Microsoft
Azure Key Vault

Integration Guide

CryptoServer HSM

SecurityServer V4.50.0.2

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	2026-03-05
Status	PUBLISHED
Document No.	IG-2025-0036

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 describes how to bring your own key from Utimaco Hardware Security Module (HSM) into the Azure Key Vault.

1.1.1 Target Audience for This Guide

This guide is intended for Azure administrators and HSM administrators.

1.1.2 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	Press the **OK** button.
`Monospaced`	File names, folder and directory names, commands, file outputs, programming code samples	You will find the file `example.conf` in the `/exmp/demo/ directory`.
Italic	References and important terms	See Chapter 3, "Sample Chapter", in the *u.trust Anchor - csadm Manual* or [CSADM].

Table 1: 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.

1.1.3 Abbreviations

The following abbreviations are used in this guide:

Abbreviation	Meaning
API	Application Programming Interface
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

Abbreviation	Meaning
SKU	Stock-Keeping-Unit

Table 2: List of abbreviations

2 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. 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 CryptoServer HSM

CryptoServer is a hardware security module developed by Utimaco IS GmbH. CryptoServer 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 Utimaco u.trust Anchor

u.trust Anchor is the next generation hardware security module platform developed by Utimaco IS GmbH. u.trust Anchor is a physically protected specialized computer unit designed for true multi-tenancy and 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.

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 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 Server 2019 Redhat8 / Centos8	BYOK tool 1.0.0	SecurityServer V4.50.0.2	CryptoServer 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 2.15 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	CryptoServer CSe-Series/Se-Series LAN with firmware SecurityServer 4.50.0.2 or higher u.trust Anchor CSAR Series LAN with firmware 4.30.0 or higher
Utimaco PCI-e HSM	CryptoServer CSe-Series/Se-Series LAN with firmware SecurityServer 4.50.0.2 or higher u.trust Anchor CSAR Series LAN with firmware 4.30.0 or higher

Table 5: Hardware requirements

3.4 Prerequisites

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

- Set up and configure CryptoServer. Refer to the CryptoServer documentation to setup the HSM.
- Create and store the MBK onto each HSM. Refer to the CryptoServer documentation to set up the MBK.
- Replace the CryptoServer Default Admin with a new admin user.

- Install and set up the operating system listed in Tested Versions.
- Install and set up SecurityServer 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 the utimaco folder under /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 the 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 CryptoServer 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 /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 p11tool2 command initialize the Slot that you want to use, and 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>
```

```
[root@Azurevm ~]# p11tool2 slot=22 Label=Azure-BYOK Login=ADMIN,/opt/utimaco/bin/ADMIN.key Inittoken=ask
Enter SO PIN:
Repeat SO PIN:
```

Figure 1 : Slot initialization output

```
[root@Azurevm ~]# p11tool2 slot=22 loginSO=ask initpin=ask
Enter SO PIN:
Enter normal user PIN:
Repeat normal user PIN:
```

4.2 On Windows

On Windows, as part of CryptoServer software installation, `cs_pkcs11_R3.cfg` will get automatically created and will be available under “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 2.46. 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 8 or CentOS Stream 8, add packages-microsoft-com-prod repository.

```
›_ Console
```

```
> dnf install -y https://packages.microsoft.com/config/rhel/8/packages-microsoft-prod.rpm
```

```
[root@Azurevm yum.repos.d]# dnf install -y https://packages.microsoft.com/config/rhel/8/packages-microsoft-prod.rpm
Last metadata expiration check: 0:00:12 ago on Fri 31 Mar 2023 08:26:18 AM UTC.
packages-microsoft-prod.rpm                                                    32 kB/s | 6.8 kB     00:00
Dependencies resolved.
================================================================================================================================
 Package                              Architecture           Version                Repository                      Size
================================================================================================================================
Installing:
 packages-microsoft-prod              noarch                 1.0-1                  @commandline                   6.8 k

Transaction Summary
================================================================================================================================
Install  1 Package

Total size: 6.8 k
Installed size: 204
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                              1/1
  Installing       : packages-microsoft-prod-1.0-1.noarch                                                         1/1
  Verifying        : packages-microsoft-prod-1.0-1.noarch                                                         1/1

Installed:
  packages-microsoft-prod-1.0-1.noarch

Complete!
```

Figure 2 : Microsoft packages

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

```
›_ Console
```

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

```
[root@Azurevm yum.repos.d]# dnf install azure-cli
packages-microsoft-com-prod                                                                 5.8 MB/s | 6.0 MB     00:01
Last metadata expiration check: 0:00:02 ago on Fri 31 Mar 2023 08:26:47 AM UTC.
Dependencies resolved.
================================================================================================================================
 Package                         Arch        Version                                  Repository                          Size
================================================================================================================================
Installing:
 azure-cli                       x86_64      2.46.0-1.el8                             packages-microsoft-com-prod         58 M
Installing dependencies:
 python39                        x86_64      3.9.6-2.module_el8.5.0+897+68c4c210      appstream                           33 k
 python39-libs                   x86_64      3.9.6-2.module_el8.5.0+897+68c4c210      appstream                          8.2 M
 python39-pip-wheel              noarch      20.2.4-6.module_el8.5.0+897+68c4c210     appstream                          1.3 M
 python39-setuptools-wheel       noarch      50.3.2-4.module_el8.5.0+897+68c4c210     appstream                          497 k
Installing weak dependencies:
 python39-pip                    noarch      20.2.4-6.module_el8.5.0+897+68c4c210     appstream                          2.0 M
 python39-setuptools             noarch      50.3.2-4.module_el8.5.0+897+68c4c210     appstream                          871 k
Enabling module streams:
 python39                                    3.9

Transaction Summary
================================================================================================================================
Install  7 Packages

Total download size: 71 M
Installed size: 946 M
Is this ok [y/N]: y
Downloading Packages:
(1/7): python39-3.9.6-2.module_el8.5.0+897+68c4c210.x86_64.rpm                  27 kB/s |  33 kB     00:01
(2/7): python39-pip-20.2.4-6.module_el8.5.0+897+68c4c210.noarch.rpm            863 kB/s | 2.0 MB     00:02
(3/7): python39-libs-3.9.6-2.module_el8.5.0+897+68c4c210.x86_64.rpm            2.9 MB/s | 8.2 MB     00:02
(4/7): python39-setuptools-wheel-50.3.2-4.module_el8.5.0+897+68c4c210.noarch.rpm 1.6 MB/s | 497 kB     00:00
(5/7): python39-pip-wheel-20.2.4-6.module_el8.5.0+897+68c4c210.noarch.rpm       644 kB/s | 1.3 MB     00:02
(6/7): python39-setuptools-50.3.2-4.module_el8.5.0+897+68c4c210.noarch.rpm      613 kB/s | 871 kB     00:01
(7/7): azure-cli-2.46.0-1.el8.x86_64.rpm                                        18 MB/s |  58 MB     00:03
--------------------------------------------------------------------------------------------------------------------------------
Total                                                                           11 MB/s |  71 MB     00:06
Running transaction check
Transaction check succeeded.
Running transaction test
```

Figure 3 : Installing Azure CLI packages

5.1.2 Azure CLI Installation on Windows

1. Download the Azure CLI latest version from the above link and install.

Figure 4 : Installing Azure CLI

2. Click on finish button to complete the setup.

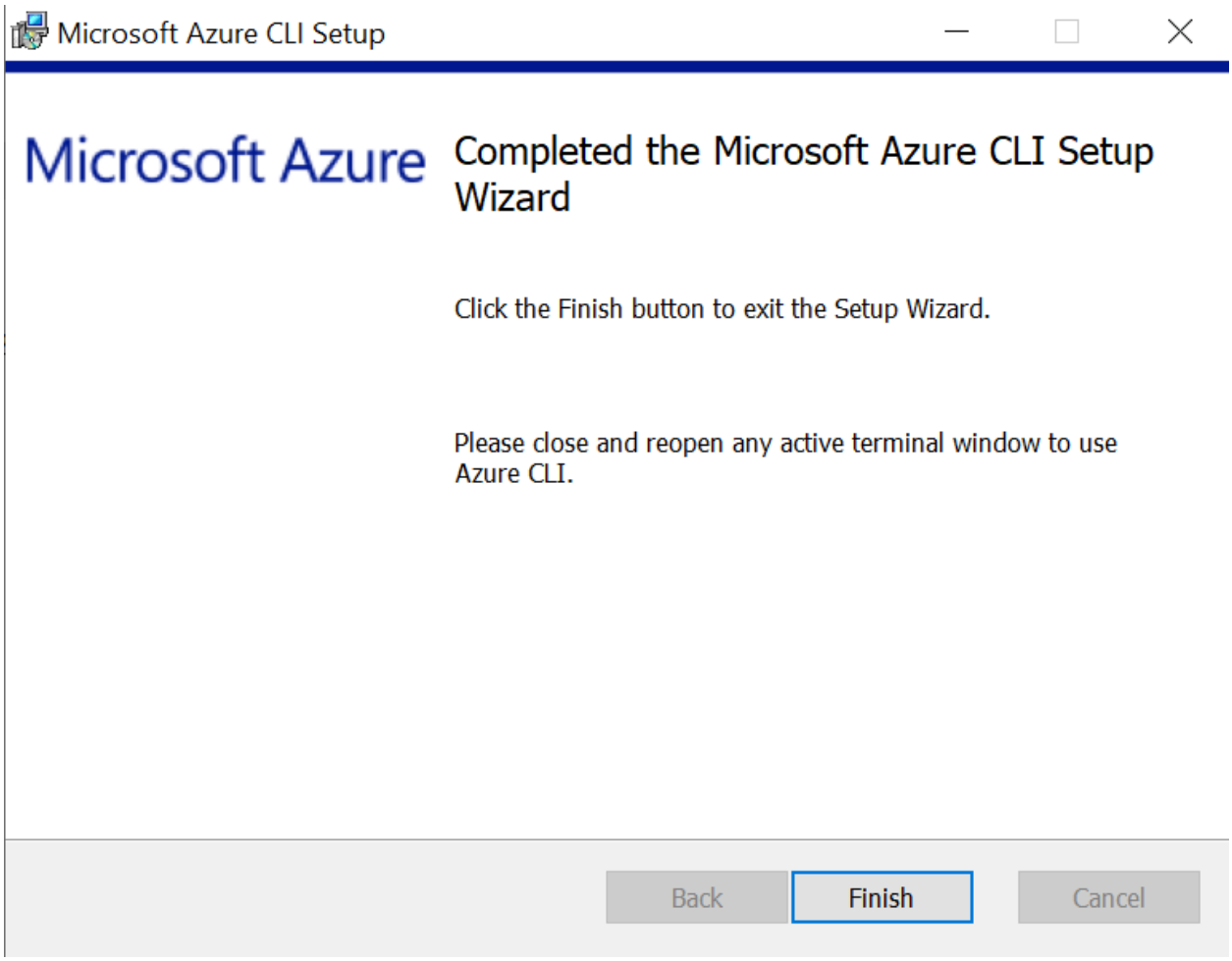


Figure 5 : Azure CLI setup

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 tool. Please reach out to Utimaco so this tool can be provided to you. The byoktool 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 Azure Key Vault with Azure Portal

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

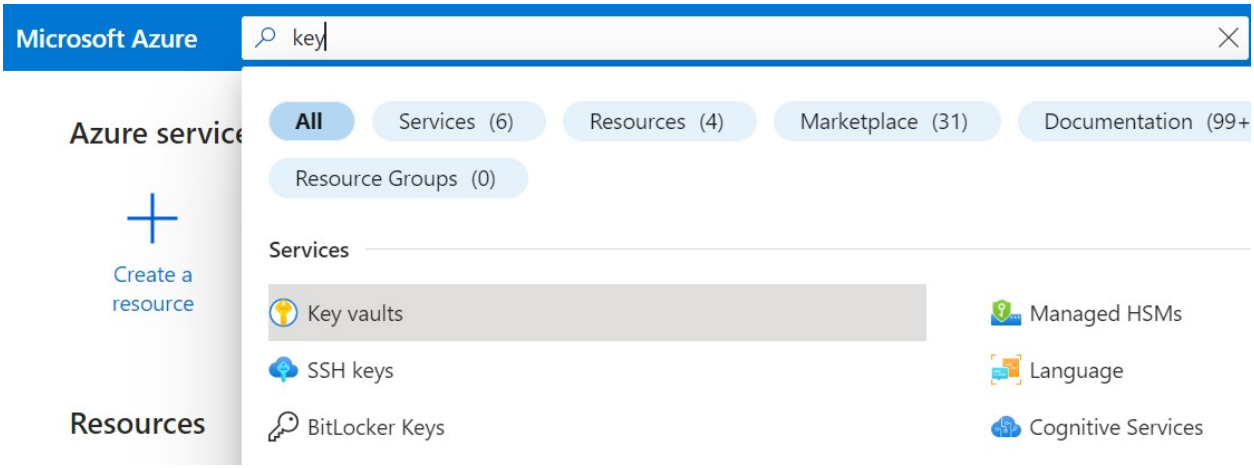


Figure 6 : Azure services

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

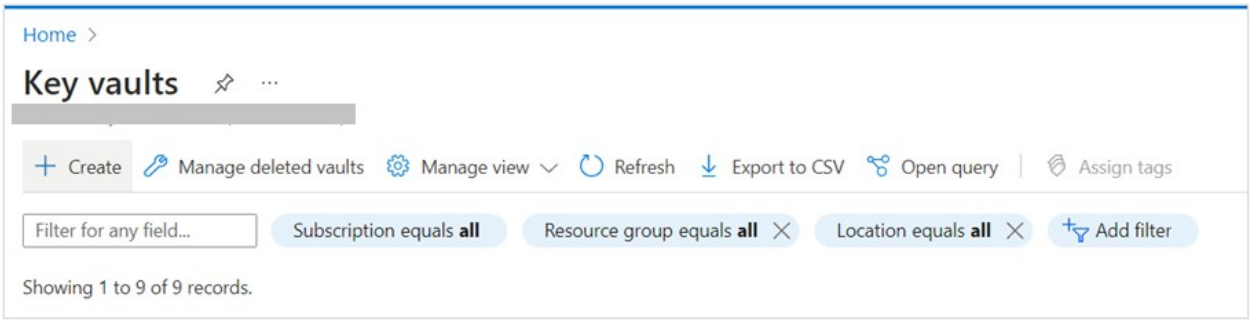


Figure 7 : Home page of Azure Key Vaults

4. Select your subscription and the resource group. Enter the name of the Key vault you will be using, as well as the Region. Make sure to select the Premium pricing tier to include support for the HSM backed keys. Set the Recovery Options according to your company policies.

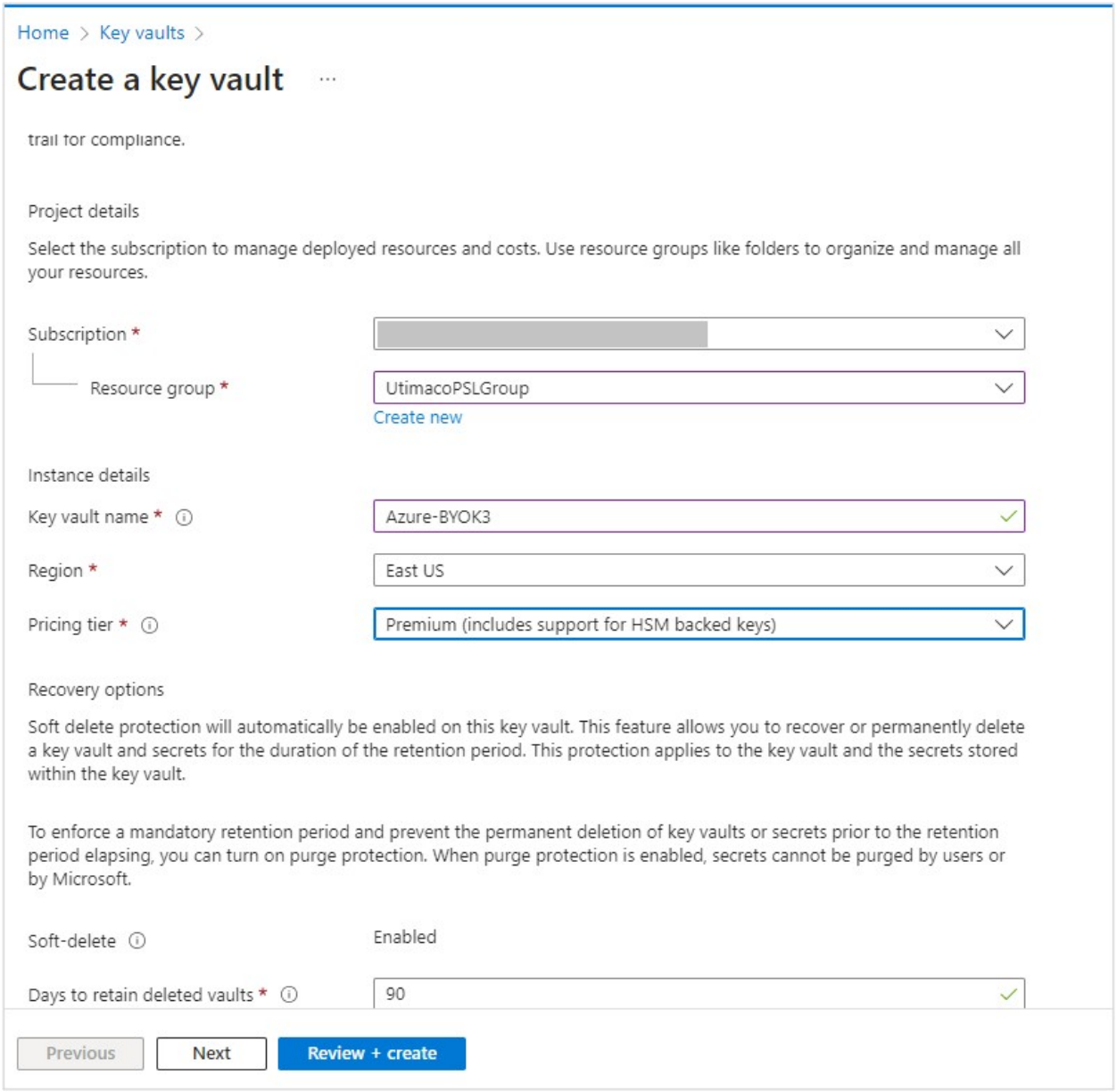


Figure 8 : Create key vault page

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

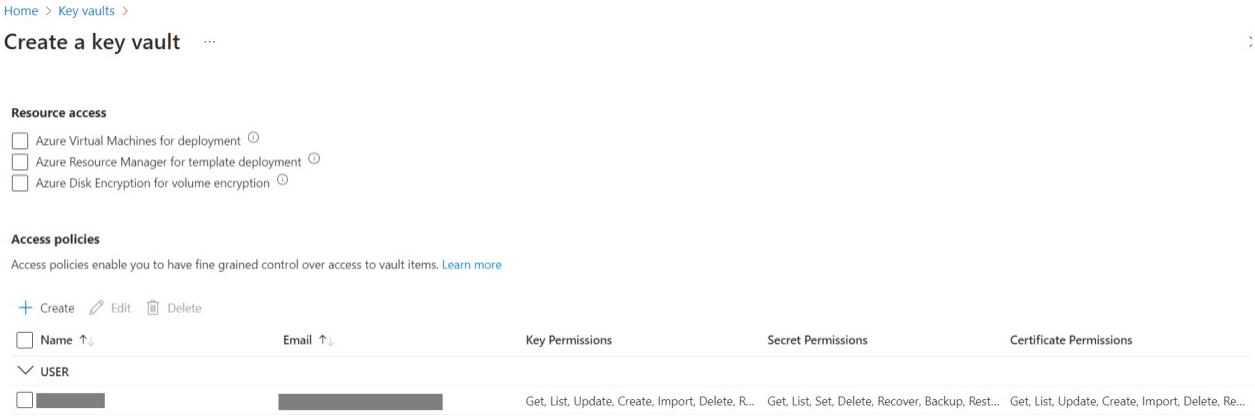


Figure 9 : Create key vault access policy page

6. Select your connectivity method.

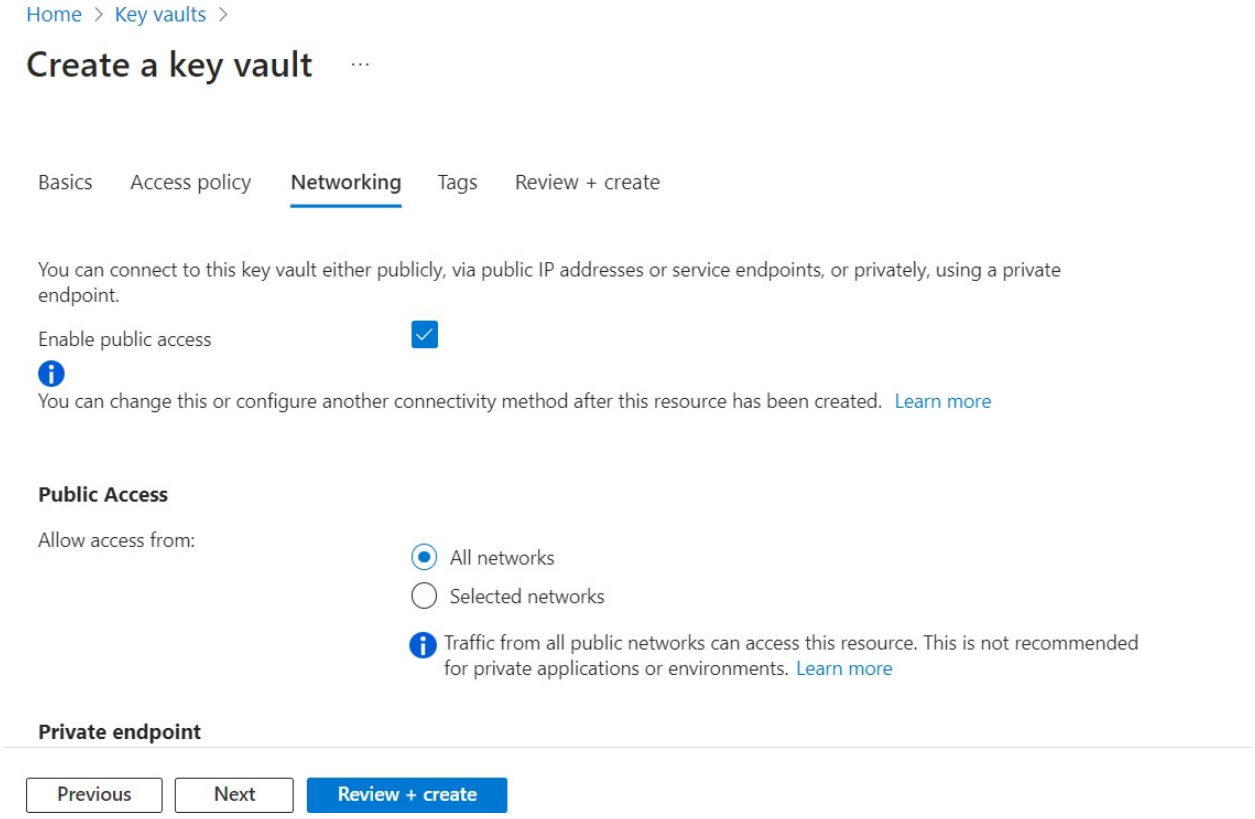


Figure 10 : Create key vault networking page

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

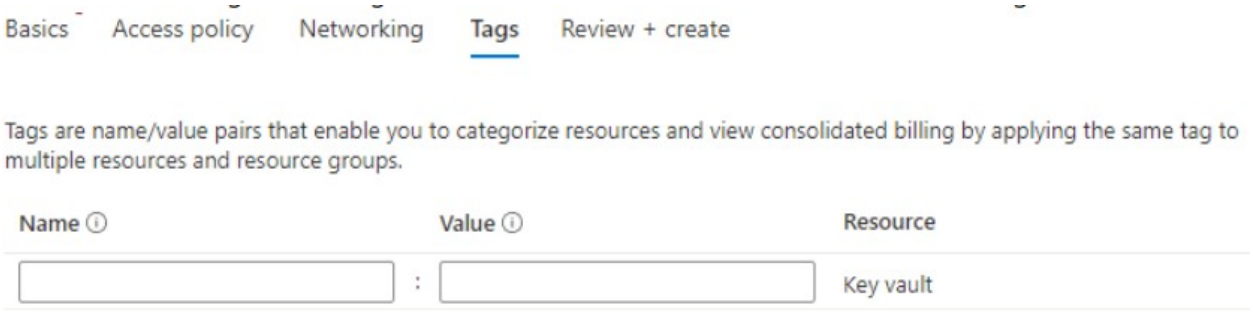


Figure 11 : Create key vault tags page

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

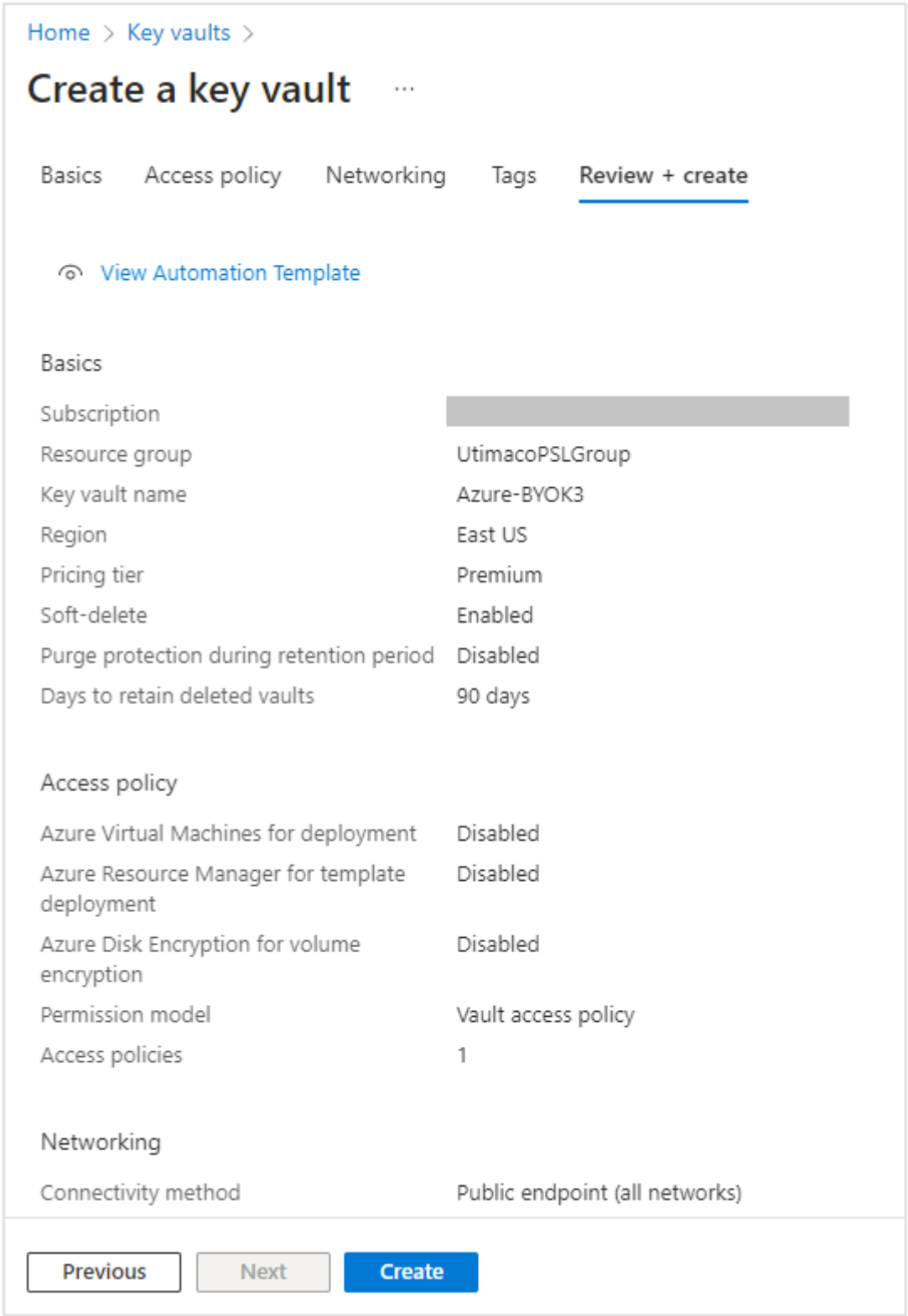


Figure 12 : Create key vault review and create page

5.4.2 Creating Azure Key Vault on Linux

1. Open the command line interpreter and login to azure with the following command:

```
›_ Console

# az login
```

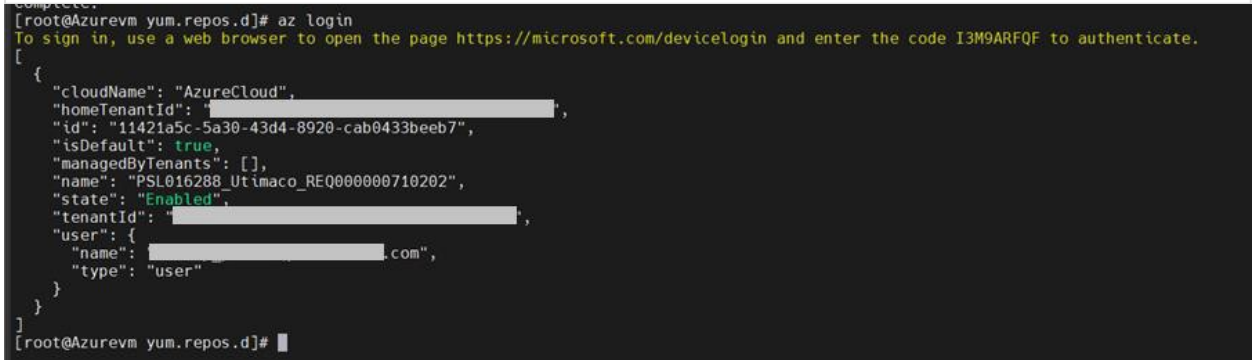


Figure 13 : Azure login

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 14 : 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
```

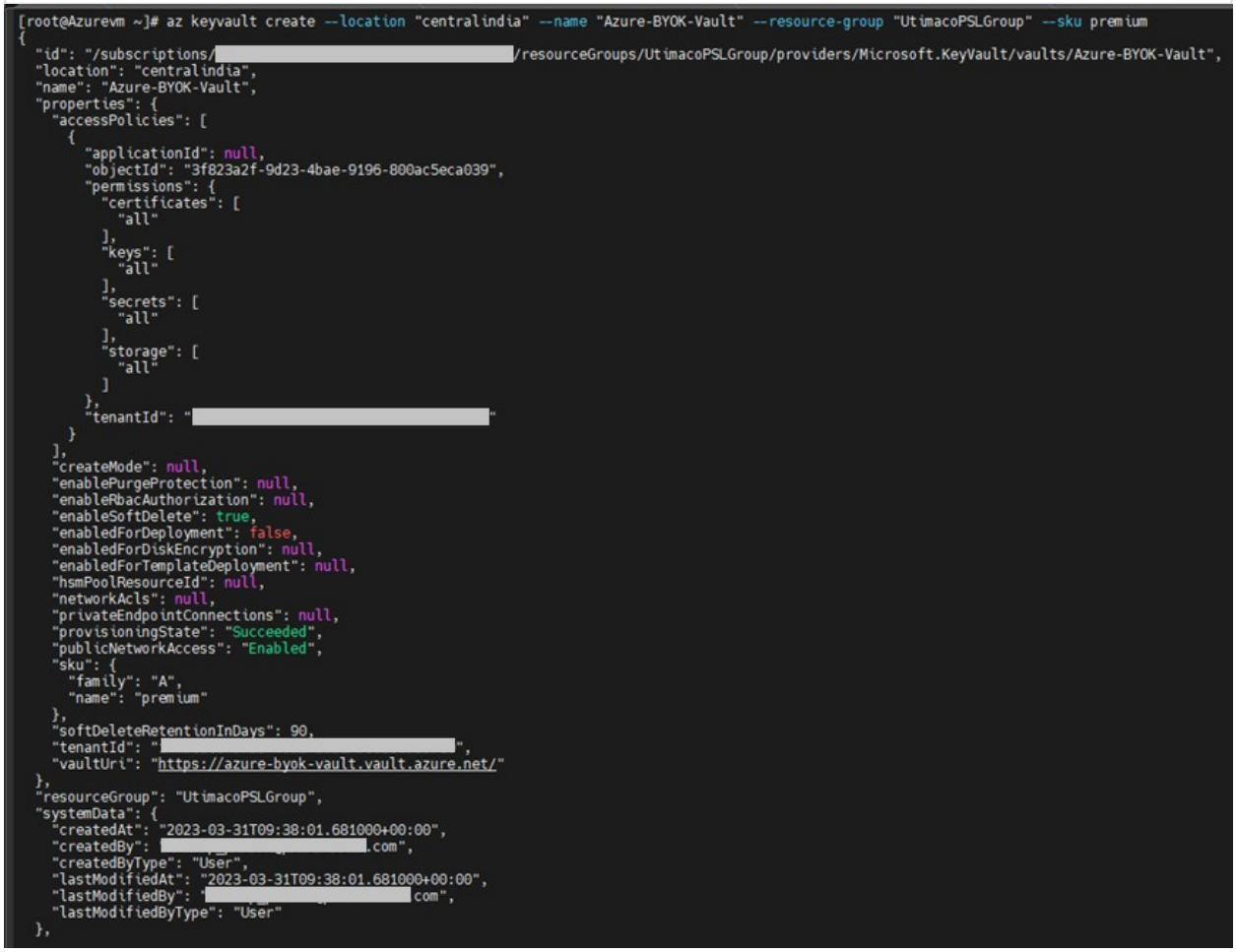


Figure 15 : Creating Azure Key Vault

5.4.3 Creating Azure Key Vault on Windows

1. Open the command line interpreter and login with the following command:

```
›_ Console
```

```
> az login
```

```
C:\Users>az login
A web browser has been opened at https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize. Please continue the login in the web browser. If no web browser is a
vailable or if the web browser fails to open, use device code flow with `az login --use-device-code`.
[
  {
    "cloudName": "AzureCloud",
    "homeTenantId": "[redacted]",
    "id": "11421a5c-5a30-43d4-8920-cab0433beeb7",
    "isDefault": true,
    "managedByTenants": [],
    "name": "PSL016288_Utimaco_REQ000000710202",
    "state": "Enabled",
    "tenantId": "[redacted]",
    "user": {
      "name": "[redacted].com",
      "type": "user"
    }
  }
]
```

Figure 16 : 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>"
```

```
C:\Users>az group create --location "centralindia" --name "UtimacoPSLGroup"
{
  "id": "/subscriptions/[redacted]/resourceGroups/UtimacoPSLGroup",
  "location": "centralindia",
  "managedBy": null,
  "name": "UtimacoPSLGroup",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
```

Figure 17 : 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
```

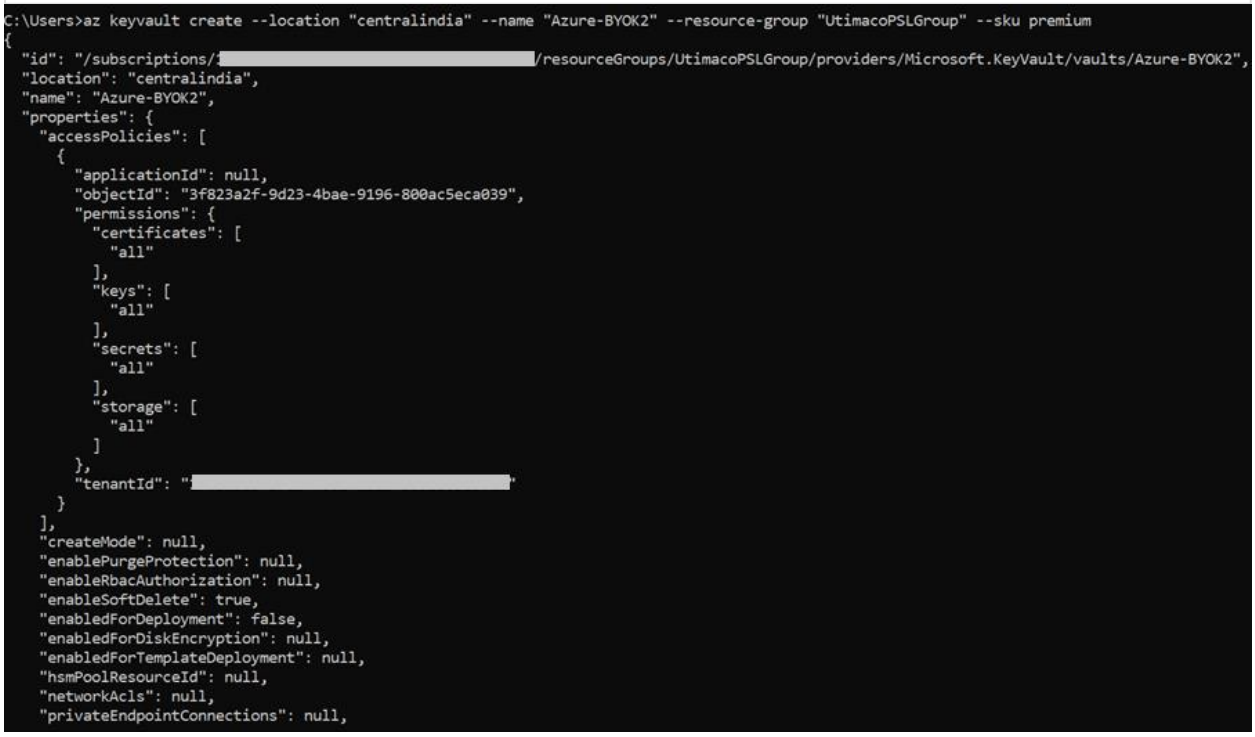
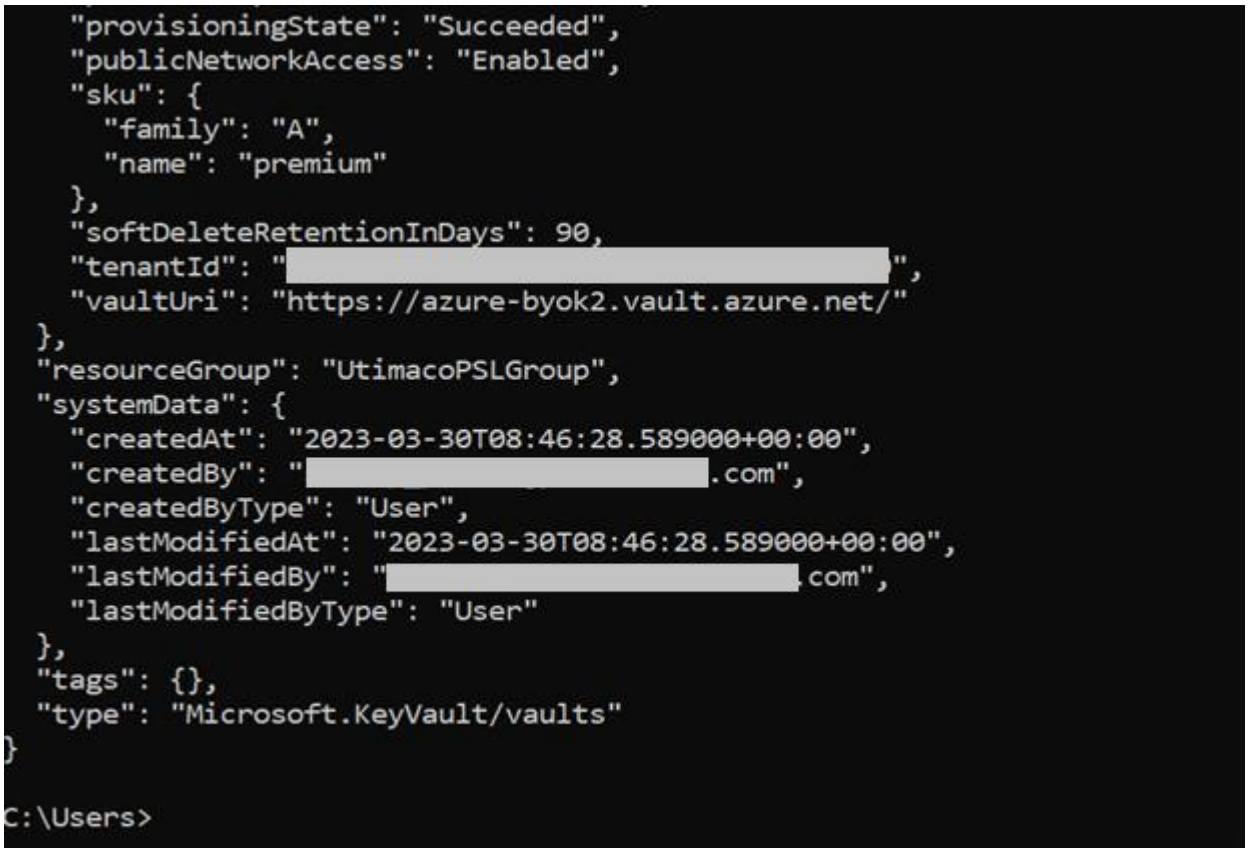


Figure 18 : Creating Azure Key Vault



5.5 Generating Key Exchange Key (KEK)

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

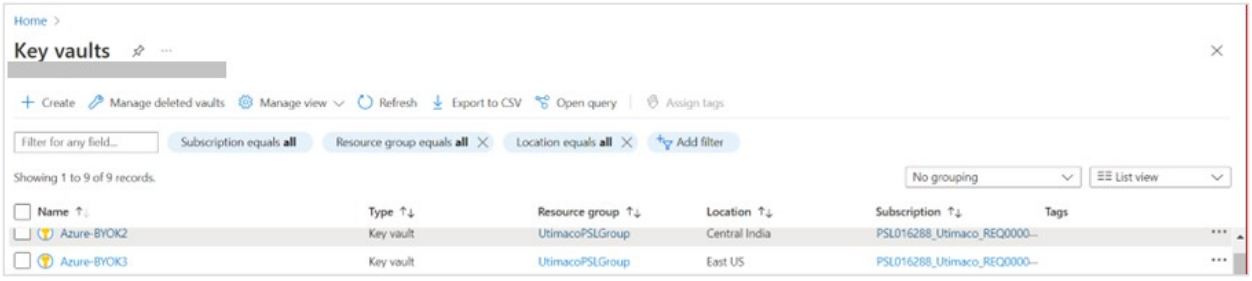


Figure 19 : Key vaults home page

1. Under the Settings menu select the Keys setting.

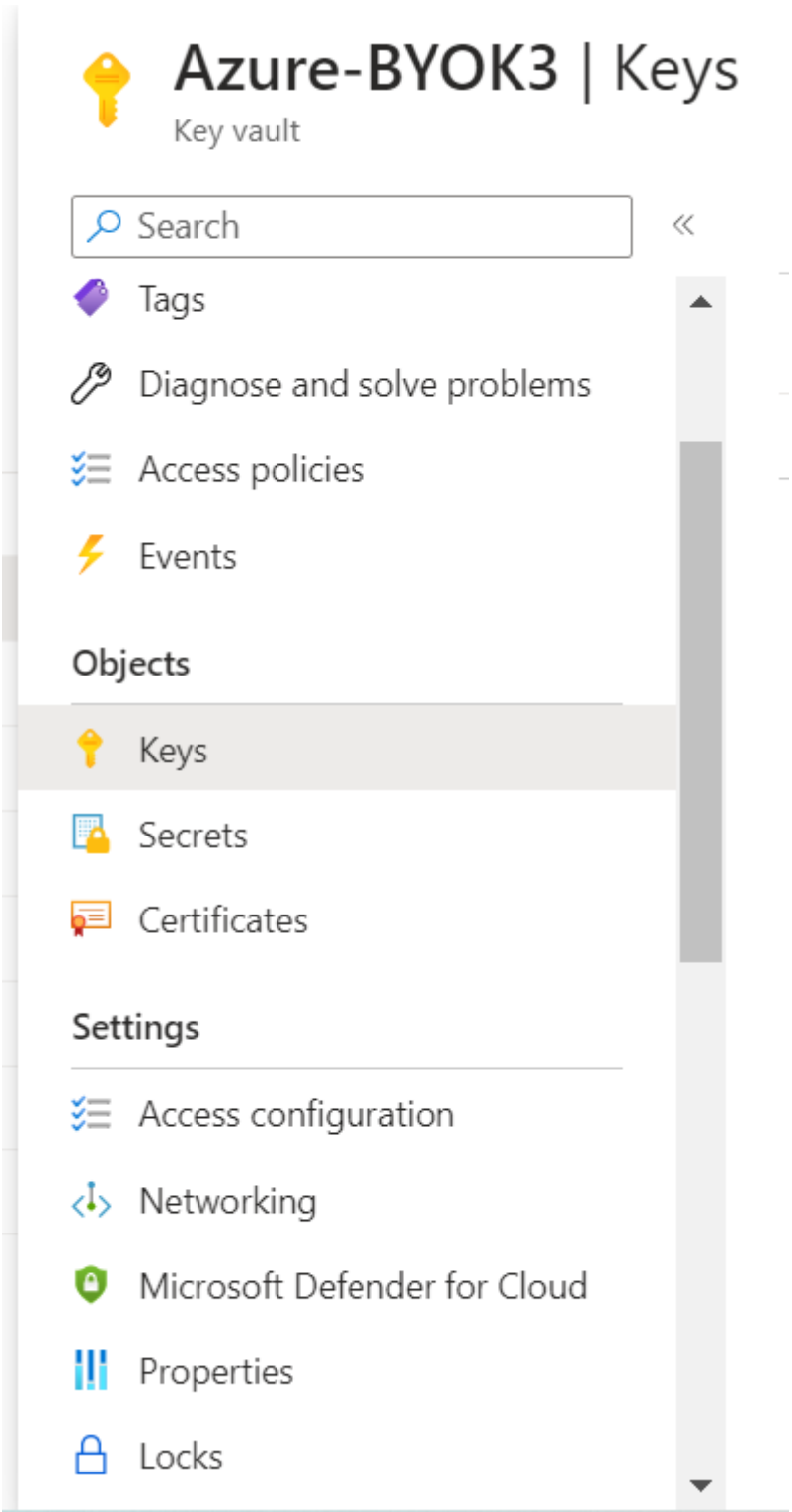


Figure 20 : Key vault menu

2. Click on Generate/Import.

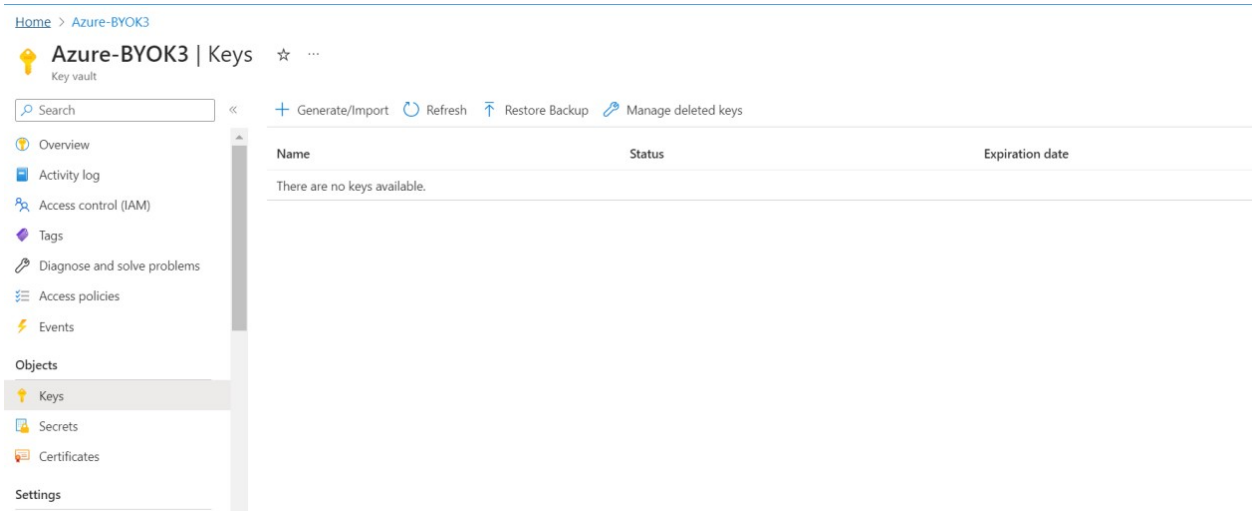


Figure 21 : Default key vault keys setting page

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.

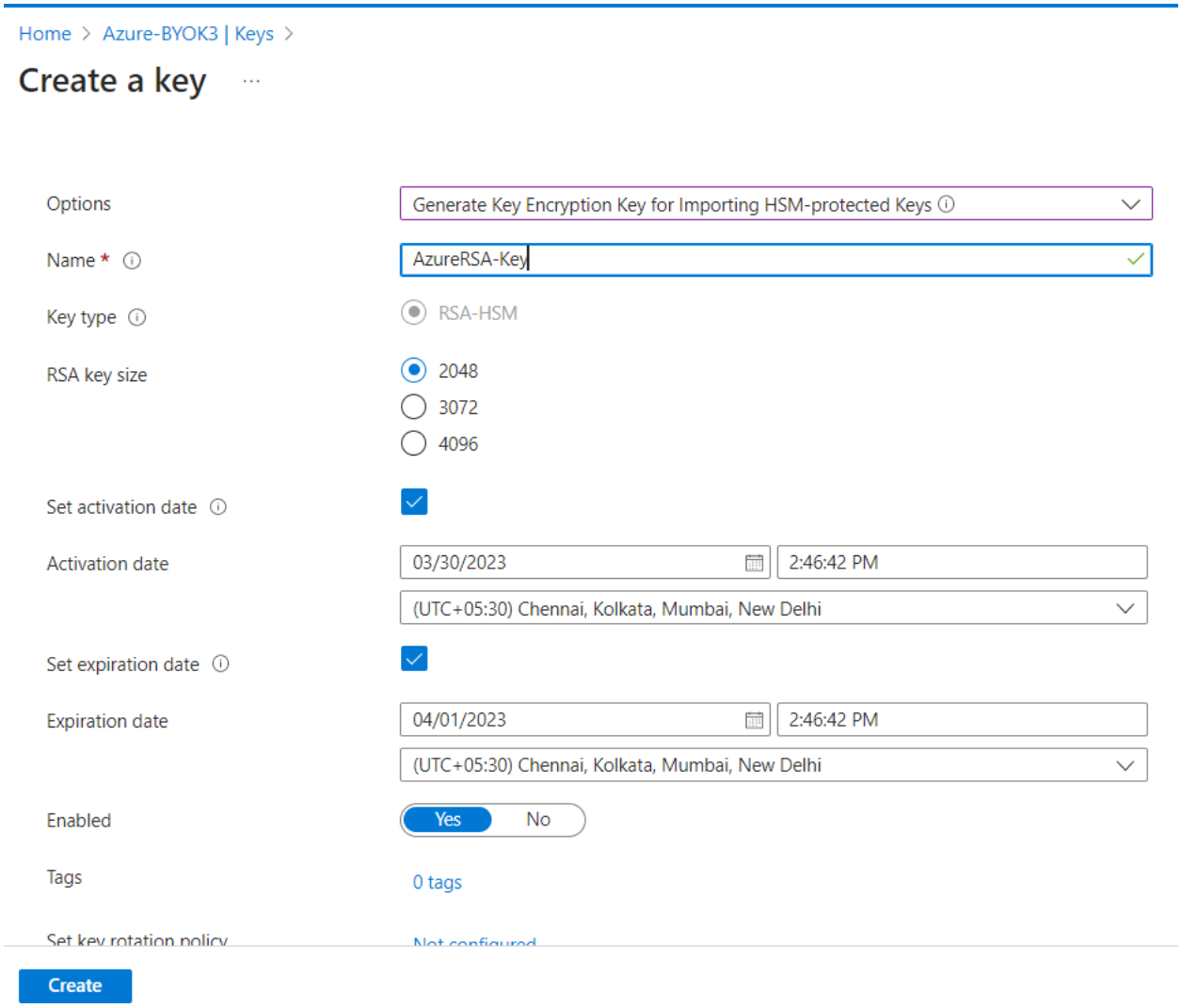


Figure 22 : Example of create a key page

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.

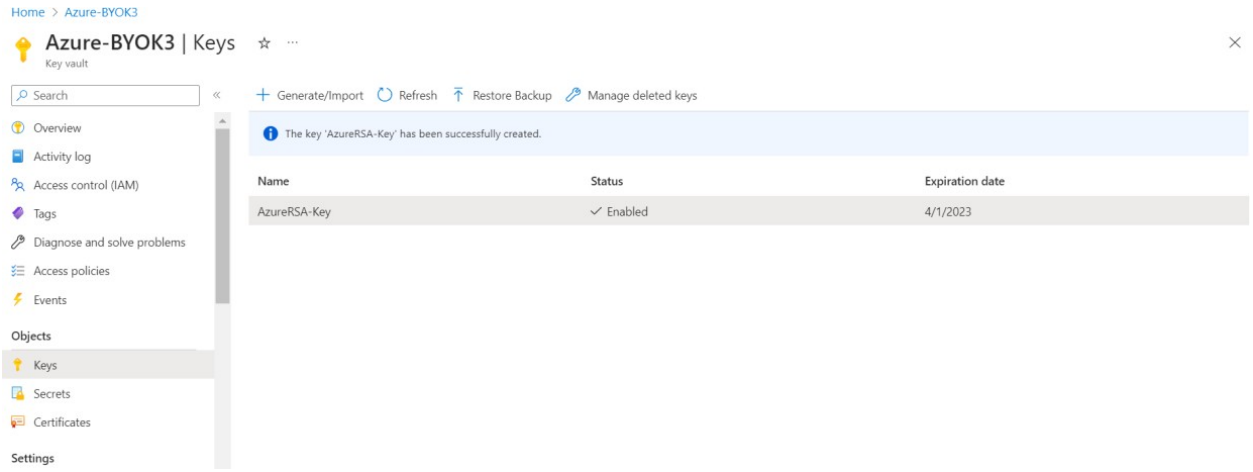


Figure 23 : Generated keys

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.

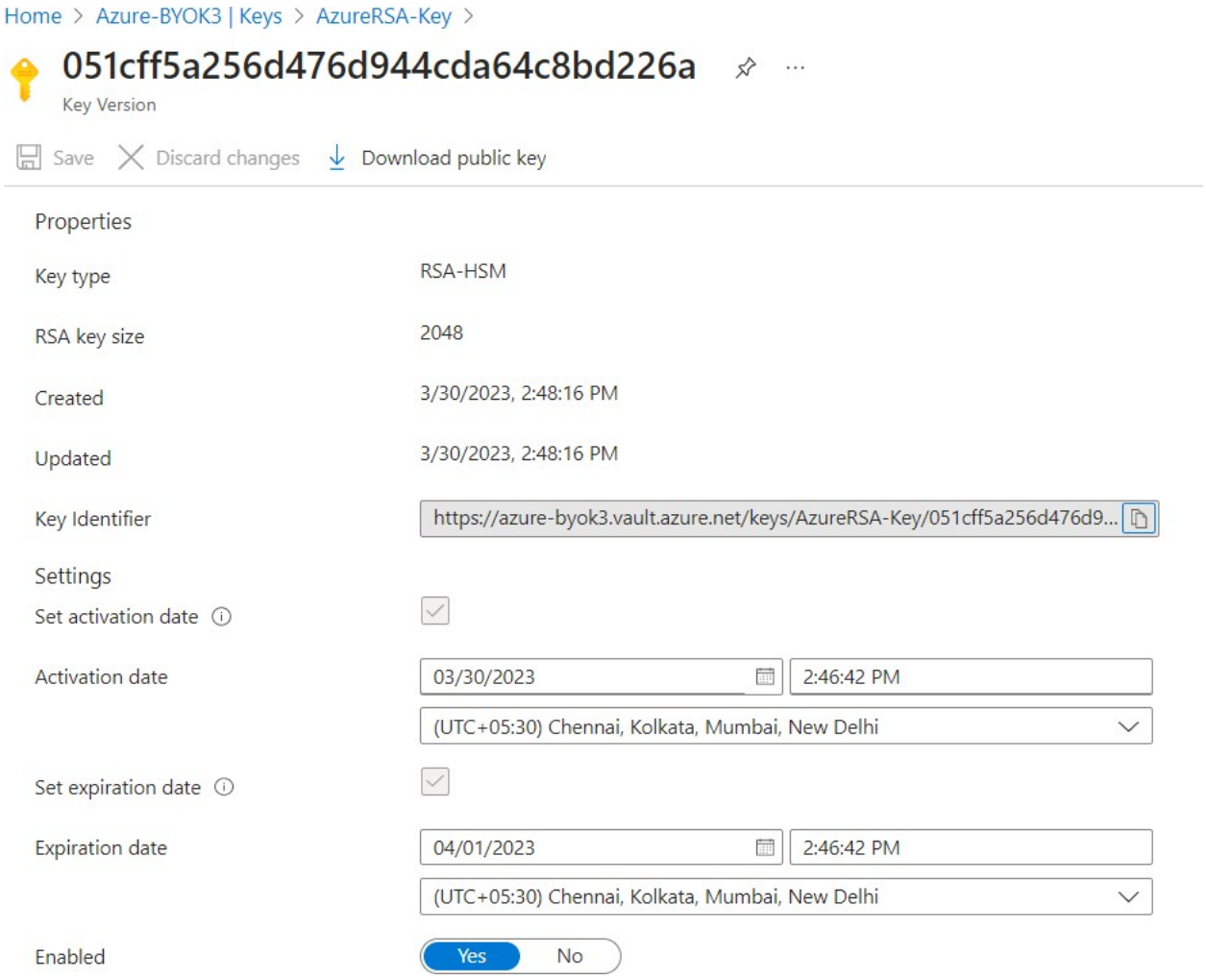


Figure 24 : Key properties

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
```

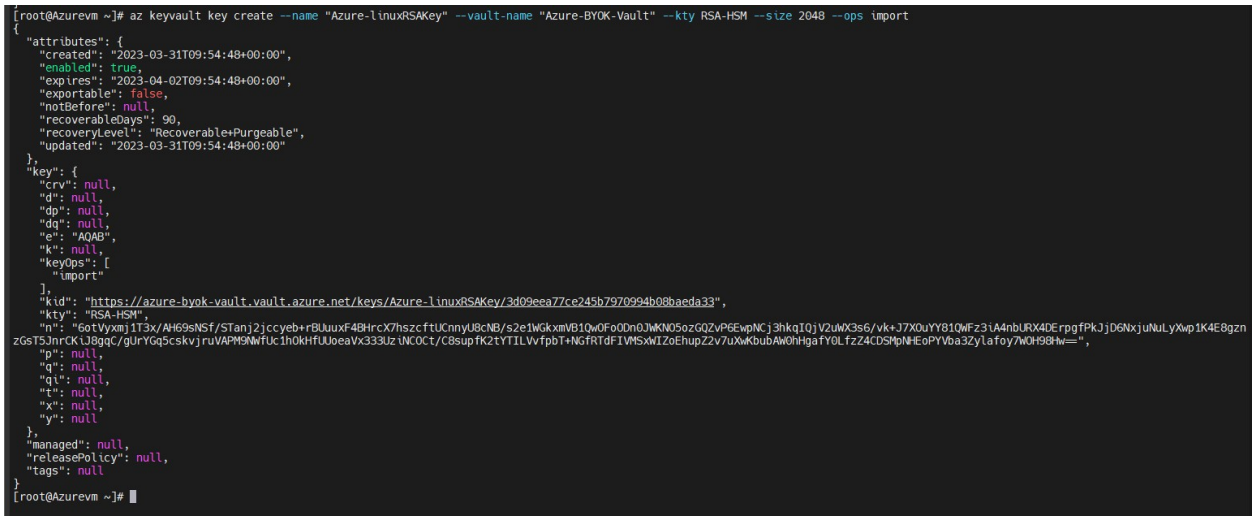


Figure 25 : Creating KEK

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
```

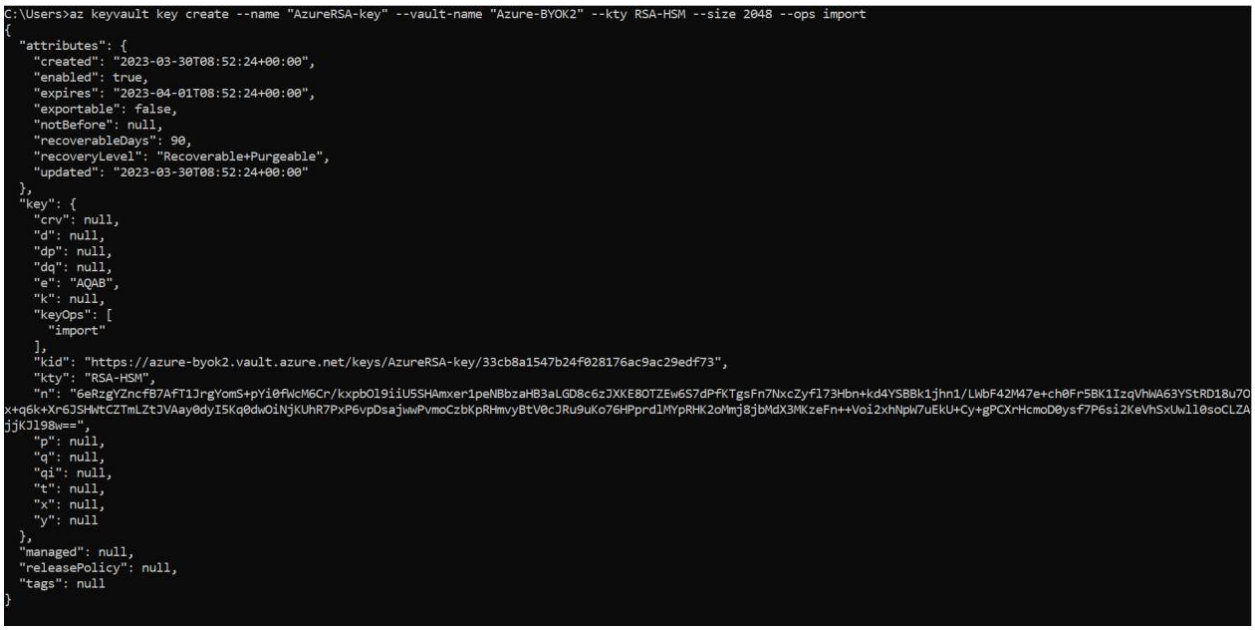


Figure 26 : Creating KEK on Windows

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.

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

Please ensure that you have administration rights when using the command line.

5.6.1 With Azure Portal

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

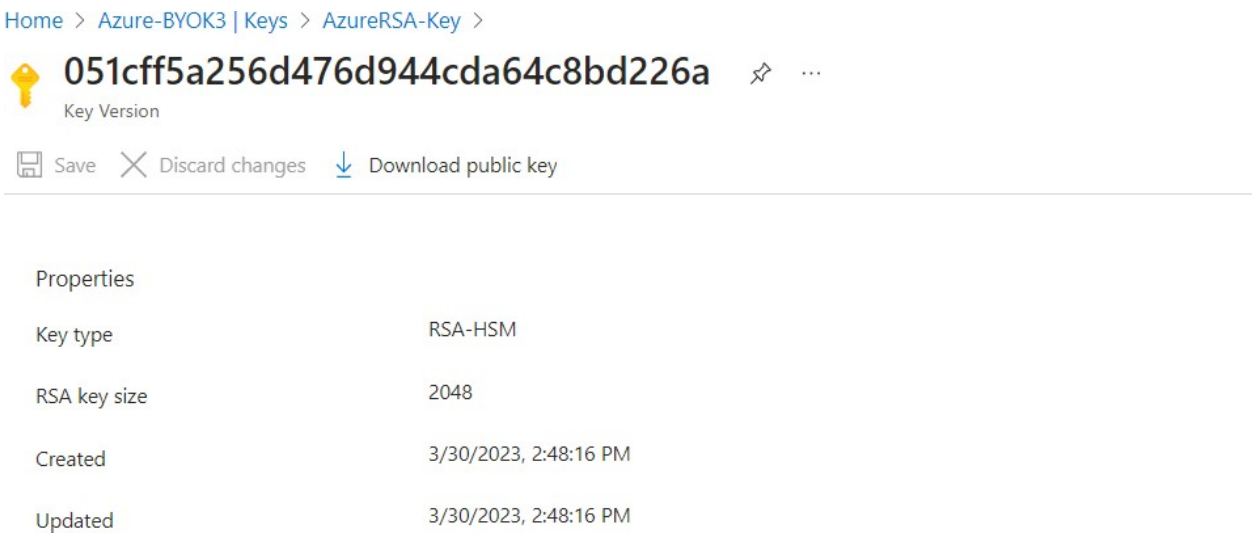


Figure 27 : Key properties

5.6.2 Downloading KEK Public Key on Linux

›_ Console

```
# az keyvault key download –name “<keyvault_key>” –vault-name “<keyvault>”
–

file <keyvault_key>.publickey.pem
```

```
[root@Azurevm ~]# az keyvault key download --name "Azure-linuxRSAKey" --vault-name "Azure-BYOK-Vault" --file RSALinux.publickey.pem
[root@Azurevm ~]# ll
total 4
-rw-r--r--. 1 root root 451 Mar 31 10:27 RSALinux.publickey.pem
[root@Azurevm ~]#
```

Figure 28 : Download key

5.6.3 Downloading KEK Public Key on Windows

›_ Console

```
> az keyvault key download –name “<keyvault_key>” –vault-name “<keyvault>”
–

file <keyvault_key>.publickey.pem
```

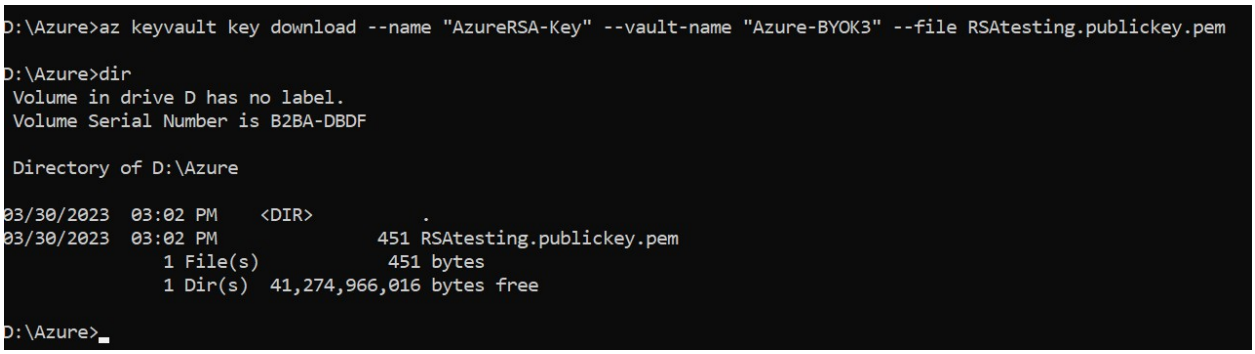


Figure 29 : Download key

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
```

```
[root@Azurevm bin]# p11tool2 slot=13 loginuser=12345678 PubKeyAttr=CKA_LABEL="tenant_public_RSAkey",CKA_MODULUS_BITS=2048 PrvK
eyAttr=CKA_LABEL="tenant_private_RSAkey",CKA_EXTRACTABLE=CK_TRUE GenerateKeyPair=RSA
[root@Azurevm bin]# p11tool2 slot=13 loginuser=12345678 listobjects


CKO_PUBLIC_KEY:

+ 1.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 113C6C8D-0EEB-4B23-AA07-151319222F03
  CKA_LABEL                     = tenant_public_RSAkey
  CKA_ID                        =



CKO_PRIVATE_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = CAA58BC2-F189-4B51-8B9C-063BDAF944EA
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_TRUE
  CKA_LABEL                     = tenant_private_RSAkey
  CKA_ID                        =
```

Figure 30 : 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.

```
[root@Azurevm bin]# p11tool2 slot=13 loginuser=12345678 PubKeyAttr=CKA_LABEL="tenant_public_ECCkey",CKA_EC_PARAMS=oid:NIST-P25
6 PrvKeyAttr=CKA_LABEL="tenant_private_ECCkey",CKA_EXTRACTABLE=CK_TRUE GenerateKeyPair=ECC
[root@Azurevm bin]# p11tool2 slot=13 loginuser=12345678 listobjects


CKO_PUBLIC_KEY:

+ 1.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = 113C6C8D-0EEB-4B23-AA07-151319222F03
  CKA_LABEL                     = tenant_public_RSAkey
  CKA_ID                        =


+ 1.2
  CKA_KEY_TYPE                  = CKK_ECDSA
  CKA_UNIQUE_ID                 = 3990E097-F060-46DD-B5C9-58DC3592AA9E
  CKA_LABEL                     = tenant_public_ECCkey
  CKA_ID                        =



CKO_PRIVATE_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = CAA58BC2-F189-4B51-8B9C-063BDAF944EA
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_TRUE
  CKA_LABEL                     = tenant_private_RSAkey
  CKA_ID                        =


+ 2.2
  CKA_KEY_TYPE                  = CKK_ECDSA
  CKA_UNIQUE_ID                 = EE9F7516-EE5D-4138-8209-306937E95343
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_TRUE
  CKA_LABEL                     = tenant_private_ECCkey
  CKA_ID                        =
```

Figure 31 : 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 RSA Key

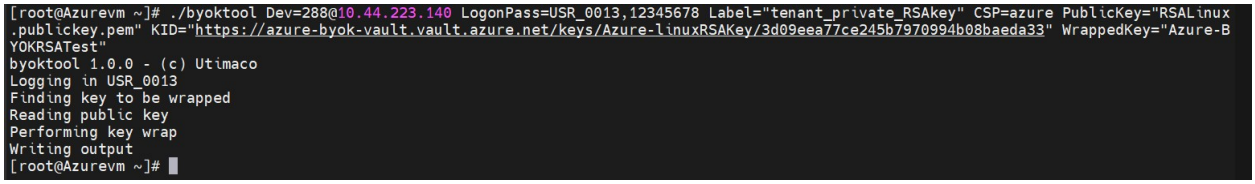


Figure 32 : Wrap the tenant key for RSA

For ECC Key

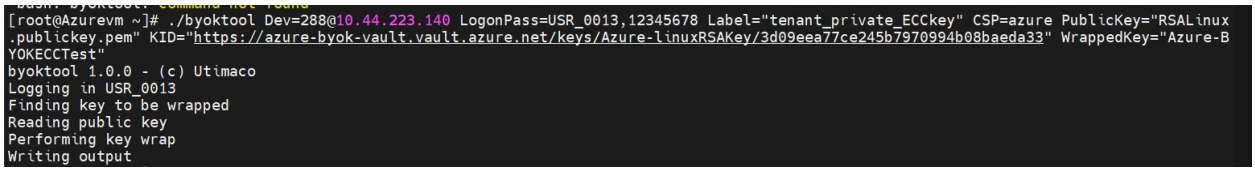


Figure 33 : Wrap the tenant key for ECC

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
```

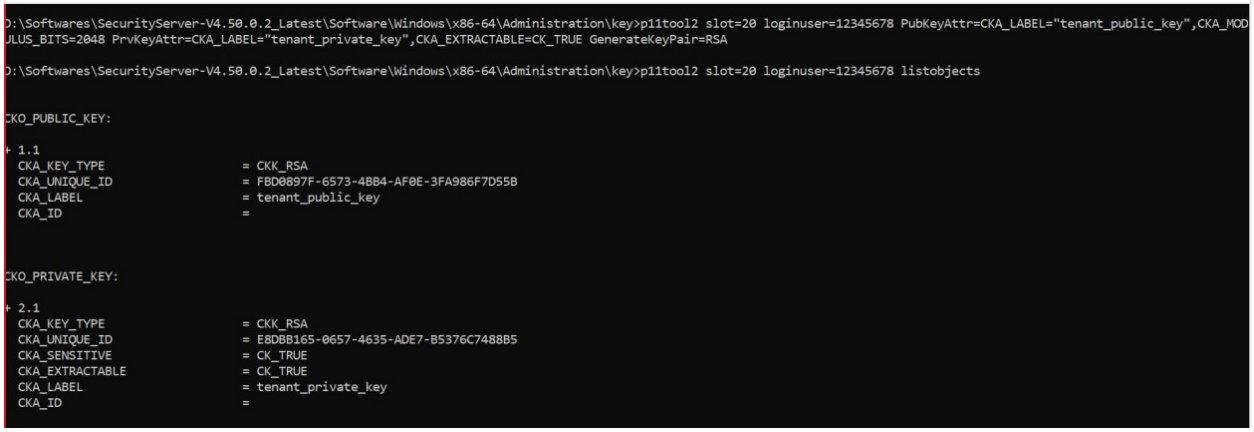


Figure 34 : List newly generated RSA keys

<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.

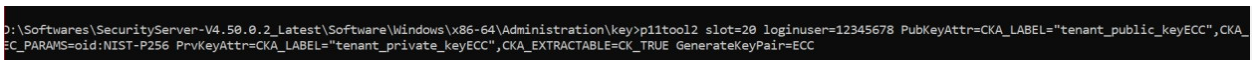


Figure 35 : List newly generated ECC keys

```
D:\Softwares\SecurityServer-V4.50.0.2_Latest\Software\Windows\x86-64\Administration\key>p11tool2 slot=20 loginuser=12345678 listobjects


CKO_PUBLIC_KEY:

+ 1.1
  CKA_KEY_TYPE                  = CKK_RSA
  CKA_UNIQUE_ID                 = FBD0897F-6573-4BB4-AF0E-3FA986F7D55B
  CKA_LABEL                     = tenant_public_key
  CKA_ID                        =


+ 1.2
  CKA_KEY_TYPE                  = CKK_ECDSA
  CKA_UNIQUE_ID                 = BC5E2B02-4EA4-405C-90A0-100ED4B43CD1
  CKA_LABEL                     = tenant_public_keyECC
  CKA_ID                        =



CKO_PRIVATE_KEY:

+ 2.1
  CKA_KEY_TYPE                  = CKK_ECDSA
  CKA_UNIQUE_ID                 = 5ACC9691-8851-4D7C-8195-0EF9CC0A6983
  CKA_SENSITIVE                 = CK_TRUE
  CKA_EXTRACTABLE               = CK_TRUE
  CKA_LABEL                     = tenant_private_keyECC
  CKA_ID                        =
```

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 RSA Key

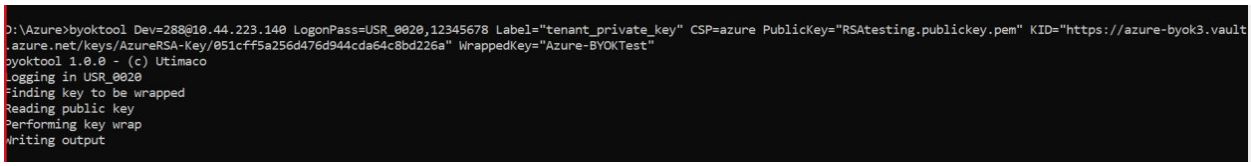


Figure 36 : Wrap the tenant key for RSA

For ECC Key

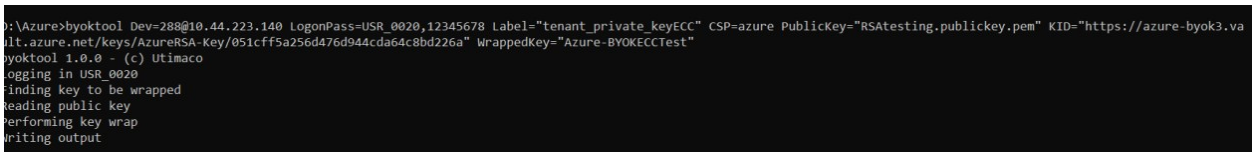


Figure 37 : Wrap the tenant key for ECC

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>"
```

```
[root@Azurevm ~]# az keyvault key import --vault-name Azure-BYOK-Vault --name Azure-linuxrsaKey --byok-file "Azure-BYOKRSATest"
{
  "attributes": {
    "created": "2023-04-06T10:34:52+00:00",
    "enabled": true,
    "expires": null,
    "exportable": false,
    "notBefore": null,
    "recoverableDays": 90,
    "recoveryLevel": "Recoverable+Purgeable",
    "updated": "2023-04-06T10:34:52+00:00"
  },
  "key": {
    "crv": null,
    "d": null,
    "dp": null,
    "dq": null,
    "e": "AAEAAQ==",
    "k": null,
    "keyOps": [
      "encrypt",
      "decrypt",
      "sign",
      "verify",
      "wrapKey",
      "unwrapKey"
    ],
    "kid": "https://azure-byok-vault.vault.azure.net/keys/Azure-linuxrsaKey/12edcdc751a74ca99676a8217a76d83d",
    "kty": "RSA-HSM",
    "n": "wsMx9hLaYvY9seU8hcjRq2H5WI20XFVy+JdBHK3r4VZ0UIY8F4SZDZNJgEMdP4AN/4vXGLK52/03q+XiPY5YzgdCf71/H4klhnh7fO4iObHiHa6YRGAhOIyzHfbfNPY1hRstvRnp8nnMTtOTwQf0aM5XGNrLARI0j8jVGVzGIfZdBObA8dzohCI45Es4WAXtKUJENhuZxjyMvvD2DDj1KWMx5kS/eNWESzXKSIWOr/ARnul56IOrwtJuxMXAX95dB1AmQjE6+AnfMaAe02E+1wT/d2iAr0ogTm/HdwajBXREa+DuQ5yg0rv+cqfQYp+/8K/uvocwLxC9CTzVozq8zw==",
    "p": null,
    "q": null,
    "qi": null,
    "t": null,
    "x": null,
    "y": null
  },
  "managed": null,
  "releasePolicy": null,
  "tags": null
}
[root@Azurevm ~]#
```

Figure 38 : Importing tenant key

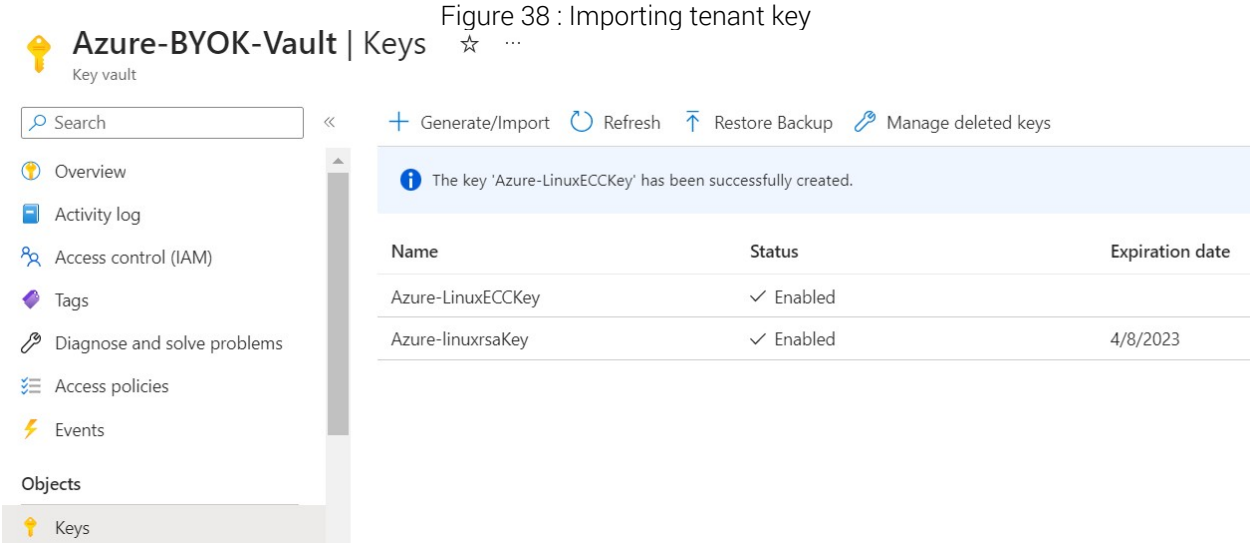


Figure 39 : List importing tenant key

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 40 : List importing tenant key

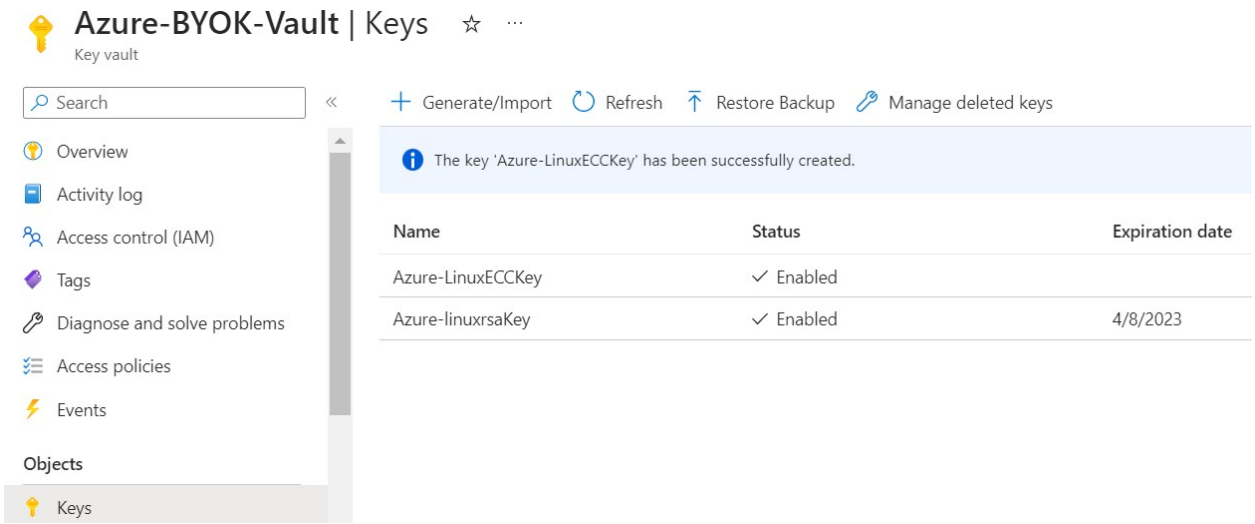


Figure 41 : List importing tenant key

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 RSA Key

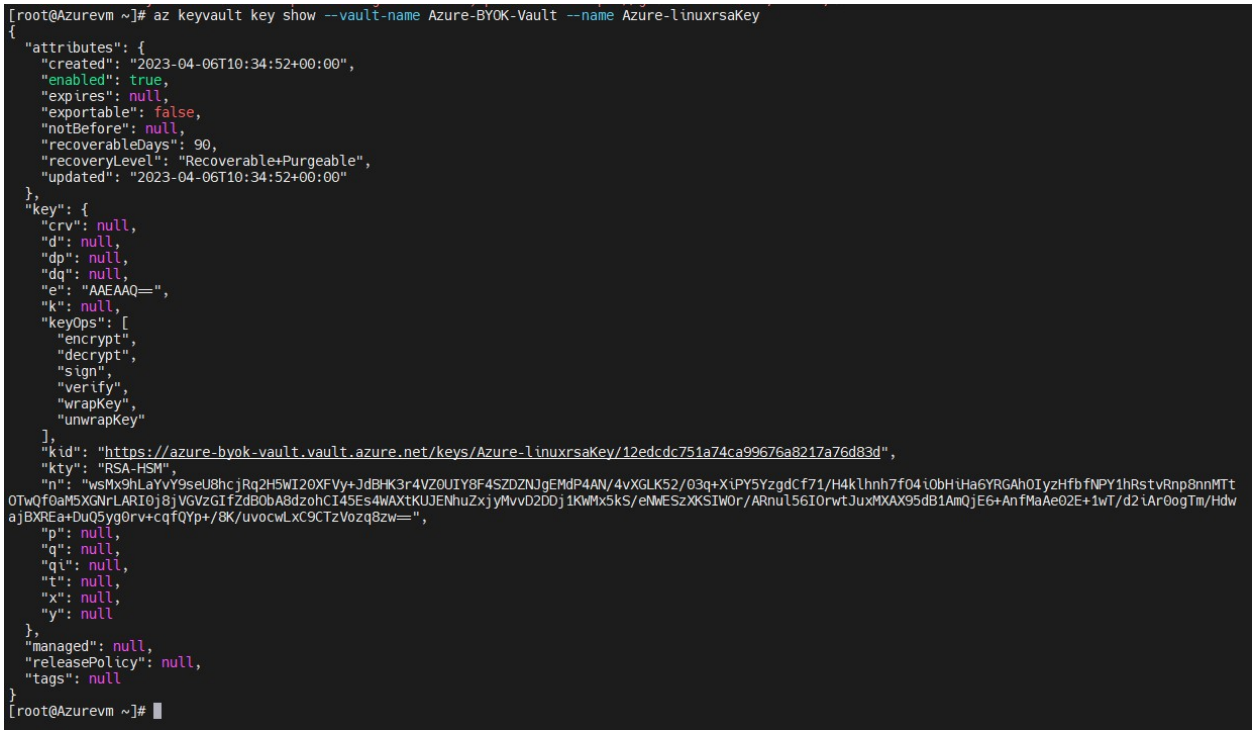


Figure 42 : List RSA key

For ECC Key

```
[root@Azurevm ~]# az keyvault key show --vault-name Azure-BYOK-Vault --name Azure-LinuxECCKey
{
  "attributes": {
    "created": "2023-04-06T10:38:58+00:00",
    "enabled": true,
    "expires": null,
    "exportable": false,
    "notBefore": null,
    "recoverableDays": 90,
    "recoveryLevel": "Recoverable+Purgeable",
    "updated": "2023-04-06T10:38:58+00:00"
  },
  "key": {
    "crv": "P-256",
    "d": null,
    "dp": null,
    "dq": null,
    "e": null,
    "k": null,
    "keyOps": [
      "sign",
      "verify"
    ],
    "kid": "https://azure-byok-vault.vault.azure.net/keys/Azure-LinuxECCKey/aa6686671dc14dbfa15d4634876e4f43",
    "kty": "EC-HSM",
    "n": null,
    "p": null,
    "q": null,
    "qi": null,
    "t": null,
    "x": "uJh3FlHgcElNroili5ko2w+Ag1MuxHX3eKr8FJsDkKc=",
    "y": "zKiHE0y2SXhLmeK7A0Hhj90UH6iZd7OYDIkh9z/8qSQ="
  },
  "managed": null,
  "releasePolicy": null,
  "tags": null
}
[root@Azurevm ~]#
```

Figure 43 : 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>"
```

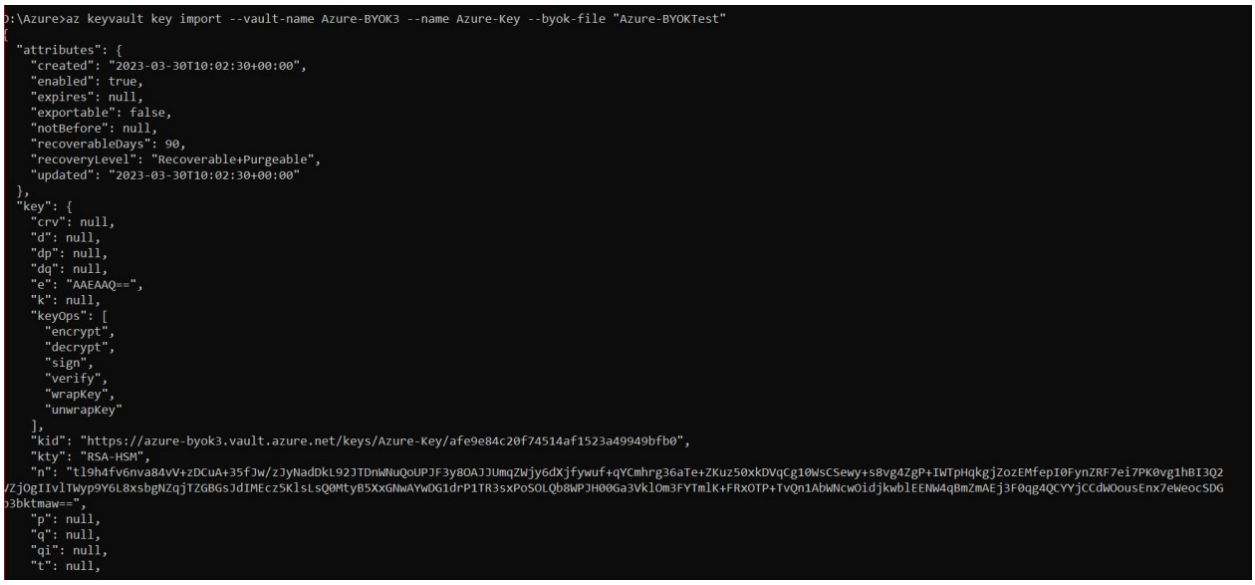


Figure 44 : Importing tenant key

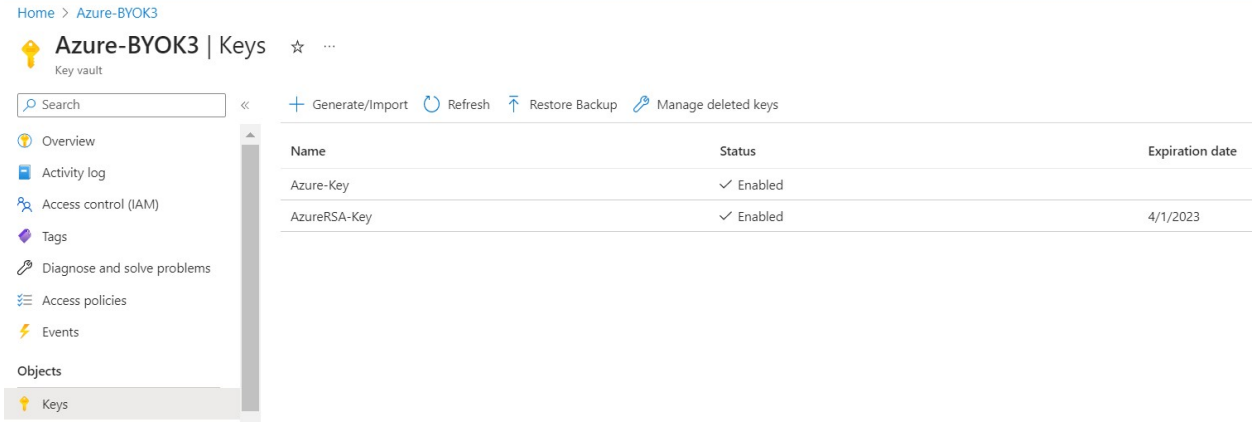


Figure 45 : List importing tenant key

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.

```
D:\Azure>az keyvault key import --vault-name Azure-BYOK3 --name Azure-keyECC --byok-file "Azure-BYOKECCTest" --kty EC --curve P-256
{
  "attributes": {
    "created": "2023-03-30T10:17:45+00:00",
    "enabled": true,
    "expires": null,
    "exportable": false,
    "notBefore": null,
    "recoverableDays": 90,
    "recoveryLevel": "Recoverable+Purgeable",
    "updated": "2023-03-30T10:17:45+00:00"
  },
  "key": {
    "crv": "P-256",
    "d": null,
    "dp": null,
    "dq": null,
    "e": null,
    "k": null,
    "keyOps": [
      "sign",
      "verify"
    ],
    "kid": "https://azure-byok3.vault.azure.net/keys/Azure-keyECC/ab2a9d5b23664bb1a39f09001429a270",
    "kty": "EC-HSM",
    "n": null,
    "p": null,
    "q": null,
    "qi": null,
    "t": null,
    "x": "h1BJYHaYa3CvElZpv7eP67chJLpSUomcWPwGnbEjpOA=",
    "y": "qiTkEzXZkD9shayABkwNePVrQoCcBrbheGiotCsCPJ4="
  },
  "managed": null,
```

Figure 46 : Importing tenant key

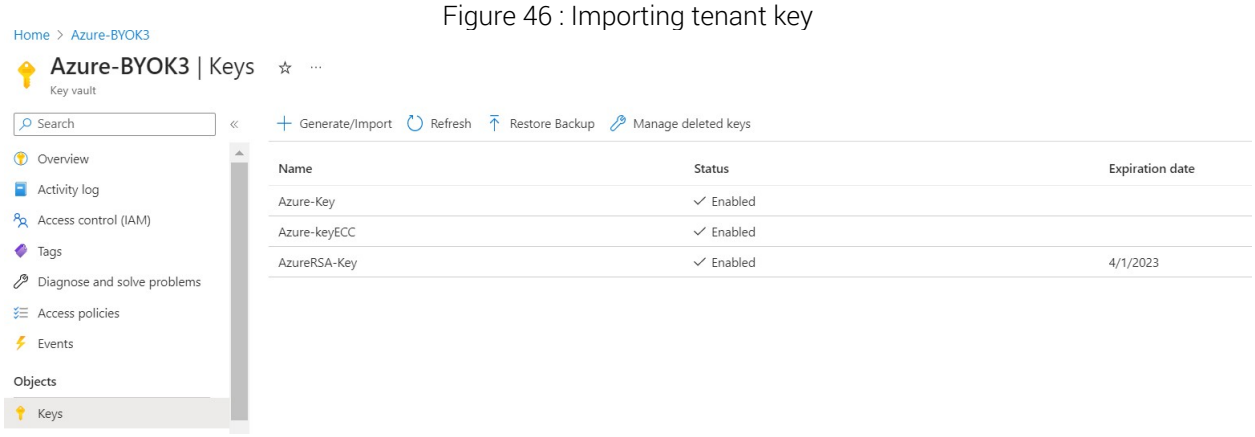


Figure 47 : List importing tenant key

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 RSA Key

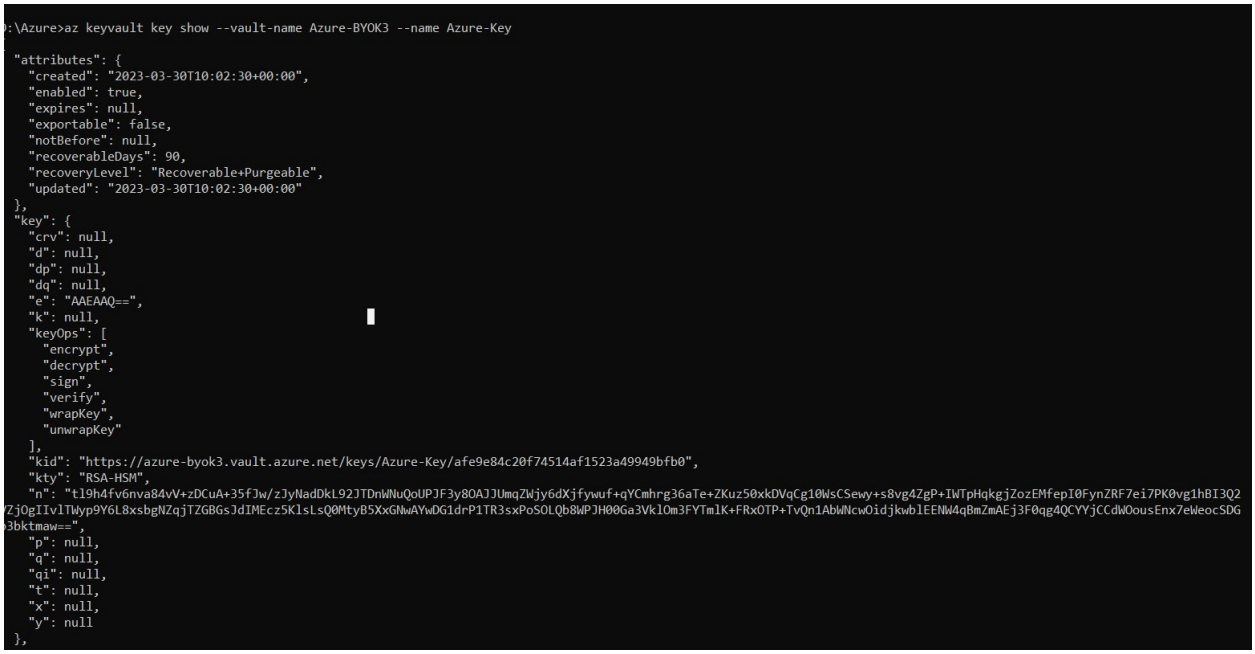


Figure 48 : List RSA key

For ECC Key

```
D:\Azure>az keyvault key show --vault-name Azure-BYOK3 --name Azure-KeyECC
{
  "attributes": {
    "created": "2023-03-30T10:17:45+00:00",
    "enabled": true,
    "expires": null,
    "exportable": false,
    "notBefore": null,
    "recoverableDays": 90,
    "recoveryLevel": "Recoverable+Purgeable",
    "updated": "2023-03-30T10:17:45+00:00"
  },
  "key": {
    "crv": "P-256",
    "d": null,
    "dp": null,
    "dq": null,
    "e": null,
    "k": null,
    "keyOps": [
      "sign",
      "verify"
    ],
    "kid": "https://azure-byok3.vault.azure.net/keys/Azure-keyECC/ab2a9d5b23664bb1a39f09001429a270",
    "kty": "EC-HSM",
    "n": null,
    "p": null,
    "q": null,
    "qi": null,
    "t": null,
    "x": "h1BJYHaYa3CvElZpv7eP67chJLpSUomcWPwGnbEjpOA=",
    "y": "qiTkEzXZkD9shayABkwNePVrQoCcBrbheGiotCsCPJ4="
  },
  "managed": null,
  "releasePolicy": null,
  "tags": null
}

D:\Azure>
```

Figure 49 : 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 see 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 CryptoServer product documentation is also available at the Utimaco IS GmbH website: https://utimaco.com/

For more information regarding Azure Key Vault, please see the following links:

Key Vault BYOK documentation - How to generate & transfer HSM-protected keys – BYOK – Azure Key Vault | Microsoft Docs

Azure CLI - Install the Azure CLI | Microsoft Docs

Azure Key Vault pricing (where Premium tier is referenced) - Pricing Details - Key Vault | Microsoft Azure, this page provides side-by-side comparison of Key Vault pricing tiers and capabilities.

8 References

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

Table 7: References

9 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
Germanusstr. 4
52080 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.