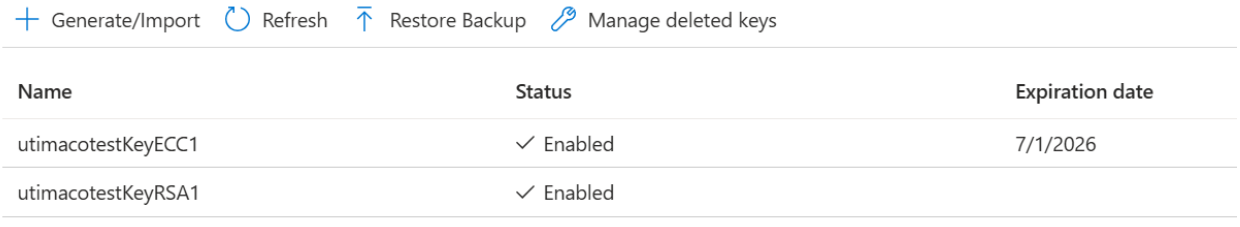

Figure 39 : List importing tenant key

2. Use the following command to check if the key has been successfully imported to the Azure Key vault:

```
›_ Console
```

```
# az keyvault key show --vault-name <keyvault> --name <WrappedKeyName>
```

You can also check if the key is visible on your Azure portal.

For the RSA Key

Figure 40 : List RSA key

For the ECC Key

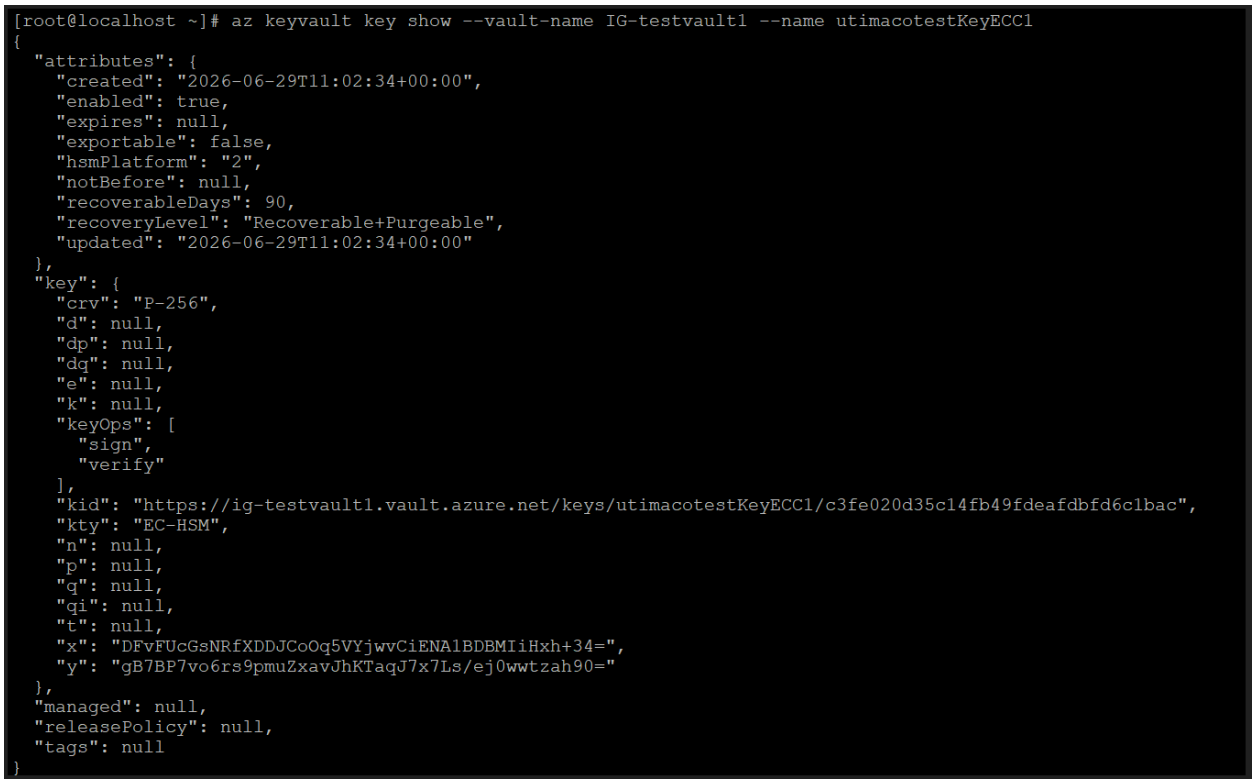

Figure 41 : List ECC key

5.8.2 Importing on Windows

Use the Azure CLI to import the wrapped key to your Azure key vault, generated in the *previous steps*.

1. Execute the following command to import an RSA tenant key:

```
›_ Console
```

```
> az keyvault key import --vault-name <keyvault> --name <WrappedKeyName> --byok-file "<wrappedkey>"
```

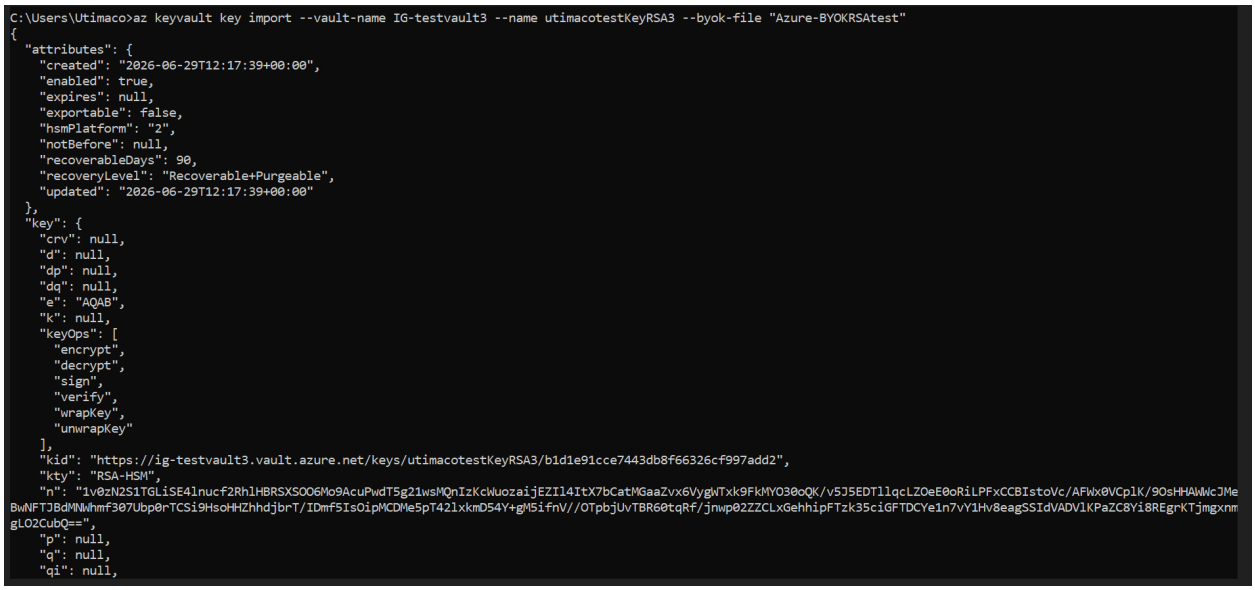

Figure 42 : Importing tenant key

Execute the following command to import an EC tenant key:

```
›_ Console
```

```
> az keyvault key import --vault-name <keyvault> --name <WrappedKeyName> --byok-file "<wrappedkey>" --kty EC --curve <curvename>
```

For `<curvename>` Azure supports P-256, P-384 and P-521. The curve name must match the actual key.

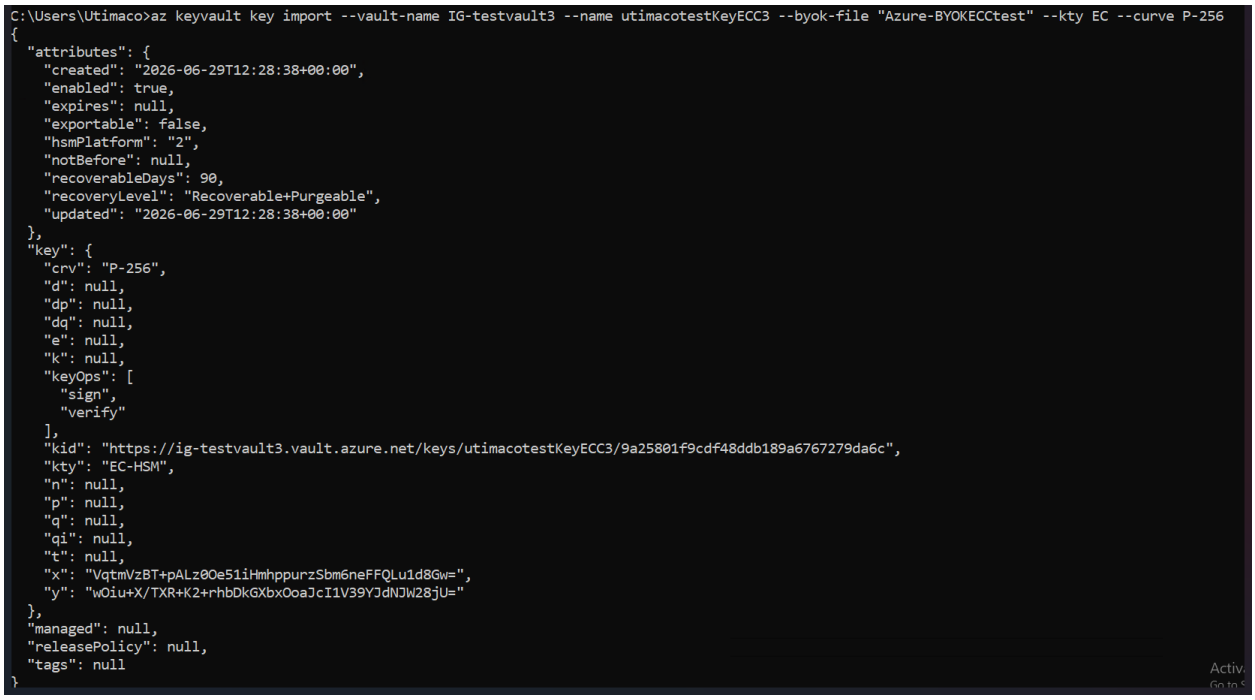

Figure 43 : Importing tenant key

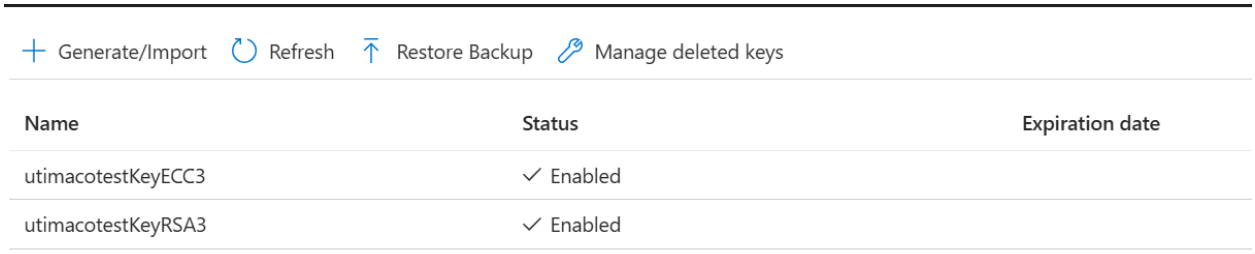

Figure 44 : List importing tenant key

2. Use the following command to check if the key has been successfully imported to the Azure Key vault:

```
›_ Console
```

```
> az keyvault key show --vault-name <keyvault> --name <WrappedKeyName>
```

You can also check if the key is visible on your Azure portal.

For the RSA Key

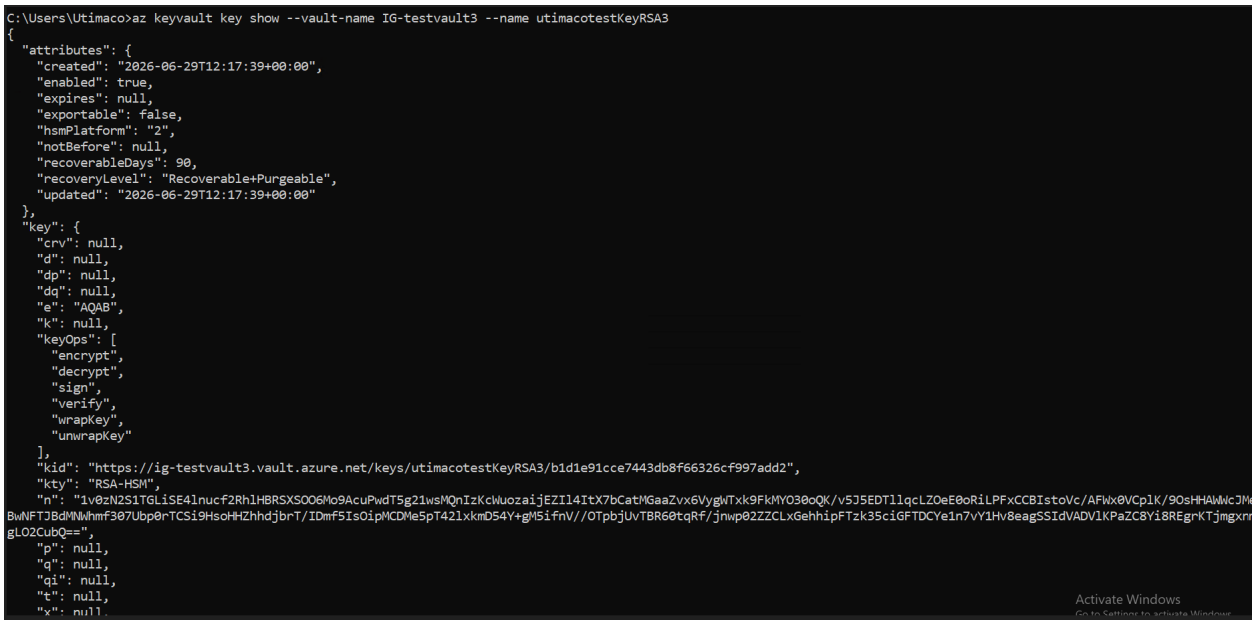

Figure 45 : List RSA key

For the ECC Key

```
C:\Users\Utimaco>az keyvault key show --vault-name IG-testvault3 --name utimacotestKeyECC3
{
  "attributes": {
    "created": "2026-06-29T12:28:38+00:00",
    "enabled": true,
    "expires": null,
    "exportable": false,
    "hsmPlatform": "2",
    "notBefore": null,
    "recoverableDays": 90,
    "recoveryLevel": "Recoverable+Purgeable",
    "updated": "2026-06-29T12:28:38+00:00"
  },
  "key": {
    "crv": "P-256",
    "d": null,
    "dp": null,
    "dq": null,
    "e": null,
    "k": null,
    "keyOps": [
      "sign",
      "verify"
    ],
    "kid": "https://ig-testvault3.vault.azure.net/keys/utimacotestKeyECC3/9a25801f9cdf48ddb189a6767279da6c",
    "kty": "EC-HSM",
    "n": null,
    "p": null,
    "q": null,
    "qi": null,
    "t": null,
    "x": "VqtmVzBT+pALz0Oe51iHmhppurzSbm6neFFQLu1d8Gw=",
    "y": "wOiu+X/TXR+K2+rhbDkGXbxOoaJcI1V39YJdNJW28jU="
  },
  "managed": null,
  "releasePolicy": null,
  "tags": null
}
```

Figure 46 : List ECC key

5.9 FIPS mode

All the steps are identical to the above, when the HSM is in FIPS 140-2 approved mode. The only difference is that the backup of the entire key database is not possible. Please refer to additional documentation on the Utimaco product CD.

6 Troubleshooting

Error	Diagnosis
07.04.2023 08:20:01.771 | [00011358:00011358] C_Login | E: Error CKR_USER_PIN_NOT_INITIALIZED occurred. 07.04.2023 08:20:01.771 | [00011358:00011358] C_Logout | T: enter C_Logout(hSession: 0x00000001) 07.04.2023 08:20:01.771 | [00011358:00011358] C_Logout | T: leave C_Logout() 07.04.2023 08:20:01.771 | [00011358:00011358] C_CloseSession | T: enter C_CloseSession(hSession: 0x00000001) 07.04.2023 08:20:01.771 | [00011358:00011358] C_CloseSession | T: leave C_CloseSession() 07.04.2023 08:20:01.771 | [00011358:00011358] C_Finalize | T: enter C_Finalize(pReserved: 0) 07.04.2023 08:20:01.771 | [00011358:00011358] C_Finalize | T: leave C_Finalize()	PKCS#11 Slot is not initialized.
The CryptoServer PKCS#11 Library R3 is not initialized. Error CKR_CRYPTOKI_NOT_INITIALIZED occurred	PKCS#11 Slot is not initialized.

Table 6: List of errors and their diagnoses

7 Further Information

This document forms a part of the information and support which is provided by the Utimaco IS GmbH. Additional documentation can be found on the product CD in the Documentation directory.

All u.trust GP HSM product documentation is also available at the Utimaco IS GmbH website: *https://utimaco.com/*.

8 Contact and Support Information

You can reach us from Monday to Friday, 09.00 a.m. to 05.00 p.m., Central European Time (CET).

Utimaco IS GmbH
Krefelder Str. 220
52070 Aachen
Germany

RMA Query

If you need to send the device back to Utimaco IS GmbH, please open a new RMA case (Return Merchandise Authorization). We request that you use the following web address. RMA cases cannot be opened by email or phone.

https://support.hsm.utimaco.com/support/rma/new

Other Support Queries

- Mail (preferred contact method)
  support@utimaco.com
  Attach the diagnostic information to your email.
- Web portal
  https://support.hsm.utimaco.com/support/cases/new/
  The diagnostic information will be requested in our response if necessary.
- By phone
  AMERICAS +1-844-UTIMACO (+1 844-884-6226)
  EMEA +49 800-627-3081
  APAC +81 800-919-1301
  The diagnostic information will be requested in our response if necessary.

9 Appendices

9.1 References

Reference	Title/Company	Document No.
[CSPKCS11RM]	CryptoServer PKCS#11 p11tool2 Reference Manual	2012-0004

Table 7: References

9.2 Command Summary

Command	Purpose
`mkdir -p /opt/utimaco/bin`	Create directory for Utimaco binaries
`mkdir -p /opt/utimaco/lib`	Create directory for Utimaco libraries
`cp ~/path_to_application_folder/lib/libcs_pkcs11_R3.so /opt/utimaco/lib`	Copy PKCS#11 library to target directory
`cp csadm p11tool2 /opt/utimaco/bin`	Copy administration tools to binary folder
`chmod +x /opt/utimaco/bin/csadm /opt/utimaco/bin/p11tool2`	Make tools executable
`mkdir /etc/utimaco`	Create configuration directory
`cp cs_pkcs11_R3.cfg /etc/utimaco`	Copy PKCS#11 configuration file

Command	Purpose
`./p11tool2 slot=<slot_no> Label=<token_label> Login=ADMIN,ADMIN.key InitToken=<SO_PIN>`	Initialize token and create Security Officer
`./p11tool2 slot=<slot_no> LoginSO=<SO_PIN> InitPin=<CryptoUser_PIN>`	Initialize user PIN for slot
`rpm --import https://packages.microsoft.com/keys/microsoft.asc`	Import Microsoft repository key
`dnf install -y https://packages.microsoft.com/config/rhel/9.0/packages-microsoft-prod.rpm`	Add Microsoft package repository
`dnf install azure-cli`	Install Azure CLI
`az login`	Authenticate to Azure account
`az group create --location "<location>" --name "<resource_group>"`	Create Azure resource group
`az keyvault create --location "<location>" --name "<keyvault>" --resource-group "<resource_group>" --sku premium`	Create Azure Key Vault (Premium)

Command	Purpose
`az keyvault key create --name "<keyvault_key>" --vault-name "<keyvault>" --kty RSA-HSM --size 2048 --ops import`	Generate Key Exchange Key (KEK)
`az keyvault key download --name "<keyvault_key>" --vault-name "<keyvault>" --file <keyvault_key>.publickey.pem`	Download KEK public key
`p11tool2 slot=<slot_no.> loginuser=<user_password> PubKeyAttr=CKA_LABEL="<tenant_public_key>",CKA_MODULUS_BITS=<keysize> PrvKeyAttr=CKA_LABEL="<tenant_private_key>",CKA_EXTRACTABLE=CK_TRUE GenerateKeyPair=RSA`	Generate RSA tenant key in HSM
`p11tool2 slot=<slot_no.> loginuser=<user_password> PubKeyAttr=CKA_LABEL="<tenant_public_key>",CKA_EC_PARAMS=oid:<curvename> PrvKeyAttr=CKA_LABEL="<tenant_private_key>",CKA_EXTRACTABLE=CK_TRUE GenerateKeyPair=ECC`	Generate ECC tenant key in HSM

Command	Purpose
`byoktool Dev=<HSM_IP> LogonPass=<user,password> Label="<tenant_private_key>" CSP=azure PublicKey="<keyvaultkey>.pem" KID="<kid>" WrappedKey="<wrappedkey>"`	Wrap tenant key for Azure import
`az keyvault key import --vault-name <keyvault> --name <WrappedKeyName> --byok-file "wrappedkey"`	Import wrapped RSA key into Azure Key Vault
`az keyvault key import --vault-name <keyvault> --name <WrappedKeyName> --byok-file "wrappedkey" --kty EC --curve <curvename>`	Import wrapped EC key into Azure Key Vault
`az keyvault key show --vault-name <keyvault> --name <WrappedKeyName>`	Verify imported key in Azure

Table 8: Commands