appviewx

SSH+ Guides

Version: 2023.1.0 FP3

Copyright AppViewX, Inc.

External Reference Links

This product includes software developed by the CentOS Project (www.centos.org).

This product includes software developed by Red Hat, Inc. (www.redhat.com).

This product includes software developed by VMware, Inc. (www.vmware.com).

Contact Information

AppViewX, Inc.

222 Broadway, FL 19

New York, NY 10038

Email: info@appviewx.com

Web: www.appviewx.com

Contents

Preface....xi

Revision History....xi

About the Documentation....xi

Audience....xi

Text Conventions....xi

Chapter 1. SSH+ Admin Guide....13

Introduction to SSH+....13

Risks of improper SSH Management....13

What Enterprises Need....14

How AppViewX Can Help....14

System Requirements....15

SSH within the Application Infrastructure....15

Hardware....15

Operating System....16

Browser....16

Installation Instructions....16

On-Premise Installation Instructions....17

SaaS Installation Instructions....20

Accessing SSH+ Features....25

Configuring Provision Settings....26

Getting Started....28

Onboarding Users....28

Discovery and Visibility....28

Access Management....29

Compliance....29

Additional Settings....29

Access Requests Hub....29

Setting ACF and ACL Permissions..... 30
Setting ACF Permissions for Roles..... 31
Setting ACL Permissions to Infra Access Groups 31
Setting ACL Permissions to Key Compliance Groups..... 33
Adding or Deleting Regex Patterns..... 34
Adding Access to Users..... 35
Overview..... 35
Approving/Rejecting Access Requests..... 35
Glossary..... 36
..... 36
Chapter 2. SSH+ User Guide..... 38
Introduction to SSH+..... 38
Risks of improper SSH Management..... 39
What Enterprises Need 39
How AppViewX Can Help..... 39
System Requirements..... 40
SSH within the Application Infrastructure..... 40
Hardware..... 40
Operating System 41
Browser..... 41
Accessing SSH+ Features..... 42
Discovering Keys..... 42
Overview..... 43
Network Scan..... 43
Managed Devices..... 52
Discovery Status..... 56
Scheduler..... 61
Managing Devices/Hosts 62
Overview..... 62

Adding Credentials....63
Host Inventory....64
Adding Server 71
Adding Cloud....76
Actions 80
Access Requests Hub....81
Access Control....82
Overview....82
Requesting Access to Terminals....83
Approving Access Requests....85
Viewing Terminal Access Control Page....86
Accessing Host Terminals....87
Configuring Provision Settings....88
Setting ACL Permissions to Resources....90
Setting ACL Permissions to Infra Access Groups 90
Setting ACL Permissions to Key Compliance Groups....91
Adding or Deleting Regex Patterns....92
Adding Infra Access Groups....93
Infra Access Group....94
Adding Infra Access Group....94
Removing Hosts from Infra Access Group....95
Viewing Infra Access Group....96
Managing Host Key and User Key Inventories....96
Overview....97
Key Inventory....97
Dashboard....109
Reports....109
Remediation Actions....114
Creating Key Policy and Group....115

Overview ... 116
Key Policy ... 116
Key Compliance Group ... 118
Creating Host Policy and Group ... 119
Creating Host Policy ... 119
Glossary ... 120
... 120
Chapter 3. SSH+ API Guide ... 122
Understanding the AppViewX SSH API ... 122
RESTful HTTPS Requests ... 122
Requests ... 123
Request Structure ... 124
Response Structure ... 124
Description of Server Responses ... 124
URI Scheme ... 125
Types of Accounts in AppViewX ... 125
Authentication ... 125
Using User Account ... 126
Authentication Using Service Account ... 133
Add New Host ... 142
Before you begin ... 142
Request Structure ... 143
Payload ... 144
Response Structure ... 147
Response ... 147
Status Codes ... 147
Sample Request/Response ... 148
Reference ... 150
What's New ... 150

Search Hosts....151
Before you begin.... 151
Request Structure.... 151
Payload.... 152
Response Structure.... 153
Status Codes.... 153
Sample Request/Response.... 154
Reference....157
What's New....158
Search Host Keys....158
Before you begin.... 158
Request Structure....159
Payload.... 160
Response Structure.... 160
Status Codes.... 161
Sample Request/Response.... 162
Reference....168
What's New....169
Search User Keys....169
Before you begin.... 169
Request Structure....169
Payload.... 170
Response Structure.... 171
Status Codes.... 172
Sample Request/Response.... 172
Reference....185
What's New....186
Search Access Groups....186
Before you begin.... 186

Request Structure....186
Payload....188
Response Structure....188
Status Codes....189
Sample Request/Response....190
Reference....191
What's New....192
SSH Create CA....192
Before you begin....192
Request Structure....192
Payload....193
Response Structure....194
Status Codes....195
Sample Request/Response....196
Reference....197
What's New....198
SSH Download CA....198
Before you begin....198
Request Structure....198
Response Structure....199
Status Codes....200
Sample Request/Response....200
Reference....201
Search CA....201
Before you begin....201
Request Structure....202
Payload....203
Response Structure....203
Status Codes....204

Sample Request/Response 205
Reference 206
What's New 206
SSH Create Certificate 207
Before you begin 207
Request Structure 207
Payload 208
Response Structure 209
Status Codes 211
Sample Request/Response 212
Reference 213
What's New 214
SSH Download KRL 214
Before you begin 214
Request Structure 214
Response Structure 215
Status Codes 216
Sample Request/Response 217
Reference 217
SSH Get Hosts From Infra Access Group 218
Before you begin 218
Request Structure 218
Response Structure 220
Status Codes 221
Sample Request/Response 221
Reference 222
What's New 223
Trigger Network Scan for Range of IP Addresses 223
Before you begin 223

Request Structure..... 223
Payload..... 224
Response Structure..... 227
Response..... 228
Status Codes..... 228
Sample Request/Response..... 228
Reference..... 231
Revoke Certificate..... 232
Before you begin..... 232
Request Structure..... 232
Payload..... 233
Status Codes..... 234
Response Structure..... 234
Revoke Response..... 234
Status Codes..... 235
Sample Request/Response..... 235
Reference..... 237

Preface

Revision History

Revision	Description	Date
4.0	Initial Release of AppViewX_v2023.1.0 SSH+ FP3	Jun 2024
3.0	Initial Release of AppViewX_v2023.1.0 SSH+ FP2	Feb 2024
2.0	Initial Release of AppViewX_v2023.1.0 SSH+ FP1	Nov 2023
1.0	Initial Release of AppViewX_v2023.1.0 SSH+	Sep 2023

About the Documentation

This guide talks about the complete functionality of the AppViewX SSH Key and Host Management solution. With the help of this guide, you can manage:

- Host discovery and onboarding
- SSH key and SSH certificate discovery
- Key compliance group
- Host compliance group
- Infra access group
- Access control
- Remediation

Audience

This guide is intended for CISO, PKI Security, and Application Teams.

All the people belonging to the groups mentioned below should use this guide.

IT Operations	CISO, CIO, CTO, Server & Application Administration
Security	IAM Administration
Risk Management	Risk, Compliance, Audit Administration

Text Conventions

The following text conventions are used in this document:

Convention	Description
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in the text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`codeblock`	Indicates commands with a paragraph, URLs, codes in examples, text that appears on the screen, or text that you enter.

Chapter 1: SSH+ Admin Guide

- Introduction to SSH+
- System Requirements
- Installation Instructions
- Accessing SSH+ Features
- Configuring Provision Settings
- Getting Started
- Access Requests Hub
- Setting ACF and ACL Permissions
- Adding Access to Users
- Glossary

Introduction to SSH+

As application infrastructures grow, so do security threats. Organizations have to find newer ways for protecting their data and granting access to the right users and devices to avoid security threats and breaches. While the traditional approach used password authentication, it proved to be insecure. This is where AppViewX SSH+ comes into play.

AppViewX SSH+ is a fully-automated application infra-access management and SSH key lifecycle management solution that allows you to centrally discover, manage, and protect SSH keys with access across hybrid multi-cloud environments. It also helps simplify access management, stay compliant and mitigate risks with SSH+.

AppViewX offers visibility and SSH access management across traditional on-premises data centers and cloud-hosted infrastructures.

Risks of improper SSH Management

Since there is no governing body to regulate the use of SSH keys, there is an element of risk involved. As SSH keys are generated on a need basis, several keys may be discarded and left unmanaged when they are no longer of use. Without an inventory, managing these keys and revoking their access pose a security threat to large organizations for potential back-door entry into the network, data theft, or breaches.

Improper SSH key management can lead to unauthorized access, compliance violations, identity and access management issues, data breaches, operational disruption, and reputation damage. To mitigate these risks, organizations should implement proper SSH key management practices, including secure key storage, regular key rotation, and access controls.

What Enterprises Need

AppViewX conducted multiple surveys to identify the core features and functionality needed to address SSH management challenges. SSH and Identity and Access Management (IAM) Administrators highlighted the following requirements:

- Discover keys from standard and non-standard locations
- Identify and report non-standard and non-compliant keys
- Visibility of keys and the users of these keys
- Revoke access to non-compliant and non-standard keys
- Rotation and distribution of keys
- Self-service SSH access requests
- Support for cloud and legacy on-premise infrastructure
- Centralized SSH Certificate Authority

How AppViewX Can Help

AppViewX SSH+ key lifecycle management is a fully automated solution that discovers and manages enterprise SSH infrastructure. It can identify and mitigate risks associated with poorly managed passwordless access management.

AppViewX SSH+ features include:

- **Centralized Discovery and Visibility**
  - The solution offers on-demand scans to discover SSH keys across multi-vendor, hybrid network infrastructures, and map trust relationships to determine access privileges.
  - The consolidated inventory provides a central console to view and manage all SSH keys and hosts.
- **Risk Scorecard and One-Click Remediation**
  - The solution proactively identifies and remediates risks associated with inactive, weak, orphan, or suspicious keys using an intuitive SSH scorecard dashboard.
  - The one-click remediation feature enables instant deletion or regeneration of keys.

System Requirements

SSH within the Application Infrastructure

Application infrastructure refers to all the components required to deliver an application and its functions and services to the customer. Although each application is unique, certain common components can be identified that are typically implemented to support application capabilities and service delivery.

One of the most common components of a typical application infrastructure is Linux hosts.

Modern application infrastructure leverages multiple hosting platforms such as on-premise data centers, private clouds, and public clouds offered by third-party hosting providers.

The application infrastructure components communicate with each other to enable service delivery. This can leverage SSH communication. Additionally, administrators of these application infrastructures will need to SSH into these hosts to perform maintenance.

Additionally, security and audit compliance requirements necessitate constant awareness of who has access to what and the maintenance of best security practices for the communications.

Hardware

Ensure that you have, at minimum, the following hardware with the given specifications before proceeding with the installation:

- **Single Node Deployment Requirements**

Node	Bare Minimum		
	CPU	RAM	Hard Disk Space
Single node	8	32GB	500GB

- **Multi-Node Deployment Requirements**

Node	Bare Minimum		
	CPU	RAM	Hard Disk Space
Multi-node (master node)	4	4GB	100GB

Note: One node for a single master installation and a minimum of three nodes for multi-master installation.

Node	Bare Minimum		
	CPU	RAM	Hard Disk Space
Multi-node (worker node)	8	32GB	500GB

- **Platform Bare Minimum Requirements**

Supported Virtualization Platforms	Versions	vCPU	RAM	HDD
VM Server, VMware ESXi	5.5 or later	8v	32GB	1TB

Operating System

Both single node and multi-node installations of AppViewX are supported on the following operating systems:

- RHEL 8.5
- RHEL 8.6
- RHEL 8.7
- Ubuntu 20.04

Browser

Following is the browser requirements to use the AppViewX SSH+ node:

Browser	Version
Firefox	v74.0.1 (64-bit) or later
Google Chrome	v85.0.4183.83 (64-bit) or later

Installation Instructions

- On-Premise Installation Instructions
- SaaS Installation Instructions

On-Premise Installation Instructions

This chapter provides the step-by-step instruction for on-premise deployment.

On-Premise Installation Steps

Step No	Step Name	Mandatory
1	Working with Prerequisites	Yes
2	Configuring Firewall	Yes
3	Configuring Elevated Access	Yes
4	Running the Prerequisite Tool	Yes
5	Deploying the AppViewX Virtual Appliance	No
6	Performing a Single Node or Standalone Installation	No
7	Performing a Multi-node or High Availability Installation	No
8	Configuring the appviewx.conf file	Yes
9	Configuring POD and Service IP CIDR	No
10	Verifying the Installation	Yes
11	Activating SSH	Yes
12	Accessing the AppViewX Graphical User Interface	Yes
13	Adding Third-party Libraries	No

- Activating SSH+
- Uninstalling AppViewX
- Troubleshooting
- Migrating CentOS to Ubuntu/RHEL

Activating SSH+

License Management Software tracks software installed throughout the enterprise and ensures legal licenses for its usage. The software helps you to obtain the license key, upload the license key, and troubleshoot the license issues. License management is an essential element of software asset management (SAM).

To access the application for SSH+, send an email to help@appviewx.com with the hostname of the node in which the application is installed.

Uninstalling AppViewX

Users can uninstall AppViewX when they want to migrate into another environment. They can also uninstall AppViewX when it is no longer required.

To uninstall an application package safely:

1. Open the terminal window.
2. To navigate to the **appviewx_kubernetes** directory, execute the following command:

   ```
   cd /home/appviewx/appviewx_kuberbetes/scripts/uninstall
   ```

3. To start the uninstallation process, execute the following command:

   ```
   /uninstall.sh
   ```

4. Enter the node's credentials when prompted.
5. Reboot all the nodes after completion of the AppViewX uninstallation.

Troubleshooting

Whenever the AppViewX installation fails, you will get an error stating that some script execution failed.

- **Prerequisites not met**

  Please check for all the items below.

  - port not opened
  - insufficient disk/CPU
  - time not in sync
  - packages not found
  - hostname incorrect in configuration
- **Error while installing the docker**

  If a customer brings in a custom OS, the Linux packages that AppViewX includes with the installer may not be compatible with the OS. In such situations, you may need to install the appropriate package to continue. This can be observed from the log messages that indicate an error while installing a package.
- **Error while installing the docker**

Occasionally, we have observed intermittent errors from the OS during the installation of Docker. If you encounter an error at this stage, please attempt to uninstall the application, reboot all nodes, and then proceed with the installation.

- **Docker gets uninstalled from the CAGateway**

  Root cause: Although we removed the "uninstall docker" commands from our scripts, we discovered that Docker relies on containerd, which is used as a runtime in the product. The scripts also include steps to remove containerd in the install, uninstall, and upgrade scripts, which cannot be avoided. This ultimately results in the removal of Docker as well. Additionally, the containerd version used in the product conflicts with the pre-existing containerd version of Docker on the server.

  Docker and the AppViewX application cannot co-exist in the same server as it is tightly coupled with containerd. The manually installed docker will be removed during every maintenance activity such as install, uninstall and infra upgrade.

- **Context deadline exceeded in consul after the FP3 patching process**

  For setups with high network latency or slow I/O, after the FP3 patch process, the consul may be stuck in 1/2 stage, causing the vault to go in a crash loop back. If you encounter this, check the consul logs using the command

  ```
  kubectl logs consul-consul-server-0 -n avx
  ```

  If the logs specify “context deadline exceeded,” then increase the timeout in consul by the following steps:

  1. Navigate to <installer location>/appviewx_kubernetes/yaml/appviewx_vaut/consul/chart/vaules.yaml
  2. Edit **consulAPITimeout: 5s** (old value) to **consulAPITimeout: 10s** (new value)
  3. Save the changes.

  Note: Increase this timeout only based on the latency.

- **Error while initializing the kube master/worker**

  In certain cases, when uninstallation does not clean up the data properly, we may observe errors while initializing kube master and worker. In such cases, perform an uninstall, reboot all the nodes and then go ahead with the install. Additionally, there are cases where the installation fails due to port connectivity issues. If a failure occurs in this stage, check if ports 6443, 10250, 2379 and 2380 are opened properly.

- **Error while initializing the mongodb chart**

This specific error occurs after a timeout of 5 minutes to initialize the mongodb charts. This error occurs when the pods are not able to communicate between themselves. Use the following commands to verify that:

```
kubectl describe statefulset -n avx mongo-shareddb
```

For any connectivity issues, the output of this command will display the specific error stating connection timed out.

- **Node is enabled with IPv6 but the application is not.**
  Verify the output of the command:

```
ifconfig | grep -i inet6
```

If an IPv6 address is displayed, it is necessary to enable IPv6 in the appviewx.conf file. Failure to do so may result in communication issues.

- **IP in IP tunneling is not enabled**

If the IP in IP traffic is disabled, which means that the IPv4 protocol is not permitted, we will encounter the same problem. The prerequisite check script does not identify this, so we need to verify it separately to confirm.

- **Error while installing the AppViewX plugins**

If an error occurs during the installation of AppViewX plugins, it is likely due to an error in the configuration file. You may observe an error such as `Upload failed: scp`, in such cases re-trigger plugins_install.sh to install the plugins. Likewise, ensure to review the configuration file carefully and proceed with the execution of plugins_install.sh to install only the plugins.

Migrating CentOS to Ubuntu/RHEL

For detailed instructions, refer to Migrating CentOS to Ubuntu/RHEL.

SaaS Installation Instructions

For detailed instructions, refer to AppViewX SaaS Setup Guides.

- AppViewX Software as a Service
- AppViewX SaaS Onboarding and Getting Started Guide
- Features of the AppViewX Cloud Connector
- System Requirements
- Setting Up the AppViewX Cloud Connector
- Prerequisites for Managing ADC Devices

- Installing the AppViewX Windows Gateway
- Managing the AppViewX Cloud Connector
- Troubleshooting the AppViewX Cloud Connector

AppViewX Software as a Service

For more information, refer to AppViewX Software as a Service and SaaS Architecture Guide.

AppViewX SaaS Onboarding and Getting Started Guide

For more information, refer to AppViewX SaaS Onboarding and Getting Started Guide.

Features of the AppViewX Cloud Connector

For more information, refer to Features of the AppViewX Cloud Connector.

System Requirements

For more information, refer to System Requirements for Setting up the AppViewX Cloud Connector.

Setting Up the AppViewX Cloud Connector

- **Methods to Set up the AppViewX Cloud Connector:** For more details, click here.
- **Setting up the AppViewX Cloud Connector via a Virtual Image:** For more details, click here.
- **Setting up the AppViewX Cloud Connector via the Native OS:** For more details, click here.
- Installing the AppViewX Cloud Connector

Installing the AppViewX Cloud Connector

Note: The following steps assume that:

- All system prerequisites are fulfilled by the host machine.
- The AppViewX Cloud Connector installer (downloaded in the above step) is securely copied via SCP/SFTP to the host machine where the AppViewX Cloud Connector is to be installed.

1. To extract the installer, from the downloaded package, extract the tar.gz file using the command given below: `tar -zxvf <filename>.tar.gz`
   For example: `tar - zxvf pesrv07-test-94-99-appviewx-appviewx-net-cloud-connector.tar.gz`
2. On the node where the AppViewX Cloud Connector agent will be installed, from the extracted installation package, run the **./install.sh** script.
   The script will check if the installation prerequisites for the AppViewX Cloud Connector have been fulfilled.

   Note:
   Ensure that the license file is placed in the same location as the **install.sh** script. If the license file is placed in another location, run the install.sh script using the following command:

   ```
   ./install.sh <complete path of the license file with the filename>
   ```

   On successful verification of the prerequisites, you will be prompted to specify if you want to manage f5 BIG-IP devices and if you need auto-enrollment of the certificates.
3. Enter the required input value:

   Important: If you choose to **not enable** any of the following features, you will have to reinstall the AppViewX Cloud Connector to enable them later.

   a. If you want to manage f5 BIG-IP devices, enter **y**/**n** for yes/no, respectively.

   b. If you need auto-enrollment of the certificate using one of the following supported auto-enrollment protocols, enter **y**/**n** for yes/no, respectively.
   - If you choose **y** (yes) here, enter the required protocol(s) name. SSH server is installed.
   - If you choose **n**(no), you will see this prompt:

     Do you want to enable SSH Terminal Server for using SSH terminal usecase (y/n)?

     If you choose y, SSH server is installed.

   c. If you want to enable Syslog receiver for a near-real time configuration updates from the devices, enter **y**/**n** for yes/no, respectively. For configuring Syslog reception, refer to Platform User guide section, Syslog Reception.

   In case you have an older version of AppViewX on cloud and want to make use of Syslog capabilities for ADC, you must manually activate the Syslog flag by setting SYSLOG_ENABLED=`true` in the path ccpath/deps/properties.

4. Enter the sudo password.

After the relevant details have been entered, the installation proceeds. Installation logs, according to the outcome of the installation, are displayed.

Given below are sample installation logs:

```
Loaded image: rancher/k3s:v1.23.3-k3s1
Loaded image: rancher/k3d-tools:5.2.2
Loaded image: rancher/mirrored-pause:3.6
[36mINFO[0m[0000] [SimpleConfig] Hostnetwork selected - disabling injection of docker host into the cluster, server load balancer and setting the api port to
 the k3s default
[33mWARN[0m[0000] No node filter specified
[33mWARN[0m[0000] No node filter specified
[33mWARN[0m[0000] No node filter specified
[36mINFO[0m[0000] Prep: Network
[36mINFO[0m[0000] Re-using existing network 'host' (8bebb4ae61001f74487d0aa6b315396405d0127c938da1206614d113295ae139)
[36mINFO[0m[0000] Created volume 'k3d-cc-images'
[36mINFO[0m[0000] Starting new tools node...
[36mINFO[0m[0000] Starting Node 'k3d-cc-tools'
[36mINFO[0m[0001] Creating node 'k3d-cc-server-0'
[36mINFO[0m[0001] Using the k3d-tools node to gather environment information
[36mINFO[0m[0001] Starting cluster 'cc'
[36mINFO[0m[0001] Starting servers...
[36mINFO[0m[0001] Starting Node 'k3d-cc-server-0'
[36mINFO[0m[0033] All agents already running.
[36mINFO[0m[0033] All helpers already running.
[36mINFO[0m[0033] Cluster 'cc' created successfully!
[36mINFO[0m[0034] You can now use it like this:
kubectl cluster-info
Cluster setup is completed. Will start the deployment shortly...
Importing the required images...
[36mINFO[0m[0000] Importing image(s) into cluster 'cc'
[36mINFO[0m[0000] Importing images from 1 tarball(s)...
[36mINFO[0m[0000] Importing images '[/home/appviewx/CCTEST/deps/tools/mid-server-docker-image/avx-mid-server-base-22.1.0.0.tar]' into node
 'k3d-cc-server-0'...
[36mINFO[0m[0024] Successfully imported image(s)
[36mINFO[0m[0024] Successfully imported 1 image(s) into 1 cluster(s)
Import in progress...
[36mINFO[0m[0000] Importing image(s) into cluster 'cc'
[36mINFO[0m[0000] Importing images from 1 tarball(s)...
```

```
[36mINFO[0m[0000] Importing images '[/home/appviewx/CCTEST/deps/tools/mid-server-docker-image/k3d-tools-5.2.2.tar]' into node 'k3d-cc-server-0'...
[36mINFO[0m[0005] Successfully imported image(s)
[36mINFO[0m[0005] Successfully imported 1 image(s) into 1 cluster(s)
Import in progress...
[36mINFO[0m[0000] Importing image(s) into cluster 'cc'
[36mINFO[0m[0000] Importing images from 1 tarball(s)...
[36mINFO[0m[0000] Importing images '[/home/appviewx/CCTEST/deps/tools/mid-server-docker-image/rancher-mirrored-coredns-coredns-1.8.6.tar]' into
 node 'k3d-cc-server-0'...
[36mINFO[0m[0007] Successfully imported image(s)
[36mINFO[0m[0007] Successfully imported 1 image(s) into 1 cluster(s)
[36mINFO[0m[0000] Importing image(s) into cluster 'cc'
[36mINFO[0m[0000] Importing images from 1 tarball(s)...
[36mINFO[0m[0000] Importing images '[/home/appviewx/CCTEST/deps/tools/mid-server-docker-image/rancher-local-path-provisioner-v0.0.21.tar]' into node
 'k3d-cc-server-0'...
[36mINFO[0m[0004] Successfully imported image(s)
[36mINFO[0m[0004] Successfully imported 1 image(s) into 1 cluster(s)
[36mINFO[0m[0000] Importing image(s) into cluster 'cc'
[36mINFO[0m[0000] Importing images from 1 tarball(s)...
[36mINFO[0m[0000] Importing images '[/home/appviewx/CCTEST/deps/tools/mid-server-docker-image/rancher-mirrored-pause-3.6.tar]' into node
 'k3d-cc-server-0'...
[36mINFO[0m[0003] Successfully imported image(s)
[36mINFO[0m[0003] Successfully imported 1 image(s) into 1 cluster(s)
Deploying the Cloud Connector...
NAME: avx-mid-server-starter
LAST DEPLOYED: Mon May 30 15:51:13 2022
NAMESPACE: cc
STATUS: deployed
REVISION: 1
NOTES:
1. It may take a couple of minutes for the Cloud Connector to be up.
  kubectl get pod --namespace cc
**************************************************************************
*    Congratulations!!! The installation completed successfully.   *
*    Please wait till the Cloud Connector is up and running.       *
**************************************************************************
(1%) Cloud Connector status: Running
[32m Cloud Connector is up and running. (B[m
```

Troubleshooting: For installation errors, refer to the Troubleshooting section.

The AppViewX Cloud Connector consists of two important components—the starter plugin and the platform. The starter plugin component is installed along with the AppViewX Cloud Connector, in the same installation process.

When installed, the starter plugin is used to initiate the download of the platform component. The platform component is used to host business use cases related to the AppViewX Cloud Connector.

When the platform component download is in progress, it is indicated by the symbol prefixed to the platform component version number in the AppViewX Cloud Connector inventory details 

. A completed download/upgrade is indicated by the symbol in the same location 

.

Note: Based on the internet bandwidth and the number of cloud connectors being installed, the downloading of the cloud connector may vary between 5 to 15 minutes.

Prerequisites for Managing ADC Devices

For more information, refer to Prerequisites for Managing ADC Devices.

Installing the AppViewX Windows Gateway

For more information, refer to Installing the AppViewX Windows Gateway.

Managing the AppViewX Cloud Connector

For more information, refer to Managing the AppViewX Cloud Connector.

Troubleshooting the AppViewX Cloud Connector

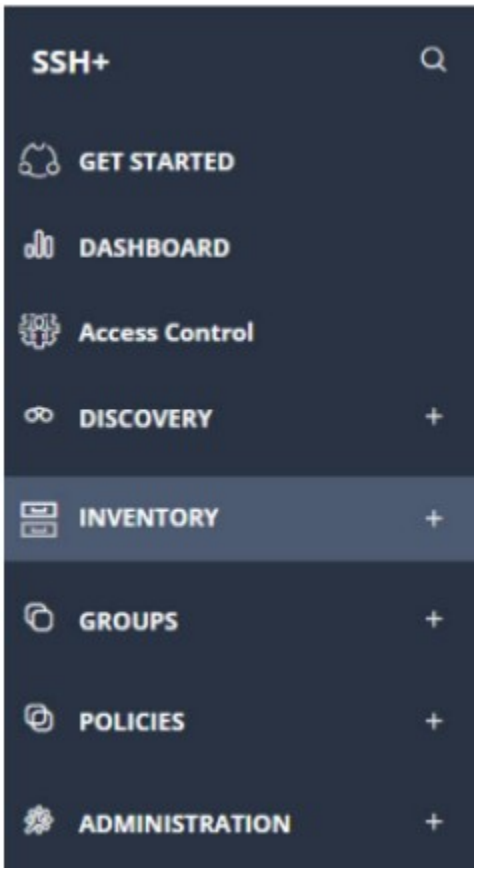
For more information, refer to Troubleshooting the AppViewX Cloud Connector.

Accessing SSH+ Features

You have to access the SSH+ node to access the various functions provided by it.

To access SSH+:

1. Log into AppViewX with valid credentials.
2. Hover the mouse pointer over (**Menu**) icon on the top-left corner of the screen.
3. From the left pane, click **SSH+**. You can now see the different menus of **SSH+** on the left hand side of the page.



4. From the left pane, expand any of the nodes to see that page.

> **Note:** For detailed functionalities, refer to the SSH+ User Guide.

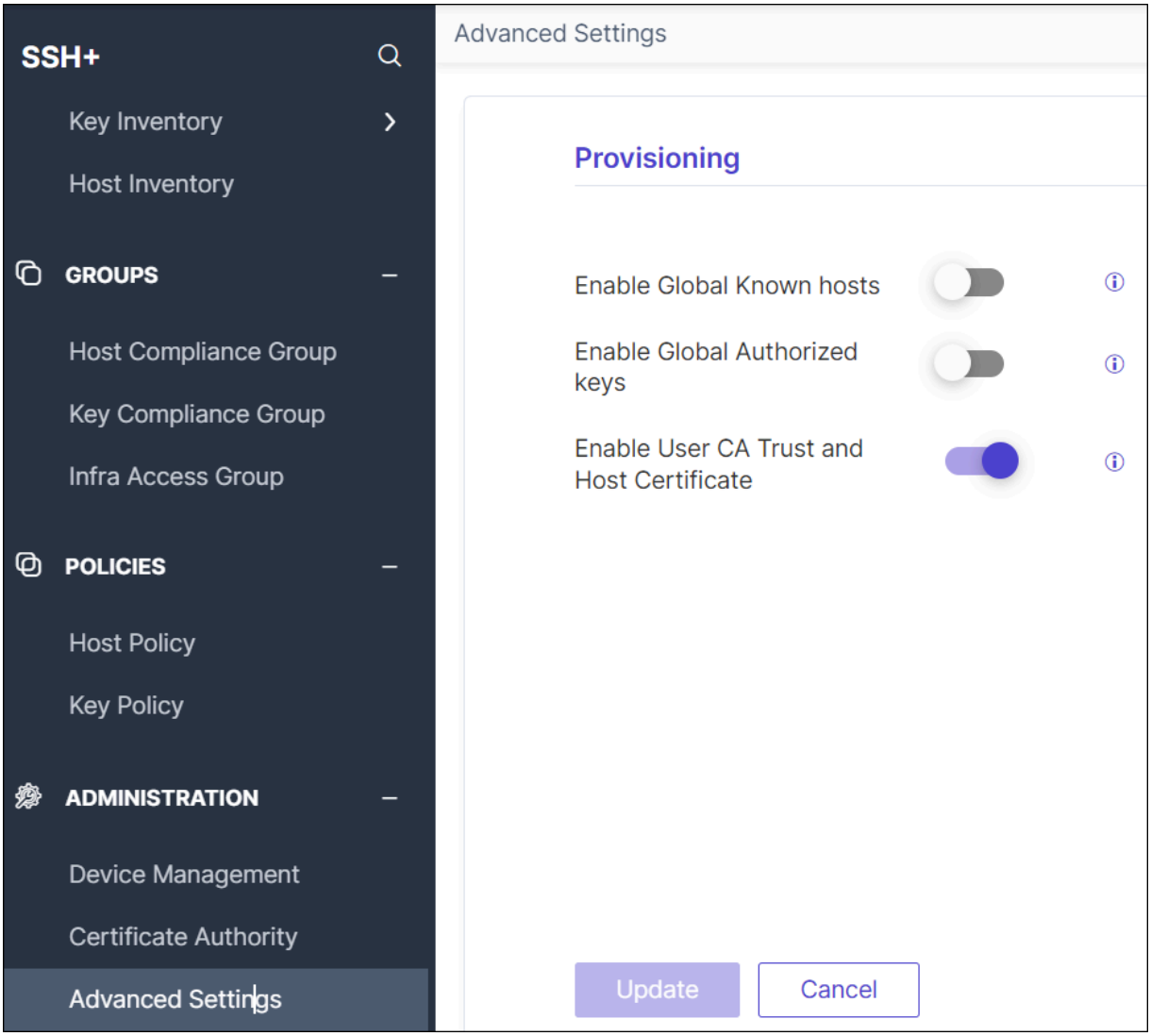
Configuring Provision Settings

You can gain comprehensive control over key provisioning with the **Provisioning** feature, which provides following options:

- **Provision to Enable Global Known hosts**
- **Provision to Enable Global Auth keys**
- **Provision to Enable User CA Trust and Host Certificate**

Procedure

1. Go to (**Menu**) icon **> SSH+ > Administration > Provisioning**.
   The **Provisioning** page is displayed.



2. Enable the **Enable Global Known hosts**. By default, this option is disabled. Enabling this option allows to generate a Global Known Hosts configurtaion, ensuring that all new keys/certificates generated during access requests, provisioning, or remediation activities (such as key rotation) are added to the Global Known Hosts repositories.
3. Enable the **Enable Global Authorized keys**. By default, this option is disabled. Enabling this option allows to establish a Global Auth Keys configurtaion, ensuring that all new keys/certificates generated during access requests, provisioning, or remediation activities (such as key rotation) are added to the Global Known Keys repositories.

4. Enable the **Enable User CA Trust and Host Certificate**. By default, this option is enabled. This option allows certificate based authentication for the devices.
5. Click **Update**. A message, *Provisioning settings saved successfully*, appears.

Getting Started

This section explains the SSH+ management workflow as seen on the Get Started page:

Onboarding Users

AppViewX offers comprehensive support for Role and Resource-Based Access Control (RBAC). RBAC is a method of restricting AppViewX functions, and managing and monitoring network resources in AppViewX based on the roles of individual users within an enterprise. It allows you to integrate with the existing identity stores such as Active Directory (AD) and Lightweight Directory Access Protocol (LDAP) to enforce authorization policies. Roles and resources can be customized to suit any organizational structure and user requirements.

For more information on configuring role and RBAC, refer to the Section, Configuring Role and Resource-Based Access Control (RBAC) in the Platform Guide.

Discovery and Visibility

You can gain full visibility of SSH user and host keys by discovering them within your network infrastructure.

- **Discover Hosts and Keys from IP network:** You can discover user or/and host keys configured on your server by creating and running scans on your network using IP address or subnet. The Discovery scans your network (on the default SSH enabled port 22) for SSH keys configured on your server. You can map the discovered keys to the selected key compliance groups and manage/monitor them.

  Clicking this link takes you to the **Discovery** page. See Using IP Range Option.

- **Risk Report Dashboard:** The risk reports are generated based on user and host configuration, key usage, and access requests once the discovery is complete. This information can help identify potential security risks, compliance issues, or other areas for improvement.

  Clicking this link takes you to the **Dashboard** page. See Reports.

- **User Key Inventory:** You can see the total number of weak, shared, orphan, and suspicious keys in the key discovery status. Click the number hyperlink to drill down on the metrics. This helps you track the progress of the key discovery efforts, identify any potential security risks, and prioritize the remediation actions.

  Clicking this link takes you to the **Key Inventory** page. See Viewing User/Host Key Inventory.

- **Host Key Inventory:** You can see the total number of weak and shared keys in the key discovery status. Click the number hyperlink to drill down on the metrics. This helps you track the progress of the key discovery efforts, identify any potential security risks, and prioritize the remediation actions.

  Clicking this link takes you to the **Key Inventory** page. See Viewing User/Host Key Inventory.

Access Management

You can ensure secure access control and authorization within your network infrastructure.

- **Onboard AWS Hosts:** Clicking this link takes you to the **Device Management > Device :: Cloud** page. See Adding Cloud.
- **Host onboarding settings:** Clicking this link takes you to the **Host Inventory** page. See Adding Host.
- **Manage Host Access Groups:** Clicking this link takes you to the **Infra Access Groups** page. See Adding Infra Access Group.
- **Request Access:** Clicking this link takes you to the **Access Control** page. See Accessing Requests

Compliance

You can use key policies for configuring key rotation to ensure secure and standardized key management practices.

- **Configure Key Compliance Policy:** Clicking this link takes you to the **Key Policy** page. See Key Policy.
- **Configure Host Configuration Policy:** Clicking this link takes you to the **Host Policy** page. See Creating Host Policy.

Additional Settings

- **Onboard Network Hosts:** Clicking this link takes you to the **Device Management > Device :: Server** page. See Adding Server.

Access Requests Hub

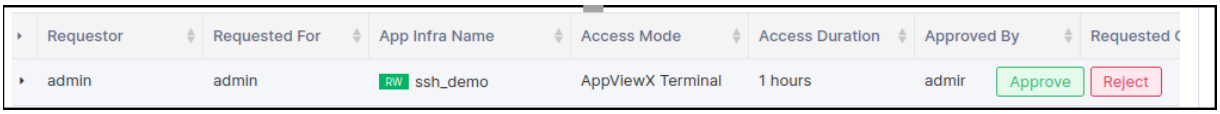
Any user with administrator privileges can use Access Request Hub if you have ACF permissions enabled. This is a full-featured tool for access request management enhancing security governance, operational efficiency, and compliance through detailed oversight and control of user access within the system.

1. Go to (**Menu**) icon **> SSH+ > Manage Access > Access Requests Hub**.

   The **Access Requests Hub** page is displayed.

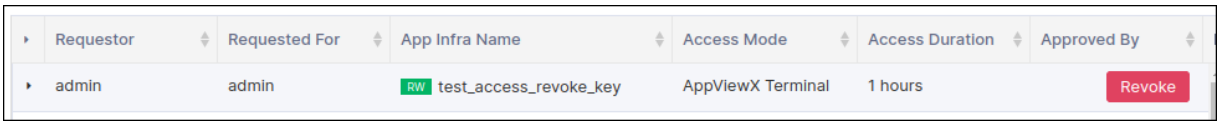
   You can view the details of the access requests such as requestor, requested for, app infra name, access mode, access duration, approver, and requested date.

2. You can approve or reject requests that are in *Pending Approval* status by hovering over the access request to see the **Approve or Reject** button as shown.



   On approving the request, the **Access Status** column changes to **Accessible**, and the **Approve or Reject** button changes to **Revoke**.

3. You can revoke access requests that are in **Accessible** status by hovering over the access request to see the **Revoke** button as shown.



4. You can use the **Search** option to search access requests by requester, requested for, app infra name, access mode, access duration, and approver.
5. You can sort and filter access requests by clicking the hyperlink on the top panel that displays the number of access requests by their statuses such as accessible, partial, pending, denied, expired, failed, and revoked. Clicking the hyperlink fetches those access requests and displays details of the access requests.

Setting ACF and ACL Permissions

This section explains how you can set authorized functions (ACF) permissions for users based on their roles. Depending on the role and the ACF permissions set, the user can access some or all of the SSH+ functionalities.

You can also assign/unassign resources access control list (ACL) permissions for infra access groups and key compliance groups. Depending on the ACL permissions set, resources will have partial, full, or no access to the groups and related actions.

- Setting ACF Permissions for Roles
- Setting ACL Permissions to Infra Access Groups

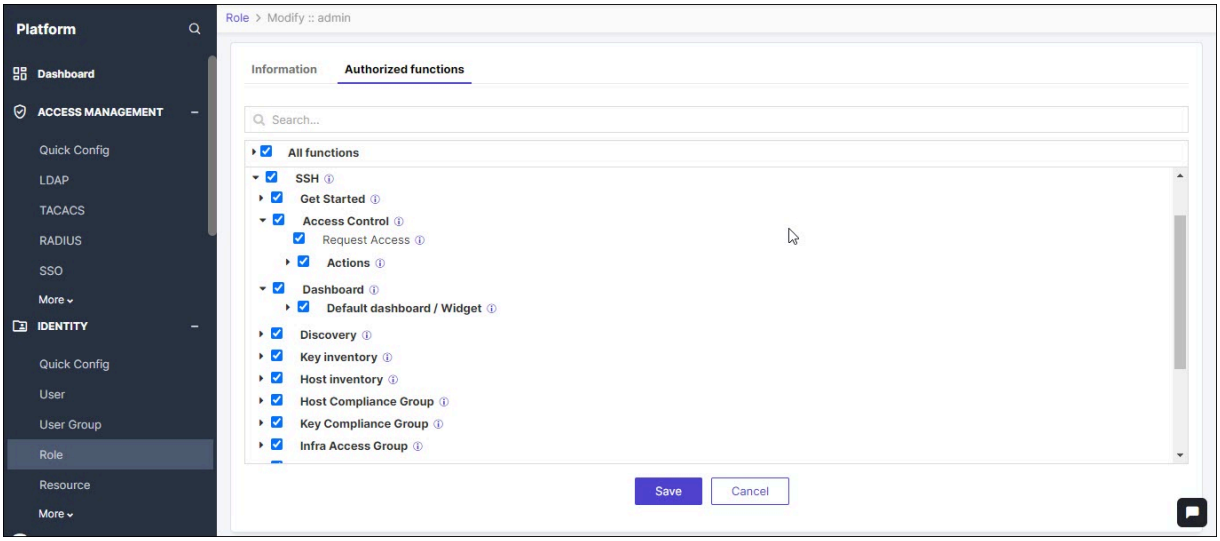
• Setting ACL Permissions to Key Compliance Groups
• Adding or Deleting Regex Patterns

Setting ACF Permissions for Roles

As administrators, you can set authorized functions (ACF) permissions for users based on their roles. Depending on the role and the ACF permissions set, the user can access some or all of the SSH+ functionalities.

To set ACF permissions for a role:

1. Go to (**Menu**) icon **> Platform > Identity > Role**.
2. Click the role for which you want to set ACF permission.
3. Select **Authorized functions.**
4. Select and expand SSH to select and unselect the functions.



5. Click **Save.**
   While most of the changes are implemented instantaneously, it takes approximately 3 minutes for changes in **Access Control** function to take effect.

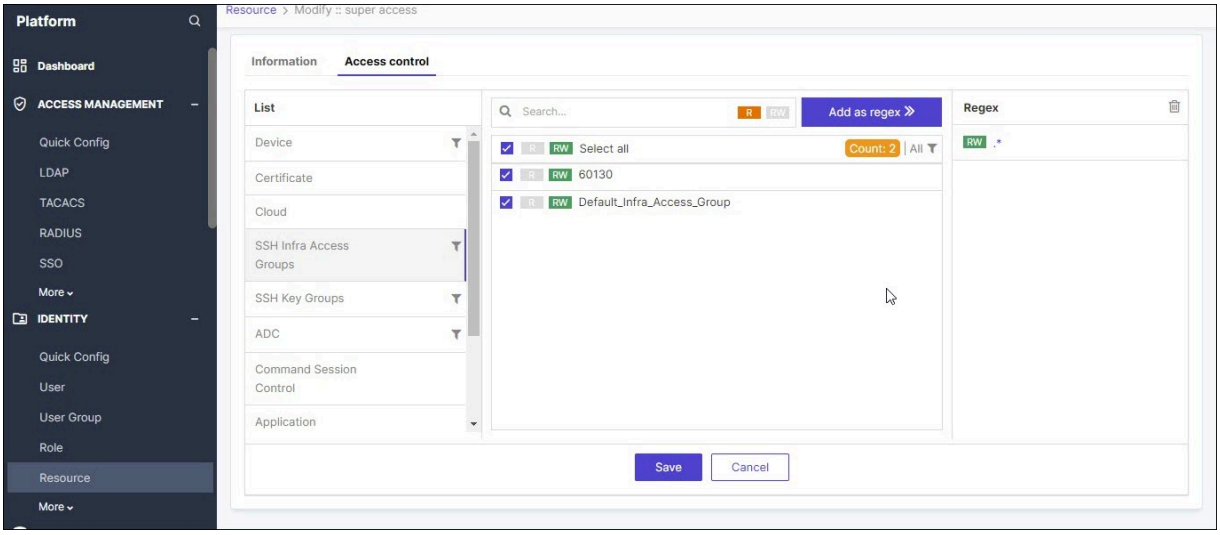
Setting ACL Permissions to Infra Access Groups

You can assign/unassign resources the following access control list (ACL) permissions to infra access groups:

- **RW** denotes that users have *Read-Write* permission to the infra access group along with hyperlinks to request access to the group on the **Access Control** page. Users can modify or delete infra access groups.
- **R** denotes that users only have *Read* permissions to the infra access groups. Users cannot request access or modify or delete infra access groups.
- If R or RW permission is not assigned, then users will not be able to view the infra access groups.

To assign ACL permission to infra access groups:

1. Go to (**Menu**) icon **> Platform > Identity > Resource**.
2. Click a resource name.
3. Click **Access Control** tab and select **SSH Infra Access Groups** from the list.



The infra access groups are displayed on the right.

4. Select the infra access group for which you want to assign permission by clicking the check box.
5. Click **R** or **RW** button to assign the read or read-write permission.
6. Click **Save**.

A message, *Successfully assigned/unassigned SSH groups data for resource super access*, appears.

Note: On deleting a resource, all ACL permissions from the associated infra access groups are also removed.

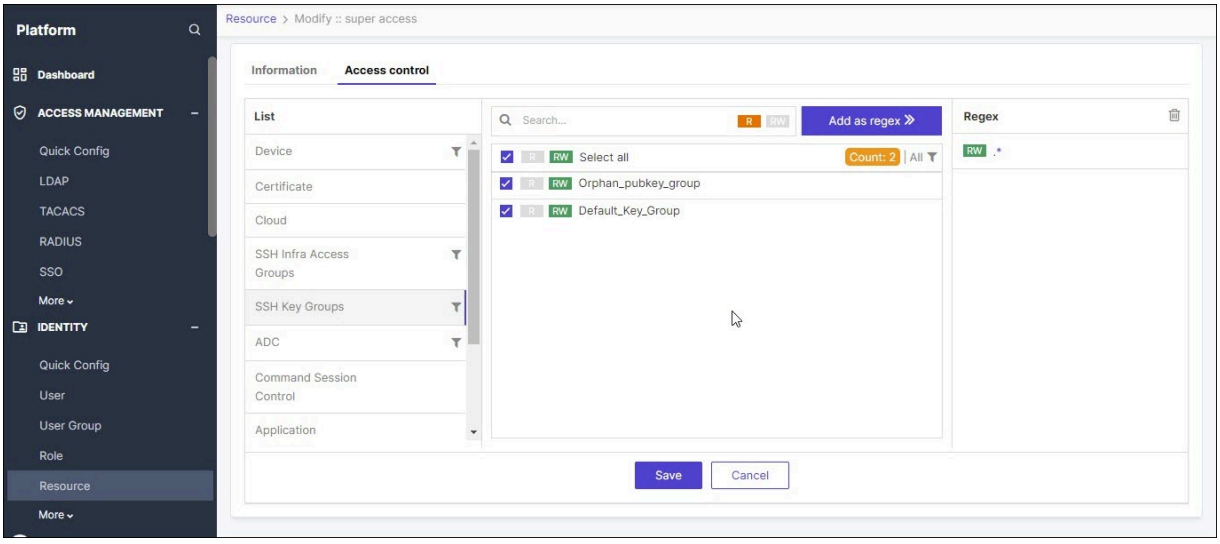
Setting ACL Permissions to Key Compliance Groups

You can assign/unassign resources the following access control list (ACL) permissions to key compliance groups:

- **RW** denotes that users have *Read-Write* permissions to the key compliance groups. Users can modify and delete key compliance groups. Users can view the keys associated with the key compliance groups and perform any of the actions in the inventory.
- **R** denotes that users have *Read* permissions to only view details displayed on the **Key Compliance Group** page. Users can only view the keys associated with the key compliance groups but cannot perform any action in the inventory.
- If R or RW permission is not assigned, then users will not be able to view the key compliance groups and the keys associated with them.

To assign ACL permission to key compliance groups:

1. Go to (**Menu**) icon **> Platform > Identity > Resource**.
2. Click a resource name.
3. Click **Access Control** tab and select **SSH Key Groups** from the list.



The key groups are displayed on the right.

4. Select the key group for which you want to assign permission by clicking the check box.
5. Click **R** or **RW** button to assign the read or read-write permission.
6. Click **Save**.

A message, *Successfully assigned/unassigned SSH groups data for resource super access*, appears.

Note: On deleting a resource, all ACL permissions from the associated key compliance groups are also removed.

Adding or Deleting Regex Patterns

Regex stands for regular expression; it is a string used to define filters. The string can contain a part of the device name or a key scan instance. These expressions are stored in the registry.

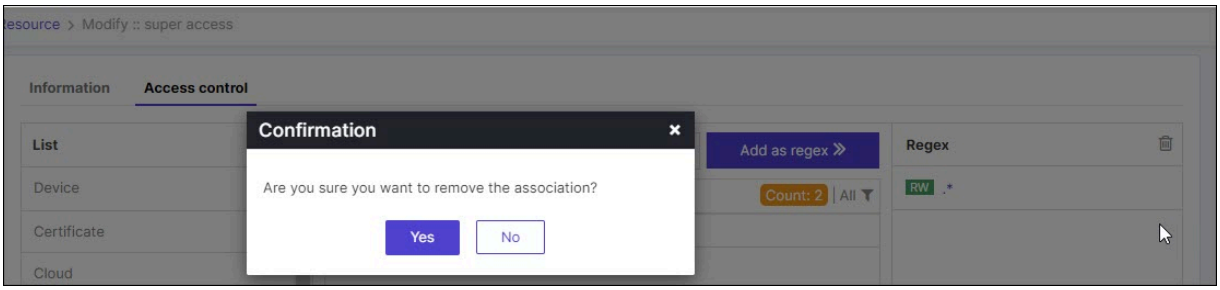
With Regex (Regular Expression) support, you can dynamically map user groups to infra access groups and key groups within the SSH+ module. This feature allows for the automatic mapping of newly created infra access groups and key groups that match specified Regex patterns, thus streamlining the process and reducing manual overhead.

To add or delete Regex patterns:

1. Go to (**Menu**) icon **> Platform > Identity > Resource**.
2. Click a resource name.
3. Click **Access Control** tab and select **SSH Infra Access Groups** or **SSH Key Groups** from the list.
4. Type a regex pattern in the **Search** box and click **Add as regex**.
   Examples of regex pattern include:
   - .*<text>
   - <text>.*
   - <text>.*<text>
   - .*

   where <text> can be alphabets, numbers, and special characters to create these entries in the pages.

   A message that the Regex pattern is successfully added for admin appears. Based on the matched regex, permission is automatically added to that resource. The newly added regex appears under the Regex column on the RHS of the page.
5. To individually delete a regex pattern, select the regex pattern from the **Regex** column and click **X** against the name. To delete all regex patterns, click (Delete) icon. A confirmation message appears as shown.



Click **Yes** and a message that it is successfully deleted from the matching groups appears.

Adding Access to Users

• Overview
• Approving/Rejecting Access Requests

Overview

With access control, you can control and manage users and user lists who have access to the infra access groups and the actions they perform on the hosts. This helps to keep the infra access hosts secure.

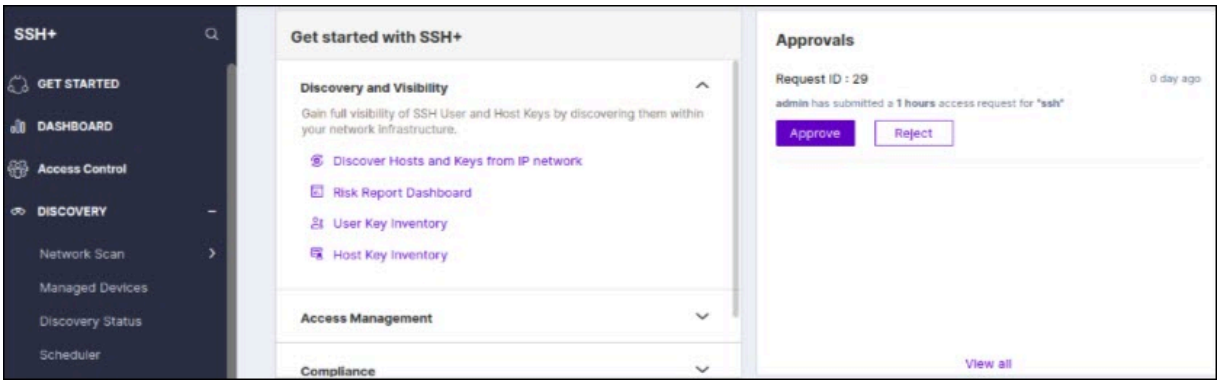
You can view all the access requests from users on the **Get Started** page in the **Approvals** section. You can approve/reject the requests efficiently and effectively based on the business justification and for the time duration requested in the access request form. On approving an access request, the user is granted access to the specified infra access group and hosts for the requested duration. The users will be able to perform authorized tasks such as running scripts, executing commands, and troubleshooting issues on those resources for that duration. On rejecting an access request, the user is denied access to the requested resources and will not be able to perform any actions.

Approving/Rejecting Access Requests

All requests for terminal access are displayed on the **Get Started** page in the **Approvals** section.

To approve or reject access request:

1. Go to (**Menu**) icon **> SSH+ > Get Started**.
   The access requests are displayed in the **Approvals** section.



2. Based on the requestors and their access rights, you can approve or reject the request.
   On approving the request, the **Access Status** column on the **Terminal Access Control** page turns to:
   - **Accessible:** Status when key provisioning is successful for all hosts in the infra access groups.
   - **Partially Accessible:** Status when key provisioning fails for some hosts in the infra access groups.
   - **Failed:** Status when key provisioning fails for all hosts in the infra access groups.

   On approving an access request, the user is granted access to the specified infra access group and hosts for the requested duration. The users will be able to perform authorized tasks on those resources. On rejecting an access request, the user is denied access to the requested resources.

> **Note:** You can approve or reject access requests from **Access Requests Hub**. For more information, see Access Requests Hub.

Glossary

Term definition

Term	Definition
SSH	Secure Socket Shell (SSH), also known as simply Secure Shell, is a cryptographic protocol used to enable secure access to remote servers and devices over the internet using SSH keys, certificates, or passwords.
SSH key	SSH keys are used to encrypt communication with a remote system. SSH keys usually come in pairs comprising a public and a private key and are used to grant access to authorized personnel to critical systems such as cloud, on-premise servers, and network devices.
Host key	A host key is a key that is used to identify the server. It is generated by the server and shared with the client during the initial connection setup. The

Term definition (continued)

Term	Definition
	client uses this key to verify the identity of the server before establishing a connection.
User key	A user key is a public key that is associated with a particular user account on the host. It is used to authenticate the user and establish a secure connection with the server.
Public key	A public key is used to encrypt data and verify digital signatures. It can be freely distributed, and anyone can use it to encrypt data or verify digital signatures. It is also used to establish a secure connection between the client and the server.
Private key	A private key is a secret key that is used to decrypt data and create digital signatures. It must be kept secret and never shared with anyone. The private key is used to authenticate the user and establish a secure connection with the server.
Suspicious key	A key without a known client association.
Shared key	A key used by more than one user.
Orphan key	A key that is found on a non-standard client file-folder path and does not have a known server.
SSH key rotation	The process of replacing the old key with a new one that adheres to the SSH key policy.
Weak key	A key that is generated using a weaker algorithm and size.

Chapter 2: SSH+ User Guide

- Introduction to SSH+
- System Requirements
- Accessing SSH+ Features
- Discovering Keys
- Managing Devices/Hosts
- Access Requests Hub
- Access Control
- Configuring Provision Settings
- Setting ACL Permissions to Resources
- Adding Infra Access Groups
- Managing Host Key and User Key Inventories
- Dashboard
- Creating Key Policy and Group
- Creating Host Policy and Group
- Glossary

Introduction to SSH+

As application infrastructures grow, so do security threats. Organizations have to find newer ways for protecting their data and granting access to the right users and devices to avoid security threats and breaches. While the traditional approach used password authentication, it proved to be insecure. This is where AppViewX SSH+ comes into play.

AppViewX SSH+ is a fully-automated application infra-access management and SSH key lifecycle management solution that allows you to centrally discover, manage, and protect SSH keys with access across hybrid multi-cloud environments. It also helps simplify access management, stay compliant and mitigate risks with SSH+.

AppViewX offers visibility and SSH access management across traditional on-premises data centers and cloud-hosted infrastructures.

Risks of improper SSH Management

Since there is no governing body to regulate the use of SSH keys, there is an element of risk involved. As SSH keys are generated on a need basis, several keys may be discarded and left unmanaged when they are no longer of use. Without an inventory, managing these keys and revoking their access pose a security threat to large organizations for potential back-door entry into the network, data theft, or breaches.

Improper SSH key management can lead to unauthorized access, compliance violations, identity and access management issues, data breaches, operational disruption, and reputation damage. To mitigate these risks, organizations should implement proper SSH key management practices, including secure key storage, regular key rotation, and access controls.

What Enterprises Need

AppViewX conducted multiple surveys to identify the core features and functionality needed to address SSH management challenges. SSH and Identity and Access Management (IAM) Administrators highlighted the following requirements:

- Discover keys from standard and non-standard locations
- Identify and report non-standard and non-compliant keys
- Visibility of keys and the users of these keys
- Revoke access to non-compliant and non-standard keys
- Rotation and distribution of keys
- Self-service SSH access requests
- Support for cloud and legacy on-premise infrastructure
- Centralized SSH Certificate Authority

How AppViewX Can Help

AppViewX SSH+ key lifecycle management is a fully automated solution that discovers and manages enterprise SSH infrastructure. It can identify and mitigate risks associated with poorly managed passwordless access management.

AppViewX SSH+ features include:

- **Centralized Discovery and Visibility**
  - The solution offers on-demand scans to discover SSH keys across multi-vendor, hybrid network infrastructures, and map trust relationships to determine access privileges.
  - The consolidated inventory provides a central console to view and manage all SSH keys and hosts.
- **Risk Scorecard and One-Click Remediation**

- The solution proactively identifies and remediates risks associated with inactive, weak, orphan, or suspicious keys using an intuitive SSH scorecard dashboard.
- The one-click remediation feature enables instant deletion or regeneration of keys.

System Requirements

SSH within the Application Infrastructure

Application infrastructure refers to all the components required to deliver an application and its functions and services to the customer. Although each application is unique, certain common components can be identified that are typically implemented to support application capabilities and service delivery.

One of the most common components of a typical application infrastructure is Linux hosts.

Modern application infrastructure leverages multiple hosting platforms such as on-premise data centers, private clouds, and public clouds offered by third-party hosting providers.

The application infrastructure components communicate with each other to enable service delivery. This can leverage SSH communication. Additionally, administrators of these application infrastructures will need to SSH into these hosts to perform maintenance.

Additionally, security and audit compliance requirements necessitate constant awareness of who has access to what and the maintenance of best security practices for the communications.

Hardware

Ensure that you have, at minimum, the following hardware with the given specifications before proceeding with the installation:

- **Single Node Deployment Requirements**

Node	Bare Minimum		
	CPU	RAM	Hard Disk Space
Single node	8	32GB	500GB

- **Multi-Node Deployment Requirements**

Node	Bare Minimum		
	CPU	RAM	Hard Disk Space
Multi-node (master node)	4	4GB	100GB
Note: One node for a single master installation and a minimum of three nodes for multi-master installation.			
Multi-node (worker node)	8	32GB	500GB

- **Platform Bare Minimum Requirements**

Supported Virtualization Platforms	Versions	vCPU	RAM	HDD
VM Server, VMware ESXi	5.5 or later	8v	32GB	1TB

Operating System

Both single node and multi-node installations of AppViewX are supported on the following operating systems:

- RHEL 8.5
- RHEL 8.6
- RHEL 8.7
- Ubuntu 20.04

Browser

Following is the browser requirements to use the AppViewX SSH+ node:

Browser	Version
Firefox	v74.0.1 (64-bit) or later
Google Chrome	v85.0.4183.83 (64-bit) or later

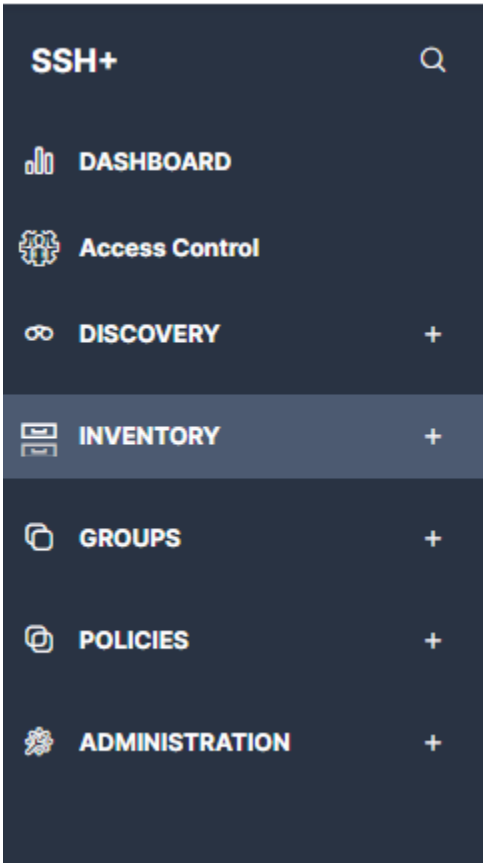
Accessing SSH+ Features

Note: The SSH+ features accessible to you is based on the ACF permissions enabled by the administrator.

You have to access the SSH+ node to access the various functions provided by it.

To access SSH+:

1. Log into AppViewX with valid credentials.
2. Hover the mouse pointer over (**Menu**) icon on the top-left corner of the screen.
3. From the left pane, click **SSH+**. You can now see the different menus of **SSH+** on the left hand side of the page.



4. From the left pane, expand any of the nodes to see that page.

Discovering Keys

- Overview
- Network Scan
- Managed Devices
- Discovery Status
- Scheduler

Overview

Before you begin: You can access this functionality only if you have the ACF permissions enabled for your role.

SSH keys are installed to grant and protect access to privileged accounts. When initially deployed on a device, the device is configured to change privileged account passwords; however, if the devices are deployed after the SSH keys are installed, changing the passwords does not stop SSH keys from working thus rendering the privileged account insecure. To make it secure, these keys must be found so you can remove them and make the accounts secure again.

From the **Discovery** page, you can:

- Discover keys configured by creating and running scans on your network using IP range or subnet option. You can map the discovered keys to the selected key compliance groups and manage/monitor them. See Network Scan.
- Discover keys on the devices you configured by creating and running scans on your devices. If the key already exists in the key inventory under another key group, then only the additional location/filename/filepath details is updated. The keys are not updated to the group that was provided in the discovery creation details. See Managed Devices.
- Fetch the details and the status of the discovery such as the discovery method, action, recurrence, status along with the start and end time. See Discovery Status.
- Create, customize, or delete the scheduler to run the discoveries. See Scheduler.
- Fetch the key discovery status of the user and host keys, risk report, details of the user and host keys and the hosts. See Viewing Discovery Summary.

Network Scan

You can discover user or/and host keys configured on your server by creating and running scans on your network using IP address or subnet. The Discovery scans your network (on the default SSH enabled port 22) for SSH keys configured on your server. You can map the discovered keys to the selected key compliance groups and manage/monitor them.

Note: Only live IPs (hosts) discovered from the IP range/subnet will be reflected in the discovery summary.

• Using IP Range Option
• Using Subnet Option

Using IP Range Option

To discover keys using the IP Range option:

1. Go to (**Menu**) icon **> SSH+ > Discovery > Network Scan > IP Range**.
   The **Discover** page is displayed.
2. Enter the following details:

Field description for Discover IP Range section

Field	Description
Discover By	
***Select**	Select one of the options: • **Instant:** To discover the keys immediately. By default, the **Instant** option is selected. • **Scheduled:** To schedule the discovery of keys on a specific date and time.
Scheduler (This section appears only if you have selected the **Discovery** option as *Scheduled*)	
***Schedule Name**	Enter a unique name. This helps you identify it easily.
Description	Enter details pertaining to the scheduling discovery purpose.
***Starts On**	Under the **Starts On**, set the time to start the run. You can customize the date, month, year, and time by clicking the Calendar icon.
***Repeat Every**	Schedule discovery can be set to repeat discovery after every 5 minutes or can be customized per your requirement.
***End Date**	Select one of the options to end the scheduled discovery:

Field	Description
	• **Never:** To keep the scheduled discovery going. • **On:** To select the end date when the scheduled discovery has to stop. • **After:** To stop the scheduled discovery after a certain number of occurrences.
Discover SSH Keys	
***Discovery Name**	This field appears on selecting the **Instant** discovery option. Enter a unique name. This helps you identify it easily.
Description	This field appears on selecting the **Instant** discovery option. Enter the details pertaining to the discovery stating the purpose.
***Start IP**	Enter the start range of the IP address for discovery.
***End IP**	Enter the end range of the IP address for discovery.
***Ip(S) Per Batch**	Select a value from the dropdown list. Based on this value, the subnet provided is split into multiple batches for the discovery process.
***Ports**	By default, the port is 22. You can enter a port number from where the keys have to be discovered.
***Access Type**	Select **Key** or **Certificate**. **Note:** The Certificate option can be enabled or disabled by configuring the **Enable User CA Trust and Host Certificate** toggle button under Advanced Settings.
*** DataCenter**	Select a datacenter to connect to the host(s).
***Credential Type**	Select one of the options: • **Manual entry:** Enter the username and password. • **Credential List:** Select the credential details that are already stored in the credential inventory page.
***Credential Name**	This field appears only if you have selected **Credential Type** as *Credential List*.
***Login Type**	Select one of the options:

Field	Description
	• **Password:** Enter username and password. • **Identity Key:** Click **Upload** and the **Upload SSH Private Key** window opens. Browse for the key file and fill out all the fields. Enter passphrase.
Sudoer User	Enable this checkbox if you want: • privileges to perform actions on discovery, provisioning, and remediation. • to discover keys for all users configured in the host.
***Access Elevation**	This field appears only on selection of **Sudoer User**.
***Discover**	Select one or both of the options: • **User Keys:** To discover user keys. • **Host Keys:** To discover host keys.
***Application Infra Access Group**	Groups with RW permission will be visible in the **Application Infra Access Group** field. Select the Application Infra Access Group(s) to which you want to map the onboarded host.
Key Compliance Group	Groups with RW permission will be visible in the **Key Compliance Group** field. Select the required Key Compliance Group to which you want to map the discovered user keys. The discovered keys are associated with the selected **Key Compliance Group**. **Note:** The key group selection simplifies the grouping of the discovered keys and checks the discovered keys for key compliance. The keys are checked for compliance based on the policy of the key group it is associated with.
***Scan Type**	Select one of the options:

Field	Description
	• **Default:** The system scans the default ssh folders. • **Full:** The system scans the entire location. You can enter the files name/path that you want to exclude from the discovery for non-standard location. Make sure to select the **Sudoer User** checkbox. • **Directory:** The system performs default scan along with directory scan in the specified directory. Enter the file name/path you want to exclude/include for non-standard location. **Note:** Changing the scan type clears the File Path table.
File Path	This field is enabled only if you select **Full** or **Directory** as your **Scan Type**. Enter the file/s name/path that you want to exclude/include scan (only for directory scan) from the discovery for non-standard location. File path should always start with /.
Operation	This field is enabled only if you select **Full** or **Directory** as your **Scan Type**. Select one of the options: • **Exclude:** Disables the scan in the file/s name/path location entered in **File Path**. • **Include:** Enables the scan only in the file/s name/path location entered in **File Path**. **Note:** Multiple folder/path entries can be entered for scan, which are displayed in the consecutive table with respect to **File Path** and **Operation**.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

3. Click **Add.**

   The **File Path** table is populated with the operation.
4. In **Inventory Action,** select one of the options:

- **Do Not Move:** To avoid the movement of newly discovered keys in the inventory.
- **Manage:** To allow the system to manage the newly discovered keys, which are moved to the inventory with **Managed** status.
- **Monitor:** To allow the system to monitor the newly discovered keys, which are moved to the inventory with **Monitored** status.

5. Click **Discover.**

   The discovery runs per the settings and the key scan instance is added to the discovery inventory with the **Status** as *In Progress* until the discovery is completed. The **Status** in the discovery inventory changes to *Successful* or *Failed* depending on the outcome of the scan.

> **Note:** Only live IPs (hosts) discovered from the IP range will be reflected in the discovery summary.

Using Subnet Option

To discover keys with the subnet option:

1. Go to (**Menu**) icon **> SSH+ > Discovery > Network Scan > Subnet**.
   The **Discover** page is displayed.
2. Enter the following details:

Field description for Discover Subnet section

Field	Description
Discover By	
***Select**	Select one of the options: • **Instant:** To discover the keys immediately. By default, the **Instant** option is selected. • **Scheduled:** To schedule the discovery of keys on a specific date and time.
Scheduler (This section appears only if you have selected the **Discovery** option as *Scheduled*)	
***Schedule Name**	Enter a unique name. This helps you identify it easily.
Description	Enter the details pertaining to the scheduling discovery purpose.
***Starts On**	Under the **Starts On**, set the time to start the run. You can customize the date, month, year, and time by clicking the Calendar icon.

Field	Description
***Repeat Every**	Scheduled discovery can be set to repeat discovery after every 5 minutes or can be customized per your requirement.
***End Date**	Select one of the options to end the scheduled discovery: • **Never:** To keep the scheduled discovery going. • **On:** To select the end date when the scheduled discovery has to stop. • **After:** To stop the scheduled discovery after a certain number of occurrences.
Discover SSH Keys	
***Discovery Name**	This field appears on selecting the **Instant** discovery option. Enter a unique name.
Description	This field appears on selecting the **Instant** discovery option. Enter the details pertaining to the discovery stating the purpose.
***Network**	Enter the IP address of the network. For example, 192.168.1.1/24
***Subnets Per Batch Of Discovery**	Select a value from the dropdown list. Based on this value, the subnet provided is split into multiple batches for the discovery process.
***Ports**	By default, the port is 22. You can enter a port number from where the keys have to be discovered.
***Access Type**	Select **Key** or **Certificate**. **Note:** The Certificate option can be enabled or disabled by configuring the **Enable User CA Trust and Host Certificate** toggle button under Advanced Settings.
***DataCenter**	Select a datacenter to connect to the host(s).
***Credential Type**	Select one of the options: • **Manual entry:** Enter the username and password. • **Credential List:** Select the credential details that are already stored in the credential inventory page.
***Credential Name**	This field appears only if you have selected **Credential Type** as *Credential List*.
***Login Type**	Select one of the options:

Field	Description
	• **Password:** Enter username and password. • **Identity Key:** Click **Upload** and the **Upload SSH Private Key** window opens. Browse for the key file and fill out all the fields. Enter passphrase.
Sudoer User	Enable this checkbox if you want: • privileges to perform actions on discovery, provisioning, and remediation. • to discover keys for all users configured in the host.
***Access Elevation**	This field appears only on selection of **Sudoer User**.
***Discover**	Select one or both of the options: • **User Keys:** To discover user keys. • **Host Keys:** To discover host keys.
***Application Infra Access Group**	Groups with RW permission will be visible in the **Application Infra Access Group** field. Select the Application Infra Access Group(s) to which you want to map the onboarded host.
Key Compliance Group	Groups with RW permission will be visible in the **Key Compliance Group** field. Select the required Key Compliance Group to which you want to map the discovered user keys. The discovered keys are associated with the selected **Key Compliance Group**. **Note:** The key group selection simplifies the grouping of the discovered keys and checks the discovered keys for key compliance. The keys are checked for compliance based on the policy of the key group it is associated with.
***Scan Type**	Select one of the options:

Field	Description
	• **Default:** The system scans the default ssh folders. • **Full:** The system scans the entire location. You can enter the files name/path that you want to exclude from the discovery for non-standard location. Make sure to select the **Sudoer User** checkbox. • **Directory:** The system performs default scan along with directory scan in the specified directory. Enter the file name/path you want to exclude/include for non-standard location. **Note:** Changing the scan type clears the File Path table.
File Path	This field is enabled only if you select **Full** or **Directory** as your **Scan Type**. Enter the file/s name/path that you want to exclude/include scan (only for directory scan) from the discovery for non-standard location. File path should always start with /.
Operation	This field is enabled only if you select **Full** or **Directory** as your **Scan Type**. Select one of the options: • **Exclude:** Disables the scan in the file/s name/path location entered in **File Path**. • **Include:** Enables the scan only in the file/s name/path location entered in **File Path**. **Note:** Multiple folder/path entries can be entered for scan, which are displayed in the consecutive table with respect to **File Path** and **Operation**.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

3. Click **Add.**

   The **File Path** table is populated with the operation.
4. In **Inventory Action,** select one of the options:

- **Do Not Move:** To avoid the movement of newly discovered keys in the inventory.
- **Manage:** To allow the system to manage the newly discovered keys, which are moved to the inventory with **Managed** status.
- **Monitor:** To allow the system to monitor the newly discovered keys, which are moved to the inventory with **Monitored** status.

5. Click **Discover.**

The discovery runs per the settings and the key scan instance is added to the discovery inventory with the **Status** as *In Progress* until the discovery is completed. The **Status** in the discovery inventory changes to *Successful* or *Failed* depending on the outcome of the scan.

Note: Only live IPs (hosts) discovered from the subnet will be reflected in the discovery summary.

Managed Devices

You can discover keys by creating and running scans on your configured devices. The Discovery scans these devices for SSH keys configured on your server. You can map the discovered keys to the selected key compliance groups and manage/monitor them.

To discover keys using managed devices option:

1. Go to (**Menu**) icon **> SSH+ > Discovery > Managed Devices**.
   The **Managed Devices > Discover** page is displayed.
2. Enter the following details:

Field description for Discover Managed Devices section

Field	Description
Discover By	
***Select**	Select one of the options: • **Instant:** To discover the keys immediately. By default, **Instant** option is selected. • **Scheduled:** To schedule the discovery of keys on a specific date and time.
Scheduler (This section appears only if you have selected the **Discovery** option as *Scheduled*)	
***Schedule Name**	Enter a unique name. This helps you identify it easily.

Field	Description
Description	Enter details pertaining to the scheduling discovery purpose.
***Starts On**	Under the **Starts On**, set the time to start the run. You can customize the date, month, year, and time by clicking the Calendar icon.
***Repeat Every**	Schedule discovery can be set to repeat discovery after every 5 minutes or can be customized per your requirement.
***End Date**	Select one of the options to end the scheduled discovery: • **Never:** To keep the scheduled discovery going. • **On:** To select the end date when the scheduled discovery has to stop. • **After:** To stop the scheduled discovery after a certain number of occurrences.
Discover SSH Keys	
***Discovery Name**	This field appears only on selecting the **Instant** discovery option. Enter a unique name. This helps you identify it easily.
Description	This field appears only on selecting the **Instant** discovery option. Enter details pertaining to the discovery stating the purpose.
A list of added and managed devices is displayed. Only devices with status as *Managed* and those that have RW permission are displayed in the list. From the list of managed device(s), select the **Managed Device(s)**. The selected device(s) is the source of discovery. To select all the managed devices, select **Select all**. All the managed devices are the source of discovery. To understand the functionality of Regex, see Using Regex Feature.	
***Ip(S) Per Batch**	Select a value from the dropdown list. Based on this value, the subnet provided is split into multiple batches for the discovery process
***Discover**	Select one or both of the options: • **User Keys:** To discover user keys. • **Host Keys:** To discover host keys.
Key Compliance Group	Select the required **Key Compliance Group** to which you want to map the discovered keys. The discovered keys are associated with the selected **Key Compliance Group**.

Field	Description
	Note: The key group selection simplifies the grouping of the discovered keys and checks the discovered keys for key compliance. The keys are checked for compliance based on the policy of the key group it is associated with.
***Scan Type**	Select one of the options: • **Default:** The system scans the default ssh folders. • **Full:** The system scans the entire location. You can enter the files name/path that you want to exclude from the discovery for non-standard location. • **Directory:** The system performs default scan along with directory scan in the specified directory. Enter the file name/path you want to exclude/include for non-standard location. **Note:** Changing the scan type clears the File Path table.
File Path	This field is enabled only if you select **Full or Directory** as your **Scan Type**. Enter the file/s name/path that you want to exclude/include scan (only for directory scan) from the discovery for non-standard location. File path should always start with '/'.
Operation	This field is enabled only if you select **Full** or **Directory** as your **Scan Type**. Select from the following options: • **Exclude:** Disables the scan in the file/s name/path location entered in **File Path**. • **Include:** Enables the scan only in the file/s name/path location entered in **File Path**. **Note:** Multiple folder/path entries can be entered for scan, which are displayed in the consecutive table with respect to **File Path** and **Operation**.

Field	Description
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

3. Click **Add.**
   The **File Path** table is populated with the operation.
4. In **Inventory Action,** select one of the options:
   - **Do Not Move:** To avoid the movement of newly discovered keys in the inventory.
   - **Manage:** To allow the system to manage the newly discovered keys, which are moved to the inventory with **Managed** status.
   - **Monitor:** To allow the system to monitor the newly discovered keys, which are moved to the inventory with **Monitored** status.
5. Click **Discover.**

   The discovery runs per the settings and the key scan instance is added to the discovery inventory with the **Status** as *In Progress* until the discovery is completed. The **Status** in the discovery inventory changes to *Completed* or *Failed* depending on the outcome of the scan.

• Using Regex Feature

Using Regex Feature

Regex stands for regular expression; it is a string used to define filters. The string can contain a part of the device name or a key scan instance. These expressions are stored in the registry and can be used to select the devices/key scan instances in future. This feature is available for discovering keys using the managed devices.

It enables you to filter your records on the strings mentioned in **Regex**.

1. Go to **Managed Devices > Discover** page.
   The **Discover Managed Devices** page is displayed.
2. On the left side is the list of the devices/key scan instances.
3. In the search bar, enter an expression.
4. Click **Add as regex >>**.

   The expression is added in the list on the right hand side.

Note: These expressions are stored in the registry and can be used to select the devices/key scan instances in future.

You can use this register of expressions (Regex) for all future managed devices.

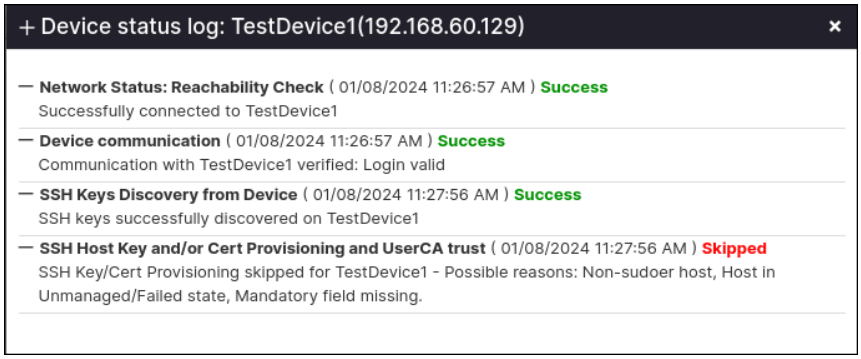
Discovery Status

To discover keys using the discovery status option:

1. Go to (**Menu**) icon **> SSH+ > Discovery > Discovery Status**.
   The **Discovery Status** page is displayed.
2. Displays the following details:

Field description for Discovery Status section

Field	Description
Discovery Name	Displays unique discovery name. This helps you identify it easily. This field is clickable and opens a discovery report for the host with the following discovery status: • **Successful:** Displays all tabs such as Hosts, Host Keys, User Keys, and Risk Report. Hosts can have statuses such as *Managed* or *Monitored*. • **Failed:** Displays only the **Hosts** tab. Each host has a *Failed* status, which is a hyperlink. Clicking it displays the reason for failure in a pop-up window. • **Partial:** Displays all tabs. The **Hosts** tab displays statuses such as *Managed*, *Monitored*, or *Failed*. The *Failed* status is a hyperlink. Clicking it displays the reason for failure in a pop-up window.
Discovery Mode	Displays the mode of discovery (IP range, subnet, or managed devices).
Discover action	Displays the discovery action as **Instant** or **Scheduled**.
Recurrence Type	Displays the frequency of the run of the discovery key instance.
Status	Displays the discovery status which are **Successful**, **Failed**, or **Partial**. You can click the status to open a pop-up window displaying a summary of the discovery results.

Field	Description
	 To check the host status, click the **Discovery Name > Hosts > Host status**.  For more details on the error messages, see Error Messages.
Start Time	Displays the **Start Time**--the time set to start the discovery run. You can customize it if the **Recurrence Type** is one of these: • Daily • Weekly • Monthly • Yearly
End Time	Displays the date and time when the discovery ends.
Description	Displays the details pertaining to the discovery stating the purpose.

What to do next:

- To schedule a discovery using the same input parameters, click **Actions > Rediscover**.
- To delete a discovery, select the checkbox against the **Discovery Name**(s) that has to be deleted and click **Actions > Delete**.

  The selected discovery is deleted from the AppViewX database.

- Viewing Discovery Summary
- Error Messages

Viewing Discovery Summary

To view the discovery summary:

1. Go to (**Menu**) icon **> SSH+ > Discovery > Discovery Status**.

   The **Discovery Status** page is displayed.
2. Click the **Discovery Name** link.

   The **Discovery Summary** of that discovery is displayed.

Summary

The summary report provides information about the discovery of the keys in the default branch. It contains the cumulative results of all successful discoveries.

Discovery summary mainly contains four reports:

- **Key Discovery Summary:** This widget gives a count of the discovered host keys and the user keys.

Color Code	Description
Orange	Displays the number of discovered host keys.
Blue	Displays the number of discovered user keys.

  Clicking the widget redirects you to the **Host Keys/User Keys** tab.

- **Hosts:** This widget gives a count of the existing hosts and the newly discovered hosts.

Color Code	Description
Orange	Displays the number of existing hosts.
Blue	Displays the number of newly discovered hosts.

Clicking the widget redirects you to the **Hosts** tab.

- **User Keys:** This widget gives a count of the newly discovered user keys, missing user keys, and unchanged user keys in the device.

Color Code	Description
Blue	Displays the number of newly discovered user keys.
Yellow	Displays the number of missing user keys.
Green	Displays the number of user keys with no changes.

Clicking the number hyperlink redirects you to the **User Keys** tab.

- **Host Keys:** This widget gives a count of the newly discovered host keys, missing host keys, and unchanged host keys in the device.

Color Code	Description
Blue	Displays the number of newly discovered host keys.
Yellow	Displays the number of missing host keys.
Green	Displays the number of host keys with no changes.

Clicking the number hyperlink redirects you to the **Host Keys** tab.

Hosts

The Hosts tab displays the total number of compliant, non-compliant, weak kex algorithm hosts, weak ciphers, and weak mac algorithm hosts in the host discovery status. Click the number hyperlink to drill down on the metrics.

This helps you monitor the progress of the host discovery efforts, identify the compliance gaps, and prioritize the remediation actions.

Host Keys

The Host Keys tab displays the total number of weak, shared, orphan, and suspicious keys in the key discovery status. Click the number hyperlink to drill down on the metrics.

This helps you track the progress of the key discovery efforts, identify any potential security risks, and prioritize the remediation actions.

User Keys

The User Keys tab displays the total number of weak, shared, orphan, and suspicious keys in the key discovery status. Click the number hyperlink to drill down on the metrics.

This helps you track the progress of the key discovery efforts, identify any potential security risks, and prioritize the remediation actions.

Risk Report

The Risk Report tab contains the same field information as the Dashboard. The only difference is that while Dashboard shows the reports for all the discovered keys and also an option to perform remediation, you can use this page to fetch all the reports for the selected discovery. See Reports.

Error Messages

Error Messages

Device Status	Status	Message
Network Status: Reachability Check	Success	Successfully connected to <deviceName>.
	Failed	Unable to connect to <deviceName>: Host not reachable.
	Failed	Connection to <deviceName> refused. Possible causes: Incorrect IP Address or Port, Service Not Running, Configuration Issues.
	Failed	FQDN <fqdn> resolution failed.
Device Communication	Success	Communication with <deviceName> verified: Login valid.
	Failed	Connection to <deviceName> failed: Invalid login credentials.
	Failed	Connection to <deviceName> failed: Invalid login credentials. <exceptionCatched>.
	Failed	Connection to <deviceName> refused. Possible causes: Incorrect IP Address or Port, Service Not Running, Configuration Issues.
SSH Keys Discovery from Device	Success	SSH keys successfully discovered on <deviceName>.

Error Messages (continued)

Device Status	Status	Message
	Failed	SSH Key discovery on <deviceName> failed. Error encountered during the discovery process: <exceptionCatched during discovery>.
SSH Host Key and/or Cert Provisioning and UserCA Trust	Skipped	SSH Key/Cert Provisioning skipped for <deviceName>. Possible reasons: Non-sudoer host, Host in Unmanaged/ Failed state, Mandatory field missing.
	Failed	SSH Host Key/Cert or UserCA Trust provisioning failed on <deviceName>. Possible reasons: Unable to create certificate for the given key, or unable to create a host certificate. Failures in building provisioning requests for the device.

Scheduler

To discover keys using the Scheduler option:

1. Go to (**Menu**) icon **> SSH+ > Discovery > Scheduler**.
   The **Scheduler** page is displayed.
2. Displays the following details:

Field description for Scheduler section

Field	description
Schedule Name	Displays unique name for the schedule.
Discovery Mode	Displays the mode of discovery (IP range, subnet, or managed devices).
Recurrence Type	Displays the frequency of the run of the scheduled discovery key instance.
Last Execution Time	Displays the date and time of the previous scheduled discovery occurrence details.
Status	Displays the discovery status which are **Completed**, **Scheduled**, and **Paused**.

Field	description
	Note: Scheduled discovery can be paused or resumed by clicking the pause or resume icon before the occurrence of the discovery.
Description	Displays the details pertaining to the discovery stating the purpose.
Next Execution Time	Displays the date and time of the next scheduled discovery occurrence details.

What to do next:

- To modify the scheduled discovery, click **Discovery Status::Scheduler > Modify**.
- To delete a discovery, select the checkbox next to the **Schedule name** that has to be deleted and click **Actions > Delete**.

  The selected discovery is deleted from the AppViewX database.

Managing Devices/Hosts

- Overview
- Adding Credentials
- Host Inventory
- Adding Server
- Adding Cloud
- Actions

Overview

Before you get started: You can access this functionality only if you have the ACF permissions enabled for your role.

You can configure and manage devices (AWS Cloud and Linux servers), enable certificate sync for AppViewX to connect with customer's accounts and discover certificates, enable SSH sync for AppViewX to connect with customer's accounts, and discover host and user keys.

From the **Managing Devices** page, you can:

- Configure devices (AWS Cloud and Linux servers). See Adding Cloud or Adding Server.
- Set user name and passwords to make the devices secure for SSH management. See Adding Credentials.
- Perform action such as export, import, manage, unmanage, or delete a server, or fetch configuration from a server. See Actions.

Adding Credentials

To add credentials for any device:

1. Go to (**Menu**) icon **> SSH+ > Administration > Device Management**.
   The **Device::Server** page is displayed.
2. Select the device from the tabs.
3. Click the  (**Credentials**) icon in the command bar.
   The **Credentials** page is displayed. If credentials are set up, a list of credential names with details is displayed in the table.
4. To add a new credential, click + (**Add**) icon in the command bar.
   The **Add Credential** page is displayed with default credentials fields for AppViewX.
   - To set credentials for **AppViewX**:

Field description for AppViewX Credential Details section

Field	Description
***Credential name**	Enter a suitable credential name.
***User name**	Enter a suitable username.
Credential type (Password)	Select one of the options: • **Password:** Enter password and secondary password. • **Identity key:** Enter identity key and passphrase.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

   - To set credentials for **CyberArk**:

Field description for CyberArk Credential Details section

Field Name	Description
***Credential name**	Enter a suitable credential name
Type	Select one of the options: • **Device:** Enter user name, App ID, and user type. • **Amazon (AWS/ELB):** Enter AWS IAM user name, App ID, and AWS access key ID.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

Note: To configure the API Settings for CyberArk, click the **Cyberark API Settings** button on the right of the screen.

- To set credentials for **Thycotic**:

Field description for Thycotic Credential Details section

Field Name	Description
***Credential name**	Enter a suitable credential name.
***API Profile**	Select the desired API profile from the dropdown list.
Secret Type	Select one of the options: • **Device:** Enter user name. • **Amazon (AWS/ELB):** Enter AWS IAM user name.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

Note: To configure the API Settings for Thycotic, click the **Thycotic API Settings** button on the right of the screen.

Host Inventory

Hosts can be discovered, added, modified, deleted, and decommissioned. The most important feature of this module is discovering the cloud host. Cloud host management helps simplify grouping and access to the host.

- You can discover the hosts that are a part of the AWS cloud.
- When you create access groups based on AWS Tags, it is easier to identify the LDAP user groups associated with the access groups of these hosts.
- The devices can be automatically grouped together based on *AWS Tags* during the run of a cloud host discovery host instance.
- The automatic grouping option can be enabled using a toggle button.
- Alternatively, you can manually create an Access Group by grouping the devices as per your discretion.

- Adding Host
- Viewing Host Inventory
- Actions on Host Inventory

Adding Host

Note: AppViewX SSH+ currently supports only adding servers as hosts.

To add a host:

1. Go to (**Menu**) icon **> SSH+ > Inventory > Host Inventory**.
2. On the command bar, click **+ Add Host**.

   The **SSH+::Host Inventory > Add Host** page is displayed.
3. Enter the following details:

Field description for Add Host section

Field	Description
General information	
***Category**	Select **Server**. Selecting the **Server** option displays the **Port** field.
***Vendor**	Select **Linux**.
***Device Name**	Enter the name of the device. Displays the port used while configuring the device.
Port	This is a non-editable field.

Field	Description
Jump Server Client	By default, this is turned off. Turning on the toggle button allows you to identify the host as a jump server. client. The application infra access groups selected also get mapped to this jump server. client.
***Access Type**	Select **Key** or **Certificate**. **Note:** The Certificate option can be enabled or disabled by configuring the **Enable User CA Trust and Host Certificate** toggle button under Advanced Settings.
***FQDN / IP Address**	Enter the FQDN or the IP address of the host.
***DataCenter**	Select a datacenter to connect to the host.
***Inventory Action**	Select one of the options: • **Do Not Move:** To avoid the movement of newly discovered keys in the inventory. • **Manage:** To allow the system to manage the newly discovered keys, which are moved to the inventory with **Managed** status. • **Monitor:** To allow the system to monitor the newly discovered keys, which are moved to the inventory with **Monitored** status.
***Discover**	Select one or both of the options: • **User Keys:** To discover user keys. • **Host Keys:** To discover host keys.
***Scan Type**	Select one of the options: • **Default:** The system scans the default ssh folders. • **Full:** The system scans the entire location. You can enter the files name/path that you want to exclude from the discovery for non-standard location. Make sure to select the **Sudoer User** checkbox. • **Directory:** The system performs default scan along with directory scan in the specified directory. Enter the file name/path you want to exclude/include for non-standard location. **Note:** Changing the scan type clears the File Path table.
File Path	This field is enabled if you select **Full** or **Directory** as your **Scan Type**.

Field	Description
	Enter the file/s name/path that you want to exclude/include scan (only for directory scan) from the discovery for non-standard location. File path should always start with /.
Operation	This field is enabled if you select **Full** or **Directory** as your **Scan Type**. Select one of the options: • **Exclude:** Disables the scan in the file/s name/path location entered in **File Path**. • **Include:** Enables the scan only in the file/s name/path location entered in **File Path**.  **Note:** Multiple folder/path entries can be entered for scan, which are displayed in the consecutive table with respect to **File Path** and **Operation**.
Click **Add.** The File Path table is populated with the results.	
Credentials	
***Credential Type**	Select one of the options: • **Manual entry:** Enter username and password. • **Credential List:** Select the credential details that are already stored in the credential inventory page.
Login Using	Select one of the options: • **Password:** Enter username and password. • **Identity Key:** Click **Upload** and the **Upload SSH Private Key** window opens. Browse for the key file and fill out all the fields. Enter passphrase.
Sudoer User	Enable this checkbox if you want: • privileges to perform actions on discovery, provisioning, and remediation. • to discover keys for all users configured in the host.
***Access Elevation**	This field appears only on selection of **Sudoer User**.
Assign group	

Field	Description
***Host Compliance Group**	Groups with RW permission will be visible in the **Host Compliance Group** field.
***Application Infra Access Group**	Groups with RW permission will be visible in the **Application Infra Access Group** field. Only users with ACF permission can create an infra access group by entering a name in the text box and pressing Enter. Select the required **Application Infra Access Group** to which you want to map the onboarded host. The onboarded hosts are associated with the selected Application Infra Access Group. The Application Infra Access Group selection simplifies the grouping of the onboarded hosts and checks the onboarded hosts for user compliance. The onboarded hosts are checked for compliance based on the policy of the Application Infra Access Group it is associated with.
***Key Compliance Group**	Groups with RW permission will be visible in the **Key Compliance Group** field.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

4. Click **Create**.

   The host is created in the host inventory.

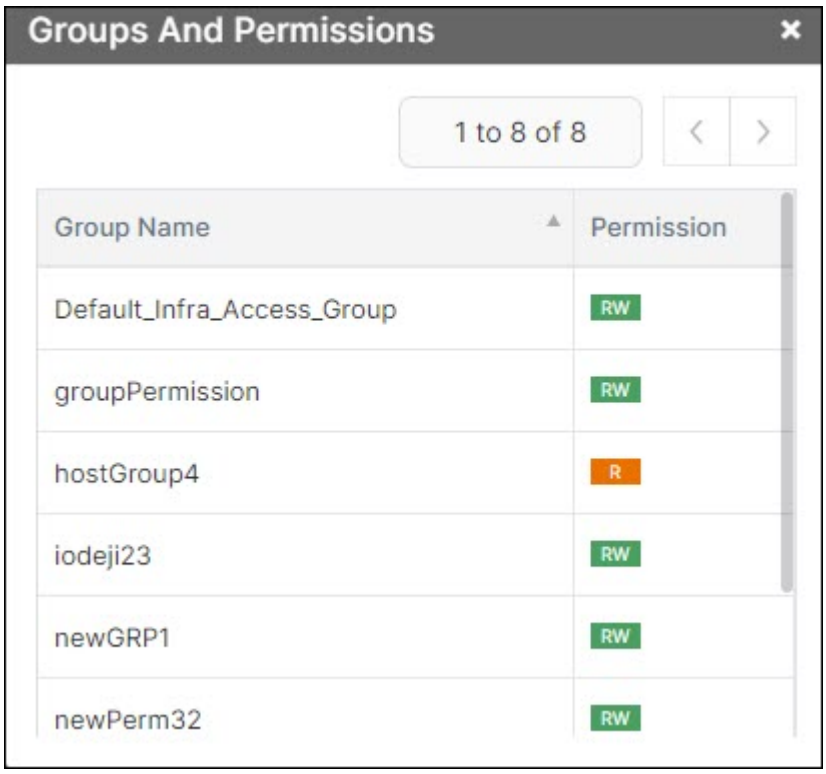
Viewing Host Inventory

The Host Inventory tab displays the total number of hosts with weak kex algorithms, weak ciphers, and weak mac algorithms in the host discovery status. Click the number hyperlink to get the details of the host with weak kex algorithms/weak ciphers/weak mac algorithms. This helps you monitor the progress of the host discovery, identify the compliance gaps, and prioritize the remediation actions.

To view the details of the host inventory:

1. Go to (**Menu**) icon **> SSH+ > Inventory > Host Inventory.**
   The **Host Inventory** page is displayed.

Field description for Host Inventory section

Field	Description
Device name	Displays the unique name provided for the device.
FQDN/IP address	Displays the FQDN/IP address of the host.
Host name	Displays the name of the host.
Host Permission	Displays the count of infra access groups for which access control list (ACL) permissions are enabled for the hosts. Hosts with R (Read) and RW (Read-Write) permissions are only displayed. Clicking the hyperlink opens up a pop-up window displaying the infra access group and the related permissions as shown:  For example, if a host has ACL permissions enabled in eight of the infra access groups, then the count is shown as 8. In this example, the host has six RW and two R permissions enabled in different infra access groups, it is shown as RW 8. This will be the case if a host has only one RW and six R permissions as RW outweighs R permission.
Group	Displays the host compliance group associated with the host.

Field	Description
Allowed Principals	Displays the count of users who have access to a particular host (certificate-based access type) based on the access provided by the administrator using **Access Control.** Clicking the count hyperlink opens the **Host Level Access Settings** window where administrator can remove host access by selecting the users. Click the confirmation message and provide a comment, which is displayed in the audit logs.
Host Status	Displays the status of the host as: • **Managed:** When SSH Host Key and/or Cert Provisioning and UserCA trust operations are completed, it displays Managed , else it displays Managed . • **In-Progress:** Status when host addition/discovery is in progress. • **Failed:** Status when host communication failed. • **Unresolved:** Status when the device is not reachable. Clicking the hyperlink displays the device status.
Jump Server Client	Displays the devices acting as a jump server. client. Client device allows the users to connect to other devices in the infra access group.
Access Type	Displays key or certificate.
User	Displays the name of the user onboarding the device.
Port	Displays the port number that is communicating with the host.
Vendor	Displays the operating system vendor associated with the host.
Last sync time	Displays the last sync date and time with the host.
Instance Id	Displays the ID automatically generated by AWS when you launch a new EC2 instance. **Note:** This is applicable only for EC2 instances discovered by adding the AWS cloud account.

Actions on Host Inventory

You can perform any of following actions by selecting the checkbox against the host name and clicking the **Actions** dropdown menu on the **Host Inventory** page:

Action description on Host Inventory page

Action	Description
Modify	Only users with RW permission can edit the selected host details.
Delete	Only users with RW permission can delete the active hosts from the host inventory.
Credentials	You can view the credentials of the host keys that are discovered on the device. You can also add, modify, or delete the credentials. See Adding Credentials. You cannot delete the default or active credentials.
Fetch Keys	Only users with RW permission can re-fetch all the keys from the host.
Export	You can export hosts from the host inventory to .csv or .xls format.
Rotate Host Certificate	**Important:** • Only users with RW permission can perform this action. • This action is enabled only for devices/hosts with host status as *Managed* with sudo access. • If the *global known host* is not present, then AppViewX will create one with the right CA definition. Rotate the host key along with the host certificate from the host inventory to update the public host key in the known hosts. A confirmation pop-up window appears warning about the implications of the rotation process. On confirmation, the existing host certificate is deleted and a new one is created in its place for the selected host in the *global known host* file.

Adding Server

Note: AppViewX SSH+ currently supports addition of only Linux servers.

To add a server:

1. Go to (**Menu**) icon **> SSH+ > Administration > Device Management**.

   The **Device::Server** page is displayed.

2. On the command bar, click + (**Add**) icon to add a new server.

   The **Device::Server > Add** page is displayed.

3. Select **Linux** from the **Vendors** list.
4. Enter the following details:

Field description for Device Details section

Field	Description
Server details	
***Server name**	Enter a unique name for the server. This helps you identify it easily.
***IP address/FQDN**	Enter the IP address/FQDN.
Data center	Select a data center from the dropdown list.
Communication mode	Select SSH.
***SSH Port**	By default, the port is 22. You can choose to enter a port number.
Cert sync	Select one of the options: • **Managed:** AppViewX connects to the customer's server account and discovers host and user keys. These keys are added to the host and key inventory. Users with the relevant permissions can then perform the required keys-related actions. • **Monitored:** AppViewX connects to the customer's server account and discovers host and user keys. These keys are added to the host and key inventory where the users are allowed to only view the keys. • **Ignored:** Certificate discovery is ignored.
Credentials	
***Credential Type**	Select one of the options: • **Manual entry:** Enter the username and password. • **Credential List - AppViewX:** Select the credential details that are already stored in the credential inventory page. • **SSH:** Enter the username, browse and upload the identity key along with its passphrase.
Service account credentials	

Field	Description
Username	Enter the user name.
Password	Enter the password.
Vendor Specific Details	
Access Elevation	By default, the value is **None**. Select a value from the dropdown list.
Discover Formats	Enter a value to filter the formats to be discovered from the device. By default, all standard formats are discovered.
Certificate details	
Certificate Directory	Provide the directory from where the certificates must be discovered. By default, the system scans for certificates from all the directories.
Scan type	Select one of the options: • **Default:** The system scans for supported certificate formats such as pem, crt, cer, der, kdb, jks, p12, p7, pfx, and adds them to the certificate inventory. • **Aggressive:** The system scans for all keystore files with non-standard extensions.
***Operation**	Select one the options: • **Exclude:** Disables the scan in the specified certificate directory. • **Include:** Enables the scan only in the specified certificate directory.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

5. Scroll down to the **SSH Details** section. By default, the **SSH Sync Enable** toggle button is turned off.
6. Click the **SSH Sync Enable** toggle button to enable SSH sync.
7. Click **Customise** to modify the default settings.
8. Enter the following fields:

Field description for SSH Details section

Field	Description
***Inventory Action**	Select one of the options:

Field	Description
	• **Do Not Move:** To avoid the movement of newly discovered keys in the inventory. • **Manage:** To allow the system to manage the newly discovered keys, which are moved to the inventory with **Managed** status. • **Monitor:** To allow the system to monitor the newly discovered keys, which are moved to the inventory with **Monitored** status.
***Discover**	Select one or both of the options: • **User Keys:** To discover user keys. • **Host Keys:** To discover host keys.
Scan Type	Select one of the options: • **Default:** The system scans the default ssh folders. • **Full:** The system scans the entire location. You can enter the files name/path that you want to exclude from the discovery for non-standard location. • **Directory:** The system performs default scan along with directory scan in the specified directory. Enter the file name/path you want to exclude/ include for non-standard location. **Note:** Changing the scan type clears the File Path table.
File Path	This field appears if you select **Full** or **Directory** as your **Scan Type**. Enter the file/s name/path that you want to exclude/include scan (only for directory scan) from the discovery for non-standard location. File path should always start with '/'.
Operation	This field appears if you select **Full** or **Directory** as your **Scan Type**. Select one of the options: • **Exclude:** Disables the scan in the file/s name/path location entered in **File Path**. • **Include:** Enables the scan only in the file/s name/path location entered in **File Path**.

Field	Description
	Note: Multiple folder/path entries can be entered for scan, which are displayed in the consecutive table with respect to **File Path** and **Operation**.
***Host Compliance Group**	Groups with RW permission will be visible in the **Host Compliance Group** field. Select the required Host Compliance Group to which you want to map the discovered hosts and host keys. The discovered hosts and host keys are associated with the selected host compliance group.
***Key Compliance Group**	Groups with RW permission will be visible in the **Key Compliance Group** field. Select the required Key Compliance Group to which you want to map the discovered keys. The discovered keys are associated with the selected key compliance group. The key group selection simplifies the grouping of the discovered keys and checks the discovered keys for key compliance. The keys are checked for compliance based on the policy of the key group it is associated with.
***Application Infra access group**	Groups with RW permission will be visible in the **Application Infra Access Group** field. Only users with ACF permission can create an infra access group by entering a name in the text box and pressing Enter. Select the Application Infra Access Group(s) to which you want to map the onboarded host.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

9. Click **Save**.

   The host is created and displayed in the host inventory.

   Multiple devices can be configured for the same vendor.

What to do next:

- To add credentials to the server, see Adding Credentials.
- To perform any of the actions such as export, import, manage, unmanage, or delete a server, or fetch configuration from a server, see Actions.

Adding Cloud

Note: AppViewX SSH+ currently supports addition of only AWS cloud devices.

1. Go to (**Menu**) icon **> SSH+ > Administration > Device Management**.
   The **Device::Server** page is displayed.
2. Click the **Cloud** tab.
   The **Device::Cloud** page is displayed.
3. On the command bar, click + (**Add**) icon to add a new cloud device.
   The **Device::Cloud > Add** page is displayed. By default, AWS is selected from the **Vendors** list.
4. Enter the following fields:

Field description for AWS Device Details section

Field	Description
Basic Information	
***Account Type**	Select **Cross or Federated** to authenticate using the assumed role.
***Account Name**	Enter a unique name. It cannot be an account name that is already in the cloud inventory. Name can be alphanumeric and contain hyphen (-) and period (.).
***Account Number**	Enter a valid AWS account number.
Account Description	Enter a description that helps identify your account from the cloud inventory.
Proxy Required	Select the checkbox if you want to create it as a proxy.
***Default Region**	Select the region from the dropdown list for API communication.
***Data Center**	Select a datacenter to connect to the host.
Credentials	
***Credential type**	Select one of the options:

Field	Description
	• **Manual Entry:** Enter username and password. • **Credential List:** Select the credential details that are already stored in the credential inventory page.
***Access Key ID**	Enter the access key ID.
***Secret Access Key**	Enter the secret access key. **Note:** To authenticate requests, use both the access key ID and the secret access key.
Discover resources	
Auto Discover Resources	By default, this is turned off. Turn on the toggle button to discover all cross or federated/child accounts of the provided master account details.
Advanced Settings	By default, this is turned off. Turn on the toggle button to customize the auto-discovery process.
***Auto Discovery Mode**	Select one or both of the options.
***Service**	Select **EC2 (EC2 instance)** from the dropdown list.
***Service Region**	Click **Fetch Region** to fetch the service regions for the provided account information.
Cert Sync	Select one of the options: • **Managed:** AppViewX connects to the customer's server account and discovers host and user keys. These keys are added to the host and key inventory. Users with the relevant permissions can then perform the required keys-related actions. • **Monitored:** AppViewX connects to the customer's server account and discovers host and user keys. These keys are added to the host and key inventory where the users are allowed to only view the keys. • **Ignored:** Certificate discovery is ignored.
Auto Sync	By default, this is turned off. Turn on the toggle button to auto sync based on trigger or schedule.
EC2 Services	
Communication mode	Keep the default selection.

Field	Description
Certificate Discovery Mode	Keep the default selection.
***S3 Deployment Type**	Enter the S3 deployment type that can be a centralized S3 bucket.
***S3 Bucket Name**	Click the **Settings** icon and fill out the mandatory fields in the **ARN Advanced Settings** window that pops up.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

5. Scroll down to the **SSH Details** section. By default, the **SSH Sync Enable** toggle button is turned off.
6. Click the **SSH Sync Enable** toggle button to enable SSH sync.
7. Click **Customise** to modify the default settings.
8. Enter the following fields:

Field description for SSH Details section

Field	Description
***Inventory Action**	Select one of the options: • **Do Not Move:** To avoid the movement of newly discovered keys in the inventory. • **Manage:** To allow the system to manage the newly discovered keys, which are moved to the inventory with **Managed** status. • **Monitor:** To allow the system to monitor the newly discovered keys, which are moved to the inventory with **Monitored** status.
***Host Compliance Group**	Groups with RW permission will be visible in the **Host Compliance Group** field. Select the required Host Compliance Group to which you want to map the discovered hosts and host keys. The discovered hosts and host keys are associated with the selected host compliance group.
***Key Compliance Group**	Groups with RW permission will be visible in the **Key Compliance Group** field. Select the required Key Compliance Group to which you want to map the discovered keys. The discovered keys are associated with the selected key compliance group.

Field	Description
	The key group selection simplifies the grouping of the discovered keys and checks the discovered keys for key compliance. The keys are checked for compliance based on the policy of the key group it is associated with.
Scan Type	Select one of the options: • **Default:** The system scans the default ssh folders. • **Full:** The system scans the entire location. You can enter the files name/path that you want to exclude from the discovery for non-standard location. • **Directory:** The system performs default scan along with directory scan in the specified directory. Enter the file name/path you want to exclude/include for non-standard location. **Note:** Changing the scan type clears the File Path table.
***Discover**	Select one or both of the options: • **User Keys:** To discover user keys. • **Host Keys:** To discover host keys.
File Path	This field appears if you select **Full** or **Directory** as your **Scan Type**. Enter the file/s name/path that you want to exclude/include scan (only for directory scan) from the discovery for non-standard location. File path should always start with '/'.
Operation	This field appears if you select **Full** or **Directory** as your **Scan Type**. Select one of the options: • **Exclude:** Disables the scan in the file/s name/path location entered in **File Path**. • **Include:** Enables the scan only in the file/s name/path location entered in **File Path**. **Note:** Multiple folder/path entries can be entered for scan, which are displayed in the consecutive table with respect to **File Path** and **Operation**.

Field	Description
	Note: Fields indicated with red asterisk (*) symbol are mandatory.

What to do next:

- To add credentials to the cloud device, see Adding Credentials.
- To perform any of the actions such as export, import, manage, unmanage, or delete a server, or fetch configuration from a server, see Actions.

Actions

To do any of the following actions on the command bar, select the checkbox against the device.

Action Descriptions on Command Bar

Icon	Action	Description
	Terminal connect	Click to open the terminal connect page of the device.
	Delete	Click **Yes** on the confirmation popup window. The device is deleted from the inventory.
	Credential	Click to add credentials for the server. See Adding Credentials.
	Manage	Click **Yes** on the confirmation popup window. Configuration fetch is triggered, and the device status is changed to *Managed* in the device inventory. **Note:** With the introduction of the Cert License Enforcement; Server Auto discovery of EC2 instances will be on-boarded with certificate status as *Monitored*, even if *Managed* state is configured in the device account configuration.
	Unmanage	Click **Yes** on the confirmation popup window. Configuration fetch is triggered, and the device status is changed to *Unmanaged* in the device inventory.

Action Descriptions on Command Bar (continued)

Icon	Action	Description
	Export	Select one of the options: • **All Columns:** Select this option if you want to export all information about the device. • **Displayed columns:** Select this option if you want to export only the information that is visible on the screen. • **Columns to modify data and import:** Select this option if you want to export device details to make modifications and then import the data into the device inventory. The data is exported to an Excel (.xls) format.
	Import	1. Click **Import** to be redirected to the import cloud page. 2. Download the sample <.csv> or <.xls> file. 3. Update the details. 4. Click to browse and upload the files. The Cloud details are updated in the cloud inventory.
	Fetch	A popup message, **Fetch config has been triggered for the device** appears. The configuration is fetched from the device.
	Device Settings	Click to change the **Certificate details** section.

Access Requests Hub

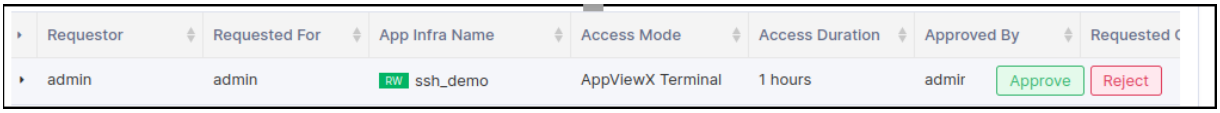
Any user with administrator privileges can use access request hub if you have ACF permissions enabled. This is a full-featured tool for access request management enhancing security governance, operational efficiency, and compliance through detailed oversight and control of user access within the system.

1. Go to (**Menu**) icon **> SSH+ > Manage Access > Access Requests Hub**.

   The **Access Requests Hub** page is displayed.

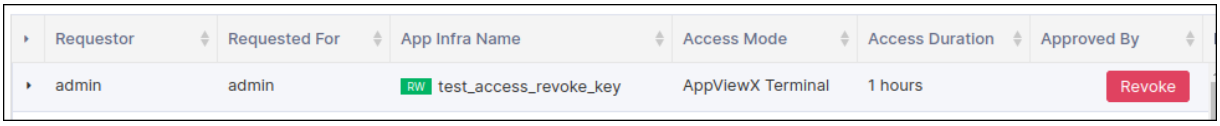
You can view the details of the access requests such as requestor, requested for, app infra name, access mode, access duration, approver, and requested date.

2. You can approve or reject requests that are in *Pending Approval* status by hovering over the access request to see the **Approve or Reject** button as shown.



On approving the request, the **Access Status** column changes to **Accessible**, and the **Approve or Reject** button changes to **Revoke**.

3. You can revoke access requests that are in **Accessible** status by hovering over the access request to see the **Revoke** button as shown.



4. You can use the **Search** option to search access requests by requester, requested for, app infra name, access mode, access duration, and approver.
5. You can sort and filter access requests by clicking the hyperlink on the top panel that displays the number of access requests by their statuses such as accessible, partial, pending, denied, expired, failed, and revoked. Clicking the hyperlink fetches those access requests and displays details of the access requests.

Access Control

• Overview
• Requesting Access to Terminals
• Approving Access Requests
• Viewing Terminal Access Control Page
• Accessing Host Terminals

Overview

Before you begin: You can access this functionality only if you have the ACF permissions enabled for your role.

With access control, users with RW permission can access terminals to manage and monitor all the hosts on your network from a single platform to perform various tasks such as running scripts, executing commands, and troubleshooting issues.

You can access terminals using the **Open with Password** option if you know the password or click the **Name** hyperlink of the infra access group once you have been granted access to the group (See Requesting Access to Terminals).

Requesting Access to Terminals

Before you begin: Only users with RW permission can access hyperlinks to infra access groups for which you can request access.

To request access to terminal:

1. Go to (**Menu**) icon **> SSH+ > Access Control**.
   The **Terminal Access Control** page is displayed.
2. Click the **Name** hyperlink of the infra access group to which you want to request access.
   The access request form is displayed.
3. Enter the following fields:

Field description for Access Request section

Field	Description
***Access Mode**	Depending on your ACF permissions, select one of the options: • **AppViewX Terminal:** Select this option if you are selecting only the infra access group. • **Jump Server: Client:** Select this option if you have mapped the infra access group to a jump server client or a set of jump server. clients. The jump servers clients are populated in the **Device name** box. You can search or select the ones you want access to. • **Unmanaged Clients:** Select this option to request access for a client not managed by AppViewX. You can select between generating keys via AppViewX or uploading your own public keys thus ensuring secure and customized access to unmanaged clients. On clicking **Download**, the SSH keys are downloaded along with the host CA trust certificate in the known host file. If the known host file exists, then copy the content from the downloaded file into it; in case the

Field	Description
	known host file is not present, then copy and paste the known_host file itself. The download option has a disabled PEM format field and an optional password field when AppViewX is selected for key generation. The signed keys are downloadable as a PEM file from the Infra Access Group inventory as long as the access request is valid. AppViewX signs the user-uploaded key with a certificate validity for the requested access duration. The user can choose to protect the downloaded keys (zip file) with a password or leave them unprotected. The download option is disabled when the PEM format field without a password for the upload key is selected.
***Application Infra Name**	This is the infra access group for which you are requesting access. This is a non-editable field.
***Access For**	Select one of the options: • **Self** if you are requesting access for yourself • **User** for requesting access for another user. On selecting this option, the **User** field is displayed. Enter a user name or select user from the dropdown list.
***Available Jump Server(s) Client(s)**	This field is displayed on selecting the **Jump Server Client** option. Select one or more jump servers clients from the section. Click the **Refresh** icon to see the latest list.
***Access Duration Type**	Select **Hours** or **Days**.
***Access Duration Value**	You can key in a value in the text field after selecting **Hours** or **Days** option. **Note:** The access duration kicks in from the time the administrator grants access to the time requested. Once the access duration elapses, the access is revoked.
Comments	Enter the reason for requesting access to the infra access group.
***Key Generation**	This field is displayed on selecting the **Unmanaged Clients** option.

Field	Description
	You can select between generating keys via AppViewX or uploading your own public keys thus ensuring secure and customized access to unmanaged clients.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

4. Click **Request Access.**
   A message, *Your access request was sent successfully for approval; once request is approved, you can access SSH terminal* appears. The Access Status changes to *Pending Approval.* Once the administrator approves the request, the Access Status changes to *Approved*; else, it changes to *Access Denied.*

What to do next

You can directly access the terminal by clicking the **Name** hyperlink of the infra access group. See Accessing Host Terminals.

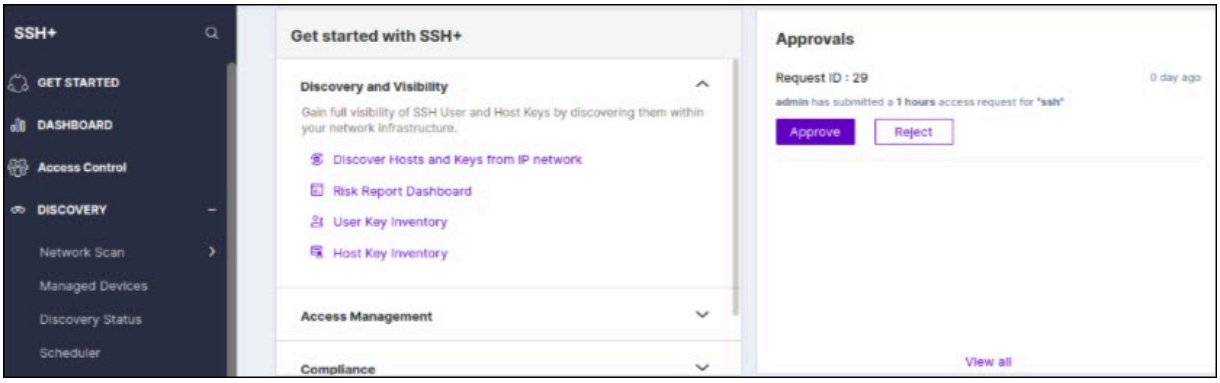
Approving Access Requests

Note: To approve or reject access requests, the ACF permissions must be enabled by the administrator.

All requests for terminal access are displayed on the **Get Started** page in the **Approvals** section.

To approve or reject access request:

1. Go to (**Menu**) icon **> SSH+ > Get Started**.
   The access requests are displayed in the **Approvals** section.



2. Based on the requestors and their access rights, you can approve or reject the request.
   On approving the request, the **Access Status** column on the **Terminal Access Control** page turns to:
   - **Accessible:** Status when key provisioning is successful for all hosts in the infra access groups.
   - **Partially Accessible:** Status when key provisioning fails for some hosts in the infra access groups.
   - **Failed:** Status when key provisioning fails for all hosts in the infra access groups.

   On approving an access request, the user is granted access to the specified infra access group and hosts for the requested duration. The users will be able to perform authorized tasks on those resources. On rejecting an access request, the user is denied access to the requested resources.

Viewing Terminal Access Control Page

To view the Terminal Access Control page:

Go to (**Menu**) icon **> SSH+ > Access Control**.
The **Terminal Access Control** page is displayed.

Field description for Terminal Access Control page

Field	Description
Name	Displays infra access groups based on your RW permission. Click the hyperlink to request access to the group. See Requesting Access to Terminals.
Host(s) Count	Displays the count of the hosts associated with the infra access group for which you have access.
Access Status	Displays the access status of the infra access group: • **N/A:** Initial status when you do not have access to the infra access group. • **Pending Approval:** Status when you have sent an access request and are awaiting approval from the administrator. When the administrator approves access to the infra access group, the status is set to **Approved**; else, it is marked as **Access Denied**. • **Provisioning In Progress:** Status when key provisioning begins for all hosts in the infra access groups. • **Accessible:** Status when key provisioning is successful for all hosts in the infra access groups.

Field	Description
	• **Partially Accessible:** Status when key provisioning fails for some hosts in the infra access groups. • **Failed:** Status when key provisioning fails for all hosts in the infra access groups. • **Expired:** Status when your access to the group is expired for the requested duration. You can request access for the same group again by raising an access request. Clicking the **Access Status** link against an infra access group displays a pop-up message.
Logs	Click **View** to open the log. The log displays details about the user who has access to the infra access group, access mode (whether SSH if it was accessed after granting approval or credential if password was used to access the terminal), status (whether access is active or expired), when the access request was initiated, when the access was terminated, and for how long the access was granted.

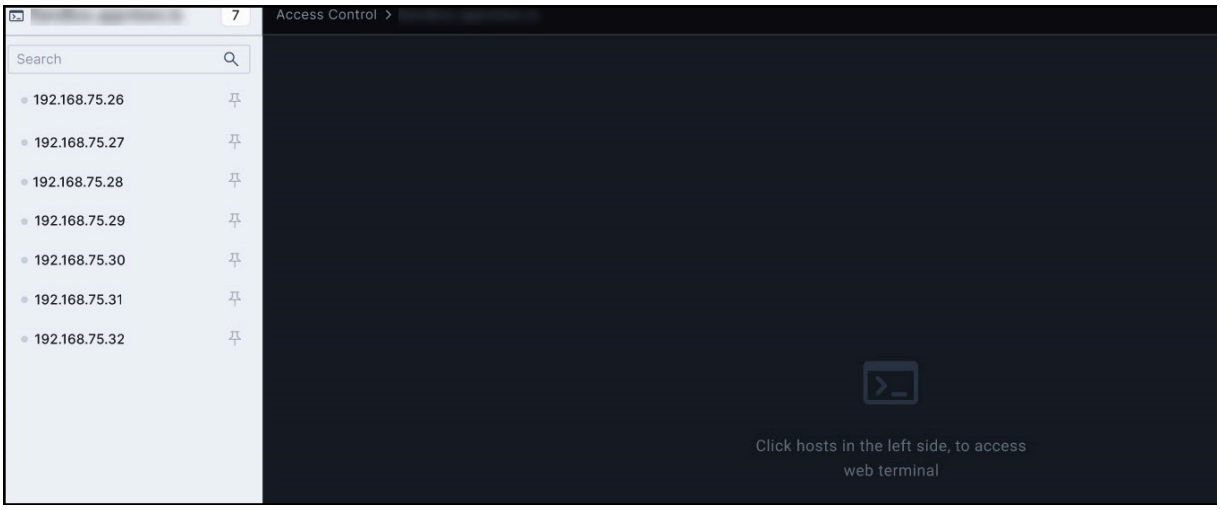
Accessing Host Terminals

You can access the host terminals in any of the following ways:

- If you have the password to the infra access server, then click the **Open with Password** option on the **Access Request** page.
- If you do not have the password to the infra access server, then request access as described in Requesting Access to Terminals. Once access is granted, you can:
  - Click the **Name** hyperlink of the infra access group.
  - Access the terminals from the **Device Management** page or the **Host Inventory** page by clicking the  (**Terminal Connect**) icon.

To access the host terminal:

1. Go to (**Menu**) icon **> SSH+ > Access Control**.
   The **Terminal Access Control** page is displayed.
2. Click the **Name** hyperlink of the infra access group.
   The terminal page opens with all the associated hosts as shown.



3. Click any of the hosts to access the web terminal.
4. Enter your credentials.
   You can now perform various tasks such as running scripts, executing commands, and troubleshooting issues.

Note:

- You can access more than one host at a time within each infra access group. On selecting multiple hosts, the hosts open in the split screen view. You can alternate between split or tab view by clicking using ( ) icon on the RHS of the page.
- You can pin only four terminals for better access and management using the (**Pin**) icon.

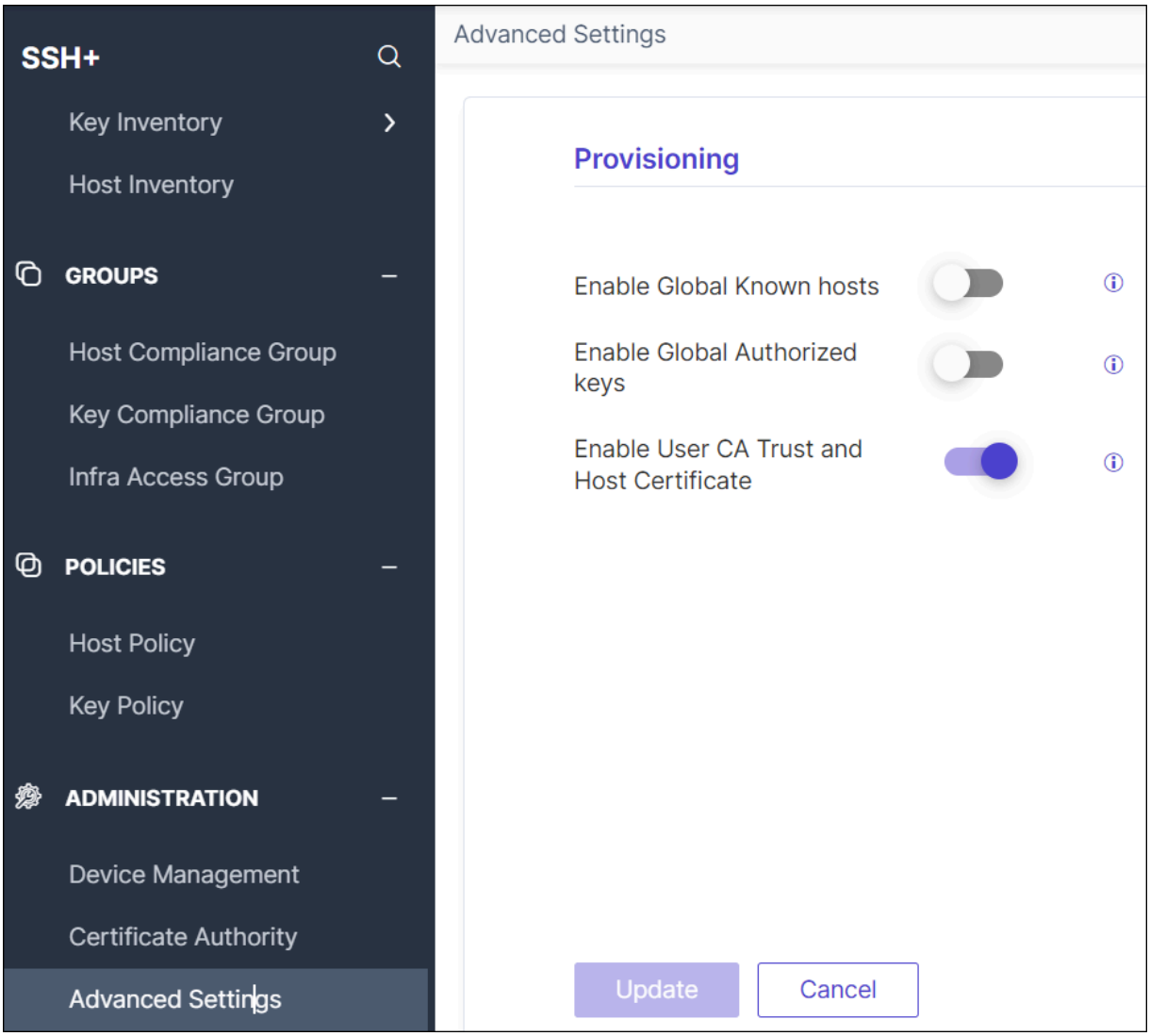
Configuring Provision Settings

You can gain comprehensive control over key provisioning with the **Provisioning** feature, which provides following options:

- **Provision to Enable Global Known hosts**
- **Provision to Enable Global Auth keys**
- **Provision to Enable User CA Trust and Host Certificate**

Procedure

1. Go to (**Menu**) icon **> SSH+ > Administration > Provisioning**.
   The **Provisioning** page is displayed.



2. Enable the **Enable Global Known hosts**. By default, this option is disabled. Enabling this option allows to generate a Global Known Hosts configurtaion, ensuring that all new keys/certificates generated during access requests, provisioning, or remediation activities (such as key rotation) are added to the Global Known Hosts repositories.
3. Enable the **Enable Global Authorized keys**. By default, this option is disabled. Enabling this option allows to establish a Global Auth Keys configurtaion, ensuring that all new keys/certificates generated during access requests, provisioning, or remediation activities (such as key rotation) are added to the Global Known Keys repositories.
4. Enable the **Enable User CA Trust and Host Certificate**. By default, this option is enabled. This option allows certificate based authentication for the devices.
5. Click **Update**. A message, *Provisioning settings saved successfully*, appears.

Setting ACL Permissions to Resources

You can also assign/unassign resources access control list (ACL) permissions for infra access groups and key compliance groups. Depending on the ACL permissions set, resources will have partial, full, or no access to the groups and related actions.

- Setting ACL Permissions to Infra Access Groups
- Setting ACL Permissions to Key Compliance Groups
- Adding or Deleting Regex Patterns

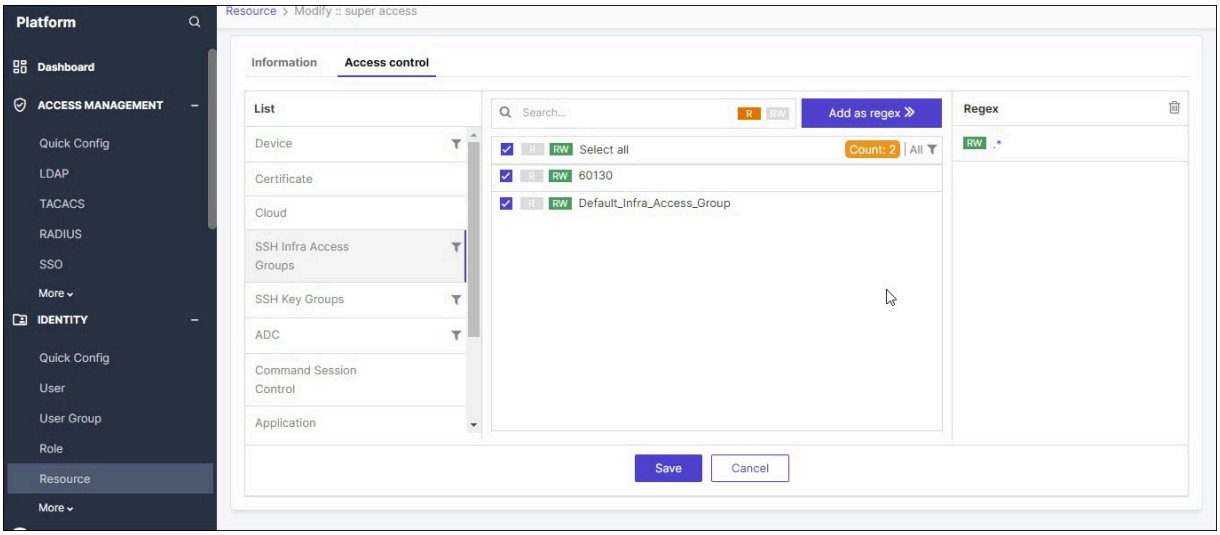
Setting ACL Permissions to Infra Access Groups

You can assign/unassign resources the following access control list (ACL) permissions to infra access groups:

- **RW** denotes that users have *Read-Write* permission to the infra access group along with hyperlinks to request access to the group on the **Access Control** page. Users can modify or delete infra access groups.
- **R** denotes that users only have *Read* permissions to the infra access groups. Users cannot request access or modify or delete infra access groups.
- If neither R nor RW permission is assigned, then users will not be able to even view the infra access group.

To assign ACL permission to infra access groups:

1. Go to (**Menu**) icon **> Platform > Identity > Resource**.
2. Click a resource name.
3. Click **Access Control** tab and select **SSH Infra Access Groups** from the list.



The infra access groups are displayed on the right.

4. Select the infra access group for which you want to assign permission by clicking the check box.
5. Click **R** or **RW** button to assign the read or read-write permission.
6. Click **Save**.

   A message, *Successfully assigned/unassigned SSH groups data for resource super access*, appears.

> **Note:** On deleting a resource, all ACL permissions from the associated infra access groups are also removed.

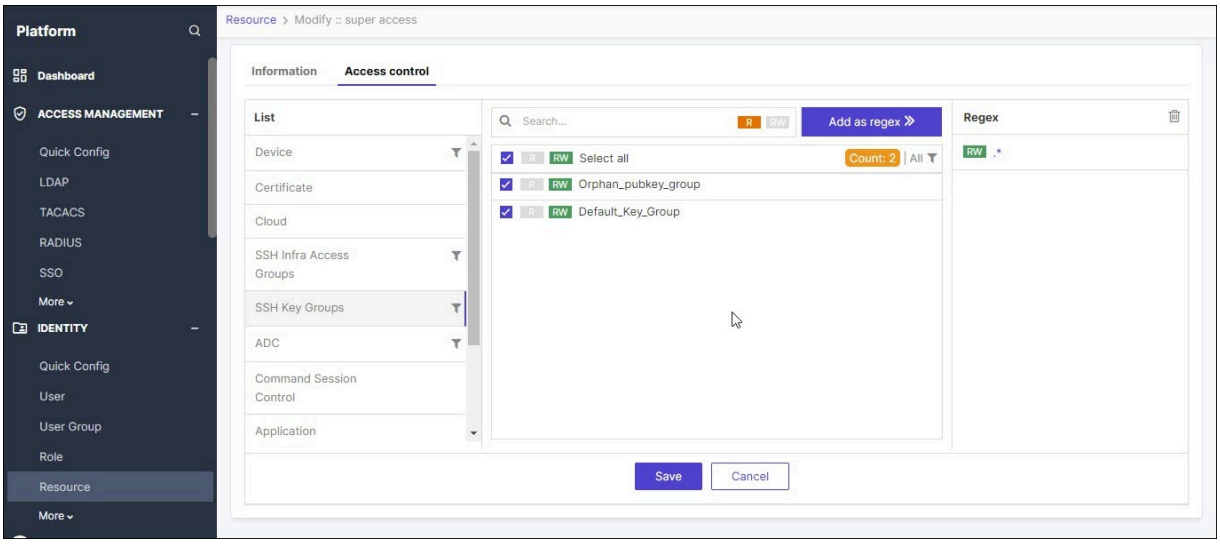
Setting ACL Permissions to Key Compliance Groups

You can assign/unassign resources the following access control list (ACL) permissions to key compliance groups:

- **RW** denotes that users have *Read-Write* permissions to the key compliance groups. Users can modify and delete key compliance groups. Users can view the keys associated with the key compliance groups and perform any of the actions in the inventory.
- **R** denotes that users have *Read* permissions to only view details displayed on the **Key Compliance Group** page. Users can only view the keys associated with the key compliance groups but cannot perform any action in the inventory.
- If R or RW permission is not assigned, then users will not be able to view the key compliance groups and the keys associated with them.

To assign ACL permission to key compliance groups:

1. Go to (**Menu**) icon **> Platform > Identity > Resource**.
2. Click a resource name.
3. Click **Access Control** tab and select **SSH Key Groups** from the list.



The key groups are displayed on the right.

4. Select the key group for which you want to assign permission by clicking the check box.
5. Click **R** or **RW** button to assign the read or read-write permission.
6. Click **Save**.

   A message, *Successfully assigned/unassigned SSH groups data for resource super access*, appears.

> **Note:** On deleting a resource, all ACL permissions from the associated key compliance groups are also removed.

Adding or Deleting Regex Patterns

With Regex (Regular Expression) support, you can dynamically map user groups to infra access groups and key groups within the SSH+ module. This feature allows for the automatic mapping of newly created infra access groups and key groups that match specified Regex patterns, thus streamlining the process and reducing manual overhead.

To add or delete Regex patterns:

1. Go to (**Menu**) icon **> Platform > Identity > Resource**.
2. Click a resource name.

3. Click **Access Control** tab and select **SSH Infra Access Groups** or **SSH Key Groups** from the list.
4. Type a regex pattern in the **Search** box and click **Add as regex**.

   Examples of regex pattern include:

   • .*<text>

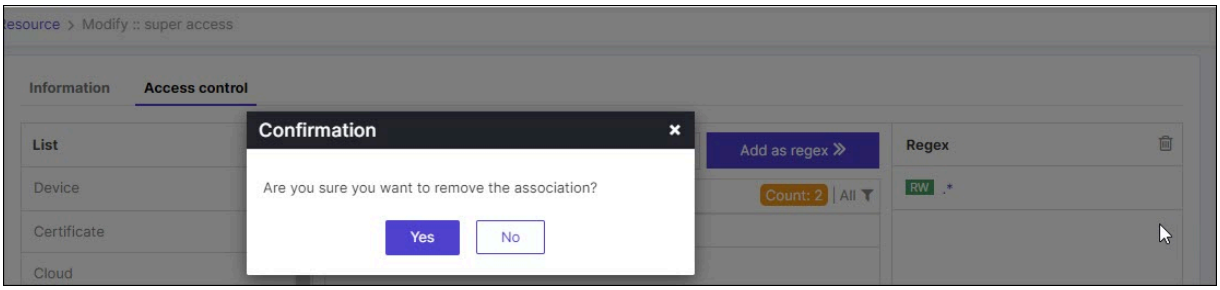
   • <text>.*

   • <text>.*<text>

   • .*

   where <text> can be alphabets, numbers, and special characters to create these entries in the pages.

   A message that the Regex pattern is successfully added for admin appears. Based on the matched regex, permission is automatically added to that resource. The newly added regex appears under the Regex column on the RHS of the page.
5. To individually delete a regex pattern, select the regex pattern from the **Regex** column and click **X** against the name. To delete all regex patterns, click  (Delete) icon. A confirmation message appears as shown.



   Click **Yes** and a message that it is successfully deleted from the matching groups appears.

Adding Infra Access Groups

• Infra Access Group

• Adding Infra Access Group

• Removing Hosts from Infra Access Group

• Viewing Infra Access Group

Infra Access Group

- Creating an access group lets you enable access to a user or a group of users to all the hosts in the access group with a single instruction/selection.
- A host can be associated with one or more access groups. A host can remain unassociated with an access group as well.
- You can create an Infra Access Group by two different methods:
  - **Auto-create during the cloud host scan**

    Auto-created access groups are created automatically by reading the AWS tags of the host during cloud host discovery. The system forms the groups dynamically based on these AWS tags. The cloud host discovery does a periodic scan. The system detects any tag change that may occur due to an action on the device. This change could be due to some action performed by the cloud administrator. Once the system identifies the change in tags of the host, it switches the host to the Infra Access Group of the host based on its new tag. The system also ensures withdrawal of access that was provisioned to the host previously. If this change causes the device to associate with an existing auto-created access group, the other devices in the new access group are automatically provisioned with access to this device.
  - **Manually created through the Infra Access Group tab**

    Manually created access groups are just groups and do not have the above-mentioned intelligence. It is meant to create access group on-premises devices as well. See Adding Infra Access Group.

Adding Infra Access Group

Before you begin: You can access this functionality only if you have the ACF permissions enabled for your role.

To add an infra access group:

1. Go to (**Menu**) icon **> SSH+ > Groups > Infra Access Group**.

   The **Infra Access Group** page is displayed.
2. On the command bar, click **+Add New Groups**.

   The **Infra Access Group > Create** page is displayed.
3. In the **General Information** section, enter the following:

Field description for General Information section

Field	Description
***Group Name**	Enter a unique name.
Description	Enter details regarding the group stating the purpose.
Managed Devices/ Instances	Select the managed devices/instances to be associated with the group.
Jump Server Client	Select the jump servers clients to be associated with the group.

Note: Fields indicated with red asterisk (*) symbol are mandatory.

4. Click **Create**.

   An Infra Access Group is created and displayed in the inventory.

5. Click the number displayed in the **Associated Machine Count** column.

   The popup window displays the **List of Devices** associated with the machine.

What to do next:

Users with RW permissions can modify or delete an infra user group by selecting the checkbox against the group name and selecting **Modify** or **Delete** from the **Actions** menu.

Removing Hosts from Infra Access Group

You can remove one or multiple hosts from an infra access group thus revoking access to those hosts (if access was granted previously) enhancing security and maintain proper access control within the system.

To remove hosts from an infra access group:

1. Go to (**Menu**) icon **> SSH+ > Groups > Infra Access Group**.

   The **Infra Access Group** page is displayed.

2. Select the infra access groups from which you want to remove the hosts by selecting the checkbox against the group.
3. On the command bar, click **Actions > Modify**.

   The **Infra Access Group > Modify** page is displayed.

4. Scroll down to **Managed Devices / instances** and select the hosts you want to revoke access.
5. Click **Update**.

A workflow is triggered in the background and once the status of the service request is displayed as *Completed*, the hosts are removed from the infra access group.

Tip: You can also remove hosts from the **Host Inventory** by selecting the hosts for which you want to revoke access. Go to **Actions > Modify** and scroll down to **Application Infra Access Group** and delete the infra access groups from which you want to remove the host.

Viewing Infra Access Group

To view infra access group:

1. Go to (**Menu**) icon **> SSH+ > Groups > Infra Access Group**. The **Infra Access Group** page is displayed.

Field description for Infra Access Group section

Field	Description
Group Name	Displays the name of the infra access group. The infra access group can have the following label based on your ACL permissions: • **RW** denotes that you have *Read-Write* permission to the infra access group along with hyperlinks to request access to the group on the **Access Control** page. You can modify or delete infra access groups. • **R** denotes that you only have *Read* permissions to the infra access groups. Users cannot request access or modify or delete infra access groups. • If R or RW permission is not assigned, then you will not be able to view the infra access groups.
Description	Displays the details of the key compliance group.
Associated Machine Count	Click the hyperlink to open a pop-up window displaying all the hosts associated with the infra access group.
Client	Displays the client associated with the infra access group.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

Managing Host Key and User Key Inventories

• Overview
• Key Inventory

Overview

Before you begin: You can access this functionality only if you have the ACF permissions enabled for your role.

SSH uses SSH keys to encrypt communication with a remote system. SSH keys usually come in pairs comprising a public and a private key and are used to grant access to authorized personnel to critical systems such as cloud, on-premise servers, and network devices. The public key (or host key) can be freely shared and is used to encrypt data sent to remote server or user; the private key (or user key) must only be with the user and be kept secret as it is used to decrypt data sent from remote server or user. It is generated by the local machine and kept in a secure location.

This chapter guides you through all the actions that can be carried out on the keys. Actions on keys such as deletion, status change, export, and upload of keys are possible.

From the **Inventory** page, you can:

- View details of the user key and host key. See Viewing User/Host Key Inventory.
- Delete, change status, export, upload user key or host key. See Actions in Key Inventory.
- Discover and add hosts to the host inventory. See Adding Host.
- View details of the hosts. See Viewing Host Inventory.
- Decommission inactive hosts or delete active hosts, add credentials to the host server, fetch host key from the host inventory, and export details of the host to a .csv or .xls format. See Actions in Key Inventory.

Key Inventory

The keys inventory displays the details about the keys. You can also perform various actions on the keys by selecting them from the **Actions** dropdown list. Selecting any action directs you to the respective action page under the **Action** section.

The **User Key Inventory** page displays:

- Key Pair with Passphrase indicated by ()
- Key Pair without Passphrase indicated by ()

• Public Key Only indicated by ()

• Private Key Only indicated by ()

The **Host Key Inventory** page displays:

• Active hosts indicated by green. These hosts can be deleted.
• Inactive hosts indicated by red. These hosts can be decommissioned.

• Viewing User/Host Key Inventory
• Actions on User Key/Host Key Inventory
• Actions on Key Groups
• Advanced Search
• Actions on Recently Deleted Keys
• Actions on Recently Rotated Keys

Viewing User/Host Key Inventory

The User/Host Keys tab displays the total number of weak, shared, orphan, and suspicious keys in the key discovery status. Click the number hyperlink to drill down on the metrics. This helps you track the progress of the key discovery efforts, identify any potential security risks, and prioritize the remediation actions.

To view the user/host key inventory:

1. Go to  (**Menu**) icon **> SSH+ > Inventory > Key Inventory**.
2. Select **User Key Inventory** or **Host Key Inventory**.
   The **SSH+::User Key** page is displayed.

Field Description in User/Host Key Inventory

Field	Description
Key name	Displays the auto-generated unique name created for the key.
Certificate Count	Displays the number of certificates associated with the key. Click the hyperlink to see a popup window with the following fields:

Field	Description
	• **Principals**: Principals contain the identities associated with a certificate. A principal can be a host or a username associated with the certificate.. • **CA name**: Displays the CA name associated with the key. • **Serial Number**: Displays the serial number of the key. • **Certificate Status**: Displays the certificate status of the key. • **Valid From**: Displays the start date of the key validity. • **Valid To**: Displays the end date of the key validity. • **Expires In**: Displays how long before the key expires. • **Extensions**: Displays the extensions of the key.
Key Compliance Group/ Host Compliance Group	Displays the name of the group associated with the key. The key/host compliance group can have the following label based on your ACL permissions: • **R** denotes that you have *Read* permission to the key/host compliance group, although there is no hyperlink available to request access to the key/host compliance group; however, you can view the logs. • **RW** denotes that you have *Read-Write* permission to the key/host compliance group along with hyperlinks to request access to the key/host compliance. You can modify and delete the key/host compliance groups, and view the logs. • If you do not have either R or RW permission, then you will not be able to view the key/host compliance group.
Encryption	This field is applicable only for the user key. Displays the encryption type of the key.
Length	Displays the bit-length of the key.
Age	Displays the age of the key. For example, if the key was created 5 days earlier, it displays as *5 Days*.
Client Endpoint(s)	Displays the count of client machines associated with the key. You can view the list of the hosts associated with the key as a client machine.
Host Endpoint(s)	Displays the count of host machines associated with the key.

Field	Description
	You can view the list of the hosts associated with the key as a host machine.
Risk Status	Displays the status of the key as weak, shared, orphan, or suspicious.
Status	Displays the status of the key. The statuses are: • Managed • Monitored
Associated Users	This field is applicable only for the user key inventory. Displays the users associated with the key.
File Path(s)	Displays where the key file is located on the host.
Comment	Displays any comments with regards to the key.
Validity	Displays the validity of the key.
Fingerprint	Displays the fingerprint of the key.

Actions on User Key/Host Key Inventory

You can perform the following actions from the **Key Inventory** page.

Action description on User/Host Key Inventory page

Action	Description
Change status	Users with RW permission can change the status of a key to **Managed** or **Monitored**.
Export	You can export the user or host key details from their respective inventory in .csv or .xls format.
Upload User SSH key	**Note:** This field appears only for User Key Inventory. **Field description for Upload SSH key section** **Field** / **Description** ***Key File** / Click **Search** icon to browse for the file.

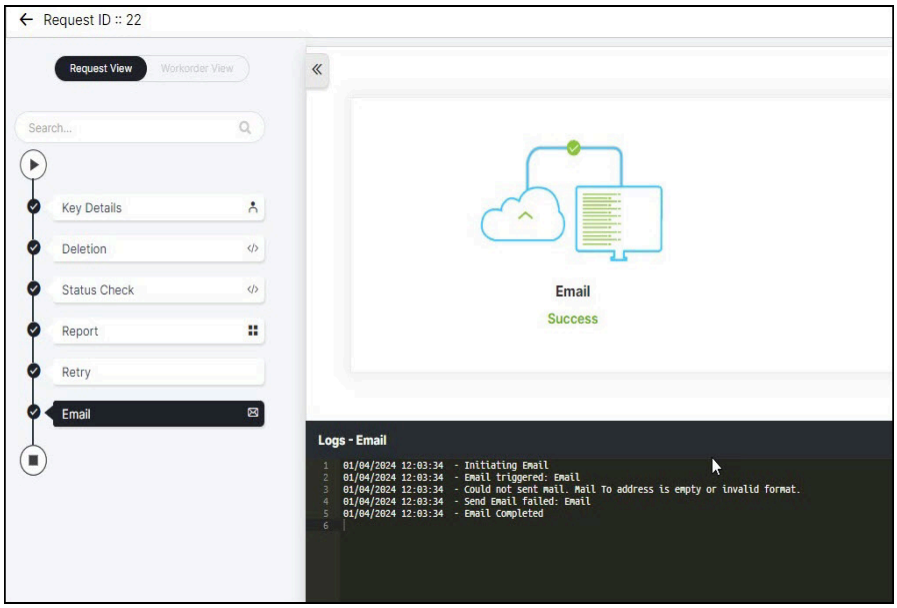
Action description on User/Host Key Inventory page (continued)

<table>
<tr><th>Action</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>*Key Group</td><td>Select key group from the dropdown list.

Note: A key is linked to a key group, and this key group is further connected to a policy. Based on the selection of the key group, it is determined if the key needs a work order approval. The key is also checked for compliance with the key policy associated with the key group.</td></tr>
<tr><td>*Key Name</td><td>Enter a unique name for the key to facilitate easy identification.</td></tr>
<tr><td>Passphrase</td><td>Enter a passphrase</td></tr>
<tr><td>Confirm Passphrase</td><td>Enter the passphrase again to confirm.</td></tr>
<tr><td>*Validity</td><td>Select validity from the dropdown list. This determines the duration for which the key is valid.</td></tr>
<tr><td>Comment</td><td>Enter remarks specific to the key.</td></tr>
<tr><td colspan="2">Note: Fields indicated with red asterisk (*) symbol are mandatory.</td></tr>
</table>
</td></tr>
<tr><td>Revoke</td><td>Note: This field appears only for User Key Inventory.

Users with RW permission can revoke certificates that are associated with keys that have a private key and key pair (public + private). If the selection has even one key that is a public key, then revoke is disabled.</td></tr>
<tr><td>Rotate</td><td>Users with RW permission can rotate selected user keys or host keys based on the rotation configuration outlined in their corresponding key policies. Keys selected for rotation are automatically backed up and stored in a secure encrypted format in the Recently Rotated Keys. The details of</td></tr>
</table>

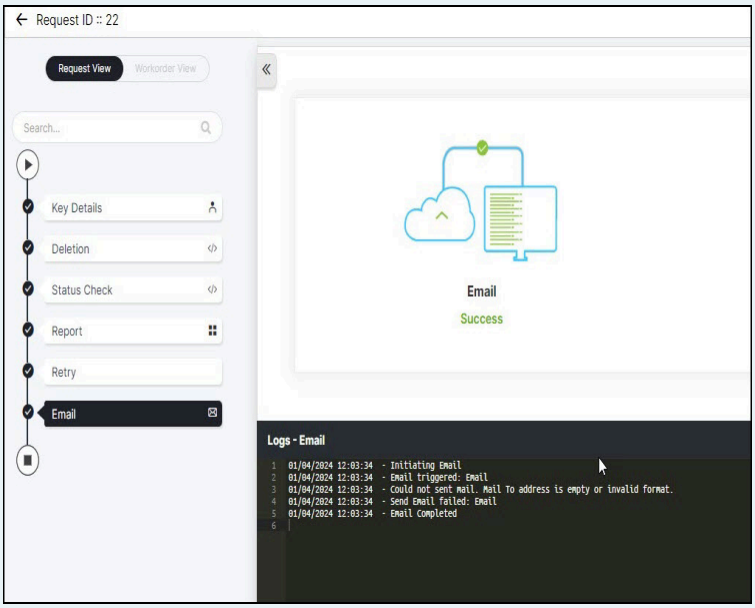
Action description on User/Host Key Inventory page (continued)

Action	Description
	backup are available in the audit log. On successful completion of backup, a message appears in the audit log, *Backup completed for the <key type> for action <action> with name <key name> with fingerprint <key fingerprint> with group name <key group name> by the user <user name>*. On selecting keys for rotation, a confirmation message appears. On confirming, the rotate operation is triggered via workflow. To check the status and reports, go to **Automation > Service Request** and select your request from All requests.  The newly rotated key adheres to the following naming convention: KEYTYPE_TIMESTAMP, where **key type** denotes the encryption algorithm of the key while **timestamp** is when you have rotated the key in the *yyyyMMdd_HHmmss_SSS_counter* format where: • *yyyy* denotes the year • *MM* denotes the month • *dd* denotes the date • *HH* denotes the hours • *mm* denotes the minutes • *ss* denotes the seconds • *SSS* denotes the milliseconds • *counter* denotes the number of keys being rotated

Action description on User/Host Key Inventory page (continued)

Action	Description
	For example, ECDSA_20230908_123456_789_1 implies that the rotated key follows the ECDSA algorithm and was generated on September 8, 2023, at 12:34:56.789 GMT. Upon successful rotation of the key, the **Comments** field is updated. **Important:** Best practices before rotating host keys: 1. If the *global known host* file is not present, then AppViewX will create one in the root folder by including all public keys from users in the global known host file. 2. Prior to host key rotation, update the *global known host* file. 3. The old public key is deleted and the new key is replaced in the *global known host* file. Best practices before rotating user keys: 1. If the *global authorized key* file is not present, then AppViewX will create one in the root folder for each login user with privileged user permission. 2. Prior to user key rotation, update the *global authorized key* file. 3. The old public key is deleted and the new key is replaced in the *global authorized key* file. **CAUTION:** Rotating keys can result in access loss and authentication problems if AppViewX does not have access to all the infrastructure information. Proceed with caution and ensure proper backup and alternative authentication methods are in place.
Delete	Users with RW permission can: • **Delete from Endpoints:** Deletes the keys from the host endpoints. Keys selected for deletion from endpoints are automatically backed up and stored in a secure encrypted format in the database. The details

Action description on User/Host Key Inventory page (continued)

Action	Description
	of backup are available in the audit log. On successful completion of backup, a message appears in the audit log, Backup completed for the <key type> for action <action> with name <key name> with fingerprint <key fingerprint> with group name <key group name> by the user <user name>. **Note:** • If you try deleting keys from hosts with only one key, then a warning message about the potential service disruption is displayed. • On selecting keys for deletion from endpoints, a confirmation message appears. On confirming, the delete operation is triggered via workflow. To check the status and reports, go to **Automation > Service Request** and select your request from All requests.  • **Delete from Inventory:** Deletes the keys from the AppViewX inventory and not from the actual hosts.

Actions on Key Groups

You can perform any of the following actions from the user key and the host key inventories:

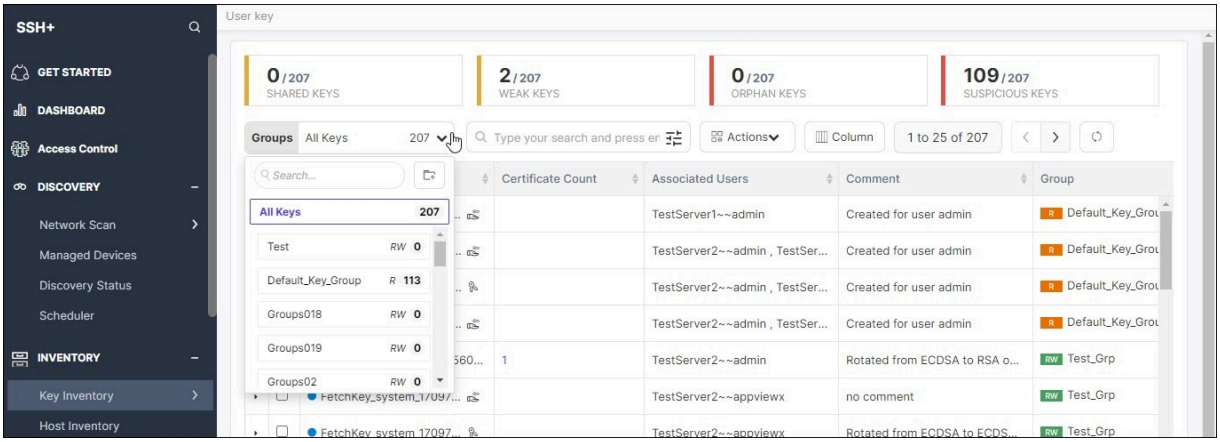
- Filter based on key groups
- Search based on key groups
- Create new key groups and associate keys to them

1. Go to (**Menu**) icon **> SSH+ > Inventory > Key Inventory**.
2. Select **User Key Inventory** or **Host Key Inventory**.

   The **SSH+::User Key Inventory/Host Key Inventory** page is displayed.

3. To filter the key groups, click the **Groups** dropdown list and select a key group from the list. By default, **All Keys** is selected. The key groups are sorted in an alphabetical order displaying key count against each of the key group.

   The filtered key group is displayed.



4. To search further among the key group results, you can provide a search criterion in the **Search** box within the **Groups** dropdown list.
5. To create a new key compliance group and associate keys with it, click (**Create a group**) icon beside the **Search** bar within the **Groups** dropdown list.

   The **Key Compliance Group > Create** page is displayed.

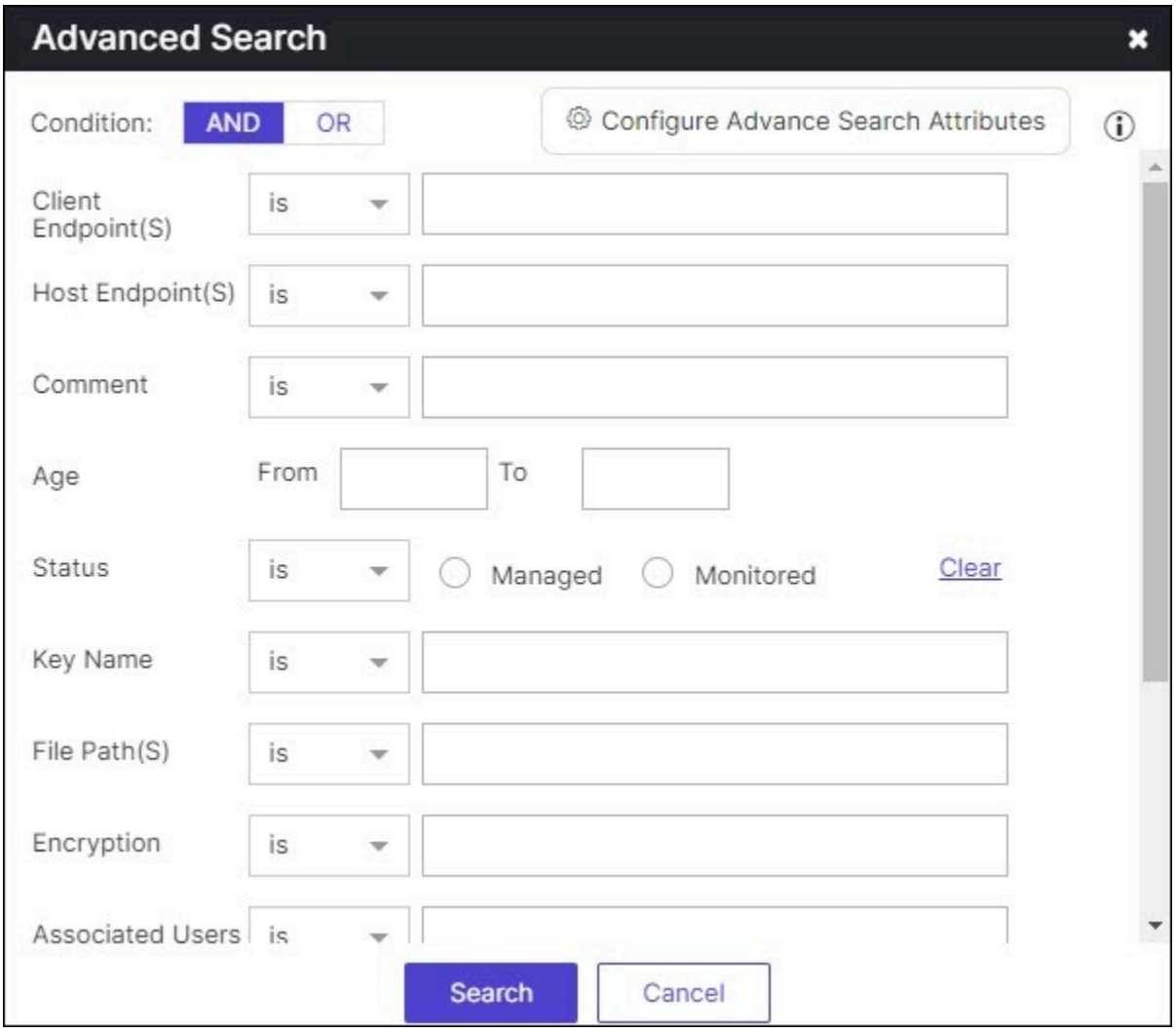
6. Follow the steps as explained in Adding Key Compliance Group to create a group.

Advanced Search

With the advanced search functionality, you can enable precise identification and management of keys. This not only enhances the platform's usability and security but also facilitates efficient key management process.

The SSH+ Key Inventory search supports more granular and flexible searches across various key attributes, including client endpoints, host endpoints, associated users, encryption types, key length, key name, group, file path, fingerprint, and comment.

1. Go to (**Menu**) icon **> SSH+ > Inventory > Key Inventory**.
2. Select **User Key Inventory** or **Host Key Inventory**.
   The **SSH+::User Key Inventory/Host Key Inventory** page is displayed.
3. Click (**Advanced Search**) icon in the **Search** bar.
   The **Advanced Search** window appears. By default, client endpoints, host endpoints, associated users, encryption types, key length, key name, file path, fingerprint, and comment are selected. To add or remove or reposition the search attributes, click **Configure Advance Search Attributes**.



4. Select the **AND** or **OR** condition. By default, **AND** is selected.
5. Select **is**, or **is not**, or **contains** from the dropdown list of each field and specify the search value. By default, **is** is selected.
   You can specify more than one search value using commas.
6. Click **Search**.
   The search results are displayed.

Actions on Recently Deleted Keys

All the keys that are deleted from the user key and host inventories are available in **Recently Deleted Keys**. You can access this functionality only if you have the ACF permissions enabled for your role.

1. Go to (**Menu**) icon **> SSH+ > Inventory > Key Inventory**.
2. Select **Recently Deleted Keys**.

   The **Recently Deleted Keys** page is displayed.

3. You can perform any of the following actions:
   - **Viewing deleted keys:** The deleted user or host keys from endpoints are made available in the **User Keys** or **Host Keys** for a period of 30 days. The inventory lists key details including associated users, key type, client and host endpoints, who deleted the key, deletion date, and the days left before a key is permanently deleted.
   - **Restoring deleted keys:** To restore keys that were deleted by mistake or are needed again, simply select them and select **Restore** from the **Actions** menu. A message appears, *You are about to restore the selected key(s) back to their original location(s). This action will reinstate the key(s) and make them active again. Are you sure you want to proceed with the restoration?* Click **Confirm Restoration** to proceed.

     Audit logs for the action when:
     - **restoration succeeded:** Restore action triggered by the user <username> for uuids <uuid list> completed successfully for backup <user or host key>
     - **restoration failed:** Restore action triggered by the user <username> for uuids <uuid list> failed for backup <user or host keys>

     Note: <uuid> is the unique ID of the key.

   - **Deleting keys permanently:** You can permanently delete keys before the 30-day window by selecting the keys and then selecting **Delete Permanently** from the **Actions** menu. A message appears, *You are about to permanently delete the selected key(s). This action cannot be undone, and the key(s) will be irrecoverably removed from the system. Are you sure you want to proceed with permanent deletion?* Click **Confirm Deletion** to proceed.

     Audit logs for the action when:

     - **delete succeeded:** Delete action triggered by the user <username> for uuids <uuid list> completed successfully for backup <user or host keys>
     - **delete failed:** Delete action triggered by the user <username> for uuids <uuid list> failed for backup <user or host keys>

     Note: <uuid> is the unique ID of the key.

Actions on Recently Rotated Keys

All the keys that are rotated from the user key and host inventories are available in **Recently Rotated Keys**. You can access this functionality only if you have the ACF permissions enabled for your role.

1. Go to (**Menu**) icon **> SSH+ > Inventory > Key Inventory**.
2. Select **Recently Rotated Keys**.

   The **Recently Rotated Keys** page is displayed.
3. You can perform any of the following actions:
   - **View rotated keys:** The rotated user or host keys from endpoints are made available in the **User Keys** or **Host Keys**. The inventory lists key details including key name, associated users, client and host endpoints, fingerprint, who rotated the key, and the date when rotated.
   - **Rollback rotated keys:** To rollback keys that were rotated by mistake, simply select them and select **Rollback** from the **Actions** menu. A message appears, *Are you sure you want to rollback to the previous key? This action will revert the selected key(s) to their former state. Please confirm to proceed.* Click **Yes** to proceed.

> **Note:** An info icon appears beside the rotated key. Clicking the info icon takes you to the user/host key inventory where you can see the active key in the device. On performing rollback, that key is pushed and the active key is removed.

Dashboard

- Reports
- Remediation Actions

Reports

The dashboard provides several components that track SSH traffic and log interaction with devices. You can use this dashboard to look at the risks associated with SSH and their severity. This dashboard provides detailed information identifying critical and high device risks for administrators to mitigate and secure before management of a device is compromised.

> **Note:** The reports visible to you is based on your assigned ACL permissions.

Risk Report

The Risk Report provides information about risks from the discovery of the default branch. It contains cumulative results of successful discovery.

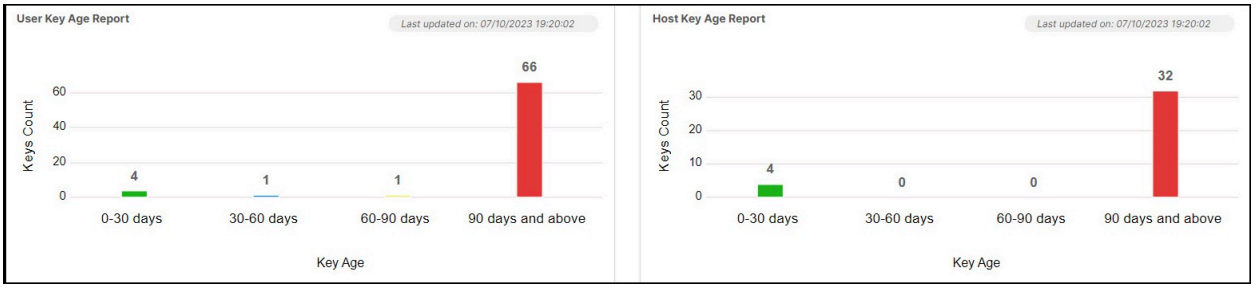
Click the **Count** hyperlink to fetch more details such as key name, group, length, age, encryption and so on based on the key type.

Note: The Count field displays key count only from key compliance groups that the user has permission to access.

You can identify and remediate weak, shared, suspicious, and orphan host/user keys from the dashboard to keep the infrastructure secure. To perform remediation actions, see Remediation Actions. For description of keys, see Glossary.

Key Age Reports

Displays the following reports in a bar chart:



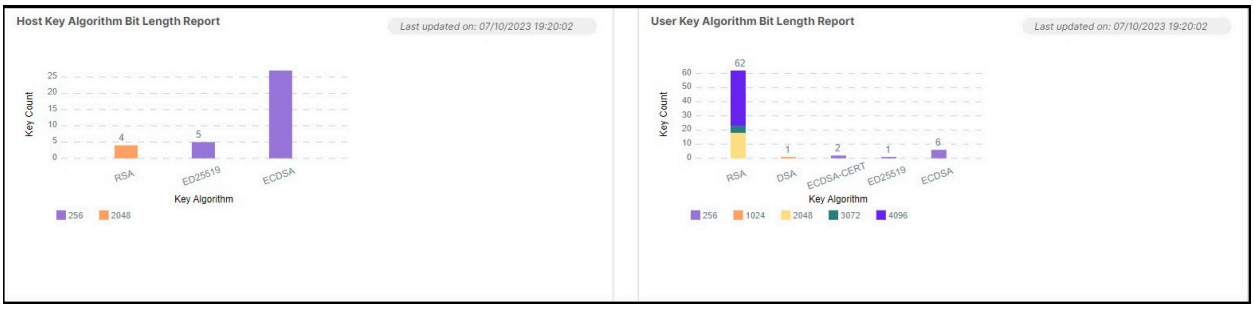
- **User Key Age Report:** A bar chart displaying the groupwise number of keys (Y-axis) based on the following user key age (X-axis):
  - 0 to 30 days
  - 30 to 60 days
  - 60 to 90 days
  - 90 days and above
- **Host Key Age Report:** A bar chart displaying the groupwise number of keys (Y-axis) based on the following host key age (X-axis):
  - 0 to 30 days
  - 30 to 60 days
  - 60 to 90 days
  - 90 days and above

Click the graph on the widget to fetch details of the client endpoints, host endpoints, associated users, key name, and the group.

You can also rotate and delete keys from hosts with multiple keys through the user and host key age report thus mitigating service disruptions. On selecting keys for rotation or deletion from an endpoint, a confirmation message appears. On confirming, the operation is triggered via workflow. To check the status and reports, go to **Automation > Service Request** and select your request from **All requests.**

Key Algorithm Bit Length Reports

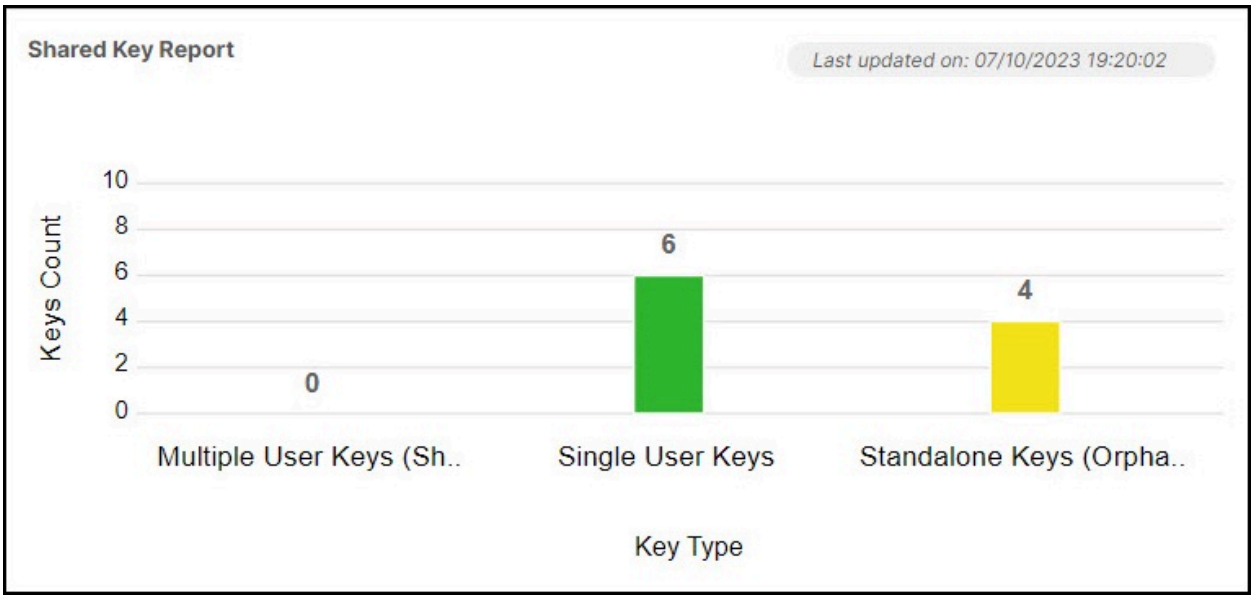
Displays a stacked bar chart to filter keys based on key bit length and key algorithm:



- **User Key Algorithm Bit Length Report:** A stacked bar chart displaying the groupwise number of key size (Y-axis) based on the user key size and algorithm (X-axis). The user key sizes are 256, 384, 521, 1024, 2048, 3072, 4096 while the key algorithms are RSA, DSA, RSA1, ED25519, ECDSA.
- **Host Key Algorithm Bit Length Report:** A stacked bar chart displaying the groupwise number of key size (Y-axis) based on the host key size and algorithm (X-axis). The host key sizes are 256 and 2048 while the key algorithms are RSA, ED25519, ECDSA.

Shared Key Report

Displays the key count (Y-axis) based on the following key type (X-axis) as a bar chart.

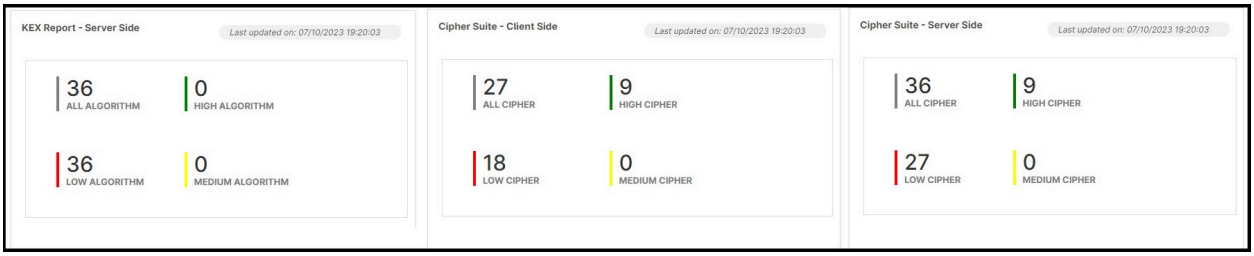


- **Multiple User Keys (Shared):** The number keys that are associated with multiple users and in the key group.
- **Single User Keys:** The number keys that are associated with a single user and in the key group.
- **Standalone Keys (Orphan):** The number of keys on standalone machines associated with the key group.

Click the graph on the widget to fetch details of the client endpoints, host endpoints, associated users, key name, and the host group.

Server-Side Reports

Displays the following reports:

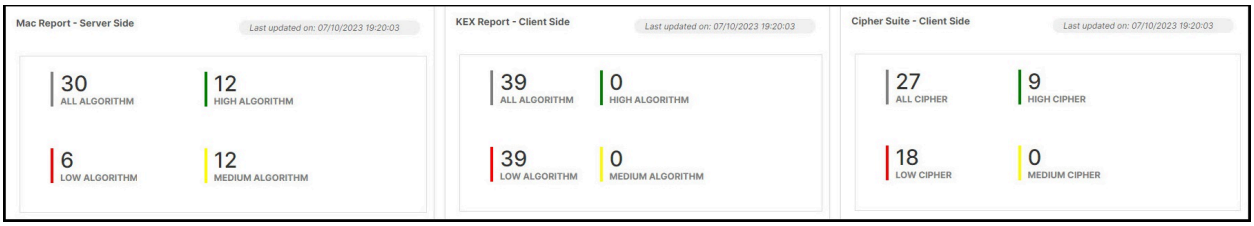


- **Cipher suite report:** This report provides the following numerical representation of All Cipher, High Cipher, Low Cipher, and Medium Cipher for the hosts added in the SSH host inventory.
- **KEX report:** This report provides the following numerical representation of All Algorithm, High Algorithm, Low Algorithm, and Medium Algorithm for the hosts added in the SSH host inventory.
- **Mac report:** This report provides the following numerical representation of All Algorithm, High Algorithm, Low Algorithm, and Medium Algorithm for the hosts added in the SSH host inventory.

Hover the mouse over the numbers and click it to fetch details of the host name, IP address/FQDN, algorithm, OS type, version, and group.

Client-Side Reports

Displays the following reports:

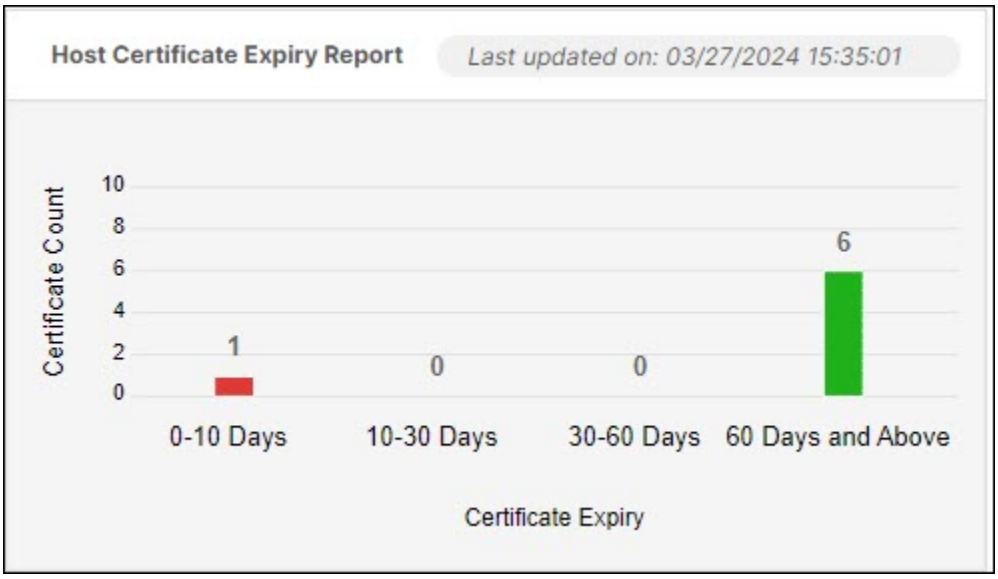


- **Cipher suite report:** This report provides the following numerical representation of All Cipher, High Cipher, Low Cipher, and Medium Cipher for the hosts added in the SSH host inventory.
- **KEX report:** This report provides the following numerical representation of All Algorithm, High Algorithm, Low Algorithm, and Medium Algorithm for the hosts added in the SSH host inventory.
- **Mac report:** This report provides the following numerical representation of All Algorithm, High Algorithm, Low Algorithm, and Medium Algorithm for the hosts added in the SSH host inventory.

Hover the mouse over the numbers and click it to fetch details of the host name, IP address/FQDN, algorithm, OS type, version, and host group.

Host Certificate Expiry Report

A bar chart displaying the number of host certificates (Y-axis) based on their expiration (X-axis):



- 0 to 10 days
- 10 to 30 days
- 30 to 60 days
- 60 days and above

Actions

- Click **Export** to export the details to .csv or .xls format.
- Click **View in Inventory** to view the inventory listing of the categories on that page.

Remediation Actions

Before you begin:

- You can access this functionality only if you have the ACF permissions enabled for your role.
- To perform any of the remediation actions, you must have RW permissions assigned to you.

You can identify and remediate weak, shared, suspicious, and orphan host/user keys from the dashboard to keep the infrastructure secure.

1. Go to (**Menu**) icon **> SSH+ > Dashboard**.
   The **Dashboard** page is displayed.
2. On the **Risk Report,** click **Remediation Option** against the host/user key category.

A popup window of the selected host/user key is displayed along with the details of the key name, host endpoints, associated users, and file path.

3. Click the checkbox against the key(s) you want to remediate.
4. You can perform one of the following actions:
   a. **Rotate:** On selecting keys for rotation, a confirmation message appears. On confirming, the rotate operation is triggered via workflow. To check the status and reports, go to **Automation > Service Request** and select your request from **All requests.**
   The selected keys are regenerated and pushed to the host endpoints.

   > **Important:**
   > Best practices before rotating host keys:
   > i. If the *global known host* file is not present, then AppViewX will create one in the root folder by including all public keys from users in the global known host file.
   > ii. Prior to host key rotation, update the *global known host* file.
   > iii. The old public key is deleted and the new key is replaced in the *global known host* file.
   > Best practices before rotating user keys:
   > i. If the *global authorized key* file is not present, then AppViewX will create one in the root folder for each login user with privileged user permission.
   > ii. Prior to user key rotation, update the *global authorized key* file.
   > iii. The old public key is deleted and the new key is replaced in the *global authorized key* file.

   b. **Delete:** On selecting keys for deletion from endpoint, a confirmation message appears. On confirming, the delete operation is triggered via workflow. To check the status and reports, go to **Automation > Service Request** and select your request from **All requests.** The selected keys are deleted from the host endpoints and the key inventory.
   c. **Acknowledge:** The selected keys are acknowledged and excluded from the risk report for the duration specified in the associated key policy. The keys, however, continue to be present in the key inventory.

   Selecting any of the actions opens a confirmation window.
5. Click **Confirm** to proceed.

Creating Key Policy and Group

• Overview
• Key Policy
• Key Compliance Group

Overview

Before you begin: You can access this functionality only if you have the ACF permissions enabled for your role.

Policy is configured to define the attributes of a key to belong to the key group. A key is compliant only if it matches the attributes defined in the associated policy.

From the **Policies** page, you can add, modify, or delete a key policy. See Creating Key Policy.

You can bring together keys under a key group. A key can be part of only one group associated with a policy (key policy for the key group). This group to policy association can be done from the key inventory or from the policy create/edit page. By default, the system has a default key group to accommodate all keys that are not manually associated with any other group. See Adding Key Compliance Group.

Key Policy

The key policy plays an important part while generating an SSH key. You can set the key to rotate automatically while creating the policy.

You cannot delete the default key policy, but you have the option to change the settings. The default policy is associated with all the key groups.

Note: You can associate a particular policy with a single key group or multiple key groups. The key configuration is based on the associated policy. The discovered keys are checked for compliance against the policy associated with them. The system marks the key as non-compliant in the inventory if the compliance check fails.

• Creating Key Policy

Creating Key Policy

To create a key policy:

1. Go to (**Menu**) icon **> SSH+ > Policies > Key Policy**.

   The **Key Policy** page is displayed.

2. On the command bar, click **+Create policy**.
3. Enter the following details:

Field description for Key Policy section

Field	Description
Policy details	
***Policy Name**	Enter a unique name for the policy.
Description	Enter details of the policy stating the purpose.
Compliance Configuration	
***Key Algorithm**	Select a value from the dropdown list. You can select more than one value.
***Key Size**	Select a value from the dropdown list. You can select more than one value.
Rotation Configuration	
***Key Rotation Period**	Select a value from the dropdown list. For example, if you select 180 days from the dropdown list, then the key will be rotated after 180 days.
***Key Algorithm**	Select a key algorithm that specifies which host key types are allowed to be used for the SSH connection.
***Key Size**	Select the size of the key used in the key algorithm.
Host Certificate Auto Rotate Settings	
***Auto Rotate Host Certificates before**	Select a value from the dropdown list to initiate certificate rotation before its expiration. By default, this value is 10 days.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

4. Click **Create**.

   A key policy is created and added to the key inventory.

What to do next:

- Modify or delete a key policy by selecting the checkbox against the policy name and selecting **Modify** or **Delete** from the **Actions** menu.
- Associate the policy with a key compliance group. See Adding Key Compliance Group.

Key Compliance Group

By default, all the LDAP users are mapped to the Default Requestor group. A user can be associated with a single user group. Whenever a requestor is added to a group, a compliance check is triggered to check if the key is compliant or not.

- Adding Key Compliance Group
- Viewing Key Compliance Group

Adding Key Compliance Group

To add a key compliance group:

1. Go to (**Menu**) icon **> SSH+ > Groups > Key Compliance Group**.

   The **Key Compliance Group** page is displayed.

2. On the command bar, click **+Add New Groups**.

   The **Key Compliance Group > Create** page is displayed.

3. In the **General Information** section, enter the following:

   Field description for General Information section

Field	Description
***Group Name**	Enter a unique name. This helps you identify it easily.
Description	Enter details regarding the group stating the purpose.
***Requestor**	Select the requestor(s) to be associated with the group.
***Requestor Policy**	Select the required policy to be associated with the requestor group.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

4. Click **Create**.

   A key compliance group is created and displayed in the inventory.

What to do next:

- Users with RW permission can modify or delete a key compliance group by selecting the checkbox against the group name and selecting **Modify** or **Delete** from the **Actions** menu.

Viewing Key Compliance Group

To view a key compliance group:

1. Go to (**Menu**) icon **> SSH+ > Groups > Key Compliance Group**. The **Key Compliance Group** page is displayed.

Field description for General Information section

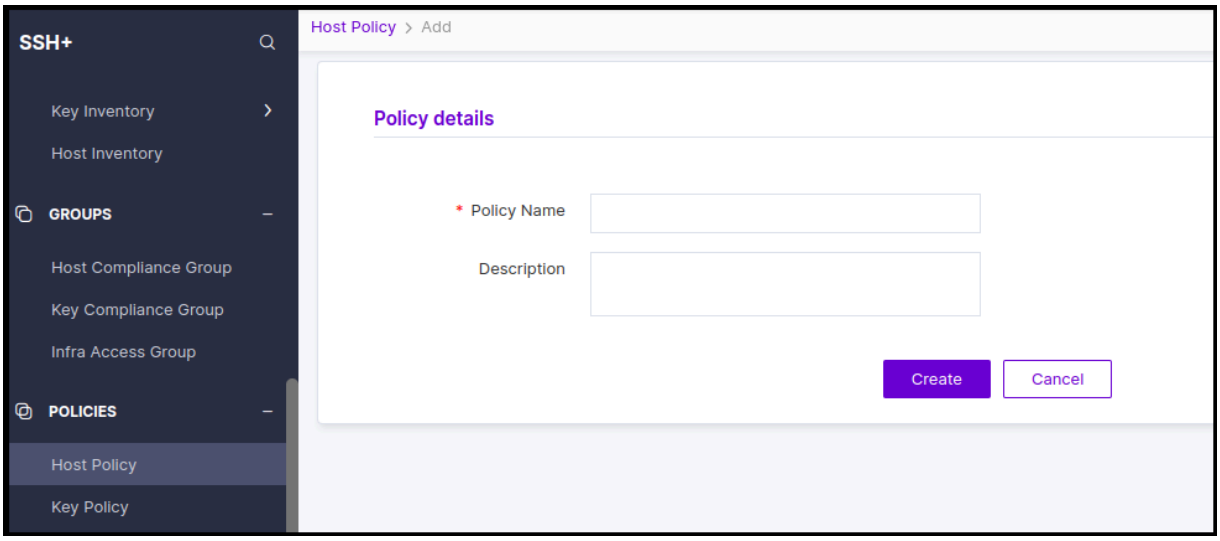
Field	Description
Group Name	Displays the name of the key compliance group.
Description	Displays the details of the key compliance group.
Count	The Count field displays key count of keys only from the Key Compliance Groups that the user has permission to access. Click the hyperlink to open a pop-up window displaying all the keys associated with the key compliance group.
Policy Associated	Displays the policy associated with the key compliance group.
Note: Fields indicated with red asterisk (*) symbol are mandatory.	

Creating Host Policy and Group

• Creating Host Policy

Creating Host Policy

1. Go to (**Menu**) icon **> SSH+ > Policies > Host Policy**.
2. Click **Create Policy**.
   The **Host Policy** page is displayed.



3. Enter a unique policy name.
4. [Optional] Add description for the policy.
   Click **Create**. The host policy is created.

> **Note:** Host policy will be available in the next release.

Glossary

Term definition

Term	Definition
SSH	Secure Socket Shell (SSH), also known as simply Secure Shell, is a cryptographic protocol used to enable secure access to remote servers and devices over the internet using SSH keys, certificates, or passwords.
SSH key	SSH keys are used to encrypt communication with a remote system. SSH keys usually come in pairs comprising a public and a private key and are used to grant access to authorized personnel to critical systems such as cloud, on-premise servers, and network devices.
Host key	A host key is a key that is used to identify the server. It is generated by the server and shared with the client during the initial connection setup. The client uses this key to verify the identity of the server before establishing a connection.

Term definition (continued)

Term	Definition
User key	A user key is a public key that is associated with a particular user account on the host. It is used to authenticate the user and establish a secure connection with the server.
Public key	A public key is used to encrypt data and verify digital signatures. It can be freely distributed, and anyone can use it to encrypt data or verify digital signatures. It is also used to establish a secure connection between the client and the server.
Private key	A private key is a secret key that is used to decrypt data and create digital signatures. It must be kept secret and never shared with anyone. The private key is used to authenticate the user and establish a secure connection with the server.
Suspicious key	A key without a known client association.
Shared key	A key used by more than one user.
Orphan key	A key that is found on a non-standard client file-folder path and does not have a known server.
SSH key rotation	The process of replacing the old key with a new one that adheres to the SSH key policy.
Weak key	A key that is generated using a weaker algorithm and size.

Chapter 3: SSH+ API Guide

- Understanding the AppViewX SSH API
- Authentication
- Add New Host
- Search Hosts
- Search Host Keys
- Search User Keys
- Search Access Groups
- SSH Create CA
- SSH Download CA
- Search CA
- SSH Create Certificate
- SSH Download KRL
- SSH Get Hosts From Infra Access Group
- Trigger Network Scan for Range of IP Addresses
- Revoke Certificate

Understanding the AppViewX SSH API

The AppViewX API is a programmatic way to get data in and out of the AppViewX subsystems. With access to RESTful AppViewX APIs, you can leverage the raw potential of AppViewX. It provides a powerful way to channel the data into native business applications. This document comprises of module-wise APIs used in AppViewX.

RESTful HTTPS Requests

Type	Description
GET	GET requests, retrieve resource representation/information only and not to modify it.

Type	Description
POST	POST APIs create new subordinate resources. For example, a file is subordinate to a directory containing it or a row is subordinate to a database table. In terms of REST, POST methods are used to create a new resource into the collection of resources.
PUT	PUT APIs are used to update existing resources (if a resource does not exist then API may decide whether to create a new resource or not).
DELETE	DELETE APIs are used to delete resources (identified by the Request-URI).

Requests

Each endpoint URL is built in the same way by the following structure:

```
http://<IP/HostName/TenantName>:<GWPORT>/avxapi/<Endpoint>?<gwsource>
```

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

Request Structure

All endpoints accept a request structure that should consist of JSON formatted data. To ensure the request is accepted, set the header **Content-Type: application/json**.

The following example shows a request to add a resource:

```
{
"payload": {
"name": "resource_1",
"description": "This is a sample resource."
}
}
```

Response Structure

The Content-Type of the response is typically determined by the Content-Type header, and for most endpoints, it will be application/json. All requests that reach the server, regardless of the response code, will retrieve a response body. A successful request will contain a body with the requested information, for example:

```
https://appviewxapi.com/avxapi/resource?gwsource=external
```

Returns the following JSON structure that a resource is added:

```
{
"response": "Resource added successfully",
"message": "Resource added successfully",
"appStatusCode": null,
"tags": null,
"headers": null
}
```

Description of Server Responses

HTTP Code	Response Message
200 OK	The request was successful (some API calls may return 201 or 202 instead).
400 Bad Request	The request is not understood or required parameters are missing.

HTTP Code	Response Message
401 Unauthorized	Authentication failed or the user doesn't have permissions for the requested operation.
409 Forbidden	Access denied.
404 Not Found	Resource not found.
429 Too many requests	The number of requests to the service has crossed the threshold.
503 Service unavailable	The client cannot communicate with the service.
504 Gateway timeout	The given request has exceeded the expected time.

URI Scheme

- **Host :** {url}
- **BasePath :** /avxapi
- **Schemes :** HTTPS
- **URL :** https://{url}/avxapi

Types of Accounts in AppViewX

There are two types of accounts in AppViewX:

- **User Accounts**: These are used by actual users.
- **Service Accounts**: These are used by system services such as web servers, automation tools, and so on.

AppViewX recommends using a Service Account for accessing APIs from automation tools. Service Accounts are supported with oAuth standard for a more secure and standard way of accessing APIs.

Note: AppViewX supports both User Account and Service Account for accessing APIs.

Authentication

- Using User Account
- Authentication Using Service Account

Using User Account

For accessing APIs with a user account, you need to get the session ID by providing a username and password in the login API. This session ID can then be used for accessing other APIs.

Note: You can also use the username and password in all API calls instead of the sessionId. However, this is not recommended.

- Retrieve session ID using login API
- Using Session ID for further API calls

Retrieve session ID using login API

This API used to retrieve the session ID using the login API for secure authentication and access to system resources.

Before you begin

- Make sure you have valid login credentials (Username and Password) for accessing the system.
- You cannot use OAuth credentials (Client ID and Client Secret) for login.
- To access the APIs using the service token, use the API with the Service Account.

Request Structure

Endpoint	`/login`
Type	`POST`
Sample URL	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/login?gwsource=external` To understand the elements of the sample URL, click here.
Headers	
Content-Type	application/json
Request timeout period	15 minutes

Understanding the sample URL

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

Input Parameters

	Description
username *Header*	(Mandatory) Use login name of the user. **Type:** String **Example:** "admin"
password *Header*	(Mandatory) Password for the username. **Type:** String **Example:** "AppViewX@123"
otp *Header*	(Mandatory only if MFA is enabled) If MFA is enabled, enter the OTP received on your registered email ID in the header. Multifactor authentication (MFA) is a security mechanism that requires users to provide two or more verification factors to gain access to a resource

Input Parameters (continued)

	Description
	If MFA is enabled, and you try to login with only the username and password, you will get the following error upon execution of the API: **MFA is enabled. We have sent an OTP to your email ID: aaa********r@appviewx.com**. In this case, ensure that the OTP is included in the header and try logging in again. **Type:** String **Example:** "OTP : **609700**"
Content-Type *Header*	(Mandatory) The parameter should be set to `application/json` to specify the nature of the data in the payload. **Type:** String **Example:** "application/json"
gwsource *Query*	(Mandatory) Source from which the request is triggered. The values can be: • web • external **Type:** String

Response Structure

- **Status Code:** 200 Ok
- **Message:** Login Successful
- **Headers:**
  - **Content-Type:** application/json

Response Parameters

Name	Description
response	The response contains the attributes needed to retrieve the session ID.
message	Success message or failure description in case of error.
appStatusCode	Application specific status code for the response. Will be non-null for failure response.

Response Parameters (continued)

Name	Description
tags	More info in case of failure response.

Name	Description
status	Indicates the overall status of the response. The values can be: • SUCCESS • FAILURE
appStatusCode	An application-specific status code, if applicable.
statusDescription	Description of the status, if available.
sessionId	Unique identifier for the session.
lockDownPeriod	Number of login attempts remaining.
termsAccepted	
passwordExpiryMsg	
emailId	

Status Codes

HTTP Code	appStatusCode	Response Message
200 OK	NA	Login successful
400 Bad request	ACCT_AUTH_001	Username or password cannot be null or empty.
401 Unauthorized	ACC_AUTH_022	Login failed. Invalid credentials.
401 Unauthorized	ACC_AUTH_006	Login failed. Invalid credentials.

Sample Request/Response

Use Case

Login to the application with a username and password.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/login?gwsource=external
```

Request Payload

```
{}
```

Sample Response


```
{
    "response": {
        "status": "SUCCESS",
        "appStatusCode": null,
        "statusDescription": null,
        "sessionId": "avx~~c73a4f56-f4ab-4cdf-aadf-6d90bf406077",
        "authCode": null,
        "lockDownPeriod": 15,
        "emailId": null,
        "termsAccepted": true,
        "passwordExpiryMsg": ""
    },
    "message": "Login successful.",
    "appStatusCode": null,
    "tags": null,
    "headers": null
}
```


Using Session ID for further API calls

The sessionID retrieved using the login API can be used in the header for making further API calls.

Before you begin

- Session ID is obtained from the login API.
- Ensure that the session ID is valid and has not expired.

Request Structure

Endpoint:	`/role`
Type:	`POST`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/role?&gwsource=external` To understand the elements of the sample URL, click here.
Headers:	

Content-Type:	application/json

Understanding the sample URL

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

Input Parameters

Name	Description
sessionId *Header*	(Mandatory) Use session ID retrieved from login API, if username and password are not provided. **Type:** *String* **Example:** "sessionId": "ce7f1a14-2bf9-4e4a-89a8-bc780a255813"
gwsource *Query*	(Mandatory) Source from which the request is triggered. The values can be: • web • external

Input Parameters (continued)

Name	Description
	Type: *String*
Payload *String*	(Mandatory) Input data for request body in application/json format. For payload details, see **Payload** section.

Payload

Name	Description
name *String*	(Mandatory) Name of the role to be added. **Example:** "role_1"
description *String*	(Optional) Description of the role to be added. **Example:** "Adding a new role"

Response Structure

- **Status Code:** 201 Created
- **Message:** Role added successfully
- **Headers:**
  - **Content-Type:** application/json

Response Parameters

Name	Description
response	Contains the response attributes for role added successfully.
message	Success message or failure description in case of error.
appStatusCode	Application specific status code for the response. Will be non-null for failure response.
tags	More info in case of failure response.

Status Codes

HTTP Code	appStatusCode	Response Message
201 Created	null	Role added successfully.

HTTP Code	appStatusCode	Response Message
409 Conflict	ACCT_RO_002	Role name already exists
400 Bad Request	VALIDATION_ERROR_0004	'name' should have at least '2' characters, Mandatory Field 'name' is missing or empty.
400 Bad Request	ACCT_RO_015	Role name invalid.

Sample Request/Response

Use Case

Using the session ID acquired from the login API to execute subsequent API calls, specifically for adding a role API.

Sample Request

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/role?gwsource=external
```

Request Payload

```
{
"payload": {
 "name":"role_01",
 "description":"Adding a new role"
}
}
```

Sample Response

```
{
"response": "Role added successfully",
"message": "Role added successfully",
"appStatusCode": null,
"tags": null,
"headers": null
}
```

Authentication Using Service Account

Authenticate into the AppViewX application with the username and password.

Request Structure

URL: /login

Type: POST

Parameters

Parameter Details

Param Type	Name	Description	Field Type	Constraints
Header	`username`	AppViewX login username	`String`	
Header	`password`	AppViewX login password	`String`	
Header	`Content-Type`	Specifies the nature of the data in the payload.	`String`	Value of the param should be `application/json`
Query	`gwsource`	Source from which the request is triggered (Eg: web, external)	`String`	

Response Structure

200 OK, returns a string of type application/JSON with the following body parameters:

Name	Description	Field Type
`response`	Contains the response for the session activation with the authentication/ session ID.	Key Value Pair
`message`	Success message along with the objectIds or failure description in case of error.	String
`appStatusCode`	Application specific status code for the response. Will be non-null for failure response.	String
`tags`	More info in case of failure response.	Key value pair

Status Codes

HTTP Status Code	appStatusCode	Message	Possible Remediation
200 OK	-	Login successful	-

HTTP Status Code	appStatusCode	Message	Possible Remediation
400 Bad request		Username or password cannot be null or empty	Check and ensure if a non-null/non-empty value is given in the header field for username or password
401 Unauthorized		Login failed. Invalid credentials.	Check and ensure if the username and password provided are valid and correct.

Sample Request/Response

Use Case

Login to the application with the user name, **appviewxuser**, and password, **appviewxpassword**. The session id received as response in the example below is ce7f1a14-2bf9-4e4a-89a8-bc780a255813.

Sample Request

```
{

}
```

Sample Response

```
{
"response":{
"status":"SUCCESS",
"appStatusCode":null,
"statusDescription":null,
"sessionId":"ce7f1a14-2bf9-4e4a-89a8-bc780a255813",
"availableLoginAttemptCount":null
},


"message":null,
"appStatusCode":null,
"tags":null,
"headers":null
}
```

- Retrieve Access Token using get-service-token API
- Using Access Token in the header for further API calls

Retrieve Access Token using get-service-token API

The API provides a streamlined process for retrieving service tokens related to account management tasks.

Before you begin

- Make sure you have valid login credentials for accessing the system.

Request Structure

Endpoint:	`/acctmgmt-get-service-token`
Type:	`POST`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/acctmgmt-get-service-token?gwsource=external` To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json
Authentication:	Yes
Request timeout period	15 minutes

Understanding the sample URL

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

Example: **31443**

- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

Input Parameters

	Description
Authorization *Header*	(Mandatory) Please form a string in this format <Client ID>:<Client Secret> and do base64 encoding. Then prepend a key 'Basic' before the encoded value. Final value should be "Basic <EncodedValue>". **Type:** *String* **Example:** "admin"
Content-Type *Header*	(Mandatory) The parameter should be set to `application/json` to specify the nature of the data in the payload. **Type:** *String* **Example:** "application/json"
grant_type *Payload*	(Mandatory) Payload Type should be "Form". The value of the param should be "Client_Credentials". **Type:** *Text*

Response Structure

- **Status Code:** 200 Ok
- **Message:** Successful
- **Headers:**
  - **Content-Type:** application/json

Response Parameters

Name	Description
response	The response contains the attributes needed to retrieve the access token.

Response Parameters (continued)

Name	Description
message	Success message or failure description in case of error.
appStatusCode	Application specific status code for the response. Will be non-null for failure response.
tags	More info in case of failure response.

Status Codes

HTTP Code	appStatusCode	Response Message
200 OK	NA	Successful
400 Bad request	ACCT_SA_003	Service account is invalid/not found::[Service account not found in the database]
400 Bad request	OAUTH_CLNT_015	Client Password is incorrect::[Invalid Client credential]
400 Bad request	ACCT_SA_001	Invalid Request::[Invalid client Id or secret]
500 Internal Server Error	avx-common-011	Error while processing.

Sample Request/Response

Use Case

Retrieve Access Token.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/acctmgmt-get-service-token?gwsource=external
```

Content-Type: application/x-www-form-urlencoded

Authorization: Basic
NTIxYzNhZDItZWE0ZS00NDdiLWE1MWItOTYyMWJiN2VhMTI2OjU1QVUjTk84JSpaaGd2TmZhWVVtdHZYMGRrRWhvZ

Request Payload

Sample Response

```
{
"response":
 "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJwbGF0Zm9ybSIsImF1ZCI6ImF2eCIsImNsaWVudElkIjoiOTcwNzRlNDEtOGFmOS00NTZkLTlhNjQtZjBj
NGJiOTA4MDQ4IiwiaXNzIjoiYXZ4IiwiZXhwIjoxNjUwMzY5MzY3LCJncmFudCB0eXBlIjoiY2xpZW50X2NyZWRlbnRpYWxzIn0.HZnkuUEjXIeqJWqpqiNWFHqIDI7
GYf4cWx6VwbjGD_0",
"message": null,
"appStatusCode": null,
"tags":{},
"headers": null
}
```


Using Access Token in the header for further API calls

The access token retrieved using the get-service-token API can be used in the header for making further API calls.

Before you begin

- Access Token is obtained from the get-service-token API.
- Ensure that the Access Token is valid and has not expired.

Request Structure

Endpoint:	`/resource`
Type:	`POST`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/resource?gwsource=external` To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Understanding the sample URL

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.

- **HostName**: A human-readable label assigned to a device (host) on a network

  The hostname will be included in the endpoint URL for an on-prem deployment.

- **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

  The tenant name will be included in the endpoint URL for a SaaS deployment.

- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**

- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

Input Parameters

Name	Description
Token *Header*	(Mandatory) Use token retrieved from login API. **Type:** *String* **Example:** eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJwbGF0Zm9ybSIsImF1ZCI6ImF2eCIsImNsaWVudElkIjoiOTcwNzRlNDEtOGFmOS00NTZkLTlhNjQtZjBjNGJiOTA4MDQ4IiwiaXNzIjoiYXZ4IiwiZXhwIjoxNjUwMzY5MzY3LCJncmFudCB0eXBlIjoiY2xpZW50X2NyZWRlbnRpYWxzIn0.HZnkuUEjXIeqJWqpqiNWFHqIDI7GYf4cWx 6VwbjGD_0
gwsource *Query*	(Mandatory) Source from which the request is triggered. The values can be: • web • external **Type:** *String*
Payload *String*	(Mandatory) Input data for request body in application/json format. For payload details, see **Payload** section.

Payload

Name	Description
name *String*	(Mandatory) Name of the resource to create. Name cannot be duplicated. **Example:** "resource_1"
description *String*	(Optional) Description of the resource. **Example:** "This is a sample resource."

Response Structure

- **Status Code:** 201 Created
- **Message:** Resource added successfully
- **Headers:**
  - **Content-Type:** application/json

Response Parameters

Name	Description
response	Contains the response attributes for resource added successfully.
message	Success message or failure description in case of error.
appStatusCode	Application specific status code for the response. Will be non-null for failure response.
tags	More info in case of failure response.

Status Codes

HTTP Code	appStatusCode	Response Message
201 Created	null	Resource added successfully
409 Conflict	RBAC_RE_005	Resource with the given name already exists
400 Bad Request	VALIDATION_ERROR_0004	'name' should have at least '2' characters, Mandatory Field 'name' is missing or empty
400 Bad Request	VALIDATION_ERROR_0004	Invalid "name".

HTTP Code	appStatusCode	Response Message
401 Unauthorized	AVX_GW_012	Unauthorized access, reason - Invalid Token
407 Proxy Authentication Required	AVX_GW_011	Session validation failed, reason - Session information is missing.

Sample Request/Response

Use Case

Add a resource using API with Access Token.

Sample Request

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/resource?gwsource=external
```

Request Payload

```
{
 "payload": {
 "name": "resource_1",
 "description": "This is a sample resource."
 }
 }
```

Sample Response

```
{
 "response": "Resource added successfully",
 "message": "Resource added successfully",
 "appStatusCode": null,
 "tags": null,
 "headers": null
 }
```

Add New Host

The API will add a new host in host inventory.

Before you begin

Before attempting to add a new host, the user has to ensure the following:

• The host does not exist already.

Request Structure

Endpoint:	/ssh/host/create
Type:	POST
Sample URL:	https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/host/create?gwsource=external To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
Payload	(Mandatory) Contains all the parameters to be sent in the request body for the post request

Request Parameters (continued)

Name	Description
Body	**Type**: Payload

Payload

Name	Description
data	(Mandatory) Contains request parameters to create and save the host. **Type**: Data

Data

Name	Description
groupName	(Mandatory) Contains request parameters to create and save the host. **Type**: String
categoryType	(Mandatory) Category type of the host **Type**: String **Possible Values:** Server
vendorType	(Mandatory) Type of the vendor **Type**: String
ipAddress	(Mandatory) IP address of the host **Type**: String
fqdn	(Mandatory) Fully qualified domain name of the host **Type**: String
dataCenter	(Mandatory) Datacenter of the host **Type**: String
isClient	(Mandatory) Is the host client machine or not **Type**: Boolean
accessType	(Mandatory) Access type of the host

Data (continued)

Name	Description
	Type: String **Possible values**: Key, Certificate
port	(Mandatory) Port number to connect to the host from **Type**: Number
deviceName	(Mandatory) Device name of the host **Type**: String
hostName	(Mandatory) Host name **Type**: String
loginType	(Mandatory) Login type for the host **Type**: String **Possible values**: Password, Identity Key
fileContent	(Mandatory) Identity Key file content, applicable only if “Identity Key” loginType is selected **Content-Type**: application/octet-stream
fileName	(Mandatory) Name of the Identity key file, applicable only if “Identity Key” loginType is selected **Type**: String
userName	(Mandatory) Username to login to the host **Type**: String
password	(Mandatory) Password to login to the host **Type**: String
isSudoUser	(Mandatory) Is sudo user or not **Type**: Boolean
credentialType	(Mandatory) Credential type for authentication to login to the host **Type**: String **Possible values**: Manual Entry, Credential List - AppViewX, Credential List - CyberArk, Credential List - Thycotic Secret

Data (continued)

Name	Description
credentialName	(Mandatory) Credential name (null for “Manual Entry” credentialType) **Type**: String
accessGroups	(Mandatory) List of access groups where the host belongs to **Type**: List
userComplianceGroup	(Mandatory) User compliance group of the host **Type**: String
inventoryAction	(Mandatory) Inventory action for the host **Type**: String **Possible values**: Manage, Monitor, Do Not Move
accessElevation	(Mandatory) Access elevation of the user **Type**: String **Possible Values**: sudo, dzdo
sshSyncKeyDetail	(Mandatory) SSH sync key detail of the host **Type**: SshSyncKeyDetail

SshSyncKeyDetails

Name	Description
sshScanType	(Mandatory) SSH scan type **Type**: String **Possible values**: Default, Full, Directory
sshScanDetails	(Mandatory) Details of SSH scan **Type**: List
discoverKeyType	(Mandatory) List of key types to be discovered **Type**: List
appInfraAccessGroup	(Mandatory) List of app infra access groups where the host belongs to

SshSyncKeyDetails (continued)

Name	Description
	Type: List

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response message **Type**: response
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

response	Host created and saved successfully. **Type**: String

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Host created and saved successfully.
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.

Status Codes and Description (continued)

HTTP Status code	appStatusCode	Message and Possible remediation
400 Bad Request	**AVX-VLDTN-001**	Mandatory field is missing or invalid values specified - <<field name>> **Possible remediation**: Check and ensure that valid value is provided for <<field name>> field in the request.

Sample Request/Response

Use Case

To add new host using **add_new_host** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/host/create?gwsource=external
```

Sample Request 1


```
{"data":
  {"groupName":"Default_Host_Group",
  "categoryType":"client",
  "vendorType":"linux",
  "ipAddress":"1.1.1.1",
  "fqdn":"",
  "dataCenter":"absecon",
  "isClient":false,
  "accessType":"Certificate",
  "port":"22",
  "deviceName":"test-device",
  "hostName":"",
  "loginType":"password",
  "userName":"appviewx",
  "isSudoUser":true,
  "credentialType":"Manual Entry",
  "credentialName":null,
  "accessGroupDeviceType":"Host",
  "accessGroups":["Default_Infra_Access_Group"],
  "userComplianceGroup":"Default_Key_Group",
```

```
    "clientServerAccessGroups":[],
    "sshSyncKeyDetail":{"sshScanType":"default","sshScanDetails":[],"discoverKeyType":["User Keys","Host
 Keys"],"appInfraAccessGroup":["Default_Infra_Access_Group"]},
    "inventoryAction":"manage","password":"dummy pwd","accessElevation":"sudo"}
}
```


Sample Request 2


```
{
    "fileContent": (binary),
    "fileName": "demo",
    "data":
    {"groupName":"Default_Host_Group",
    "categoryType":"client",
    "vendorType":"linux",
    "ipAddress":"1.1.1.1",
    "fqdn":"",
    "dataCenter":"absecon",
    "isClient":false,
    "accessType":"Certificate",
    "port":"22",
    "deviceName":"test-device",
    "hostName":"",
    "loginType":"file",
    "userName":"appviewx",
    "isSudoUser":true,
    "credentialType":"Manual Entry",
    "credentialName":null,
    "accessGroupDeviceType":"Host",
    "accessGroups":["Default_Infra_Access_Group"],
    "userComplianceGroup":"Default_Key_Group",
    "clientServerAccessGroups":[],
    "sshSyncKeyDetail":{"sshScanType":"default","sshScanDetails":[],"discoverKeyType":["User Keys","Host
 Keys"],"appInfraAccessGroup":["Default_Infra_Access_Group"]},
    "inventoryAction":"manage","password":"dummy pwd","accessElevation":"sudo"
}
```


Sample Response

```
{
  "response": "Host created and saved successfully.",
  "message": null,
  "appStatusCode": null,
  "tags": {},
  "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

What's New

- Search Hosts
- SSH Create CA

Search Hosts

The API will search hosts and its information from the host inventory.

Before you begin

Before attempting to search hosts from the host inventory, the user has to ensure the following:

- Hosts must be added in AppViewX.

Request Structure

Endpoint:	/ssh/search/hosts
Type:	POST
Sample URL:	https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/search/hosts?gwsource=external To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String

Request Parameters (continued)

Name	Description
	Constraints: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
Payload *Body*	(Mandatory) Contains all the parameters to be sent in the request body for the post request **Type**: Payload

Payload

Name	Description
input	(Optional) Input parameters to find host information **Type**: Input
filter	(Mandatory) Filter parameters to find host information **Type**: Filter

Input

Name	Description
freeSearch	(Optional) Search text to find host information **Type**: String

Filter

Name	Description
sortColumn	(Mandatory) Column name to be sorted **Type**: String
sortOrder	(Mandatory) Order to be sorted **Possible values**: asc, desc
start	(Mandatory) Start count of the hosts to be fetched **Type**: String

Filter (continued)

Name	Description
max	(Mandatory) Count of the hosts to be fetched **Type**: String

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the hosts **Type**: response
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

Name	Description
data	List of host information which matches the search criteria **Type**: List
iTotalDisplayRecords	Total number of host available for the search criteria

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Host information retrieved successfully

Status Codes and Description (continued)

HTTP Status code	appStatusCode	Message and Possible remediation
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**AVX-VLDTN-001**	Mandatory field is missing or invalid values specified - <<field name>> **Possible remediation**: Check and ensure that valid value is provided for <<field name>> field in the request.

Sample Request/Response

Use Case

To search for hosts using **search_hosts** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/search/hosts?gwsource=external
```

Sample Request

```
{
 "input": {
   "freeSearch": "pe-cert-apvx-node05"
 },
 "filter": {
   "sortColumn": "none",
   "sortOrder": "desc",
   "start": "0",
   "max": "100"
 }
}
```

Sample Response

```
{
 "response": {
```

```
"data": [
 {
  "sshHost": {
   "credentialType": "Manual Entry",
   "fqdn": "pe-cert-apvx-node05.lab.appviewx.net.",
   "ipAddress": "192.168.60.133",
   "userName": "appviewx",
   "loginType": "password",
   "groupName": "Default_Host_Group",
   "deviceName": "pe-cert-apvx-node05",
   "port": "22",
   "accessElevation": "sudo",
   "hostName": "",
   "fingerPrint": "",
   "status": "Unresolved",
   "displayName": "192.168.60.133",
   "vendorType": "linux",
   "category": "Host",
   "keyWords": [
    "Unresolved",
    "22",
    "NA",
    "192.168.60.133",
    "pe-cert-apvx-node05",
    "appviewx",
    "Default_Host_Group",
    "Host",
    "sam"
   ],
   "groupIds": [
    "5767bcdc3465bfbf73e44726"
   ],
   "createdTime": 1716221229762,
   "adcAccess": false,
   "isClient": false,
   "isBastion": false,
   "deviceType": [
```

```
    "NA"
  ],
  "deviceStatusList": [
    {
      "title": "Network Status: Reachability Check",
      "message": "FQDN pe-cert-apvx-node05.lab.appviewx.net. resolution failed.",
      "status": "Failed",
      "timeStamp": 1716221234111
    }
  ],
  "isSudoUser": true,
  "isValidSudo": false,
  "lastSyncTime": 1716221229526,
  "sshKeysCount": 0,
  "certificatesCount": 0,
  "communicationMode": "SSH",
  "accessGroups": [
    "sam"
  ],
  "accessGroupDeviceType": "Host",
  "clientServerAccessGroups": [],
  "categoryType": "server",
  "userComplianceGroup": "Default_Key_Group",
  "inventoryAction": "manage",
  "sshSyncKeyDetail": {
    "autoCreateHPGroup": false,
    "sshSyncEnabled": false,
    "appInfraAccessGroup": [
      "sam"
    ],
    "sshScanType": "default",
    "sshScanDetails": [],
    "discoverKeyType": [
      "User Keys",
      "Host Keys"
    ],
    "fromSSHSubsystem": false
```



```
        },
        "dataCenter": "absecon",
        "accessType": "Certificate",
        "discoverySeqId": 0,
        "discoveryBatchNo": 0,
        "allowedPrincipalCount": 0,
        "sshOrigin": true,
        "update": false,
        "new": false,
        "selected": false,
        "active": true,
        "_id": "664b752d6c117a54017bc2f3"
      },
      "groups": [
        {
          "Default_Host_Group": "RW"
        }
      ],
      "displayName": "192.168.60.133",
      "fingerPrint": "",
      "permission": "RW",
      "groupPermissions": {
        "sam": "RW"
      },
      "_id": "664b752d6c117a54017bc2f3"
    }
  ],
  "iTotalDisplayRecords": 1
 },
 "message": "Host information retrieved successfully",
 "appStatusCode": null,
 "tags": {},
 "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

What's New

- Search Host Keys
- Search User Keys
- SSH Create CA

Search Host Keys

The API will search host keys and its information from the host key inventory.

Before you begin

Before attempting to searching host keys from the host key inventory, the user has to ensure the following:

- Host keys must be present in AppViewX.

Request Structure

Endpoint:	`/ssh/search/hostKeys`
Type:	`POST`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/search/hostKeys?gwsource=external` To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
Payload *Body*	(Mandatory) Contains all the parameters to be sent in the request body for the post request **Type**: Payload

Payload

Name	Description
input	(Optional) Input parameters to fetch host keys **Type**: Input
filter	(Mandatory) Filter parameters to fetch host keys **Type**: Filter

Input

Name	Description
freeSearch	(Optional) Search text to find host key information **Type**: String
keywordSearch	(Optional) Keyword and value to search and retrieve host key information Example: {"keyname":"RotateKeys_admin_1716367661908-B0-001"}

Filter

Name	Description
sortColumn	(Mandatory) Column name to be sorted **Type**: String
sortOrder	(Mandatory) Order to be sorted **Possible values**: asc, desc
start	(Mandatory) Start count of the host keys to be fetched **Type**: String
max	(Mandatory) Count of the host keys to be fetched **Type**: String

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the host keys **Type**: response
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

Name	Description
data	List of host key information which matches the search criteria **Type**: List
iTotalDisplayRecords	Total number of host key available for the search criteria

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Host keys retrieved successfully
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**AVX-VLDTN-001**	Mandatory field is missing or invalid values specified - <<field name>>

Status Codes and Description (continued)

HTTP Status code	appStatusCode	Message and Possible remediation
		Possible remediation: Check and ensure that valid value is provided for <<field name>> field in the request.

Sample Request/Response

Use Case

To search for host keys using **search_host_keys** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/search/hostKeys?gwsource=external
```

Sample Request 1

```
{
  "input": {
    "freeSearch": "RotateKeys_admin_1716367184410-B0-001"
  },
  "filter": {
    "sortColumn": "none",
    "sortOrder": "desc",
    "start": "0",
    "max": "100"
  }
}
```

Sample Request 2

```
{
  "input": {
    "keywordSearch":{"keyname":"RotateKeys_admin_1716367184410-B0-001"}
  },
  "filter": {
    "sortColumn": "none",
    "sortOrder": "desc",
    "start": "0",
    "max": "100"
  }
```

```
}
```

Sample Response


```
{
  "response": {
    "data": [
      {
        "sshkey": {
          "type": "ECDSA",
          "name": "RotateKeys_admin_1716367184410-B0-001",
          "bitLength": "256",
          "passPhrase": null,
          "passPhraseCryptKey": null,
          "comment": "Created for host 192.168.60.130",
          "privateKey": null,
          "cryptKeyForPrivateKey": null,
          "publicKey": null,
          "fingerPrint": "+1cEqAhMLynje99v9hjeB3QEooh1sqEPQH1303d5P8o",
          "keyStatus": "Managed",
          "compliance": null,
          "groupName": "Default_Key_Group",
          "validity": null,
          "period": "lifetime",
          "hsmDeviceName": null,
          "uuid": "81da80bf-e20f-358e-8606-34b63d8d61c1",
          "keyWords": [
            "discoveredKeys",
            "Created for host 192.168.60.130",
            "256",
            "pe-cert-apvx-node02",
            "RotateKeys_admin_1716367184410-B0-001",
            "ECDSA",
            "Compliant"
          ],
          "active": true,
          "fileName": null,
          "privateKeyFilePermission": null,
          "publicFilePermission": null,
```



```
"createdBy": "discoveredKeys",
"displayName": null,
"awsDiscoveredKey": true,
"userName": null,
"associatedUsers": [],
"clientFingerprint": null,
"modifiedBy": null,
"modifiedTime": 1716367224,
"fingerPrints": null,
"isModified": null,
"colorCode": null,
"oldPassPhrase": null,
"oldPassPhraseCryptKey": null,
"awsFingerPrint": null,
"sourceIp": [],
"destIp": [
  "192.168.60.130"
],
"createdTime": 1716366952000,
"isPushAutomatically": null,
"isRotateAutomatically": null,
"autoRotate": null,
"workOrderDetail": null,
"groupIds": [
  "5767bcef3465bfbf73e44727"
],
"readWrite": false,
"keyPath": null,
"renewDate": 0,
"expiryDate": 0,
"isExpired": false,
"hostUserName": null,
"workOrderMandate": null,
"initiatedTime": 1716366952000,
"complianceDescription": null,
"agentUuid": null,
"keyType": null,
```



```
"filePaths": [
  "pe-cert-apvx-node02~~/etc/ssh/appviewxssh/ssh_host_ECDSA_key",
  "pe-cert-apvx-node02~~/etc/ssh/appviewxssh/ssh_host_ECDSA_key.pub"
],
"symLinks": [],
"privateKeyFileProps": {
  "readable": "true",
  "executable": "false",
  "writable": "true"
},
"publicKeyFileProps": {
  "readable": "true",
  "executable": "false",
  "writable": "true"
},
"clientMachineNames": [],
"serverMachineNames": [
  "pe-cert-apvx-node02"
],
"keyFoundInDiscoverySeqIdRanges": [
  {
    "from": 30,
    "to": 30
  }
],
"firstDiscovery": 1716367184461,
"previousDiscovery": 0,
"currentDiscoveredTime": 1716367184461,
"certificate": [
  {
    "fingerPrint": "+1cEqAhMLynje99v9hjeB3QEooh1sqEPQH1303d5P8o",
    "certType": "Host",
    "rawFPString": "ECDSA-CERT SHA256:+1cEqAhMLynje99v9hjeB3QEooh1sqEPQH1303d5P8o",
    "signingCA": "ECDSA SHA256:k/k8+W/SlzdaK0ajyat/l3FVpnWpPpndaP0qTM7lf7M (using ecdsa-sha2-nistp256)",
    "keyId": "pe-cert-apvx-node02.lab.appviewx.net",
    "serialNumber": "3515119686734784",
    "rawCertType": "ecdsa-sha2-nistp256-cert-v01@openssh.com host certificate",
```

```
      "validFrom": 1716366944000,
      "validTo": 1747902944000,
      "validity": 364,
      "validityUnit": "days",
      "expiresIn": "364 days",
      "principals": [
        "192.168.60.130",
        "pe-cert-apvx-node02.lab.appviewx.net"
      ],
      "cryptKeyForCertContent": "opj82wtc1bylx4igskt7ra724",
      "certStatus": "Active",
      "filePaths": [
        {
          "hostName": "pe-cert-apvx-node02",
          "paths": [
            "pe-cert-apvx-node02~~/etc/ssh/appviewxssh/ssh_host_ECDSA_key-cert.pub"
          ]
        }
      ],
      "onlyForWebTerminal": false
    }
  ],
  "sharedType": "single",
  "excludeFromSharedKeyReportEndTime": 0,
  "excludeFromWeakKeyReportEndTime": 0,
  "excludeFromOrphanKeyReportEndTime": 0,
  "excludeFromSuspiciousKeyReportEndTime": 0,
  "discoveryIdWithNewState": {
    "30": true
  },
  "eligibleForRollback": false,
  "backupData": null,
  "sharedKey": false,
  "weakKey": false,
  "riskKey": false,
  "discovered": true,
  "accessRequest": false,
```

```
    "new": true,
    "keyDownload": false,
    "privateKeyDeleted": false,
    "publicKeyDeleted": false,
    "keyFilePermission": [
     {
      "user": null,
      "userHomeDirectory": null,
      "userGroup": null,
      "filePath": null,
      "deviceName": null,
      "fileProperties": null
     }
    ],
    "selected": false,
    "upload": false,
    "passphraseValidated": false,
    "hasPrivateKey": true,
    "_id": "664daf7868cf79570aab88b5"
   },
   "age": "0 day",
   "clientMachineNames": null,
   "serverMachineNames": null,
   "groupPermission": [
    {
     "Default_Key_Group": "RW"
    }
   ],
   "permission": "RW",
   "compliance": "Compliant",
   "createdTime": 0,
   "displayName": "RotateKeys_admin_1716367184410-B0-001",
   "hostComplianceGroup": null,
   "joinedHostGroups": null,
   "hostName": null,
   "associatedUsers": null,
   "colorCode": "newKeys",
```



```
      "complianceDescription": "",
      "hostInfos": null,
      "keyComplianceGroup": null,
      "accessGroup": null,
      "selected": false,
      "_id": null
    }
  ],
  "iTotalDisplayRecords": 1,
  "serverTime": 1716377205029
 },
 "message": "User keys retrieved successfully",
 "appStatusCode": null,
 "tags": null,
 "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL

- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

What's New

- Search Hosts
- Search User Keys
- SSH Create CA

Search User Keys

The API will search user keys and its information from the user key inventory.

Before you begin

Before attempting to searching user keys from the user key inventory, the user has to ensure the following:

- User keys must be present in AppViewX.

Request Structure

Endpoint:	/ssh/search/userKeys
Type:	POST
Sample URL:	https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/search/userKeys?gwsource=external To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username	(Mandatory) AppViewX login username

Request Parameters (continued)

Name	Description
Header	**Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
Payload *Body*	(Mandatory) Contains all the parameters to be sent in the request body for the post request **Type**: Payload

Payload

Name	Description
input	(Optional) Input parameters to fetch user keys **Type**: Input
filter	(Mandatory) Filter parameters to fetch user keys **Type**: Filter

Input

Name	Description
freeSearch	(Optional) Search text to find user key information **Type**: String
keywordSearch	(Optional) Keyword and value to search and retrieve user key information

Input (continued)

Name	Description
	Example: {"keyname":"RotateKeys_admin_1716367661908-B0-001"}

Filter

Name	Description
sortColumn	(Mandatory) Column name to be sorted **Type**: String
sortOrder	(Mandatory) Order to be sorted **Possible values**: asc, desc
start	(Mandatory) Start count of the user keys to be fetched **Type**: String
max	(Mandatory) Count of the user keys to be fetched **Type**: String

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the user keys **Type**: response
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

Name	Description
data	List of user key information which matches the search criteria **Type**: List
iTotalDisplayRecords	Total number of user key available for the search criteria

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	User keys retrieved successfully
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**AVX-VLDTN-001**	Mandatory field is missing or invalid values specified - <<field name>> **Possible remediation**: Check and ensure that valid value is provided for <<field name>> field in the request.

Sample Request/Response

Use Case

To search for user keys using **search_user_keys** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/search/userKeys?gwsource=external
```

Sample Request 1

```
{
 "input": {
  "freeSearch": "FetchKey_admin_1716211635851-B0-022"
 },
 "filter": {
  "sortColumn": "none",
```

```
    "sortOrder": "desc",
    "start": "0",
    "max": "100"
  }
}
```


Sample Request 2


```
{
  "input": {
    "keywordSearch":{"keyname":"FetchKey_admin_1716211635851-B0-022"}
  },
  "filter": {
    "sortColumn": "none",
    "sortOrder": "desc",
    "start": "0",
    "max": "100"
  }
}
```


Sample Response


```
{
  "response": {
    "data": [
      {
        "sshkey": {
          "type": "RSA",
          "name": "FetchKey_admin_1716211635851-B0-022",
          "bitLength": "1024",
          "passPhrase": null,
          "passPhraseCryptKey": null,
          "comment": "demo_may7",
          "privateKey": null,
          "cryptKeyForPrivateKey": null,
          "publicKey": null,
          "fingerPrint": "HXtVvirhDbslJottYGjpQsFgKL/Q1KtSxWx4A+dIMhI",
          "keyStatus": "Managed",
          "compliance": null,
          "groupName": "Default_Key_Group",
```

```
"validity": null,
"period": "lifetime",
"hsmDeviceName": null,
"uuid": "14383066-7481-3cbc-947f-39cc19419278",
"keyWords": [
  "FetchKey_admin_1716211635851-B0-022",
  "RSA",
  "discoveredKeys",
  "1024",
  "pe-cert-apvx-node02",
  "demo_may7",
  "Compliant"
],
"active": true,
"fileName": null,
"privateKeyFilePermission": null,
"publicFilePermission": null,
"createdBy": "discoveredKeys",
"displayName": null,
"awsDiscoveredKey": true,
"userName": null,
"associatedUsers": [
  "pe-cert-apvx-node02~~appviewx"
],
"clientFingerprint": null,
"modifiedBy": null,
"modifiedTime": 1716367224,
"fingerPrints": null,
"isModified": null,
"colorCode": null,
"oldPassPhrase": null,
"oldPassPhraseCryptKey": null,
"awsFingerPrint": null,
"sourceIp": [
  "192.168.60.130"
],
"destIp": [],
```



```
"createdTime": 1715086510000,
"isPushAutomatically": null,
"isRotateAutomatically": null,
"autoRotate": null,
"workOrderDetail": null,
"groupIds": [
  "5767bcef3465bfbf73e44727"
],
"readWrite": false,
"keyPath": null,
"renewDate": 0,
"expiryDate": 0,
"isExpired": false,
"hostUserName": null,
"workOrderMandate": null,
"initiatedTime": 1715086510000,
"complianceDescription": null,
"agentUuid": null,
"keyType": "REGULAR",
"filePaths": [
  "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7",
  "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7.pub"
],
"symLinks": [
  {
    "deviceName": "pe-cert-apvx-node02",
    "sourcePath": "/home/appviewx/.ssh/demomay7",
    "targetPath": "/home/appviewx/.ssh/appviewxssh/id_RSA_20240507_125500_639_0"
  },
  {
    "deviceName": "pe-cert-apvx-node02",
    "sourcePath": "/home/appviewx/.ssh/demomay7.pub",
    "targetPath": "/home/appviewx/.ssh/appviewxssh/id_RSA_20240507_125500_639_0.pub"
  }
],
"privateKeyFileProps": {
  "readable": "true",
```

```
  "executable": "false",
  "writable": "true"
 },
 "publicKeyFileProps": {
  "readable": "true",
  "executable": "false",
  "writable": "true"
 },
 "clientMachineNames": [
  "pe-cert-apvx-node02"
 ],
 "serverMachineNames": [],
 "keyFoundInDiscoverySeqIdRanges": [
  {
   "from": 21,
   "to": 21
  },
  {
   "from": 26,
   "to": 26
  },
  {
   "from": 28,
   "to": 30
  }
 ],
 "firstDiscovery": 1716211635872,
 "previousDiscovery": 1716366860849,
 "currentDiscoveredTime": 1716367184461,
 "certificate": [],
 "sharedType": "single",
 "excludeFromSharedKeyReportEndTime": 0,
 "excludeFromWeakKeyReportEndTime": 0,
 "excludeFromOrphanKeyReportEndTime": 0,
 "excludeFromSuspiciousKeyReportEndTime": 0,
 "discoveryIdWithNewState": {
  "21": true,
```

```
    "26": true,
    "28": true,
    "29": true,
    "30": true
  },
  "eligibleForRollback": false,
  "backupData": null,
  "sharedKey": false,
  "weakKey": true,
  "riskKey": true,
  "discovered": true,
  "accessRequest": false,
  "new": true,
  "keyDownload": false,
  "privateKeyDeleted": false,
  "publicKeyDeleted": false,
  "keyFilePermission": [
   {
     "user": "appviewx",
     "userHomeDirectory": "/home/appviewx",
     "userGroup": "appviewx",
     "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7.pub",
     "deviceName": "pe-cert-apvx-node02",
     "fileProperties": [
       {
         "readable": true,
         "writable": false,
         "executable": false,
         "permissionFor": "Group"
       },
       {
         "readable": true,
         "writable": false,
         "executable": false,
         "permissionFor": "Others"
       },
       {
```

```
      "readable": true,
      "writable": true,
      "executable": false,
      "permissionFor": "Owner"
    }
  ]
},
{
  "user": "appviewx",
  "userHomeDirectory": "/home/appviewx",
  "userGroup": "appviewx",
  "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7",
  "deviceName": "pe-cert-apvx-node02",
  "fileProperties": [
    {
      "readable": true,
      "writable": true,
      "executable": false,
      "permissionFor": "Owner"
    },
    {
      "readable": false,
      "writable": false,
      "executable": false,
      "permissionFor": "Group"
    },
    {
      "readable": false,
      "writable": false,
      "executable": false,
      "permissionFor": "Others"
    }
  ]
},
{
  "user": "appviewx",
  "userHomeDirectory": "/home/appviewx",
```



```
    "userGroup": "appviewx",
    "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7",
    "deviceName": "pe-cert-apvx-node02",
    "fileProperties": [
      {
        "readable": true,
        "writable": true,
        "executable": false,
        "permissionFor": "Owner"
      },
      {
        "readable": false,
        "writable": false,
        "executable": false,
        "permissionFor": "Others"
      },
      {
        "readable": false,
        "writable": false,
        "executable": false,
        "permissionFor": "Group"
      }
    ]
  },
  {
    "user": "appviewx",
    "userHomeDirectory": "/home/appviewx",
    "userGroup": "appviewx",
    "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7",
    "deviceName": "pe-cert-apvx-node02",
    "fileProperties": [
      {
        "readable": false,
        "writable": false,
        "executable": false,
        "permissionFor": "Others"
      },
```

```
    {
      "readable": true,
      "writable": true,
      "executable": false,
      "permissionFor": "Owner"
    },
    {
      "readable": false,
      "writable": false,
      "executable": false,
      "permissionFor": "Group"
    }
  ]
},
{
  "user": "appviewx",
  "userHomeDirectory": "/home/appviewx",
  "userGroup": "appviewx",
  "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7.pub",
  "deviceName": "pe-cert-apvx-node02",
  "fileProperties": [
    {
      "readable": true,
      "writable": true,
      "executable": false,
      "permissionFor": "Owner"
    },
    {
      "readable": true,
      "writable": false,
      "executable": false,
      "permissionFor": "Group"
    },
    {
      "readable": true,
      "writable": false,
      "executable": false,
```



```
      "permissionFor": "Others"
    }
  ]
},
{
  "user": "appviewx",
  "userHomeDirectory": "/home/appviewx",
  "userGroup": "appviewx",
  "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7.pub",
  "deviceName": "pe-cert-apvx-node02",
  "fileProperties": [
    {
      "readable": true,
      "writable": false,
      "executable": false,
      "permissionFor": "Group"
    },
    {
      "readable": true,
      "writable": true,
      "executable": false,
      "permissionFor": "Owner"
    },
    {
      "readable": true,
      "writable": false,
      "executable": false,
      "permissionFor": "Others"
    }
  ]
},
{
  "user": "appviewx",
  "userHomeDirectory": "/home/appviewx",
  "userGroup": "appviewx",
  "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7.pub",
  "deviceName": "pe-cert-apvx-node02",
```



```
    "fileProperties": [
      {
        "readable": true,
        "writable": false,
        "executable": false,
        "permissionFor": "Group"
      },
      {
        "readable": true,
        "writable": false,
        "executable": false,
        "permissionFor": "Others"
      },
      {
        "readable": true,
        "writable": true,
        "executable": false,
        "permissionFor": "Owner"
      }
    ]
  },
  {
    "user": "appviewx",
    "userHomeDirectory": "/home/appviewx",
    "userGroup": "appviewx",
    "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7",
    "deviceName": "pe-cert-apvx-node02",
    "fileProperties": [
      {
        "readable": true,
        "writable": true,
        "executable": false,
        "permissionFor": "Owner"
      },
      {
        "readable": false,
        "writable": false,
```

```
      "executable": false,
      "permissionFor": "Others"
    },
    {
      "readable": false,
      "writable": false,
      "executable": false,
      "permissionFor": "Group"
    }
  ]
},
{
  "user": "appviewx",
  "userHomeDirectory": "/home/appviewx",
  "userGroup": "appviewx",
  "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7",
  "deviceName": "pe-cert-apvx-node02",
  "fileProperties": [
    {
      "readable": false,
      "writable": false,
      "executable": false,
      "permissionFor": "Others"
    },
    {
      "readable": true,
      "writable": true,
      "executable": false,
      "permissionFor": "Owner"
    },
    {
      "readable": false,
      "writable": false,
      "executable": false,
      "permissionFor": "Group"
    }
  ]
```

```
    },
    {
      "user": "appviewx",
      "userHomeDirectory": "/home/appviewx",
      "userGroup": "appviewx",
      "filePath": "pe-cert-apvx-node02~~/home/appviewx/.ssh/demomay7.pub",
      "deviceName": "pe-cert-apvx-node02",
      "fileProperties": [
        {
          "readable": true,
          "writable": false,
          "executable": false,
          "permissionFor": "Group"
        },
        {
          "readable": true,
          "writable": false,
          "executable": false,
          "permissionFor": "Others"
        },
        {
          "readable": true,
          "writable": true,
          "executable": false,
          "permissionFor": "Owner"
        }
      ]
    }
  ],
  "selected": false,
  "upload": false,
  "passphraseValidated": false,
  "hasPrivateKey": true,
  "_id": "664b4fd9e8eaab668f8a51d9"
},
"age": "14 days",
"clientMachineNames": null,
```

```
      "serverMachineNames": null,
      "groupPermission": [
       {
         "Default_Key_Group": "RW"
       }
      ],
      "permission": "RW",
      "compliance": "Compliant",
      "createdTime": 0,
      "displayName": "FetchKey_admin_1716211635851-B0-022",
      "hostComplianceGroup": null,
      "joinedHostGroups": null,
      "hostName": null,
      "associatedUsers": null,
      "colorCode": "lifetime",
      "complianceDescription": "",
      "hostInfos": null,
      "keyComplianceGroup": null,
      "accessGroup": null,
      "selected": false,
      "_id": null
    }
  ],
  "iTotalDisplayRecords": 1,
  "serverTime": 1716376908710
 },
 "message": "User keys retrieved successfully",
 "appStatusCode": null,
 "tags": null,
 "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

What's New

- Search Host Keys
- SSH Create CA

Search Access Groups

The API will search access groups and its information from the access group inventory.

Before you begin

Before attempting to searching access groups from the access groups inventory, the user has to ensure the following:

- Access groups should be present in AppViewX.

Request Structure

Endpoint:	`/ssh/app-infra-group/list`

Type:	POST
Sample URL:	https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/app-infra-group/list?gwsource=external To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
Payload *Body*	(Mandatory) Contains all the parameters to be sent in the request body for the post request **Type**: Payload

Payload

Name	Description
input	(Optional) Input parameters to fetch access groups **Type**: Input
filter	(Mandatory) Filter parameters to fetch access groups **Type**: Filter

Input

Name	Description
freeSearch	(Optional) Search text to find access group information **Type**: String

Filter

Name	Description
sortColumn	(Mandatory) Column name to be sorted **Type**: String
sortOrder	(Mandatory) Order to be sorted **Possible values**: asc, desc
start	(Mandatory) Start count of the access groups to be fetched **Type**: String
max	(Mandatory) Count of the access groups to be fetched **Type**: String

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the access groups **Type**: response

Response Parameters (continued)

Name	Description
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

Name	Description
data	List of access group information which matches the search criteria **Type**: List
iTotalDisplayRecords	Total number of access group available for the search criteria

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Access groups retrieved successfully
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**AVX-VLDTN-001**	Mandatory field is missing or invalid values specified - <<field name>> **Possible remediation**: Check and ensure that valid value is provided for <<field name>> field in the request.

Sample Request/Response

Use Case

To search for access groups using **search_access_groups** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/app-infra-group/list?gwsource=external
```

Sample Request

```
{
  "input": {
    "freeSearch": "Default_Infra_Access_Group"
  },
  "filter": {
    "sortColumn": "none",
    "sortOrder": "desc",
    "start": "0",
    "max": "100"
  }
}
```

Sample Response

```
{
  "response": {
    "data": [
      {
        "name": "Default_Infra_Access_Group",
        "description": "Default Infra Access group",
        "creationType": "Auto",
        "associatedHostCount": 1,
        "rolePermissionMap": {
          "super access": "RW"
        },
        "keyWords": [
          "Auto",
          "Default Infra Access group",
          "Default_Infra_Access_Group"
        ],
        "userCAName": "Default-Infra-Access-Group.user.ca",
```

```
      "hostCAName": "appviewx.ssh.host.ca",
      "permission": "RW",
      "selected": false,
      "_id": "66027e3b6a91df51fa2c5d62"
    }
  ],
  "iTotalDisplayRecords": 1
 },
 "message": "Access groups retrieved successfully",
 "appStatusCode": null,
 "tags": null,
 "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

What's New

- Search CA
- SSH Create CA
- SSH Download CA
- SSH Get Hosts From Infra Access Group

SSH Create CA

The API will create CA based on the given request.

Before you begin

Before attempting to creating CAs, the user has to ensure the following:

- Proper values given in the request.

Request Structure

Endpoint:	`/ssh/ca/create`
Type:	`POST`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/ca/create?gwsource=external` To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.

Request Parameters (continued)

Name	Description
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
Payload *Body*	(Mandatory) Contains all the parameters to be sent in the request body for the post request **Type**: Payload

Payload

Name	Description
name	(Mandatory) Name of the CA **Type**: String
validityValue	(Mandatory) Validity of the certificate **Type**: String
validityUnit	(Mandatory) Validity unit of the certificate **Type**: String
caType	(Mandatory) Type of the CA **Type**: String
comment	(Optional) Comments **Type**: String
bitLength	(Mandatory) Bit length for CA

Name	Description
	Type: String
algorithm	(Mandatory) Algorithm for CA **Type**: String

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the CAs **Type**: response
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

Name	Description
id	ID of the CA **Type**: String
name	Name of the CA **Type**: String
caType	CA Type of the CA **Type**: String
comment	Comment **Type**: String

Response (continued)

Name	Description
renewEnabled	Renewal enabled for CA **Type**: String
publicKey	Public key of the CA **Type**: String
keyRevocationListStatus	KRL status of the CA **Type**: String
algorithm	Algorithm of the CA **Type**: String
bitLength	Bit length of the CA **Type**: String
validityValue	Validity of the certificate **Type**: String
validityUnit	Validity unit of the CA **Type**: String
expiryDate	Expiry date of the CA **Type**: String

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Success
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**ERR-SSH_NB-262**	Invalid Key Algorithm or BitLength

Status Codes and Description (continued)

HTTP Status code	appStatusCode	Message and Possible remediation
		Possible remediation: Make sure to provide proper key type and bit length
	ERR-SSH_NB-212	CA name already exists **Possible remediation**: Please make sure to provide different CA name

Sample Request/Response

Use Case

To create CA using **create_CA** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/ca/create?gwsource=external
```

Sample Request

```
{
    "name": "dev-ssh.user.ca",
    "validityValue": "12",
    "validityUnit": "months",
    "caType": "user",
    "comment": "This is a sample env user CA",
    "bitLength": "256",
    "algorithm": "ECDSA"
}
```

Sample Response

```
{
    "response": {
        "id": "664fc8ea0eea9f4a4438d1f0",
        "name": "dev-ssh.user.ca",
        "validityValue": 12,
        "validityUnit": "months",
        "caType": "User",
```

```
      "comment": "This is a sample env user CA",
      "renewEnabled": false,
      "publicKey": "dummy key",
      "cryptKeyForPublicKey": "dummy key",
      "keyRevocationListStatus": "Not created",
      "algorithm": "ECDSA",
      "bitLength": 256,
      "status": "Active",
      "expiryDate": 1748044799000
    },
    "message": "Success",
    "appStatusCode": "SSH-NB-200",
    "tags": {},
    "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL

- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

What's New

- SSH Download CA
- SSH Create Certificate

SSH Download CA

The API will download CA based on the given request.

Before you begin

Before attempting to downloading CAs, the user has to ensure the following:

- CA should be present in AppViewX.

Request Structure

Endpoint:	`/ssh/ca/download`
Type:	`GET`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/ca/download?gwsource=external` To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String

Request Parameters (continued)

Name	Description
	Constraints: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
caName *queryParam*	(Mandatory) Name of the CA **Type**: String
encode *queryParam*	(Mandatory) Encode response **Type**: Boolean

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the CA content **Type**: response
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String

Response Parameters (continued)

Name	Description
tags	More info in case of failure response

Response

Name	Description
content	Content of the CA **Type**: String

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	CA downloaded successfully
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**ERR-SSH_NB-247**	Invalid query param found for <FieldName> **Possible remediation**: Please make sure to provide valid values
	ERR-SSH_NB450	Failed to download a SSH CA Certificate. **Possible remediation**: Please make sure to provide proper values in request

Sample Request/Response

Use Case

To download CA using **download_CA** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/ca/download?gwsource=external
```

Sample URL

https://pe-ssh-apvx-n4.lab.appviewx.net:31443/avxapi/ssh/ca/download?caName=dev-ssh.user.ca&encode=false&gwsource=internal

Sample Response

CA content will be downloaded

Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

Search CA

The API will search CAs and its information.

Before you begin

Before attempting to searching CAs, the user has to ensure the following:

- CAs should be present in AppViewX.

Request Structure

Endpoint:	/ssh/ca/list
Type:	POST
Sample URL:	https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/ca/list?gwsource=external To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
Payload *Body*	(Mandatory) Contains all the parameters to be sent in the request body for the post request **Type**: Payload

Payload

Name	Description
input	(Optional) Input parameters to fetch CAs **Type**: Input
filter	(Mandatory) Filter parameters to fetch CAs **Type**: Filter

Input

Name	Description
freeSearch	(Optional) Search text to find CA information **Type**: String

Filter

Name	Description
sortColumn	(Mandatory) Column name to be sorted **Type**: String
sortOrder	(Mandatory) Order to be sorted **Possible values**: asc, desc
start	(Mandatory) Start count of the CAs to be fetched **Type**: String
max	(Mandatory) Count of the CAs to be fetched **Type**: String

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the CAs **Type**: response

Response Parameters (continued)

Name	Description
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

Name	Description
data	List of CA information which matches the search criteria **Type**: List
iTotalDisplayRecords	Total number of CAs available for the search criteria

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	CAs retrieved successfully
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**AVX-VLDTN-001**	Mandatory field is missing or invalid values specified - <<field name>> **Possible remediation**: Check and ensure that valid value is provided for <<field name>> field in the request.

Sample Request/Response

Use Case

To search for CAs using **search_CAs** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/ca/list?gwsource=external
```

Sample Request

```
{
  "input": {
    "freeSearch": "testVendor.user.ca"
  },
  "filter": {
    "sortColumn": "none",
    "sortOrder": "desc",
    "start": "0",
    "max": "100"
  }
}
```

Sample Response

```
{
  "response": {
    "data": [
      {
        "id": "664b4f18e8eaab668f8a5199",
        "name": "testVendor.user.ca",
        "validityValue": 5,
        "validityUnit": "years",
        "caType": "User",
        "renewEnabled": false,
        "keyRevocationListStatus": "Not created",
        "algorithm": "ECDSA",
        "bitLength": 256,
        "status": "Active",
        "expiryDate": 1874015999000
      }
```

```
  ],
  "iTotalDisplayRecords": 1
 },
 "message": "CAs retrieved successfully",
 "appStatusCode": "SSH-NB-200",
 "tags": {},
 "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

What's New

- Search CA

SSH Create Certificate

The API will create certificates based on the given request.

Before you begin

Before attempting to creating certificates, the user has to ensure the following:

- CAs should be present in AppViewX.

Request Structure

Endpoint:	/ssh/cert/create
Type:	POST
Sample URL:	https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/cert/create?gwsource=external To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String

Request Parameters (continued)

Name	Description
	Constraints: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
Payload *Body*	(Mandatory) Contains all the parameters to be sent in the request body for the post request **Type**: Payload

Payload

Name	Description
publicKey	(Mandatory) Public key to create certificate **Type**: String
validityValue	(Mandatory) Validity of the certificate **Type**: String
validityUnit	(Mandatory) Validity unit of the certificate **Type**: String
certificateIdentity	(Mandatory) Identity of the certificate **Type**: String
caName	(Mandatory) CA name of the certificate **Type**: String
principals	(Mandatory) Principals of the certificate **Type**: List
certType	(Mandatory) Type of the certificate **Type**: String (User or Host)
validFrom	(Optional) Certificate valid from value **Type**: Long
validTo	(Optional) Certificate valid to value

Name	Description
	Type: Long

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the certificate **Type**: response
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

Name	Description
fingerPrint	Fingerprint of the certificate **Type**: String
certType	Type of the certificate **Type**: String
rawFPString	Raw fingerprint of the certificate **Type**: String
signingCA	Signing CA of the certificate **Type**: String
keyId	Key ID of the certificate **Type**: String

Response (continued)

Name	Description
serialNumber	Serial number of the certificate **Type**: String
rawCertType	Raw cert type of the certificate **Type**: String
validFrom	Valid from value of the certificate **Type**: String
validTo	Valid to value of the certificate **Type**: String
validity	Validity of the certificate **Type**: String
validityUnit	Validity unit of the certificate **Type**: String
principals	Principals of the certificate **Type**: String
extensions	Extensions of the certificate **Type**: String
certContent	Content of the certificate **Type**: String
caName	CA name of the certificate **Type**: String
certStatus	Status of the certificate **Type**: String

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Success
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**ERR-SSH_NB-268**	Valid certificate already exists for the given Key **Possible remediation**: Valid certificate is available for the given public key. Please make sure to use different public key
	ERR-SSH_NB-268	Valid certificate already exists for the given Key **Possible remediation**: Valid certificate is available for the given public key. Please make sure to use different public key
	ERR-SSH_NB-267	Selected CA is not in Active status **Possible remediation**: Please make sure to provide active CA name
	ERR-SSH_NB-263	Selected CA cannot sign requested Cert Type **Possible remediation**: Please provide proper CA name
	ERR-SSH_NB-266	Validity start cannot exceed validity end **Possible remediation**: Possible remediation: Make sure to provide start value less than end value
	ERR-SSH_NB-269	validityValue must be greater than zero **Possible remediation**: validityValue field should not be 0 or less than 0
	ERR-SSH_NB-264	Certificate validity cannot exceed CA validity

Status Codes and Description (continued)

HTTP Status code	appStatusCode	Message and Possible remediation
		Possible remediation: Make sure to give the certificate validity less than CA validity
	ERR-SSH_NB-273	Certificate validity end cannot be past date **Possible remediation**: Make sure to give the certificate validity in future
404 Not Found	**ERR-SSH-NB-350**	No CA found with given name or ID **Possible remediation**: Please make sure to provide the available CA name

Sample Request/Response

Use Case

To create certificate using **create_certificate** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/cert/create?gwsource=external
```

Sample Request

```
{
    "publicKey": "dummy key",
    "validityValue" : 1,
    "validityUnit" : "months",
    "certificateIdentity":"admin",
    "caName": "testVendor.user.ca",
    "principals": ["admin"],
    "certType" : "User"
}
```

Sample Response

```
{
   "response": {
      "fingerPrint": "AwQpO14LR7d1e9BubBGWcDmIoifmmrxC3M2xyceDO2s",
      "certType": "User",
      "rawFPString": "ECDSA-CERT SHA256:AwQpO14LR7d1e9BubBGWcDmIoifmmrxC3M2xyceDO2s",
```

```
    "signingCA": "ECDSA SHA256:A/Pfc4Se53vBzIIstXbTWVmRy5u7n8mBiySryI+UZgI (using ecdsa-sha2-nistp256)",
    "keyId": "admin",
    "serialNumber": "3515398573506688",
    "rawCertType": "ecdsa-sha2-nistp256-cert-v01@openssh.com user certificate",
    "validFrom": 1716503060000,
    "validTo": 1719181460000,
    "validity": 30,
    "validityUnit": "days",
    "principals": [
      "admin"
    ],
    "extensions": {
      "permitX11Forwarding": true,
      "permitAgentForwarding": true,
      "permitPortForwarding": true,
      "permitPty": true,
      "permitUserRc": true
    },
    "certContent": "dummy cert",
    "caName": "testPermG1.user.ca",
    "certStatus": "Active",
    "filePaths": [],
    "onlyForWebTerminal": false
  },
  "message": "Success",
  "appStatusCode": "SSH-NB-200",
  "tags": {},
  "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

What's New

- SSH Download KRL
- SSH Get Hosts From Infra Access Group

SSH Download KRL

The API will download KRL based on the given request.

Before you begin

Before attempting to downloading KRL, the user has to ensure the following:

- KRL should be present in AppViewX.

Request Structure

Endpoint:	`/ssh/krl/download`

Type:	`GET`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/krl/download?caName=dev-ssh.user.ca&gwsource=external` To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
caName *Body*	(Mandatory) Name of the CA **Type**: String

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the KRL file **Type**: response
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

Name	Description
content	Content of the KRL **Type**: String
name	Name of the CA **Type**: String

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	KRL file downloaded successfully
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**ERR-SSH_NB-247**	Invalid query param found for <FieldName> **Possible remediation**: Please make sure to provide valid values

Status Codes and Description (continued)

HTTP Status code	appStatusCode	Message and Possible remediation
	ERR-CA-515	KRL file does not exist or file size mismatch **Possible remediation**: Please make sure to provide proper values in request

Sample Request/Response

Use Case

To download KRL using **download_KRL** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/krl/download?gwsource=external
```

Sample URL

https://pe-ssh-apvx-n4.lab.appviewx.net:31443/avxapi/ssh/krl/download?caName=dev-ssh.user.ca&gwsource=internal

Sample Response


```
{
   "response": {
      "name": "dev-ssh.user.ca",
      "content": "content"
   },
   "message": "",
   "appStatusCode": "",
   "tags": {},
   "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

SSH Get Hosts From Infra Access Group

The API will be used to retrieve the host information from the access group.

Before you begin

Before attempting to get the host information from access group, the user has to ensure the following:

- Access group is present in the request.

Request Structure

Endpoint:	`/ssh/app-infra-group/hosts`
Type:	`GET`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/app-infra-group/hosts?gwsource=external` To understand the elements of the sample URL, click here.
Headers:	

Content-Type:	application/json

Request structure values

Parameter	Value
URL	/ssh/app-infra-group/hosts
Type	GET

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
groupName *queryParam*	(Mandatory) Access group name **Type**: String
isClient *queryParam*	(Mandatory) Fetch client hosts **Type**: Boolean

Request Parameters (continued)

Name	Description
fetchTerminalPreference *queryParam*	(Mandatory) Fetch terminal preference **Type**: Boolean

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response attributes for the host attributes **Type**: response
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

Name	Description
hosts	List of host information **Type**: List
appInfraAccessGroupName	Name of the access group **Type**: String

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Success
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**ERR-SSH_NB-247**	Invalid Key Algorithm or BitLength **Possible remediation**: Make sure to provide proper key type and bit length

Sample Request/Response

Use Case

To get hosts from infra access groups using **get_hosts_from_infra_access_groups** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/app-infra-group/hosts?gwsource=external
```

Sample URL

https://pe-ssh-apvx-n4.lab.appviewx.net:31443/avxapi/ssh/app-infra-group/hosts?groupName=Default_Infra_Access_Group&isClient=true&fetchTerminalPreference=true&gwsource=external

Sample Response


```
{
   "response": {
      "hosts": [
         {
            "deviceName": "pe-cert-apvx-node01.lab.appviewx.net",
            "hostName": "pe-cert-apvx-node01.lab.appviewx.net",
            "ipAddress": "192.168.60.129",
            "status": "Managed",
            "isSudoUser": true,
            "isValidSudo": true,
```

```
            "isClient": true,
            "accessType": "Certificate",
            "vendor": "CentOS Linux",
            "vendorType": "linux",
            "communicationMode": "SSH",
            "pinned": false
        }
    ],
    "appInfraAccessGroupName": "Default_Infra_Access_Group"
  },
  "message": null,
  "appStatusCode": null,
  "tags": {},
  "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL

- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

What's New

- Search Access Groups

Trigger Network Scan for Range of IP Addresses

The API will initiate network scans for specified IP address ranges.

Before you begin

N/A

Request Structure

Endpoint:	`/ssh/discovery/create`
Type:	`POST`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/discovery/create?gwsource=external` To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password	(Mandatory) AppViewX login password

Request Parameters (continued)

Name	Description
Header	**Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type *Header*	(Mandatory) Specifies the nature of the data in the payload **Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
isUpdate *queryParam*	(Mandatory) Is this an update to already existing discovery or not **Type**: Boolean
Payload *Body*	(Mandatory) Contains all the parameters to be sent in the request body for the post request **Type**: Payload

Payload

Name	Description
data	Contains request parameters to trigger a new discovery. **Type**: Data

Data

Name	Description
name	(Mandatory) Name of the discovery to be triggered **Type**: String
description	(Optional) Description of the discovery to be triggered **Type**: String
discoveryMode	(Mandatory) Mode of the discovery **Type**: String

Data (continued)

Name	Description
	Possible values: ipRange, subnet
scheduleType	(Mandatory) Schedule type of the discovery **Type**: String **Possible values**: instant, scheduled
keyGroupName	(Mandatory) Name of the Key compliance group to which the hosts in discovery should be added **Type**: String
inventoryAction	(Mandatory) Inventory action for the host **Type**: String **Possible values**: Manage, Monitor, Do Not Move
ipRangeBean	(Mandatory) Contains the details of IP ranges to be discovered **Type**: IpRangeBean
hostGroupNames	(Mandatory) List of host compliance group names the hosts in discovery belong to **Type**: List
isSudoUser	(Mandatory) Is sudo user or not **Type**: Boolean
accessElevation	(Mandatory) Access elevation of the user **Type**: String **Possible values**: sudo, dzdo
accessType	(Mandatory) Access type of the hosts to be discovered **Type**: String **Possible values**: Key, Certificate
sshSyncKeyDetail	(Mandatory) SSH sync key detail of the host **Type**: SshSyncKeyDetail

IpRangeBean

Name	Description
startIp	(Mandatory) Start IP of the IP range to be discovered **Type**: String
endIp	(Mandatory) End IP of the IP range to be discovered **Type**: String
ipPerBatch	(Mandatory) Number of IP addresses to be discovered per batch **Type**: Number **Possible values**: 1, 2, 4, 8, 16, 32, 64, 128
isSelectPort	(Mandatory) List of app infra access groups where the host belongs to **Type**: String
port	(Mandatory) Port number to connect to the host from **Type**: Number
userName	(Mandatory) Username to login to the host **Type**: String
loginType	(Mandatory) Login type for the host **Type**: String **Possible values**: Password, Identity Key
fileContent	(Mandatory) Identity Key file content, applicable only if "*Identity Key*" loginType is selected **Content-Type**: application/octet-stream
fileName	(Mandatory) Name of the Identity key file, applicable only if "*Identity Key*" loginType is selected **Type**: String
password	(Mandatory) Password to login to the host **Type**: String
credentialType	(Mandatory) Credential type for authentication to login to the host **Type**: String

IpRangeBean (continued)

Name	Description
	Possible values: Manual Entry, Credential List - AppViewX, Credential List - CyberArk, Credential List - Thycotic Secret
credentialName	(Mandatory) Credential name (null for “Manual Entry” credentialType) **Type**: String
dataCenter	Name of the data center the hosts belong to **Type**: String

SshSyncKeyDetails

Name	Description
sshScanType	(Mandatory) SSH scan type **Type**: String **Possible values**: Default, Full, Directory
sshScanDetails	(Mandatory) Details of the SSH scan **Type**: List
discoverKeyType	(Mandatory) List of key types to be discovered **Type**: List
appInfraAccessGroup	(Mandatory) List of app infra access groups where the host belongs to **Type**: List

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response message **Type**: String

Response Parameters (continued)

Name	Description
message	Success message of the action or failure description in case of error. Will be non-null for failure response **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response **Type**: String
tags	More info in case of failure response

Response

response	Scheduler has been triggered successfully. **Type**: String

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Host created and saved successfully.
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**AVX-VLDTN-001**	Mandatory field is missing or invalid values specified - <<field name>> **Possible remediation**: Check and ensure that valid value is provided for <<field name>> field in the request.

Sample Request/Response

Use Case

To trigger network scan for range of IP addresses using **trigger_network_scan_for_range_of_IP_addresses** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/discovery/create?gwsource=external
```

Sample Request 1

```
{
"data":{
    "name":"test_2",
    "description":"",
    "discoveryMode":"ipRange",
    "scheduleType":"instant",
    "keyGroupName":"Default_Key_Group",
    "inventoryAction":"manage",
    "ipRangeBean":
 {"startIp":"1.1.1.1",
    "endIp":"1.1.1.4",
    "ipPerBatch":"1",
    "isSelectPort":"custom",
    "port":"22",
    "userName":"appviewx",
    "loginType":"password",
    "password":"dummy pwd",
    "credentialType":"Manual Entry",
    "credentialName":null,
    "dataCenter":"absecon"},
    "hostGroupNames":["Default_Host_Group"],
    "isSudoUser":true,
    "accessElevation":"sudo",
    "source":"IP_Range_Discovery",
    "accessType":"Certificate",
    "sshSyncKeyDetail":
    {"sshScanType":"default",
    "sshScanDetails":[],
    "discoverKeyType":["User Keys","Host Keys"],
    "appInfraAccessGroup":["Default_Infra_Access_Group"]
}
```

Sample Request 2


```
{
  "data": {
    "name": "test",
    "description": "",
    "discoveryMode": "subnet",
    "scheduleType": "instant",
    "keyGroupName": "Default_Key_Group",
    "inventoryAction": "manage",
    "subnetBean": {
      "network": "10.12.11.0/24",
      "superNet": "25",
      "isSelectPort": "custom",
      "port": "22",
      "userName": "appviewx",
      "loginType": "password",
      "password": "dummyPwd",
      "credentialType": "Manual Entry",
      "credentialName": null,
      "dataCenter": "absecon"
    },
    "hostGroupNames": [
      "Default_Host_Group"
    ],
    "isSudoUser": true,
    "accessElevation": "sudo",
    "source": "Subnet_Scan_Discovery",
    "accessType": "Certificate",
    "sshSyncKeyDetail": {
      "sshScanType": "default",
      "sshScanDetails": [],
      "discoverKeyType": [
        "User Keys",
        "Host Keys"
      ],
      "appInfraAccessGroup": [
```



```
            "Default_Infra_Access_Group"
        ]
    }
  }
}
```

Sample Response

```
{
    "response": "Scheduler has been triggered successfully.",
    "message": null,
    "appStatusCode": null,
    "tags": null,
    "headers": null
}
```

Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL

- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.

Revoke Certificate

The API will trigger revocation of the SSH certificate.

Before you begin

Before attempting to revoke the certificate, make sure that the certificate is in the *Active* state.

Request Structure

Endpoint:	`/ssh/cert/revoke`
Type:	`POST`
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/cert/revoke?gwsource=external` To understand the elements of the sample URL, click here.
Headers:	
Content-Type:	application/json

Request Parameters

Name	Description
sessionId *Header*	(Mandatory) Session ID received after login. **Type**: String **Constraints**: Required if **username** and **password** are not provided.
username *Header*	(Mandatory) AppViewX login username **Type**: String **Constraints**: Required if **sessionId** is not provided.
password *Header*	(Mandatory) AppViewX login password **Type**: String **Constraints**: Required if **sessionId** is not provided.
Content-Type	(Mandatory) Specifies the nature of the data in the payload

Request Parameters (continued)

Name	Description
Header	**Type**: String **Constraints**: The value of the param should be **'application/json'**.
gwsource *Query*	(Mandatory) Source from which the request is triggered. (E.g. external) **Type**: String
Payload *Body*	(Mandatory) Contains all the parameters to be sent in the request body for the post request **Type**: Payload

Payload

Name	Description
caName	(Mandatory) Name of the CA that signed the certificate. **Type**: List
groupName	(Mandatory) Name of the key compliance group to which the certificate is assigned. **Type**: String
serialNumber	(Optional) Serial number of the certificate **Type**: String **Constraints**: Required if **certificate** is not provided
certificate	(Optional) Certificate content **Type**: String **Constraints**: Required if **serialNumber** is not provided
reason	(Optional) Reason for revocation **Type**: String

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Host created and saved successfully.
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**AVX-VLDTN-001**	Mandatory field is missing or invalid values specified - <<field name>> **Possible remediation**: Check and ensure that valid value is provided for <<field name>> field in the request.

Response Structure

200 OK returns string of type application/json with the following body params.

Response Parameters

Name	Description
response	Contains the response message of the API. **Type**: Revoke Response
message	Success message of the action or failure description in case of error. Will be non-null for failure response. **Type**: String
appStatusCode	Application-specific status code for the response. Will be non-null for failure response. **Type**: String
tags	More info in case of failure response.

Revoke Response

revokedRequests	List of revoked requests

	Type: List
failedRequests	List of failed requests **Type**: List

Status Codes

Status Codes and Description

HTTP Status code	appStatusCode	Message and Possible remediation
200 OK	**NA**	Host created and saved successfully.
401 Unauthorized	**AVX_GW_003**	Authentication failed, reason - Invalid Credentials **Possible remediation**: Ensure that valid **username** and **password** or valid **sessionId** is provided as the header param.
400 Bad Request	**VALIDATION_ERROR_0004**	Mandatory field <<field name>> is missing or or empty **Possible remediation**: Check and ensure that valid value is provided for <<field name>> field in the request.
400 Bad Request	**ERR-SSH_NB-270**	Max request size exceeded::[Max request size allowed is 25] **Possible remediation**: Check and ensure that the number of requests specified in the payload is less than 25.
417 Expectation Failed	**ERR-SSH-NB-298**	One or more groups do not have permission to perform this action. **Possible remediation**: Check and ensure that the user has access to the specified key compliance group.

Sample Request/Response

Use Case

To revoke SSH certificate using **revoke_certificate** API.

Request URL

```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/ssh/cert/revoke?gwsource=external
```

Sample Request

```
[
  {
    "serialNumber":"3527182657550928",
    "caName":"Default-Infra-Access-Group.user.ca",
    "reason":"test",
    "groupName":"Default_Key_Group"
  },
  {
    "certificate":"<actual certificate content should be populated here>",
    "caName":"TestHostCA",
    "reason":"test",
    "groupName":"Default_Key_Group"
  }
]
```

Sample Response

```
{
  "response": {
    "revokedRequests": [
      {
        "caIdentifier": "Default-Infra-Access-Group.user.ca",
        "error": null,
        "certificate": "<actual certificate content will be present here>",
        "serialNumber": "3527182657550928",
        "revoked": true
      }
    ],
    "failedRequests": [
      {
        "caIdentifier": "TestHostCA",
```

```
            "error": "No CA found with given name or Id",
            "certificate": “<actual certificate content will be present here>”,
            "serialNumber": null,
            "revoked": false
        }
    ]
  },
  "message": "Success",
  "appStatusCode": "SSH-NB-200",
  "tags": {},
  "headers": null
}
```


Reference

Understanding the sample URL:

- **IP/HostName/TenantName**: Replace with the actual IP address, hostname, or tenant name based on the specific configuration in AppViewX.
  - **IP**: A unique identifier assigned to each device connected to a computer network that uses the Internet Protocol for communication

    The IP address will be included in the endpoint URL for an on-prem deployment.
  - **HostName**: A human-readable label assigned to a device (host) on a network

    The hostname will be included in the endpoint URL for an on-prem deployment.
  - **TenantName**: An identifier label for a tenant given to indicate which tenant's data the API request will access/modify

    The tenant name will be included in the endpoint URL for a SaaS deployment.
- **GWPORT**: AppViewX gateway port

  A gateway port refers to a network port through which data is sent and received to communicate with a gateway in an on-prem deployment.

  Example: **31443**
- **avxapi**: Path parameter value (static) that is part of the endpoint's URL
- **Endpoint**: Endpoint of the API, for example: **execute-hook**
- **gwsource**: Source or origin of a gateway, for example: **external**.