



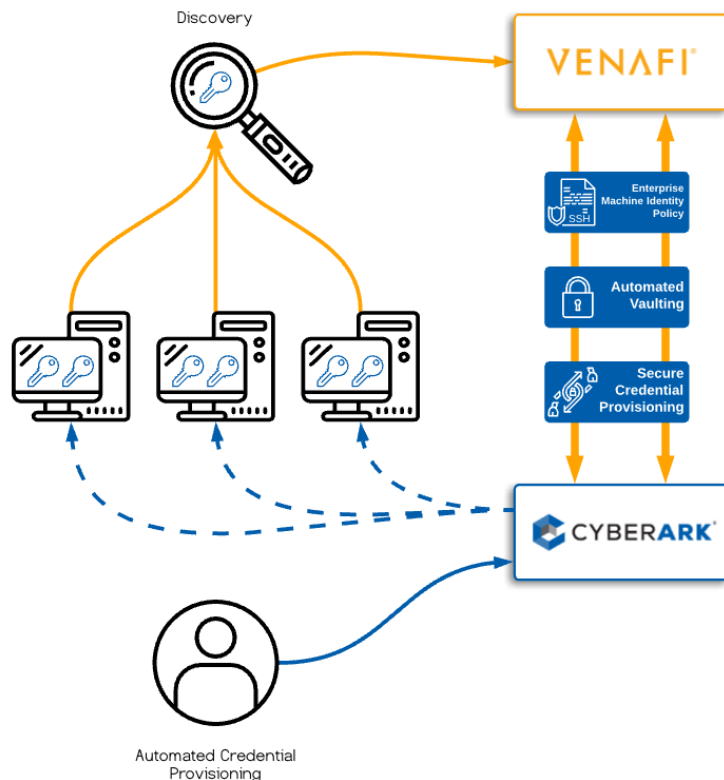
Venafi CyberArk SSH Key Discovery Module

Version 1.0



Description

The CyberArk Discovery Module has integrated with the Venafi Trust Protection Platform (TPP) to provide a comprehensive SSH Key Lifecycle Management capability. Utilizing the Venafi Network Discovery capability to conduct a scan for the environment's unmanaged SSH keys. The CyberArk Discovery Module receives a list of Device GUIDs from the discovery results to initiate a sequence of logic to identify and provision SSH keys to be provisioned into the CyberArk Vault using the CyberArk WebServices SDK.



The user has the ability to assign what usernames associated to the SSH Keys are added to the Vault through the DiscoverConfig.yml. The module will create a Self Service Key within TPP to identify the physical location of the Keys provisioned in the Vault. Once a Self Service Key is added to TPP, the process of extracting the key's sensitive data begins. The key material is extracted from the Venafi TPP solution in PuTTY format to ensure CyberArk Session Manager capabilities can be performed with the keys, then the CyberArk Discovery module provisions the key using the Add Accounts WebServices SDK api endpoint with parameters identified in the DiscoverConfig.yml

Prerequisites

The Discovery Module resides on the same platform as the SCIM Server, and has the same system requirements as the SCIM server. For further information on setting up the SCIM Server please refer to the SCIM Implementation guide:

<https://cyberark-customers.force.com/mplace/s/#a3550000000EiAfAAK-a3950000000jT7AAI>

Accounts for Discovery Module

- CyberArk account that authenticates to the WebServices SDK with permissions to Add Accounts
- Account to authenticate to Venafi VedSDK solution
- Adding **KeySetID** as a new Server File Category

System Requirements

- ** 8 Core Processors
- ** 32GB of RAM
- Microsoft Windows Server 2012 R2 (also supports Server 2016)
- Java Virtual Machine version 1.8

** These are suggested requirements for the SCIM Server but if you are using this exclusively for Discovery this much processing may not be needed.

CyberArk Requirements

- CyberArk SCIM Server 1.1.4
 - CyberArk Vault Server v9.8 or higher
 - AIM Credential Provider v9.8 or higher

SSL Communication Requirements

- Ensure the SCIM Server's Java Keystore is properly configured
- Ensure the Root certificate that signed the TPP Web Cert is properly configured in the Java Truststore

Implementation

Now that the SCIM Server is installed and functioning correctly, it will be necessary to build out the configurations for the Discovery module.

Account Objects for Discovery Module

For information regarding the process of adding accounts, here is the link to the CyberArk documentation for this process:

<https://docs.cyberark.com/Product-Doc/OnlineHelp/PAS/Latest/en/Content/PASIMP/Onboarding-Accounts-and-SSH-Keys.htm>

Adding Server File Category

The Discovery module adds the KeySet ID to the SSH Key account object to be used for communication with Venafi in future integrations with the Venafi product.

1. Access and authenticate to the **PrivateArk Client**
2. **File -> Server File Category**
3. Click **New**
4. In the **Name** field input **KeySetID -> OK**

WebServices SDK

This account will be used to authenticate to the WebServices SDK and used to create the account objects found in the Discovery process.

5. Select the **Accounts** tab
6. In the **Accounts View**, click **Add Account**
7. In the **Select System Type** window select **Application**, click Next
8. In the **Assign to platform** window, select the **Platform** modified in the last section, click Next
NOTE: If it is a local CyberArk Vault account, assign the CyberArk Vault Platform for proper management of the credential
9. Select or Create a **Safe** to store the object, click Next
NOTE: Storing in the SCIM Config safe provides a centralized location for easier management
10. Provide a **Username**
11. Provide the **Address**
NOTE: If you are using a local Vault account this is going to be the address of the Vault
12. Provide a **Password**
NOTE: You can customize the account name to make it easier for placing it in the config files

13. Scroll down to the **Optional** fields and in the **WebServiceURL** field enter the WebServices SDK **baseURL**:

https://<PVWA_URL>/PasswordVault

14. Select **Add**

Venafi VedSDK Account Object

This account will be used to authenticate to the Venafi solution to retrieve scan results and interact with the VedSDK:

1. Adding of the "**KeySetID**" **Server File Category** will be required
2. Navigate to the **Platform Management** view
3. Select **Import Platform**
4. In the Windows Explorer window locate and select the zip file downloaded from the MarketPlace, click Open
5. The Venafi Discovery SSH Keys platform should be imported
6. Select the **Accounts** tab
7. In the **Accounts View**, click **Add Account**
8. In the **Select System Type** window select **Operating System**, click Next
9. In the **Assign to platform** window, select the **Venafi Discovery SSH Keys** platform, click Next
10. Select or Create a **Safe** to store the object, click Next

NOTE: Storing in the SCIM Config safe provides a centralized location for easier management

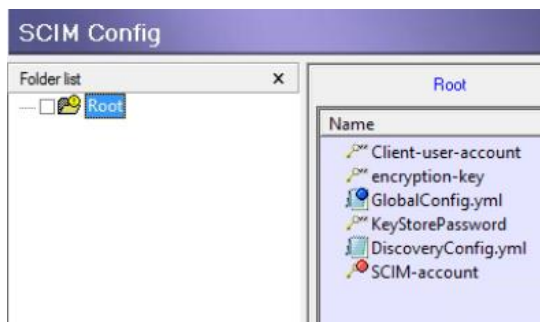
11. Provide the **Username**
12. Provide the **Address** of the **Venafi** solution
13. Provide a **Password**

NOTE: You can customize the account name to make it easier for entering in the config files

14. Scroll down to the **Optional** fields and if you added the **WebServiceURL** field, enter the Venafi VedSDK API **baseURL**
15. Select **Add**

Discovery Configuration

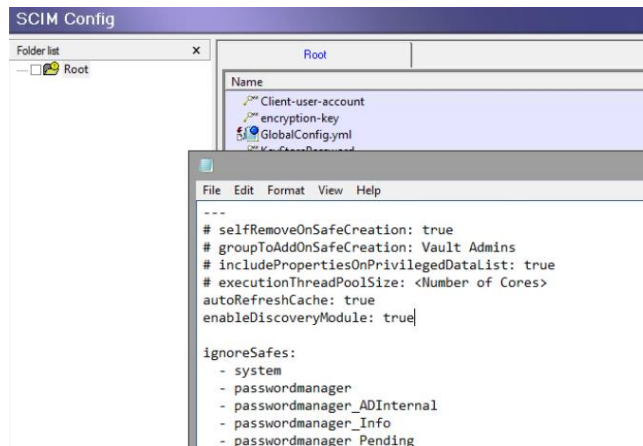
All of the necessary files for the Discovery module are located in the **SCIM Config** safe in the CyberArk Vault.



GlobalConfig.yml

This yaml file is the overall configuration for the entire Discovery/SCIM capability. Add the following syntax to enable the Discovery module:

1. Using the PrivateArk Client authenticate to the Vault navigate to the **SCIM Config** safe.
2. Retrieve the GlobalConfig.yml file and edit it
3. Add the following syntax to towards the top of the file as outlined below
enableDiscoveryModule: true



4. **Save** the file and **Return** it back to the safe

DiscoveryConfig.yml

This yaml file defines the work for the entire Discovery module. In the **CyberArk SCIM** folder in the SCIM installation zip file there is an example DiscoveryConfig.yml to modify. Below is an explanation of the parameters in the Discovery configuration.

webservice – Defines the Vaulted Account to authenticate to the CyberArk WebServices SDK API

connectionAccount	defines the object used for authenticating to the WebServices SDK	
	safe	Safe where the account object will be stored
	folder	Folder where the account object resides
	object	Object Name of the account object
	baseUrl	The base URL for WebServices SDK Example: https://components.cyberarkdemo.com/PasswordVault

discoveryEndpoints – Defines how the Discovery Module will interact with the Venafi endpoint

name	Name given to the Discovery job – venafi	
connectionAccount	Vaulted account object used to authenticate to VedSDK API	
	safe	Safe where the account object will be stored
	folder	Folder where the account object resides
	object	Object Name of the account object
privilegeDefinition	Parameters to define what keys will be vaulted	
	addBasicSearch	Boolean value indicating the inclusion of the following regex string: <code>(.*admin.*)((.*root.*)((.*superuser.*)((.*supervisor.*)((toor) (baron) (avatar) (adm) (wheel)</code>
	folderID	The GUID associated to the folder in the Venafi Trust Protection Platform that will house the information on where the key physically resides
	privKeyUsername	List of usernames the Discovery Module will identify for provisioning into the Vault.
	endpointThreshold	Integer value
	deleteFromSource	Boolean value to indicate if the Discovery Module will communicate back to Venafi that the keys identified and provisioned into the Vault will be deleted from the target they were discovered on
addDiscoveredKeys	Configurations for storing the discovered keys in the CyberArk Vault	
	skipDuplicates	Boolean value that enacts a workflow that identifies if the discovered keys are already in the Vault and will not Provision in the Vault or create a SelfServiceKey entry in TPP
	safe	The Safe in the CyberArk Vault where the discovered keys will be provisioned to
	platformID	The policyID value for the Platform assigned to the keys being provisioned
	folder	The Folder within the Safe where the Keys will be stored. If left empty, it will be stored in the Root folder
	disableAutoMgmt	Boolean value to indicate if the discovered Key Object will be rotated by the CPM

	disableAutoMgmtReason	If disableAutoMgmt is set to True, then this parameter will need a free text reason for why management will be disabled
	venafiOwner	User within the Venafi Trust Protection Platform that has access to the SelfServiceKeys housed in the folderID assigned above
properties	Defines the free text File Category values assigned to the Platform managing the Discovered Keys	
	Port	SSH Key Platforms have Port as a File Category default is 22
	Comment	Free text Comment field available in the UnixSSHKeys Platform

Java Keystore and Truststore Commands

This is an **example** of taking a p12 certificate file and storing it in the Java Keystore as a jks and then converting it to a PKCS12

```
C:\Program Files (x86)\Java\jre1.8.0_xxx\bin>keytool.exe -importkeystore -
destkeystore <MY_KEYSTORE>.jks -srckeystore <MY_FILE>.pfx/p12 -srcstoretype PKCS12 -
deststoretype jks -deststorepass <PASSWORD> -srcalias <ALIAS> -destalias <NEW_ALIAS>
```

```
keytool.exe -importkeystore -srckeystore <MY_KEYSTORE>.jks -destkeystore
<MY_KEYSTORE>.p12 -srcstoretype JKS -deststoretype PKCS12 -deststorepass <PASSWORD>
```

This is an example of storing a Root certificate and storing it in the **cacerts** truststore. You will be prompted for a password for this command and the OOB truststore password is "changeit"

```
C:\Program Files (x86)\Java\jre1.8.0_xxx\bin>keytool.exe -import -trustcacerts -file
C:\Certificates\RootCA.crt -alias CYBERARK -keystore cacerts
```