User Guide – KeyScaler SAT API

INSTALL AND USER GUIDE

Security Level:	**External**
Author:	**Nirmal Misra**
Last Edit Date:	**19 August 2024**

Table of Contents

1 This Document

1.1 Document History

Version	Description	Date	Who
1.0	Initial Document Creation	28/06/2024	Nirmal Misra
2.0	Add KeyScaler Configuration to Appendices	08/07/2024	Nirmal Misra
3.0	Changes to App User interface, updated screenshots	12/08/2024	Nirmal Misra

Referenced Documents

#	Document Name	Comment
1	SAT API One Pager	
2	Credential Manager User Guide	
3	Generate HSM Key API Document	SAC API - HSM Generate Key Pair.pdf

Glossary of Terms

Term	Description
API	Application Programming Interface
CP	KeyScaler Control Panel
CM	Credential Manager
DAE	Device Authority Engine
EC Keys	Elliptic Curve
KS	KeyScaler
HSM	Hardware Security Module
IoT	Internet Of Things
MQTT	Message Queuing Telemetry Transport
PAM	Privileged Access Management
PId	Password Identity
PSecret	Password Secret
RSA Keys	Rivest–Shamir–Adleman Key
SAC	Service Access Controller
SAT	Secure Asset Transfer
URL	Universal Resource Locator
UUID	Universal Unique Identifier aka GUID – Globally Unique Identifier

Pre-Requisites

1. Access to KeyScaler System version 7.0.4 or above configured with a SAC MQTT as the front end.
2. An RSA or EC Key available in the KeyStore for Signing the assets. Please Refer to Appendix A for how this can be done.
3. Freemarker Service Connector configured. See Appendix B for configuration guide
4. KeyScaler Configured with SAT Policy and Signing Key. For configuration guide , see Appendix C
5. Cyber Ark Account and access to CyberArk Vault to store credentials
6. Device Authority's Credential Manager agent software configured (MQTT Mode enabled) running on the IoT device. For Device Configuration and Registration please Refer to Credential Manager User Guide.

7. Sample Script available . Some sample scripts are available from Device Authority. How to configure scripts are described in Appendix D.

2 Introduction

This user guide document is intended for System Administrators who would like to install, configure and use KeyScaler's Synchronous Asset Transfer (SAT) feature to securely deliver an asset (for example, a Unix- or Windows- script) to a remote IoT device.

Sample applications to use KeyScaler's SAT API are provided, and their installation to a System Administrator's Windows computer is explained in this document. Sample Unix scripts are also provided. For other platforms, please contact cyberarkcustomer@deviceauthority.com.

2.1 SAT Feature Overview

Device Authority's SAT feature for KeyScaler integrated with CyberArk's Privileged Access Management (PAM) product enables CyberArk customers to easily and securely conduct PAM operations on IoT devices. The primary use case for this is to grant field service engineers temporary access, and then revoke such access, through password rotation. This is achieved by securely executing scripts on these IoT devices to set, and verify, user passwords.
This is enabled by a new KeyScaler API (the SAT API) that enables users to securely send scripts for immediate execution on specified remote IoT devices. The device returns the execution output log to KeyScaler.

The transfer is visible in KeyScaler's Control Panel (CP) as a device job and, when successful, a corresponding log showing the script's execution output.

This SAT feature enables CyberArk users to:

1. Store script templates (with placeholders for execution-time values) in KeyScaler via KeyScaler's CP.
2. Designate target device(s), nominate the script template, and provide payload values (to be merged with the script template to replace placeholders) via the SAT API.
3. Deliver the composed script to the target device(s) via KeyScaler's MQTT channel.
4. View device execution output log (returned to KeyScaler via the MQTT channel) in KeyScaler's CP.

Sample Unix scripts available are:

- Change User Password.
- Verify User Password.

An overview of the SAT feature flow is shown in figure 1 below:

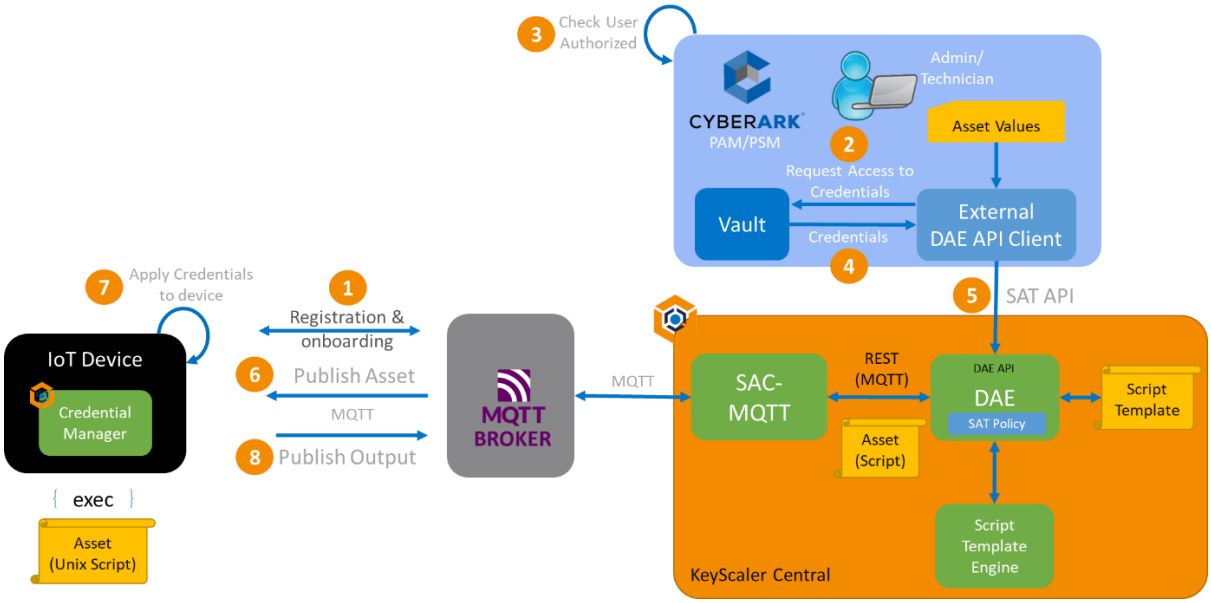


Figure 1- System overview diagram for KeyScaler's SAT feature

3 Installation

To install the Windows base KeyScaler SAT API application (this is the "External DAE API Client" shown in figure 1), download the file ***CyberArk_KsAPI_Setup-1.0.0.0.0.exe*** from the CyberArk Marketplace to your Windows machine:

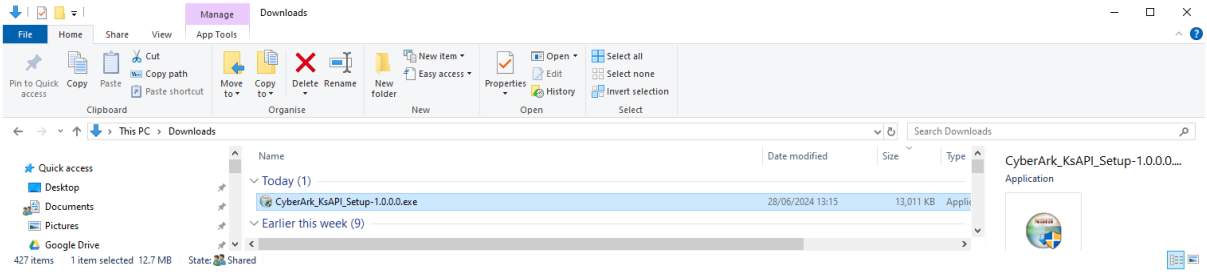


Figure 2 – KeyScaler SAT API exe file

Next, Double Click on the file to start the installation process. You may need to allow your windows firewall to add the file as an exception.

Click on 'YES' button to continue with the installation and follow the steps below:

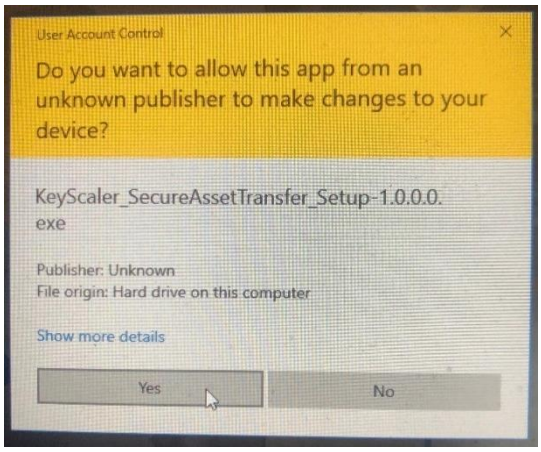


Figure 3 – Allow Windows to install the application

Figure 4 – Click Next to start the installation process

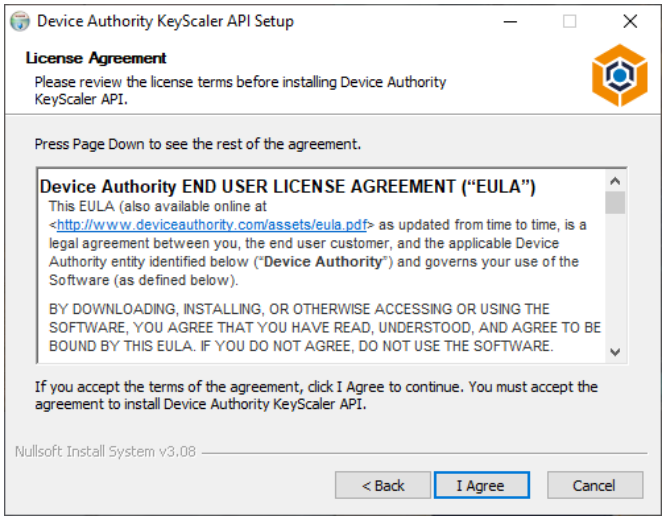


Figure 5 – Click 'I Agree' to the End User License Agreement

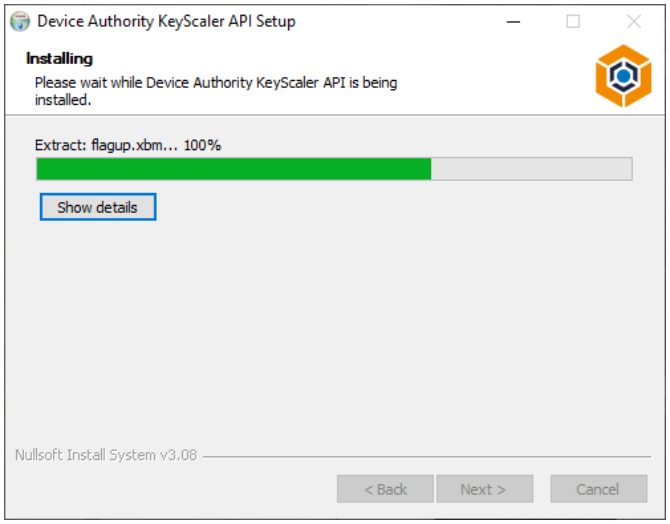


Figure 6 – Wait for the installation process to complete – it should only take a few seconds

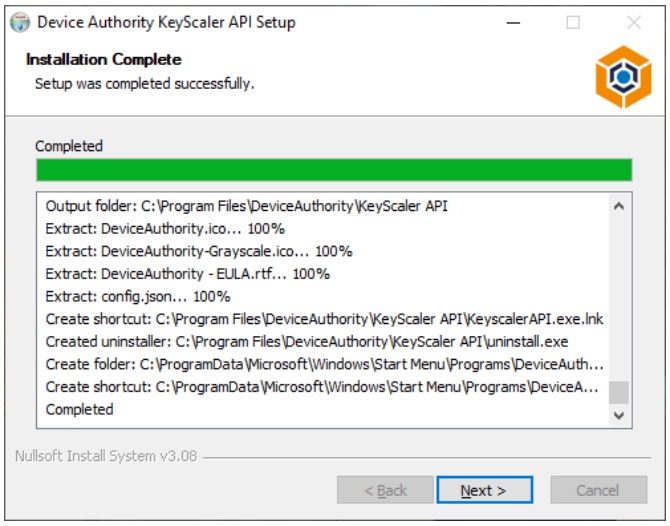


Figure 7 – Click on 'Show Details' to view the files being installed

Figure 8 – Click on 'Finish' once Setup is completed

3.1 Configuration

Next, navigate to the following folder:

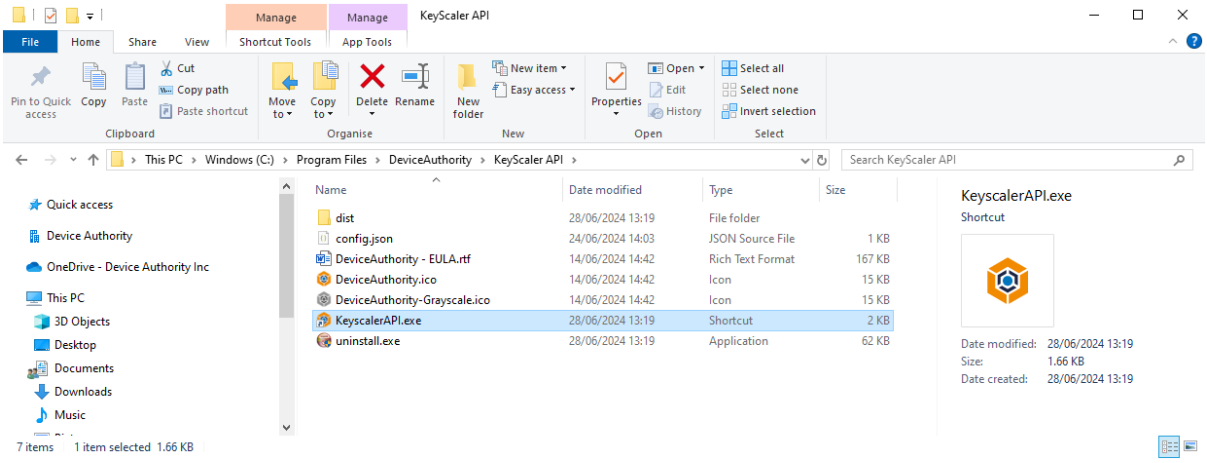


Figure 9 – File Manager showing the installation files for the SAT API application.

Next, edit the ***config.json*** file to modify the log location and file to match your environment:

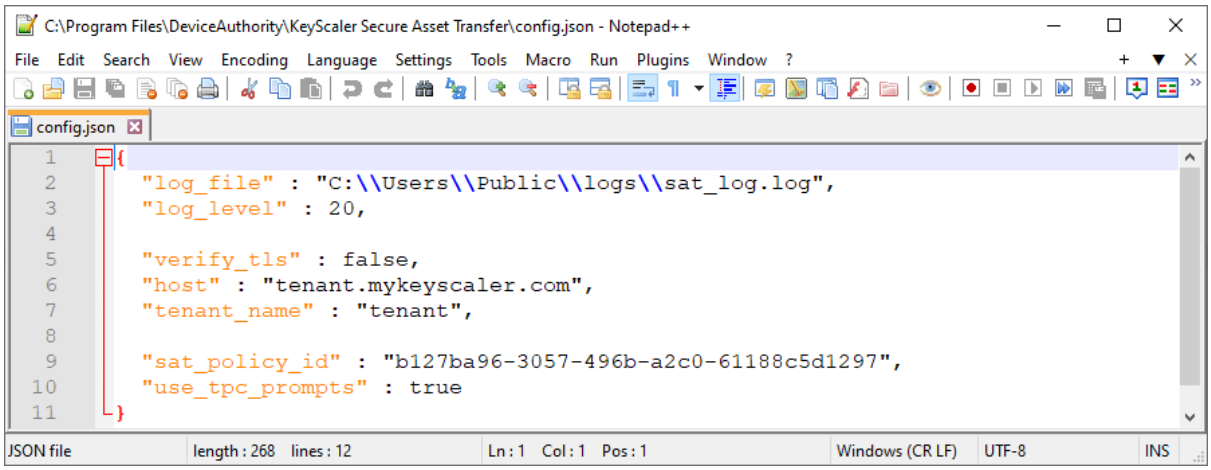


Figure 10 – Edit config.json file to match your environment

For the fields ***host***, ***tenant*** and ***sat_policy_id***, these will be available once your KeyScaler system is available.

The host will be the tenant domain, for example:

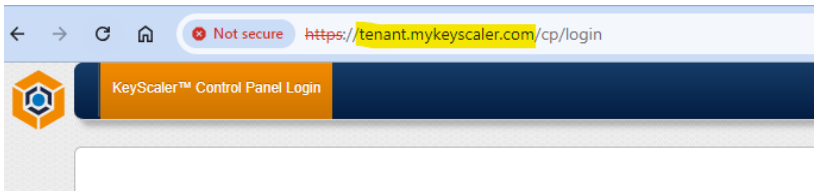


Figure 11 – Host value for config.json

The ***sat_policy_id*** is created after a SAT policy is created in the KeyScaler tenant, for example:

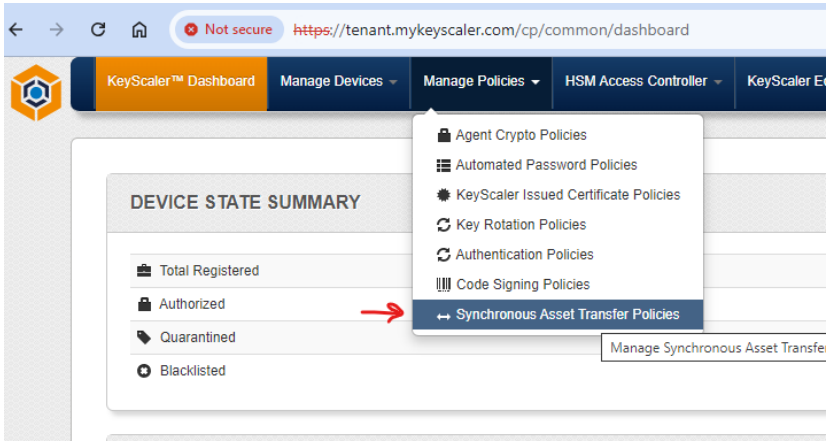


Figure 12 – KeyScaler Control Panel – View sat_policy_id

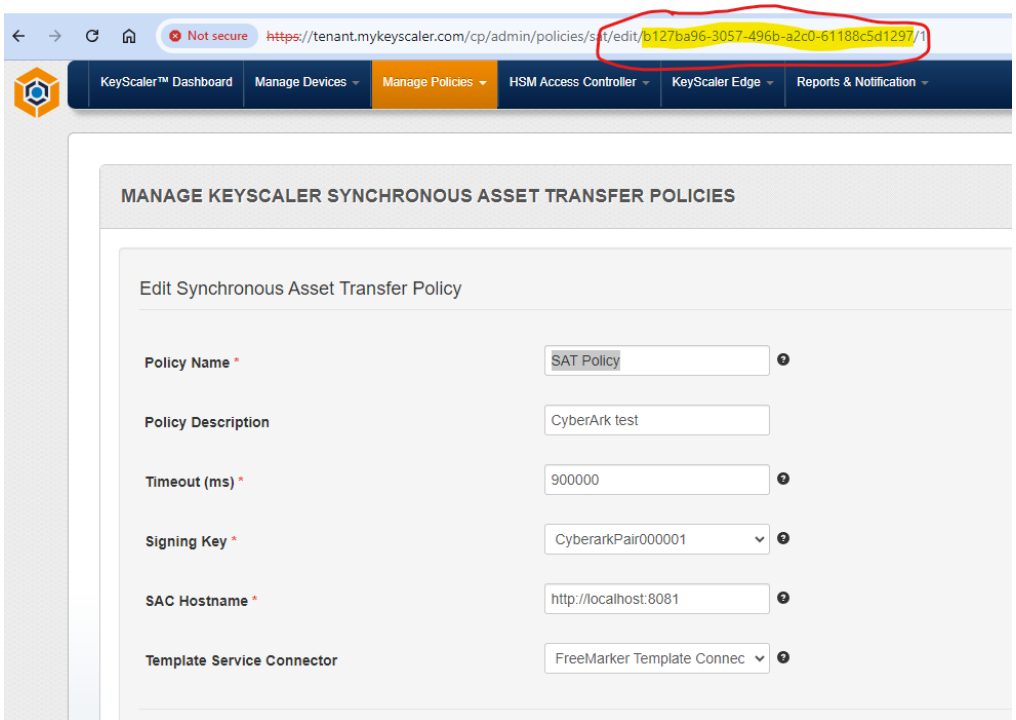


Figure 13 – KeyScaler Control Panel View sat_policy_id

Sample, **config.json**:

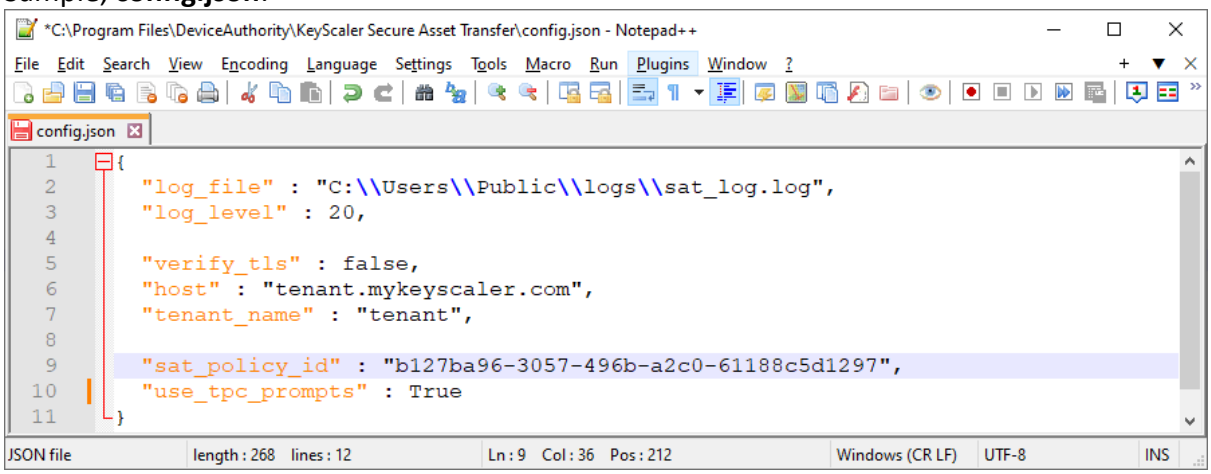


Figure 14 – Sample config.json file

Next, Save and exit the config.json file.

3.2 Run the Application

To run the application on the System Administrator's computer that will operate the KeyScaler SAT feature, double click on the ***KeyScaler_SecureAssetTransfer.exe*** file:

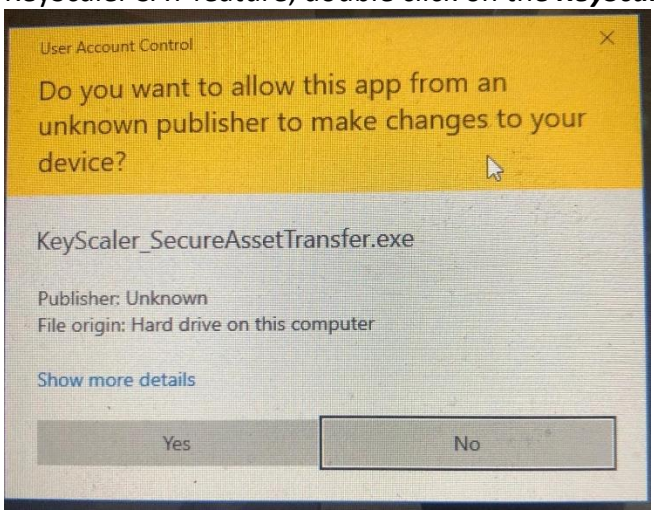


Figure 15 – Click on 'Yes' button to continue

A terminal window pop up will appear, prompting the user for some inputs:

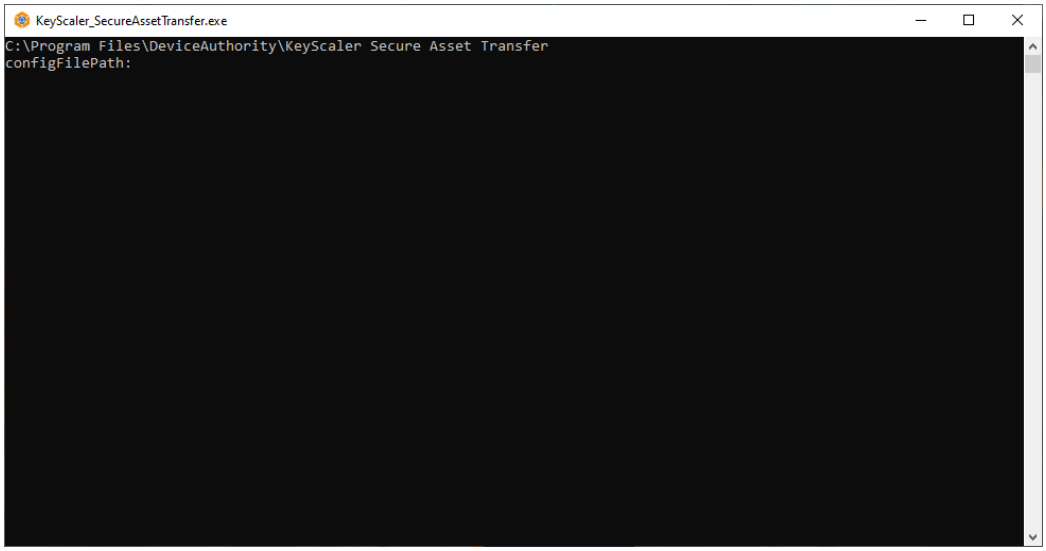


Figure 16 – Launch the SAT API application

Enter the config file config.json or just hit return to use the default config file that was edited above:

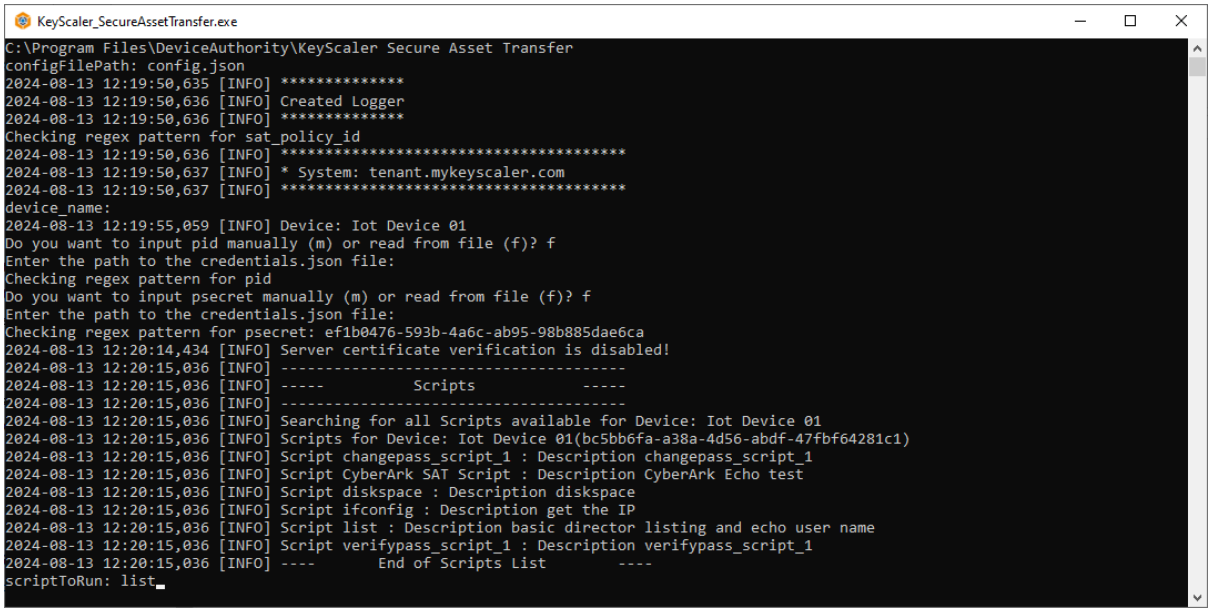


Figure 17 – Enter the values prompted by the application

Once the **config file**, **device-name**, have been entered, the App prompts the user to enter the **pid** and **psecret,** either manually (m) or from file (f). If (f) is selected, user can just hit enter to use the default file ***credentials.json*** to enter the Pid/Psecret from file.

The SAT API Application then returns a list of possible scripts that can be executed on the target device.

The ***pid*** and ***psecret*** value may also be obtained from the KeyScalar system. Once logged into the Control Panel, navigate to **Account Settings**:

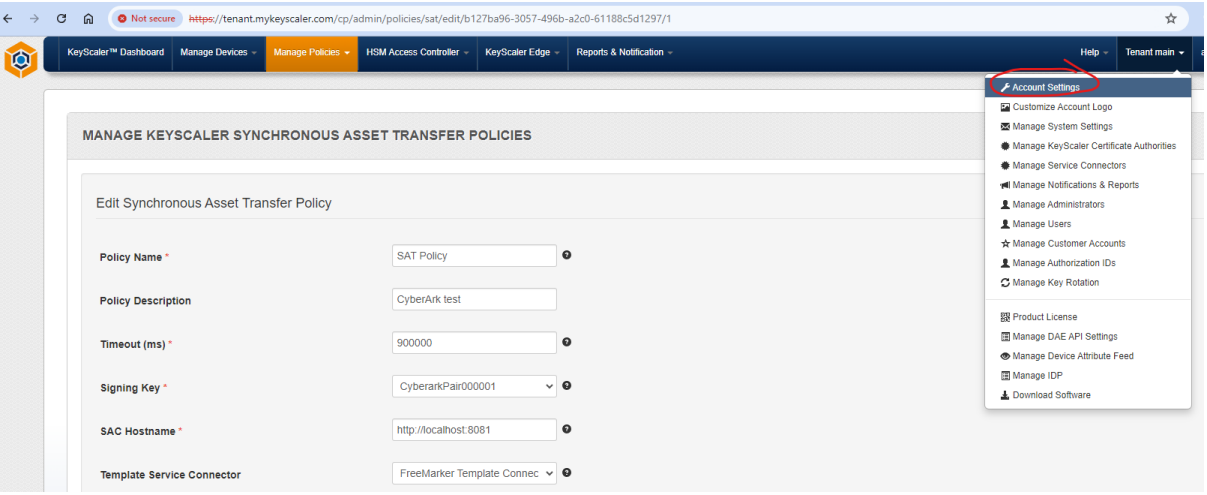


Figure 18 – KeyScaler Control Panel – Navigate to Account Settings

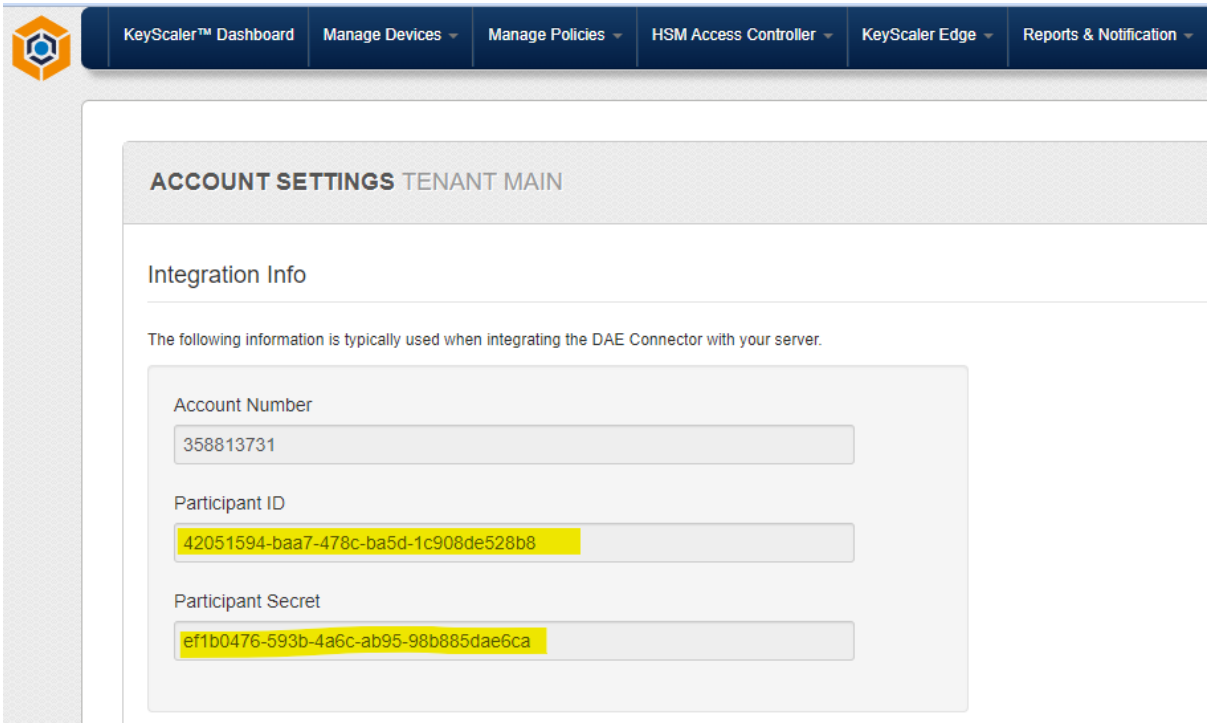


Figure 19 – KeyScaler Control Panel – Pid and psecret settings

Select a script to run, for example 'list', and enter the **target_username**, and **target_password** (note these are not required for the list command) a sample output is shown below:

```
KeyScaler_SecureAssetTransfer.exe
C:\Program Files\DeviceAuthority\KeyScaler Secure Asset Transfer
configFilePath: config.json
2024-08-13 12:19:50,635 [INFO] ***************
2024-08-13 12:19:50,636 [INFO] Created Logger
2024-08-13 12:19:50,636 [INFO] ***************
Checking regex pattern for sat_policy_id
2024-08-13 12:19:50,636 [INFO] ****************************************
2024-08-13 12:19:50,637 [INFO] * System: tenant.mykeyscaler.com
2024-08-13 12:19:50,637 [INFO] ****************************************
device_name:
2024-08-13 12:19:55,059 [INFO] Device: Iot Device 01
Do you want to input pid manually (m) or read from file (f)? f
Enter the path to the credentials.json file:
Checking regex pattern for pid
Do you want to input psecret manually (m) or read from file (f)? f
Enter the path to the credentials.json file:
Checking regex pattern for psecret: ef1b0476-593b-4a6c-ab95-98b885dae6ca
2024-08-13 12:20:14,434 [INFO] Server certificate verification is disabled!
2024-08-13 12:20:15,036 [INFO] --------------------------------------
2024-08-13 12:20:15,036 [INFO] -----          Scripts            -----
2024-08-13 12:20:15,036 [INFO] --------------------------------------
2024-08-13 12:20:15,036 [INFO] Searching for all Scripts available for Device: Iot Device 01
2024-08-13 12:20:15,036 [INFO] Scripts for Device: Iot Device 01(bc5bb6fa-a38a-4d56-abdf-47fbf64281c1)
2024-08-13 12:20:15,036 [INFO] Script changepass_script_1 : Description changepass_script_1
2024-08-13 12:20:15,036 [INFO] Script CyberArk SAT Script : Description CyberArk Echo test
2024-08-13 12:20:15,036 [INFO] Script diskspace : Description diskspace
2024-08-13 12:20:15,036 [INFO] Script ifconfig : Description get the IP
2024-08-13 12:20:15,036 [INFO] Script list : Description basic director listing and echo user name
2024-08-13 12:20:15,036 [INFO] Script verifypass_script_1 : Description verifypass_script_1
2024-08-13 12:20:15,036 [INFO] ----       End of Scripts List        ----
scriptToRun: list
target_username: sat_user
Enter Password: net1234
Enter value for 'path': .
Script: list - ID 39f3c31c-0e31-48e6-aa4c-c21c91e757bd - True
2024-08-13 12:32:55,712 [INFO] Supplied parameters from user: {'user': 'sat_user', 'path': '.', 'password': '*********'}
--------------------------------------
-----     Running SAT Script      -----
--------------------------------------
2024-08-13 12:32:55,749 [INFO] Server certificate verification is disabled!
```

Figure 20 – SAT API Application Execution

3.3 Script Output

The SAT API application returns the script output (as executed on the target device) to the application's console and to the log file (if configured – see section 3.5). An example of the console output is shown below, continuing with the previous section's use of the "list" script:

```
KeyScaler_SecureAssetTransfer.exe
Enter value for 'path': .
Script: list - ID 39f3c31c-0e31-48e6-aa4c-c21c91e757bd - True
2024-08-13 12:32:55,712 [INFO] Supplied parameters from user: {'user': 'sat_user', 'path': '.', 'password': '*********'}
---------------------------------------
-----     Running SAT Script      -----
---------------------------------------
2024-08-13 12:32:55,749 [INFO] Server certificate verification is disabled!
2024-08-13 12:33:49,625 [INFO] SAT Response Output: {'req_id': '7fbdbeb2-6f5e-4e99-9379-cbc75dbb2406', 'response_ts': 1723548829687, 'http_code': 200, '
status_code': 0, 'response_data': {'data': 'sat_user\ntotal 124\ndrwxr-xr-x  23 root root  4096 Apr 25 19:16 .\ndrwxr-xr-x  23 root root  4096 Apr 25 19
:16 ..\n-rwxrwxrwx   1 root root 12976 Apr 25 19:16 auth_test\n-rw-rw-rw-   1 root root  2404 Apr 25 19:16 auth_test.c\ndrwxr-xr-x   2 root root  4096 N
ov 17  2022 bin\ndrwxr-xr-x   4 root root  4096 Nov 17  2022 boot\ndrwxr-xr-x  15 root root  4060 Apr 24 13:53 dev\n-rw-rw-rw-   1 root root  3714 Sep 1
7  2021 deviceCert.cert\n-rw-rw-rw-   1 root root  1675 Sep 17  2021 deviceKey.pem\ndrwxr-xr-x 102 root root  4096 Jul 30 17:48 etc\n-rw-rw-rw-   1 root
 root     0 Apr 25 18:59 gcc_output.txt\ndrwxr-xr-x   6 root root  4096 Apr 24 13:53 home\nlrwxrwxrwx   1 root root    32 Aug 11  2022 initrd.img -> boo
t/initrd.img-5.4.0-1089-azure\nlrwxrwxrwx   1 root root    32 Aug 11  2022 initrd.img.old -> boot/initrd.img-5.4.0-1086-azure\ndrwxr-xr-x  22 root root
 4096 Nov 17  2022 lib\ndrwxr-xr-x   2 root root  4096 Nov 17  2022 lib64\ndrwx------   2 root root 16384 Jan 21  2020 lost+found\ndrwxr-xr-x   2 root r
oot  4096 Jan 21  2020 media\ndrwxr-xr-x   3 root root  4096 Apr 24 13:53 mnt\ndrwxr-xr-x   2 root root  4096 Jan 21  2020 opt\ndr-xr-xr-x 167 root root
     0 Apr 24 13:52 proc\ndrwx------   7 root root  4096 Aug 12 20:06 root\ndrwxr-xr-x  24 root root   860 Aug 12 13:40 run\ndrwxr-xr-x   2 root root 12
288 Nov 17  2022 sbin\ndrwxr-xr-x   2 root root  4096 Jan 29  2020 snap\ndrwxr-xr-x   2 root root  4096 Jan 21  2020 srv\ndr-xr-xr-x  12 root root     0
 Apr 24 13:52 sys\ndrwxrwxrwxrwt   9 root root  4096 Aug 13 06:27 tmp\ndrwxr-xr-x  10 root root  4096 Jan 21  2020 usr\ndrwxr-xr-x  14 root root  4096 Nov
 6  2022 var\nlrwxrwxrwx   1 root root    29 Aug 11  2022 vmlinuz -> boot/vmlinuz-5.4.0-1089-azure\nlrwxrwxrwx   1 root root    29 Aug 11  2022 vmlinuz.
old -> boot/vmlinuz-5.4.0-1086-azure\n', 'success': True}}
Result:
sat_user
total 124
drwxr-xr-x  23 root root  4096 Apr 25 19:16 .
drwxr-xr-x  23 root root  4096 Apr 25 19:16 ..
-rwxrwxrwx   1 root root 12976 Apr 25 19:16 auth_test
-rw-rw-rw-   1 root root  2404 Apr 25 19:16 auth_test.c
drwxr-xr-x   2 root root  4096 Nov 17  2022 bin
drwxr-xr-x   4 root root  4096 Nov 17  2022 boot
drwxr-xr-x  15 root root  4060 Apr 24 13:53 dev
-rw-rw-rw-   1 root root  3714 Sep 17  2021 deviceCert.cert
-rw-rw-rw-   1 root root  1675 Sep 17  2021 deviceKey.pem
drwxr-xr-x 102 root root  4096 Jul 30 17:48 etc
-rw-rw-rw-   1 root root     0 Apr 25 18:59 gcc_output.txt
drwxr-xr-x   6 root root  4096 Apr 24 13:53 home
lrwxrwxrwx   1 root root    32 Aug 11  2022 initrd.img -> boot/initrd.img-5.4.0-1089-azure
lrwxrwxrwx   1 root root    32 Aug 11  2022 initrd.img.old -> boot/initrd.img-5.4.0-1086-azure
drwxr-xr-x  22 root root  4096 Nov 17  2022 lib
drwxr-xr-x   2 root root  4096 Nov 17  2022 lib64
drwx------   2 root root 16384 Jan 21  2020 lost+found
drwxr-xr-x   2 root root  4096 Jan 21  2020 media
drwxr-xr-x   3 root root  4096 Apr 24 13:53 mnt
drwxr-xr-x   2 root root  4096 Jan 21  2020 opt
dr-xr-xr-x 167 root root     0 Apr 24 13:52 proc
drwx------   7 root root  4096 Aug 12 20:06 root
drwxr-xr-x  24 root root   860 Aug 12 13:40 run
drwxr-xr-x   2 root root 12288 Nov 17  2022 sbin
drwxr-xr-x   2 root root  4096 Jan 29  2020 snap
drwxr-xr-x   2 root root  4096 Jan 21  2020 srv
dr-xr-xr-x  12 root root     0 Apr 24 13:52 sys
drwxrwxrwxrwt   9 root root  4096 Aug 13 06:27 tmp
drwxr-xr-x  10 root root  4096 Jan 21  2020 usr
drwxr-xr-x  14 root root  4096 Nov  6  2022 var
lrwxrwxrwx   1 root root    29 Aug 11  2022 vmlinuz -> boot/vmlinuz-5.4.0-1089-azure
lrwxrwxrwx   1 root root    29 Aug 11  2022 vmlinuz.old -> boot/vmlinuz-5.4.0-1086-azure

---------------------------------------
-----         Completed           -----
---------------------------------------
Press a key to close...
```

Figure 21 – SAT API application returns the script output as executed on the device

3.4 Change Password Script

Other scripts available are for changing a password and verifying the password change. Sample outputs of these two scripts are shown below:

```
KeyScaler_SecureAssetTransfer.exe
C:\Program Files\DeviceAuthority\KeyScaler Secure Asset Transfer
configFilePath: config.json
2024-07-01 20:42:20,434 [INFO] **************
2024-07-01 20:42:20,434 [INFO] Created Logger
2024-07-01 20:42:20,434 [INFO] **************
2024-07-01 20:42:20,434 [INFO] ****************************************
2024-07-01 20:42:20,434 [INFO] * System: tenant.mykeyscaler.com
2024-07-01 20:42:20,434 [INFO] ****************************************
device_name:
KeyScaler Participant ID:
KeyScaler Participant Secret:
2024-07-01 20:42:26,355 [INFO] Server certificate verification is disabled!
2024-07-01 20:42:26,956 [INFO] ---------------------------------------
2024-07-01 20:42:26,956 [INFO] -----          Scripts            -----
2024-07-01 20:42:26,957 [INFO] ---------------------------------------
2024-07-01 20:42:26,958 [INFO] Searching for all Scripts available for Device: Iot Device 01
2024-07-01 20:42:26,958 [INFO] Scripts for Device: Iot Device 01(bc5bb6fa-a38a-4d56-abdf-47fbf64281c1)
2024-07-01 20:42:26,958 [INFO] Script changepass_script_1 : Script changepass_script_1
2024-07-01 20:42:26,958 [INFO] Script CyberArk SAT Script : Script CyberArk Echo test
2024-07-01 20:42:26,958 [INFO] Script diskspace : Script diskspace
2024-07-01 20:42:26,958 [INFO] Script ifconfig : Script get the IP
2024-07-01 20:42:26,958 [INFO] Script list : Script basic director listing and echo user name
2024-07-01 20:42:26,958 [INFO] Script verifypass_script_1 : Script verifypass_script_1
2024-07-01 20:42:26,958 [INFO] ----       End of Scripts List        ----
scriptToRun: changepass_script_1
target_username: sat_user
target_password: net1234$
Enter value for 'someOtherPlaceHolder1':
Enter value for 'someOtherPlaceHolder2':
Script: changepass_script_1 - ID ba4529a3-4e65-4814-ba80-6df50552ebeb - True
2024-07-01 20:43:22,462 [INFO] Supplied parameters from user: {'password': '*********', 'user': 'sat_user', 'someOtherPlaceHolder1': '', 'someOtherPlaceHolder2': ''}
---------------------------------------
-----     Running SAT Script      -----
---------------------------------------
2024-07-01 20:43:22,472 [INFO] Server certificate verification is disabled!
2024-07-01 20:44:05,382 [INFO] SAT Response Output: {'req_id': 'a66c0835-f5c8-4624-8c8a-977ee9312977', 'response_ts': 1719863046459, 'http_code': 200, 'status_code':
0, 'response_data': {'data': '{"message": "verifying user sat_user"}\n{Password change successful}\n', 'success': True}}
Result:
{"message": "verifying user sat_user"}
{Password change successful}

---------------------------------------
-----         Completed           -----
---------------------------------------
Press a key to close...
```

Figure 22 – Output for a sample change password script

```
KeyScaler_SecureAssetTransfer.exe
configFilePath: config.json
2024-07-01 20:46:06,162 [INFO] **************
2024-07-01 20:46:06,162 [INFO] Created Logger
2024-07-01 20:46:06,162 [INFO] **************
2024-07-01 20:46:06,162 [INFO] ****************************************
2024-07-01 20:46:06,162 [INFO] * System: tenant.mykeyscaler.com
2024-07-01 20:46:06,162 [INFO] ****************************************
device_name:
KeyScaler Participant ID:
KeyScaler Participant Secret:
2024-07-01 20:46:10,120 [INFO] Server certificate verification is disabled!
2024-07-01 20:46:10,676 [INFO] ---------------------------------------
2024-07-01 20:46:10,676 [INFO] -----          Scripts            -----
2024-07-01 20:46:10,676 [INFO] ---------------------------------------
2024-07-01 20:46:10,676 [INFO] Searching for all Scripts available for Device: Iot Device 01
2024-07-01 20:46:10,676 [INFO] Scripts for Device: Iot Device 01(bc5bb6fa-a38a-4d56-abdf-47fbf64281c1)
2024-07-01 20:46:10,676 [INFO] Script changepass_script_1 : Script changepass_script_1
2024-07-01 20:46:10,676 [INFO] Script CyberArk SAT Script : Script CyberArk Echo test
2024-07-01 20:46:10,676 [INFO] Script diskspace : Script diskspace
2024-07-01 20:46:10,676 [INFO] Script ifconfig : Script get the IP
2024-07-01 20:46:10,676 [INFO] Script list : Script basic director listing and echo user name
2024-07-01 20:46:10,676 [INFO] Script verifypass_script_1 : Script verifypass_script_1
2024-07-01 20:46:10,676 [INFO] ----       End of Scripts List        ----
scriptToRun: verifypass_script_1
target_username: sat_user
target_password: net1234$
Enter value for 'verificationuser': testuser
Script: verifypass_script_1 - ID 3e55d699-63f7-40c1-b3d4-bea0c59e2dca - True
2024-07-01 20:48:15,489 [INFO] Supplied parameters from user: {'user': 'sat_user', 'password': '*********', 'verificationuser': 'testuser'}
---------------------------------------
-----     Running SAT Script      -----
---------------------------------------
2024-07-01 20:48:15,520 [INFO] Server certificate verification is disabled!
2024-07-01 20:48:44,082 [INFO] SAT Response Output: {'req_id': 'c0e8270f-f9bb-4eaa-b6dd-b3e303ba23af', 'response_ts': 1719863325165, 'http_c
ode': 200, 'status_code': 0, 'response_data': {'data': '{"message": "verifying user sat_user"}\n{\'success\': true}\n{\'Password is correct\
'}\n', 'success': True}}
Result:
{"message": "verifying user sat_user"}
{'success': true}
{'Password is correct'}

---------------------------------------
-----         Completed           -----
---------------------------------------
Press a key to close...
```

Figure 23 – Output for a sample verify password script

3.5 Log Files

Log files can be viewed in the file defined in the config.json configuration file:

"log_file" : "C:\\Users\\Public\\Logs\\log.log",

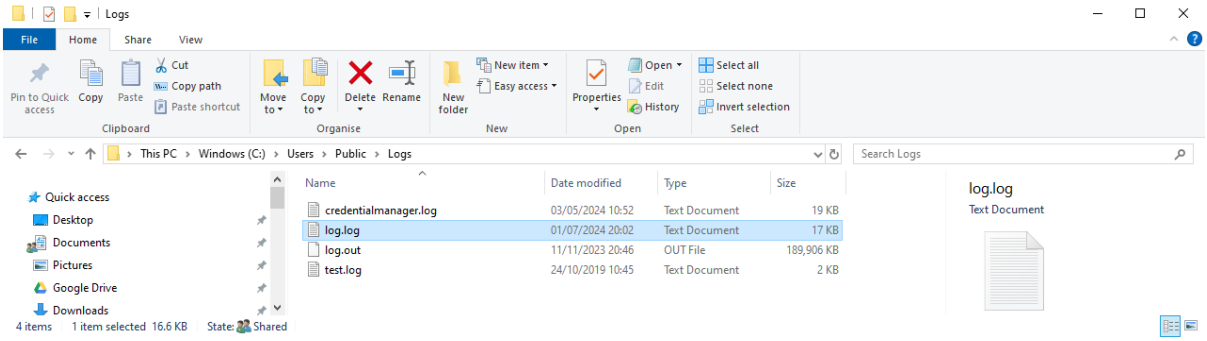


Figure 24 – Location of SAT API app log file

The default log level in **config.json** is 10, however this can be configured to the following levels for additional troubleshooting:

- CRITICAL = 50
- ERROR = 40
- WARNING = 30
- INFO = 20
- DEBUG = 10
- NOTSET = 0

Sample content of a log file is shown in the next figure.

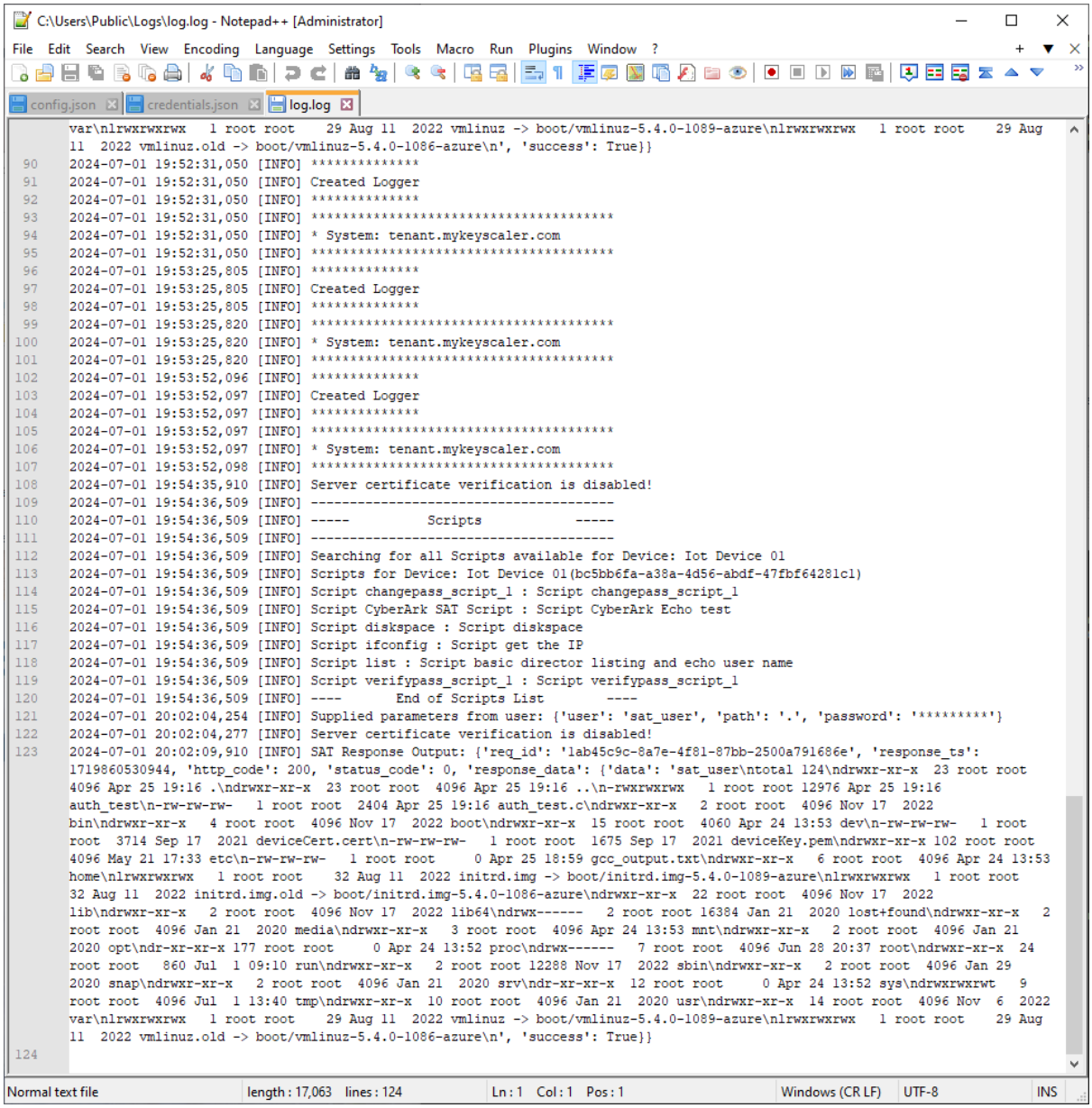


Figure 25 – Sample log file output for the "List" script

4 KeyScaler Control Panel View

On KeyScaler System, Login using login Credentials, and navigate to Managed Devices page:

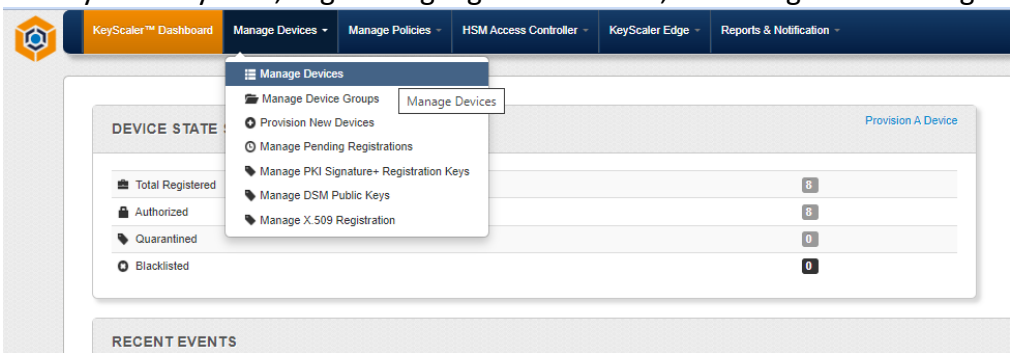


Figure 26- KeyScaler Control Panel – Managed Devices

For a specific device, the jobs for that device can be viewed by clicking on the ellipse, as shown:

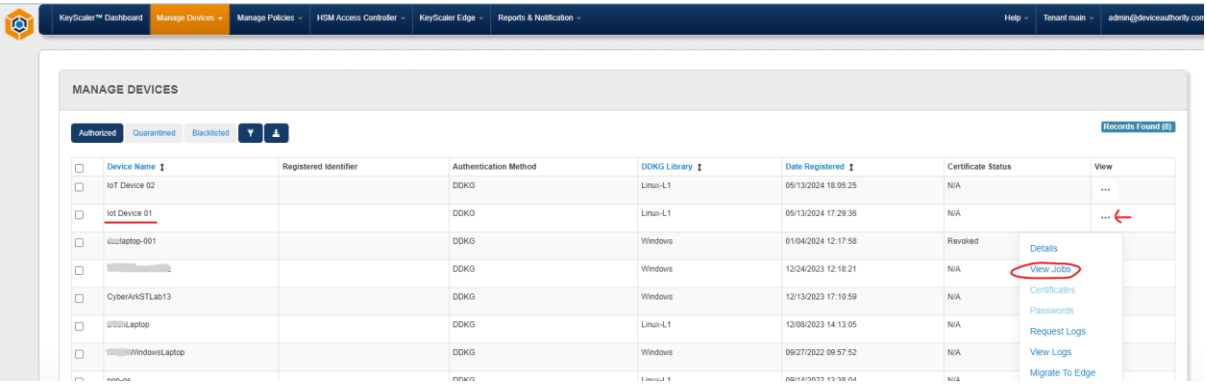


Figure 27- KeyScaler Control Panel – View Jobs for Device

Clicking on the 'view jobs' provides a list of status of those jobs for the elected device:

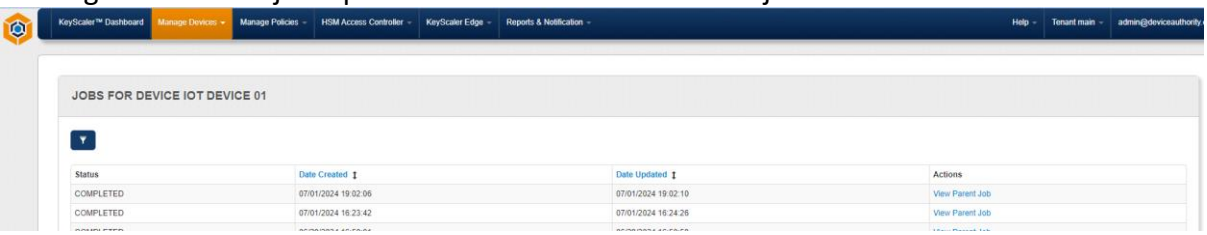


Figure 28- KeyScaler Control Panel – View Status of Jobs

Also, the Logs for the device can also be seen:

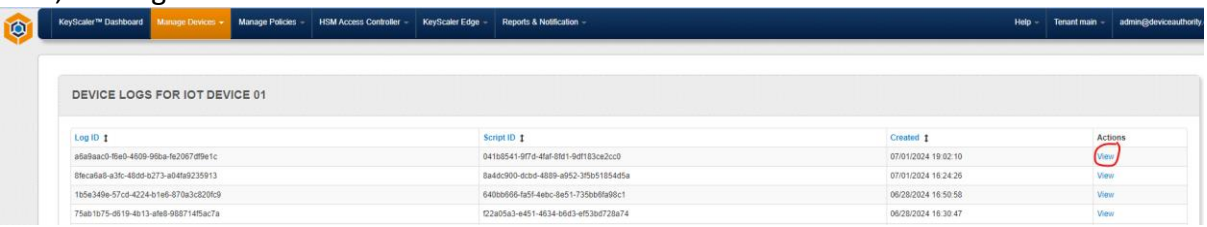


Figure 29 – KeyScaler Control Panel – View Logs

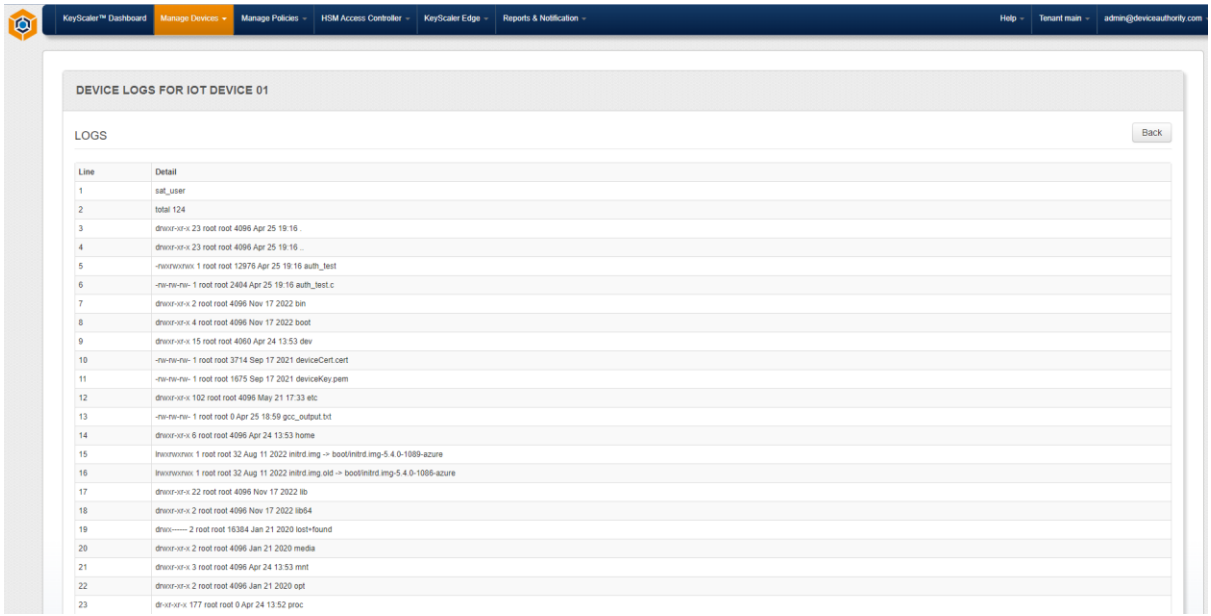


Figure 30 – KeyScaler Control Panel – View logs of a specific Log ID

5 Application Uninstallation Process

To uninstall the SAT API application, double click the uninstall file:

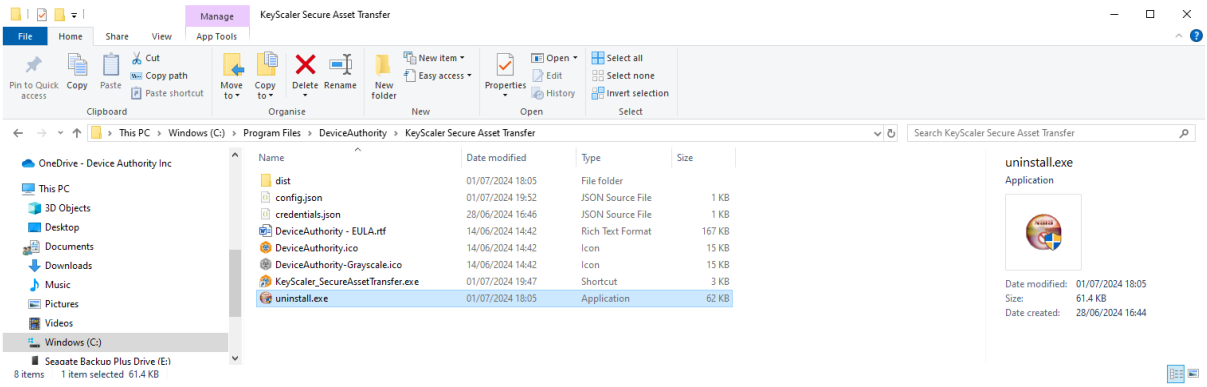


Figure 31 – Uninstall the SAT API application

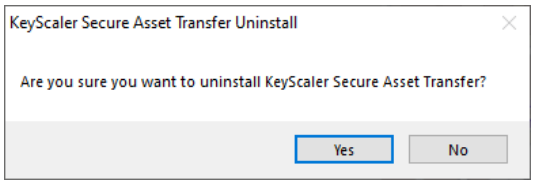


Figure 32 – Click 'YES' to continue the Uninstall of the application

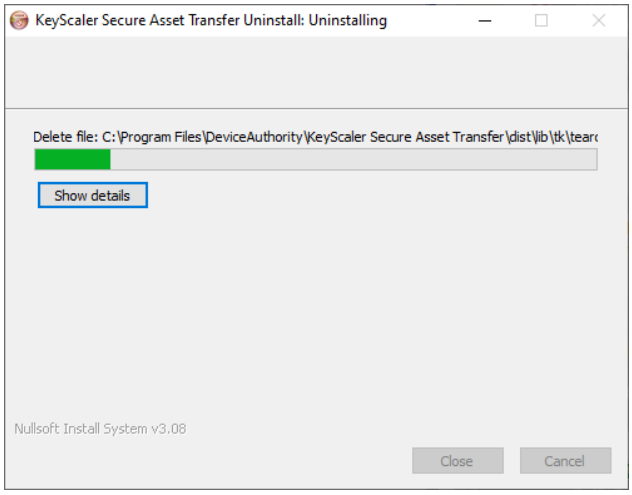


Figure 33 – The SAT API App files start to get deleted from the system

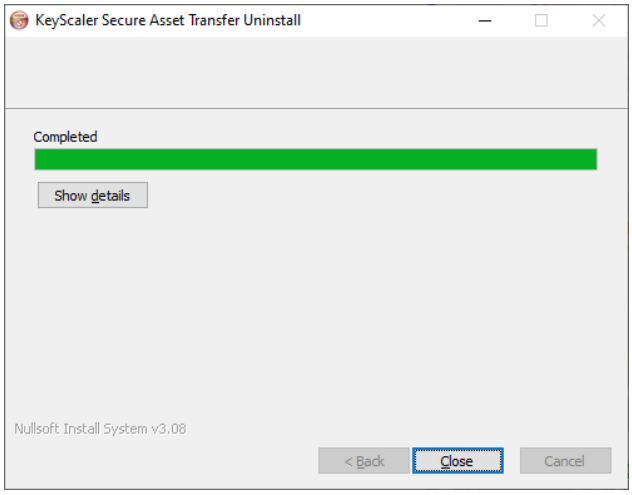


Figure 34 – Click on 'show details'

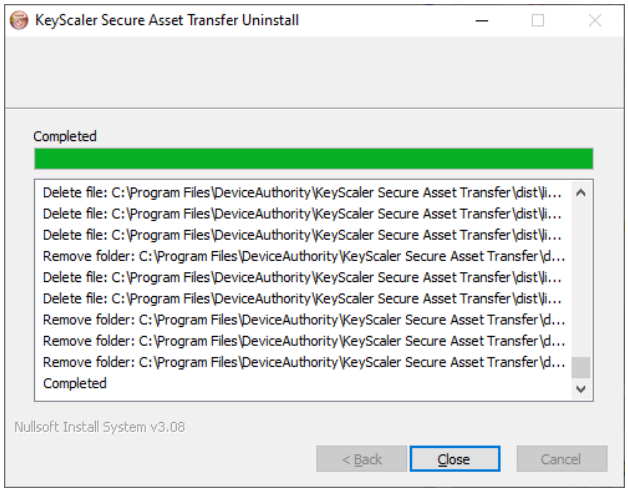


Figure 35 – Details of the files being uninstalled

This completes the uninstall process.

6 Appendix A – Create a HSM Signing Key

On KeyScaler CP, navigate to HSM Access Controller> Key Access Policies and create a new HSM Policy:

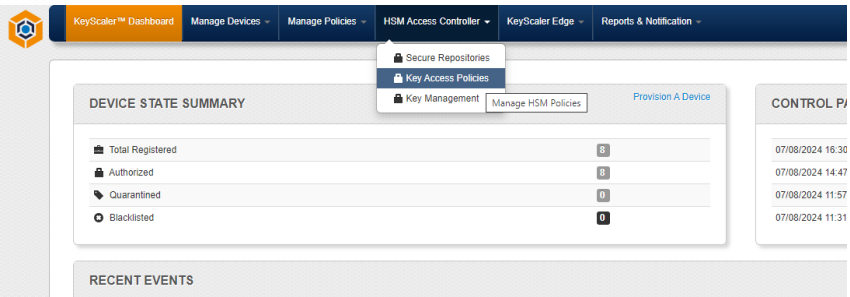


Figure 36- KeyScaler Control Panel – HSM Access Controller>Key Access Policies

In the sample below, a couple of policies are listed:

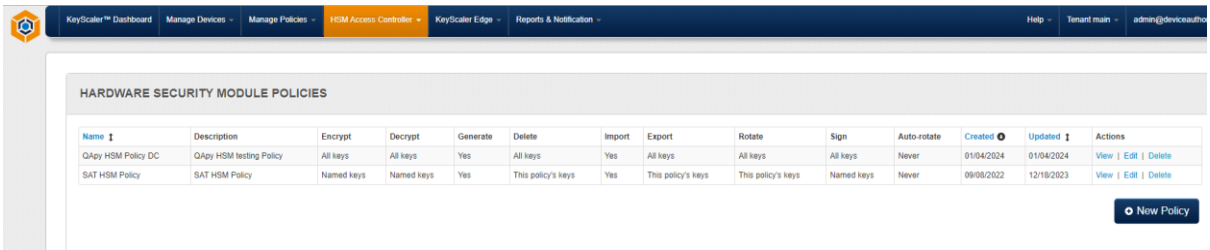


Figure 37- KeyScaler Control Panel – HSM Access Controller>Key Access Policies>List Policies

Click on **+New Policy** button to give a page as shown below:

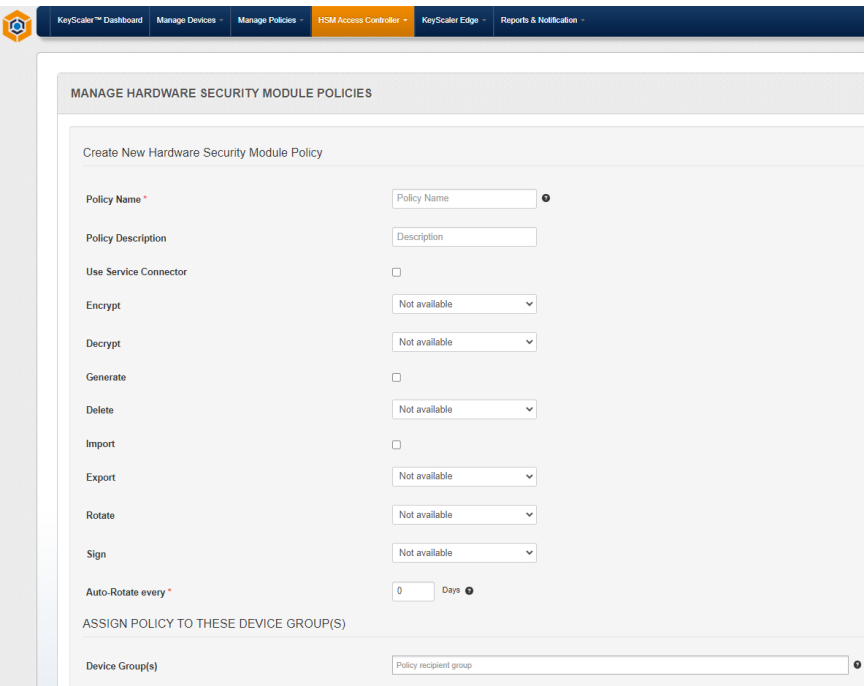


Figure 38- KeyScaler Control Panel – HSM Access Controller>Key Access Policies> Create New Policy

This policy allows users to control what access and control can be given to devices. In this case, we need to give 'Sign' permissions, so fill in the form as shown in the example below:

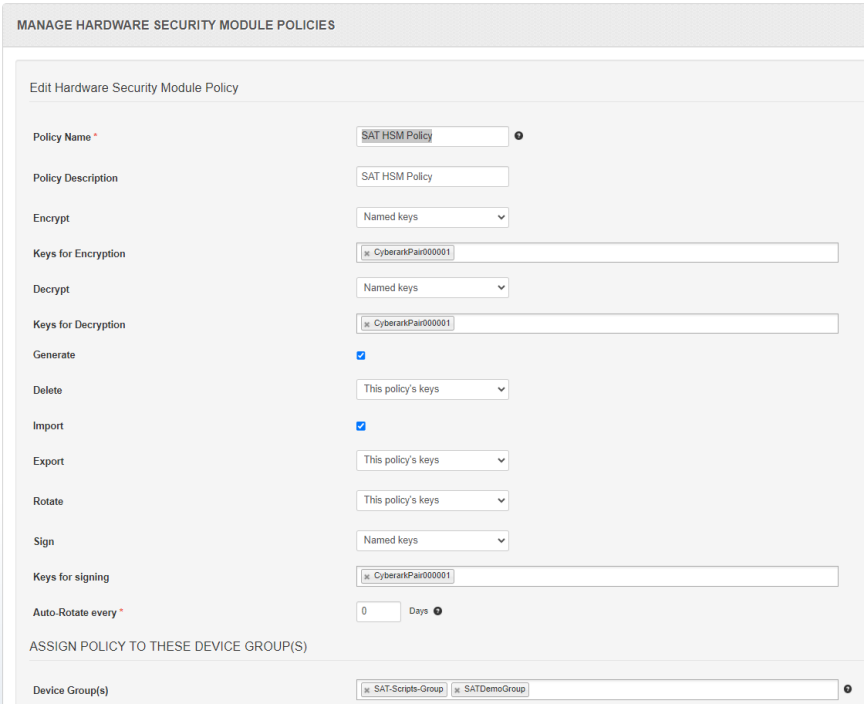


Figure 39- KeyScaler Control Panel – HSM Access Controller>Key Access Policies> Create new Policy

The 'Named Keys' as shown, can be added later by editing the policy. For now, select the option 'This Policies Keys' from the pull-down menu for 'Sign'.

Next, navigate to HSM Access Controller>Key Management.

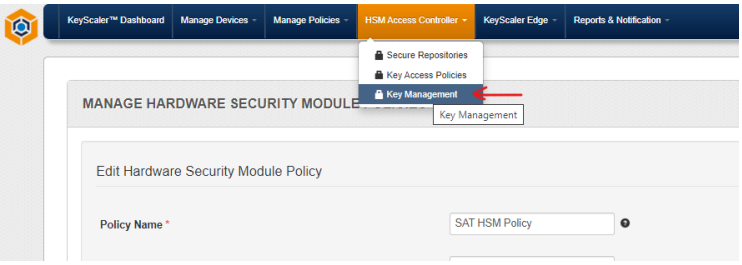


Figure 40- KeyScaler Control Panel – HSM Access Controller>Key Management

In the sample below a couple of keys are already Listed. For generating a new HSM key, this can only be done by REST API, and not via the KeyScaler Control Panel as the key need to be tied to a specific device(s) that have registered with KeyScaler. For generating a Signing Key follow the HSM Generate Key Pair API documentation. In summary, the following inputs are required:

- Key Alias
- Type of Key, either RSA or EC
- Size of Key
- HSM Policy UUID

The HSM Policy UUID can be taken from the URL as highlighted below:

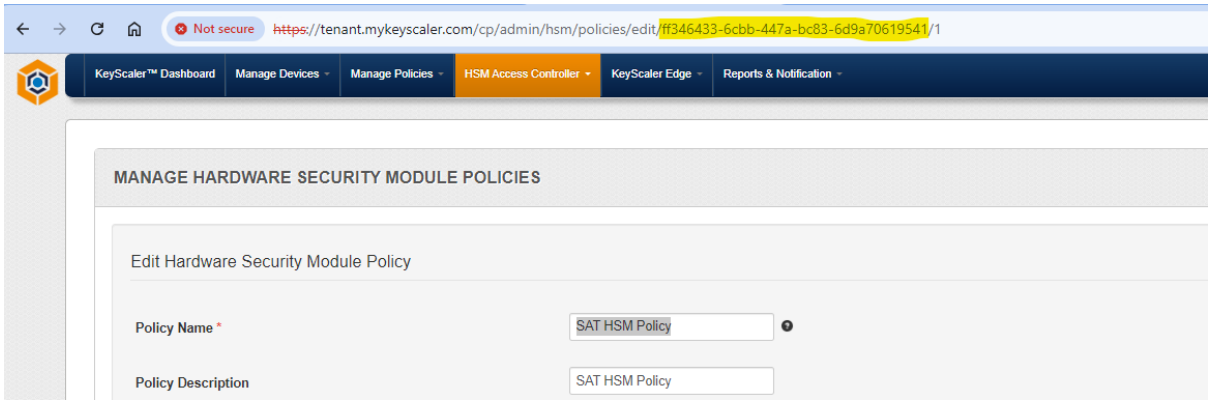


Figure 41Key – Scaler Control Panel – HSM Access Controller>HSM Policy UUID

Then refresh this page to view the new key.

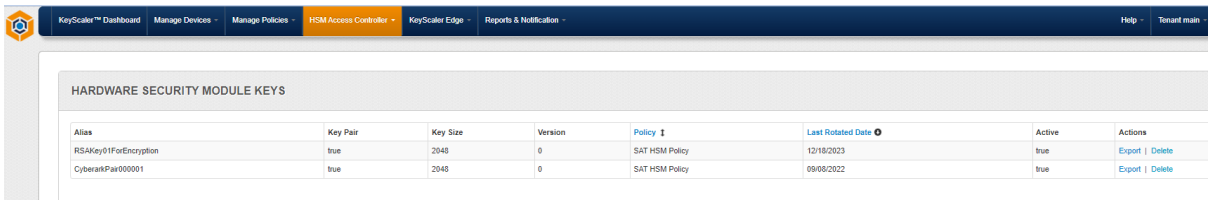


Figure 42- KeyScaler Control Panel – HSM Access Controller>List Keys

This completes the HSM Key Generation.

Appendix B – Create a new FreeMarker Service Connector

On KeyScaler CP, navigate to Tenant Main> Manage Service Connectors:

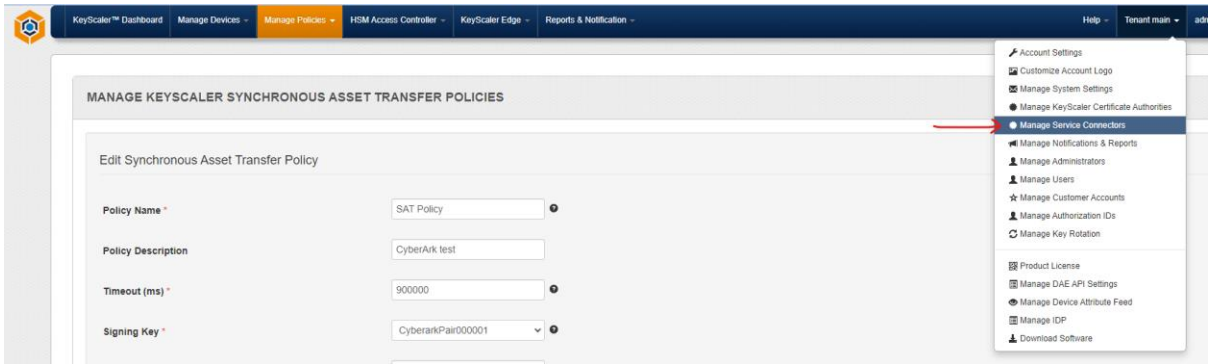


Figure 43- KeyScaler Control Panel – Manage Service Connectors

Next, click on **+Add New**, to create a new service connector:

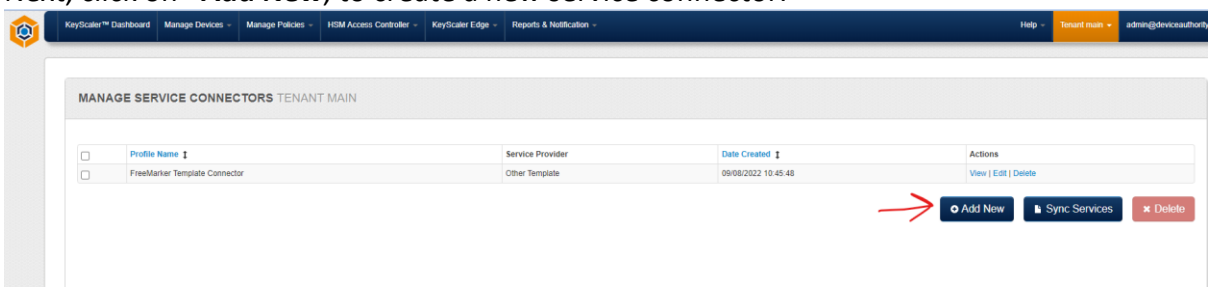


Figure 44- KeyScaler Control Panel – Manage Service Connectors>List Connectors

This will provide a form to fill-in with the service connector details.

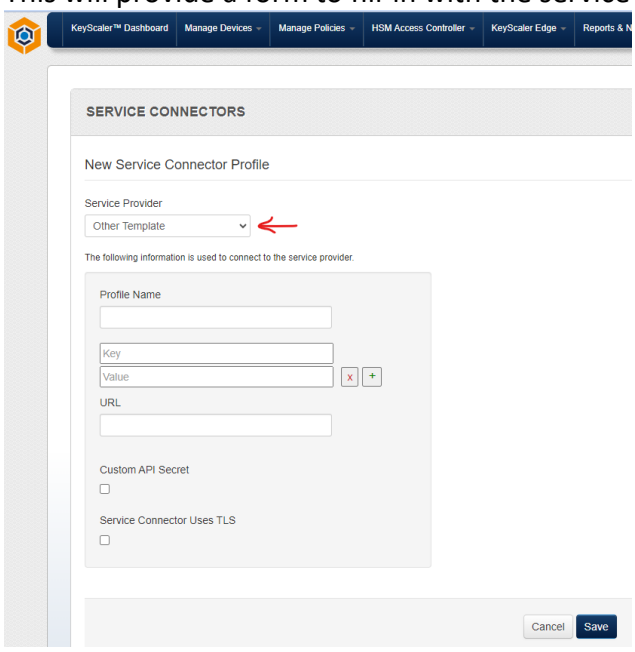


Figure 45- KeyScaler Control Panel – Manage Service Connectors>Create New Connector

A sample form is provided below:

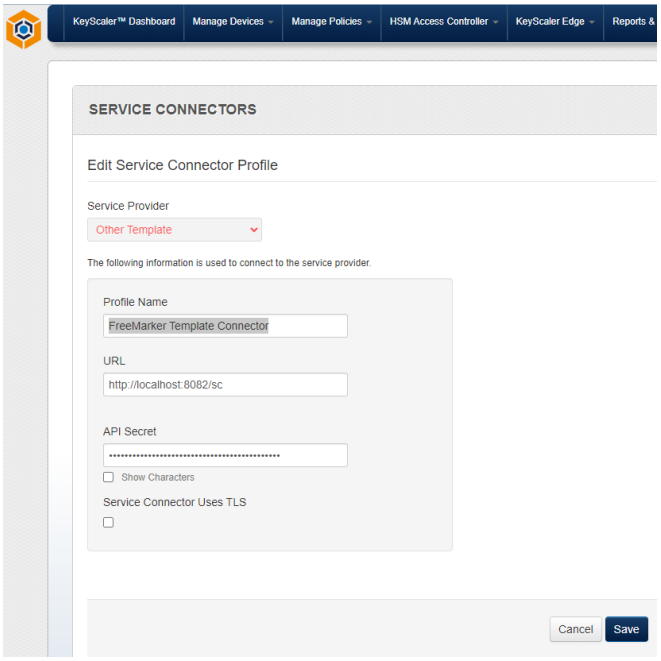


Figure 46- KeyScaler Control Panel – Manage Service Connectors>Create New Connector

- **Service Connector**: Select **Other Template**
- **Profile Name**: Free text for the service connector name
- **URL**: The URL of where the FreeMarker Service Connector[1] application is deployed. This could be the same VM as KeyScaler System, in which case use local host, as shown above.
- **API Secret**: This is used by the Service connector to authenticate to KeyScaler System. You can use your own API Secret, however, leaving it blank and saving, KeyScaler will generate one automatically for you.
- **Service Connector Uses TLS**: This should be enabled for Production Systems. When enabled, enter the service connector TLS certificate in the pop-up text field:

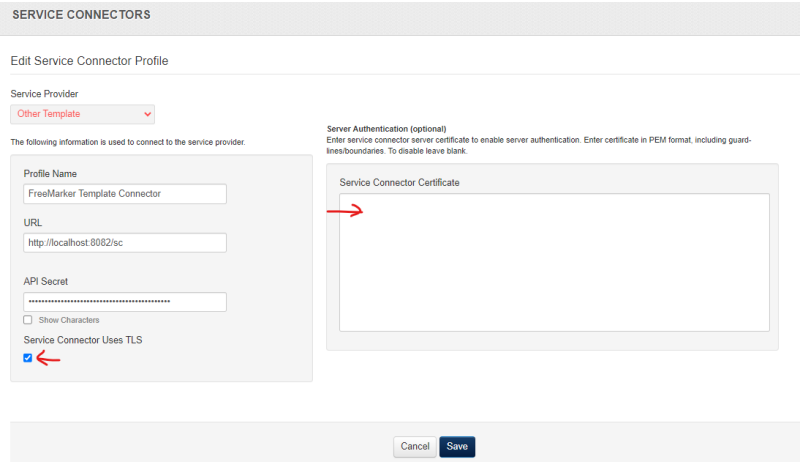


Figure 47 – KeyScaler Control Panel – Manage Service Connectors>Create New Connector> Enable TLS

Next, click on **Save** button to complete the service connector configuration.

[1] For any question and support on the connector, please contact cyberarkcustomer@deviceauthority.com

7 Appendix C – Create a new SAT Policy

To create a new Secure Asset Transfer Policy, login to KeyScaler CP and navigate to Manage Policies>Synchronous Asset Transfer Policies, as shown below:

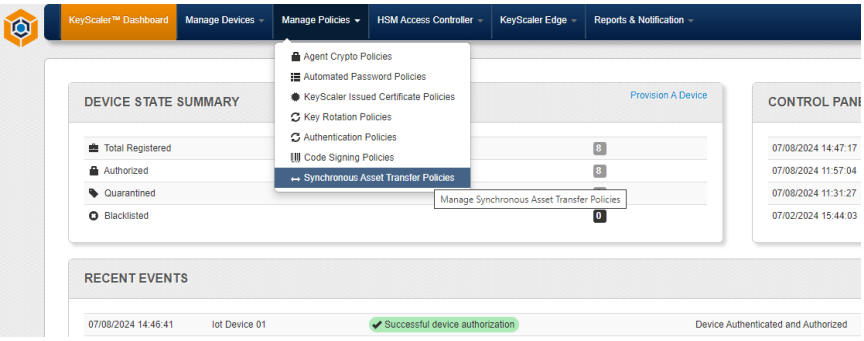


Figure 48 – KeyScaler CP> Manage Policies > SAT Policies

Next, click on **+New Policy** Button:

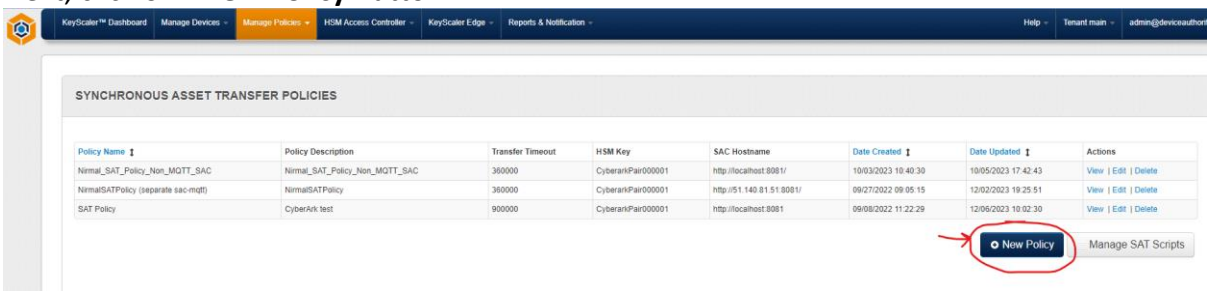


Figure 49 – KeyScaler CP> Manage Policies > SAT Policies> Create New Policy

Fill in the SAT Policy form as shown in the example below, selecting the signing key generated in Appendix A and Template Service connector from Appendix C from the pull-down menus:

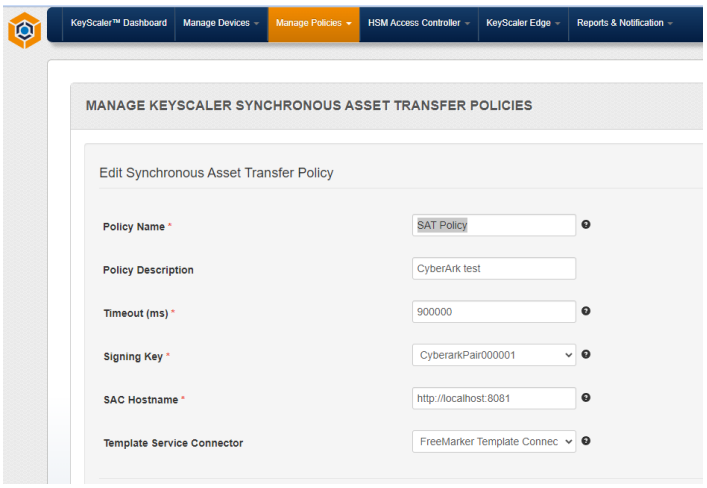


Figure 50 – KeyScaler CP> Manage Policies > SAT Policies> Fill in New SAT Policy form

For SAC Hostname[2], this could be hosted on the same VM as KeyScaler, in which case enter local host, as shown above.

[2] For any question and support on SAC Hostname, please contact cyberarkcustomer@deviceauthority.com

Click on the '?' to get additional help for each field, for example for Timeout (ms)

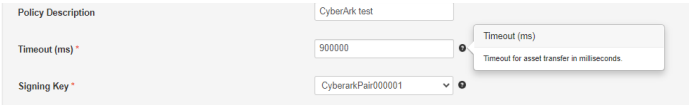
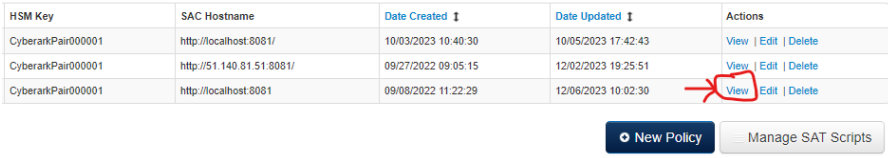


Figure 51 – KeyScaler CP> Manage Policies > SAT Policies> Popup Help for filling in SAT policy form

Click on Save Policy, to complete the SAT policy set up.
Next, view the policy, just created:



HSM Key	SAC Hostname	Date Created	Date Updated	Actions
CyberarkPair000001	http://localhost:8081/	10/03/2023 10:40:30	10/05/2023 17:42:43	View | Edit | Delete
CyberarkPair000001	http://51.140.81.51:8081/	09/27/2022 09:05:15	12/02/2023 19:25:51	View | Edit | Delete
CyberarkPair000001	http://localhost:8081	09/08/2022 11:22:29	12/06/2023 10:02:30	View | Edit | Delete

Figure 52 – KeyScaler CP> Manage Policies > SAT Policies>View Policy

And make a note of the SAT policy UUID, highlighted below, this will be needed for the SAT API application.

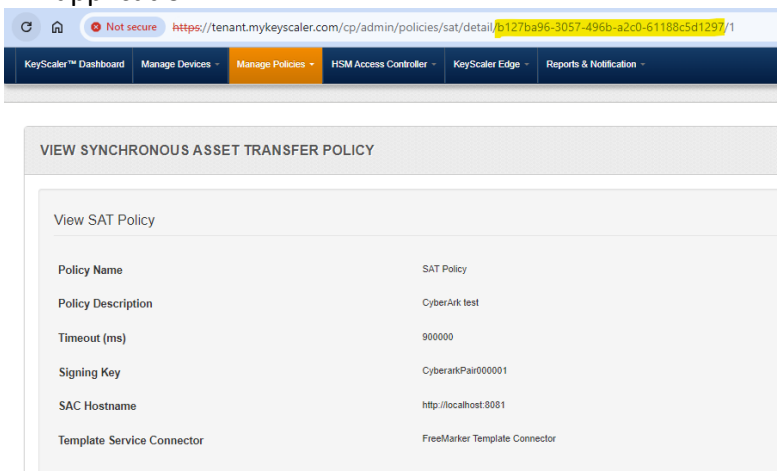


Figure 53 – KeyScaler CP> Manage Policies > SAT Policies>SAT Policy UUID

8 Appendix D – SAT Scripts Configuration

On KeyScaler CP, from the SAT Policy list Page, click on the **Manage SAT Scripts** button:

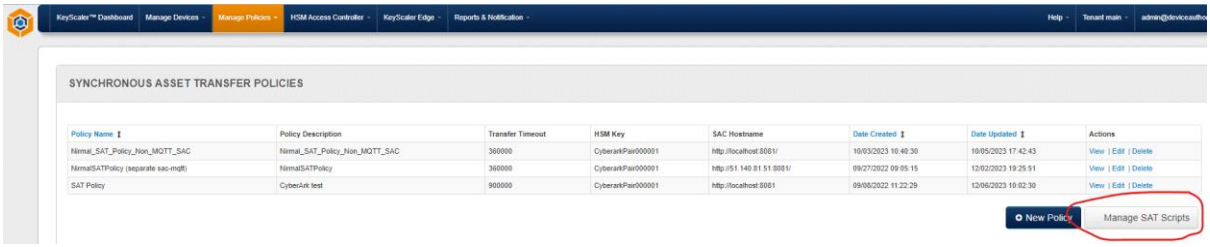


Figure 54 – KeyScaler Control Panel> Manage SAT Policy>Manage SAT Scripts

Next Click on +New Script, to give the following form to fill-in for the script set up:

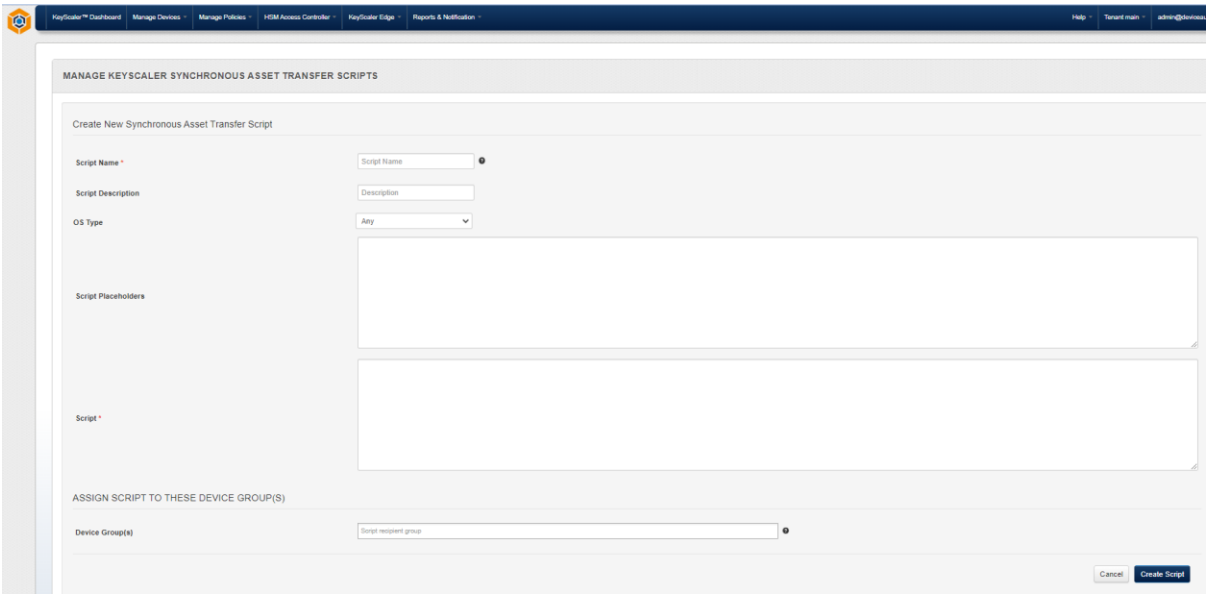


Figure 55 – KeyScaler Control Panel> Manage SAT Policy>Manage SAT Scripts>New Script

Fill in the form as shown in the example below, for list script.

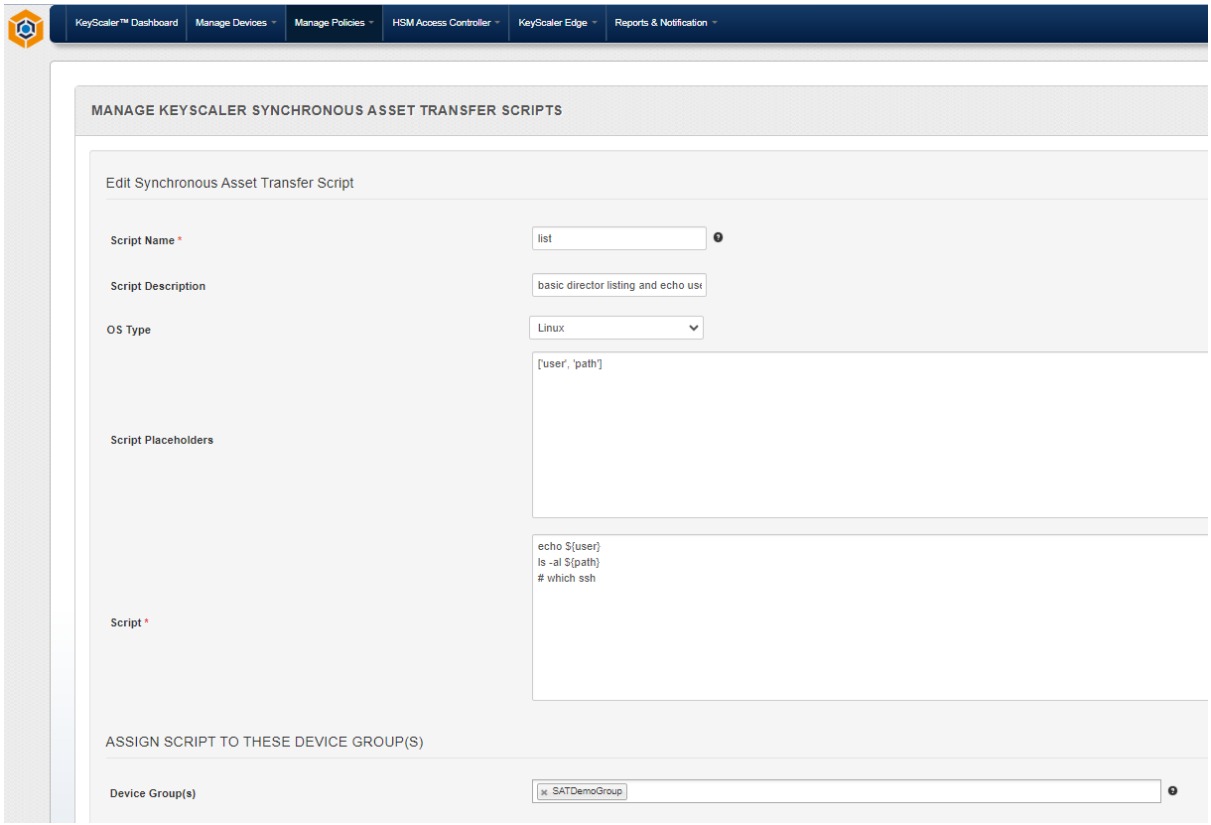


Figure 56 – KeyScaler Control Panel> Manage SAT Policy>Manage SAT Scripts>New Script for List

- **Script Name**: Free text field to enter a name for the script
- **Script Description**: Free text field to enter the description of what the script does
- **OS Type**: Pull down list of the supported OS where the script can be executed, e.g. Linux
- **Script Placeholders**: The script fields to be substituted by the service connector from the data model defined in the SAT API.
- **Script**: The script itself to be executed on the device via the SAT API.
- **Device Group(s)**: The Device Group (whose devices) have permission to have these scripts executed on them.

Click the **Save** button to complete the script configuration.

---------- **End of Document** ----------