appviewx

PKI+ Guides

Version: 2024.3.0.0

Copyright AppViewX, Inc.

Trademarks

Contact Information

AppViewX, Inc.

222 Broadway, FL 19

New York, NY 10038

Email: info@appviewx.com

Web: www.appviewx.com

Contents

Preface

Revision History

Revision	Description	Date
1.0	Release of AppViewX PKI+ for Thames FP3	Sep 2025

About this Guide

This guide explains the capabilities of AppViewX PKI+. This guide provides step-by-step instructions to configure and manage AppViewX PKI+.

Audience

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

Third-Party Software Acknowledgments

This section serves as a placeholder to document the third-party components referenced in this guide, along with their associated trademark information.

For example:

- This document includes software details developed by VMware, Inc. (www.vmware.com).

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 text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`codeblock`	Indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Chapter 1: PKI+ User Guide

This guide explains the capabilities of AppViewX PKI+. This guide provides step-by-step instructions to configure and manage AppViewX PKI+.

Introduction

What is AVX One Platform?

AVX One Platform refers to a unified solution offered by **AppViewX** designed to simplify and automate the management of various infrastructure components, such as **PKI (Public Key Infrastructure)**, **SSL/TLS certificates**, **load balancers**, and **application delivery controllers** (ADCs), among other network and security resources.

Key features of **AVX One Platform** include:

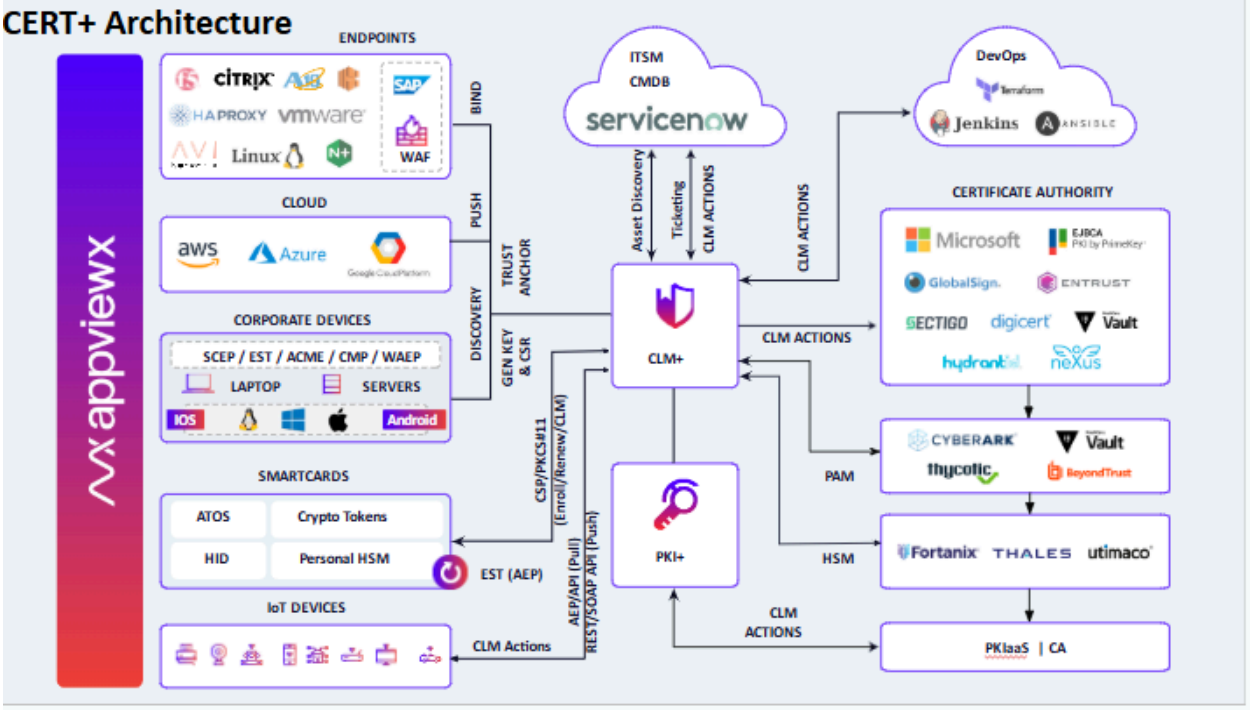
1. **Centralized Management:** AVX One provides a centralized platform for managing multiple components, including certificates, keys, and security policies, making it easier for organizations to maintain security across their infrastructure.
2. **Automation:** It offers automation capabilities for tasks such as certificate lifecycle management, certificate discovery, renewal, and revocation. This reduces manual errors and improves operational efficiency.
3. **Multi-cloud & Hybrid Cloud Support:** The platform supports both on-premises and cloud-based deployments, allowing organizations to manage their infrastructure across multi-cloud and hybrid cloud environments.

4. **PKI Management:** AVX One simplifies the deployment, configuration, and operation of **Public Key Infrastructure (PKI)**, enabling organizations to create, manage, and deploy certificate authorities (CAs), and handle certificate issuance, revocation, and validation.
5. **Security & Compliance:** The platform helps ensure compliance with various security standards and industry regulations, particularly related to encryption and certificate management.
6. **Integrations:** AVX One integrates with various third-party systems, including cloud services, load balancers, and network security appliances, offering flexibility for businesses with complex IT environments.

In summary, **AVX One Platform** is an integrated, scalable solution designed to streamline and secure the management of infrastructure resources, particularly focusing on PKI, certificates, and related security operations.

CERT Architecture

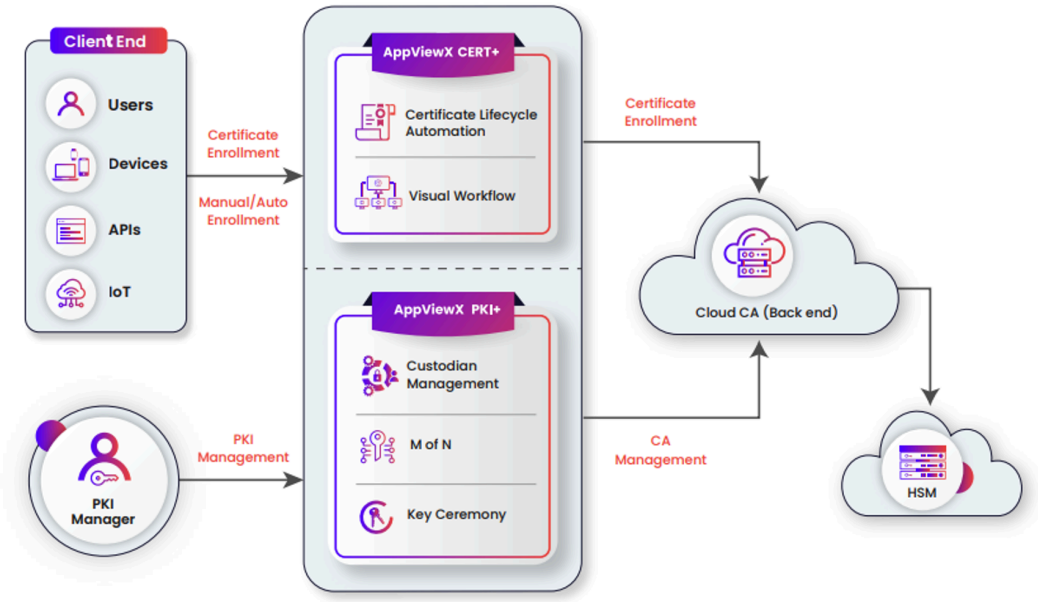
The following diagram illustrates the CERT architecture:



What is AppViewX PKI?

AppViewX PKI is a comprehensive solution for managing Public Key Infrastructure (PKI) and digital certificates within an organization's IT environment. It simplifies the entire lifecycle of digital certificates,

providing tools for automated certificate management, secure key management, and ensuring compliance with security policies.



AppViewX Ecosystem

What is AppViewX Native CA?

AppViewX introduces its own Certificate Authority (CA)--AppViewX Native CA--for PKI initialization enabling the customers to take advantage of Post-Quantum Cryptography (PQC) capabilities, to ensure the infrastructure remains secure in the era of quantum computing.

> **Note:**
> PKI can be initialized using either the AppViewX CA backend (PKIaaS Native) or a Cloud CA backend (Standard).

Key Features

- **Post-Quantum Cryptography (PQC) support:** Supports NIST-standardized PQC algorithms and selected algorithms from the fourth round of NIST's post-quantum standardization process.
- **Customizable Certificate Templates:** Allows users to create tailored certificate authorities (CAs) with custom templates and offers pre-configured templates for different types of end certificates.
- **CA Key Storage Mechanisms:**

- **CA Key with On-Prem HSM (BYOD):** Seamlessly integrates with external HSM vendors for cryptographic operations. HSMs supported by AppViewX Native CA are:
  - Fortanix with FIPS
  - Fortanix without FIPS
  - Thales DPOD
  - Thales GPN
  - Utimaco
  - Entrust
- **CA Key with Cloud HSM (BYOD or AVX Provided):** Requires connectivity 443 to the external Cloud HSM Provider.
- **AVX Managed Key:** This is a key management service provided by **AppViewX**, which is part of the **PKI** solution. The **AVX Managed Key** feature is designed to streamline and automate the generation, storage, and lifecycle management of cryptographic keys used in PKI, SSL/TLS certificates, and other security operations.

- **Enhanced Security – Airgapped Root CA:** Enhances security with offline Root CA deployment support.
- **Revocation List Management – Custom CRLDP & OCSP:** Provides customizable Certificate Revocation List Distribution Points (CRLDP) and Online Certificate Status Protocol (OCSP) services.
- **Auto-Enrollment Support:** Simplifies certificate enrollment with support for SCEP, EST, ACME, WAEP, and Microsoft Intune protocols.
- **Support for Short-Lived Certificates:** Short-lived certificates refer to SSL/TLS certificates that are issued with a very short validity period, typically ranging from a few days to a few months. With shorter validity periods, the use of automated tools (like **ACME protocol** for certificate management and many more MDM tools) becomes more common. This encourages the automation of certificate renewal, which reduces human errors and increases operational efficiency. They are more secure as attack surface is minimized because certificates are rotated more frequently. If a certificate is compromised, revocation becomes more effective because the certificate will expire quickly anyway.
- **PKI Dashboard:** Features an intuitive dashboard for streamlined certificate and CA management.
- **Security:** AppViewX Native CA provides Quantum-Resilient Security by implementing algorithms such as Dilithium, Falcon, and Sphincs Plus to protect data against quantum attacks.

Note:

AppViewX PKIaaS offers 99.5% availability by default as a multi-tenant SaaS solution.

Key Types, Hash Functions, and Key Sizes supported by AppViewX Native CA

Key Types	Hash Functions	Key Sizes
SPHINCS PLUS (SLH-DSA)	SHAKE256, HARAKA256, SHA256	256, 384, 512
DILITHIUM (ML-DSA)	SHAKE256	10496, 15616, 20736
FALCON (Beta)	SHAKE256	7176, 14344
EC	SHA160, SHA224, SHA256, SHA3-224, SHA3-256, SHA384, SHA512	160, 163, 191, 192, 193, 224, 233, 239, 256, 283, 320, 359, 384, 409, 431, 512, 521, 571
DSA	SHA160, SHA224, SHA256, SHA3-224, SHA3-256, SHA384, SHA512	1024, 2048
RSA	SHA160, SHA224, SHA256, SHA3-224, SHA3-256, SHA384, SHA512	1024, 2048, 3072, 4096, 7680, 8192

Supported Signature Algorithms for CA Certificate Creation

This section delineates the signature algorithms supported for certificate generation. Each algorithm comprises three fundamental elements:

- **Key type and scheme**: (for example, RSA, EC, DSA, Falcon, Dilithium, SPHINCS+)
- **Key size / security parameter:** (for example, 2048, 3072, 4096 bits for RSA, or parameter sets for PQC algorithms)
- **Hash or digest function**: (for example, SHA-256, SHA-384, SHA-512, SHAKE256) The hash function specified within each algorithm is used for hashing the certificate during its certificate signing process, as mentioned below

Examples of Hash functions used in Supported Algorithms

- **SHA-256:** Generates a 256-bit output, which is widely adopted and serves as the baseline for most contemporary PKI systems.
- **SHA-384:** Generates a 384-bit output, offering an elevated security posture compared to SHA-256.
- **SHA-512**: Generates a 512-bit output, particularly well-suited for high-security environments.
- **SHAKE256**: An extendable-output hash function from the SHA-3 family, engineered for versatility and post-quantum applications.

The selection of the hash function directly influences the certificate signing.

Traditional Algorithm

- **RSA Algorithm: RSA** (Rivest–Shamir–Adleman) is one of the most widely used asymmetric cryptographic algorithms.
  - **Algorithm Format**

```
RSA_<SCHEME>_<KEYSIZE>_<HASH>
```

> **Note:**
>
> The algorithm name, scheme, and key size correspond to the **CA Certificate key** generated during the **Create CA** operation. The specified hash algorithm is used by the CA to sign child certificates.

  - **Scheme:**
    - **PKCS1:** Based on the **PKCS #1** standard. It specifies the padding and encoding methods for RSA signatures. Commonly used and well-supported.
    - **PSS:** Stands for **Probabilistic Signature Scheme**. A modern, more secure padding method for RSA signatures, recommended by NIST.
  - **Key Size (2048, 3072, 4096):** Dictates the security strength. Larger keys enhance security but may lead to diminished performance.Bit length of the RSA key.
  - **Examples**
    - **RSA_PKCS1_2048_SHA256:**Uses PKCS#1 v1.5 padding, a 2048-bit key, and the SHA-256 hash algorithm.
    - **RSA_PKCS1_4096_SHA512:** A high-security configuration with PKCS#1 padding, 4096-bit key, and SHA-512.
    - **RSA_PSS_3072_SHA256:** Uses PSS padding, a 3072-bit key, and SHA-256 for enhanced security.
- **EC (Elliptic Curve) Algorithm:** Elliptic Curve Cryptography (ECC) offers comparable security to RSA with reduced key sizes, thereby facilitating faster operations.
  - **Algorithm Format**

```
EC_<CURVE>_<HASH>
```

> **Note:**
>
> The algorithm name and EC curve pertain to the CA Certificate key generated during the Create CA operation. The Hash indicates the algorithm the CA will employ for signing child certificates.

Examples:

- **EC_P256_SHA256:** Uses the NIST P-256 curve with SHA-256 hashing.
- **EC_P384_SHA384:** Uses the NIST P-384 curve with SHA-384 hashing.
- **EC_P521_SHA512**: Uses the NIST P-521 curve with SHA-512 hashing.

- **DSA with PSS:** DSA (Digital Signature Algorithm) with PSS padding is less prevalent but remains supported within certain environments.
  - **Algorithm Format**

```
DSA_PSS_<KEYSIZE>_<HASH>
```

> **Note:**
>
> The algorithm name, scheme, and key size pertain to the CA Certificate key generated during the Create CA operation. The Hash indicates the algorithm the CA will employ for signing child certificates.

  - **Examples**
    - **DSA_PSS_1024_SHA256:** A 1024-bit DSA implementation with PSS padding and SHA-256.
    - **DSA_PSS_2048_SHA256:** A 2048-bit DSA offering enhanced security.

Post-Quantum Cryptography (PQC) (Enabled only for PKIaaS Native CA)

NIST is actively engaged in standardizing algorithms resilient to quantum attacks. The subsequent algorithms represent post-quantum signature schemes.

- **Dilithium (Lattice-based):** Dilithium is recognized as a leading candidate for standardization. Larger parameter sets denote superior security.
  - **Algorithm Format**

```
DILITHIUM_<KeySize>_<HASH>
```

> **Note:**
>
> The algorithm name and key size pertain to the CA Certificate Key generated during the Create CA operation. The SHAKE256 indicates the algorithm the CA uses for signing child certificates.

  - **Supported Cryptography**

- DILITHIUM_10496_SHAKE256
- DILITHIUM_15616_SHAKE256
- DILITHIUM_20736_SHAKE256

- **Sphincs+ (Lattice-based):** SPHINCS+ is considered a conservative, hash-based approach, though it tends to produce larger signature sizes compared to Falcon/Dilithium.
  - **Algorithm Format**

```
SPHINCSPLUS_<SECURITY_LEVEL><MODE>_<HASH>
```

> **Note:**
>
> The algorithm name, security level, and generation mode pertain to the CA Certificate key generated during the Create CA operation. The SHAKE256 indicates the algorithm the CA will employ for signing child certificates.

  - **Security level:** 128, 192, 256 bits.
  - **Mode:** F = *Fast* (characterized by smaller signatures and quicker signing), S = *Small* (characterized by smaller public keys and slower performance)
  - **Hash function:** SHAKE256 or the SHA-2 family (SHA-256, SHA-384, SHA-512).
  - **Supported Cryptography**
    - SPHINCSPLUS_128F_256_SHAKE256
    - SPHINCSPLUS_128S_256_SHAKE256
    - SPHINCSPLUS_192F_384_SHAKE256
    - SPHINCSPLUS_192S_384_SHAKE256
    - SPHINCSPLUS_256F_512_SHAKE256
    - SPHINCSPLUS_256S_512_SHAKE256
    - SPHINCSPLUS_128F_256_SHA256
    - SPHINCSPLUS_128S_256_SHA256
    - SPHINCSPLUS_192F_384_SHA256
    - SPHINCSPLUS_192S_384_SHA256
    - SPHINCSPLUS_256F_512_SHA256
    - SPHINCSPLUS_256S_512_SHA256

Getting Started

Know your Deployments for PKI+

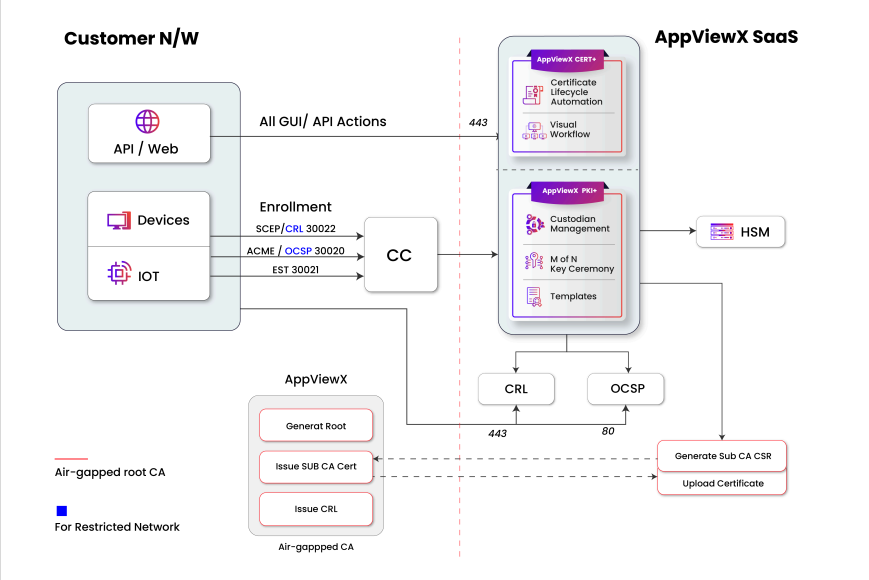
We support multiple deployment options to cater to various customer needs and infrastructure preferences. Our solutions can be deployed in On-Premises environments and as a Software as a Service (SaaS).

SaaS Deployment (Highly Secure and Hassle-Free)

Our Software as a Service (SaaS) offering is designed for organizations that prioritize security, simplicity, and efficiency. In this deployment mode, we manage all aspects of application hosting, maintenance, and scaling, providing a worry-free experience for our customers. Our SaaS platform is built with cutting-edge security measures, including robust encryption, multi-factor authentication, and continuous monitoring to ensure your data and operations are protected at all times.

Choosing SaaS not only reduces the burden on your IT teams but also ensures that you benefit from the latest updates, features, and security enhancements without any additional effort. This option is ideal for organizations of all sizes, particularly those looking to quickly access our services with the assurance of enterprise-grade security and compliance. For additional information about saas deployment, click here.

AVX ONE PKI+ – SaaS Deployment

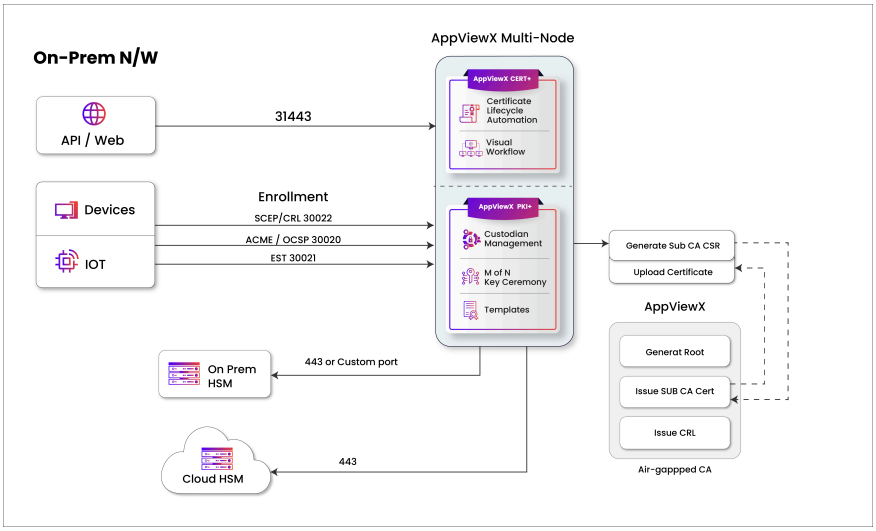


- **Cloud-Based Installation:** AVX PKI+ can be deployed in a SaaS model, with both root and intermediate CAs hosted in the cloud. CA private keys are securely stored in a Cloud HSM integrated with the solution.
- **Custodian Governance:** Key operations follow the M-of-N custodian approach with custodian users managed either directly by AVX or via enterprise SSO.
- **Key Storage Options:**
  - **AVX-Managed Keys:** Secured using a master key within the AVX vault system.
  - **On-Prem HSM:** Integrated using the AVX Cloud Connector, which enables secure connectivity between the SaaS platform and on-premise HSMs (via AVX Cloud Connector).
  - **Cloud HSM:** Supported, if needed, and packaged along with the solution.
- **Certificate Issuance:** All end-entity certificates can be issued directly from this SaaS-hosted instance.

On-Premises Deployment

On-Premises deployment enables organizations to install and operate our applications on their own infrastructure. This approach offers the highest level of control and customization, making it particularly suitable for organizations with strict security, compliance, or performance needs. It is best suited for enterprises with dedicated IT resources and the expertise to manage complex infrastructure. For additional information about on-premises deployment, click here.

AVX ONE PKI+ – Air-Gapped Root CA Deployment



- **On-Premise Installation:** AVX ONE PKI+ can be deployed within the customer's on-premise infrastructure. AVX provides the required installation packages in OVA or other supported formats.
- **Root CA Generation:** The deployed instance can be used to create the root CA, which is recommended to remain offline with restricted access to the application.
- **Custodian-Controlled Key Operations:** Key operations are managed using an M-of-N custodian model. Custodians can be AVX-managed users or authenticated via SSO and are configured directly within the system.

- **Flexible Key Storage Options:**
  - **AVX-Managed Keys:** Protected using a master key secured in the integrated vault system.
  - **On-Prem HSM:** Uses existing or newly procured HSMs within the customer environment.
  - **Cloud HSM:** Supported by AVX, though storing root CA keys in a cloud HSM is generally discouraged for security reasons.
- **Extensibility:** The same instance can optionally be used to host intermediate CAs and issue end-entity certificates, if required.

AVX ONE PKI+ – Hybrid Model with Air-Gapped Root CA and SaaS-based Issuing CA

- **Issuing CA Deployment:** AVX PKI+ is deployed in the SaaS environment with the intermediate CA (issuing CA) signed by an offline, air-gapped root CA.
- **Offline Signing Workflow:** The issuing CA key is generated within an HSM. Its CSR is exported and signed by the air-gapped Root CA. The resulting certificate is then imported back into the AVX SaaS instance.
- **Custodian-Driven Security:** All key operations are governed by the M-of-N custodian model using AVX-managed or SSO-authenticated users.
- **Comprehensive Key Storage Support:**
  - AVX-Managed Keys
  - On-Prem HSM (via AVX Cloud Connector)
  - Cloud HSM
- **End-Entity Certificate Services:** This configuration supports issuance of all end-entity certificates through the SaaS-based issuing CA.

- System Requirements
- Demo Mode
- Start your Onboarding Journey

System Requirements

System Requirements

Hardware

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

Deployment 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 8.7
- RHEL 8.8
- RHEL 8.10
- RHEL 9.2
- RHEL 9.3
- RHEL 9.4

System Requirements (continued)

	• Ubuntu 20.04 • Ubuntu 22.04
Browser Requirements	• **Firefox:** v74.0.1 (64-bit) or later • **Google Chrome:** v85.0.4183.83 (64-bit) or later

Demo Mode

AppViewX provides a demo mode for users to explore the features of AppViewX PKI+ with sample data in the read-only mode. By default, the demo mode is turned off. You can use the toggle button to turn it on.

For more details, see Navigating PKI Menu.

Start your Onboarding Journey

Key Highlights of PKI Onboarding Trial Journey

1. **Seamless Module Navigation:** Users can easily transition from the trial version of CERT+ to explore the features of PKI. During this process, they have the option to designate specific user groups that will have access to PKI upon activation.

   Any CERT+ trial customers can click **Manage PKI** to enable the PKI product and designate/invite the users into AppViewX adding the user to a specific User group and assigning them to access the PKI.

2. **Guided Activation Experience:** Once PKI is activated, the super admin or designated users can fully explore the entire product. The experience is enhanced with a simplified guided navigation, ensuring that users can quickly understand and use all the available features.

   The journey begins by creating a PKI hierarchy which allows the user to create the Root Certificate Authority.

3. **Flexible PKI Hierarchy Setup:** Users have the ability to configure both classical Public Key Infrastructure (PKI) and Post-Quantum Cryptography (PQC) hierarchies. This flexibility allows organizations to choose the most suitable security framework for their needs.
4. **Streamlined PKI Hierarchy Setup and Certificate Issuance:** The setup process for creating a PKI hierarchy and issuing certificates is designed to be efficient. Users can achieve this with minimal input,

ensuring that the security protocols such as custodian addition and key ceremonies are maintained without compromising on security standards.

5. **Collaboration and Access Sharing:** Users can invite others to join their designated user group, enabling them to experience the product firsthand. This feature promotes collaboration and allows teams to explore the product's capabilities together.

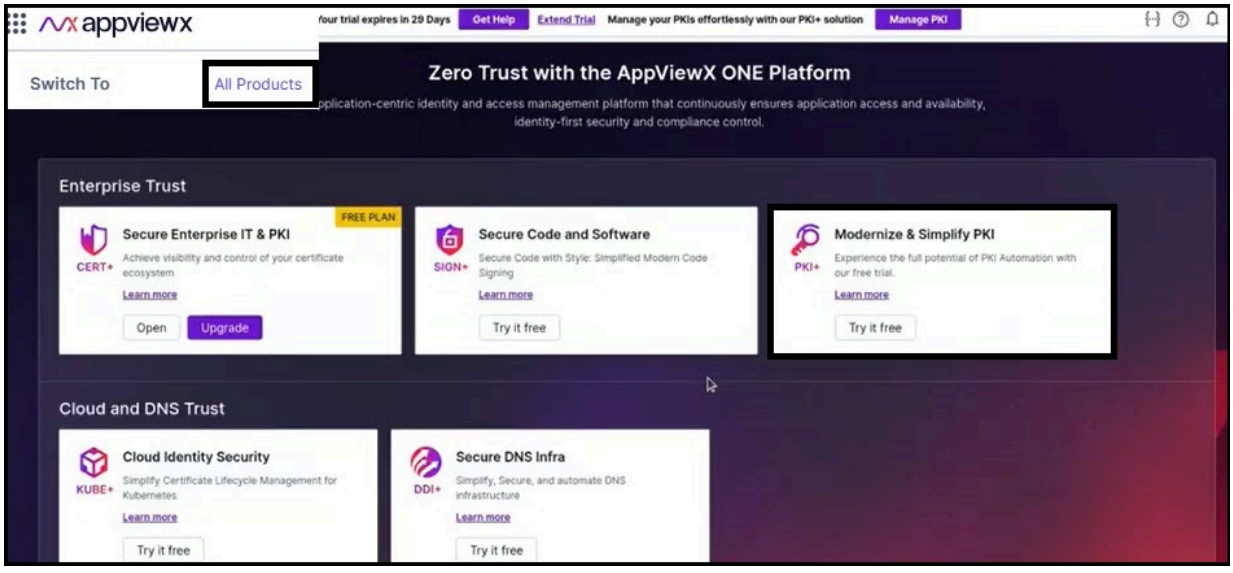
The process for managing PKI differs between SaaS trial/new customers and existing customers as mentioned:

- **SaaS trial/New Customers:** Directed to the **CERT+** > **Insights** > **Summary** page, where a pop-up window appears outlining the Onboarding Custodians.
- **Existing Customers:** Directly access the PKI page by going to (**Menu**) icon **> PKI+**.
- Onboarding Journey

Onboarding Journey

A freemium customer can access PKI+ either by:

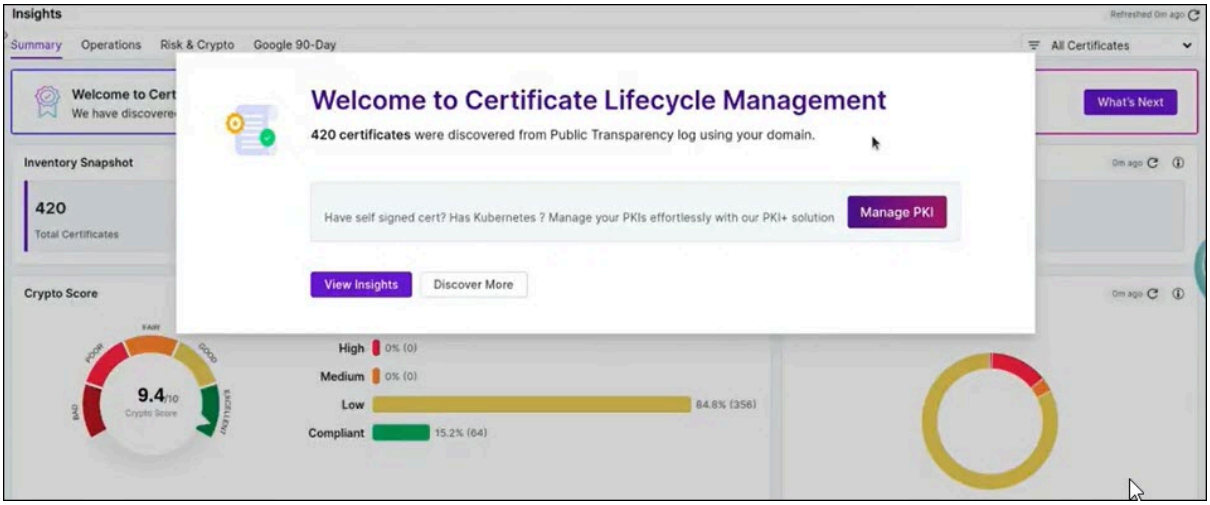
- Going to (**Menu**) icon **> All Products.** Click **Try it free** from **Modernize & Simplify PKI.** A message, *Your license activation is in progress. Please wait..,* is displayed.



-OR-

- From the **Insights > Summary** page by any of the following ways:

- After completing your first certificate discovery, you will see this pop-up window:



-OR-

- Scroll down to **Certificates by Issuing CAs** and click **Start Free PKI Trial.**

-OR-

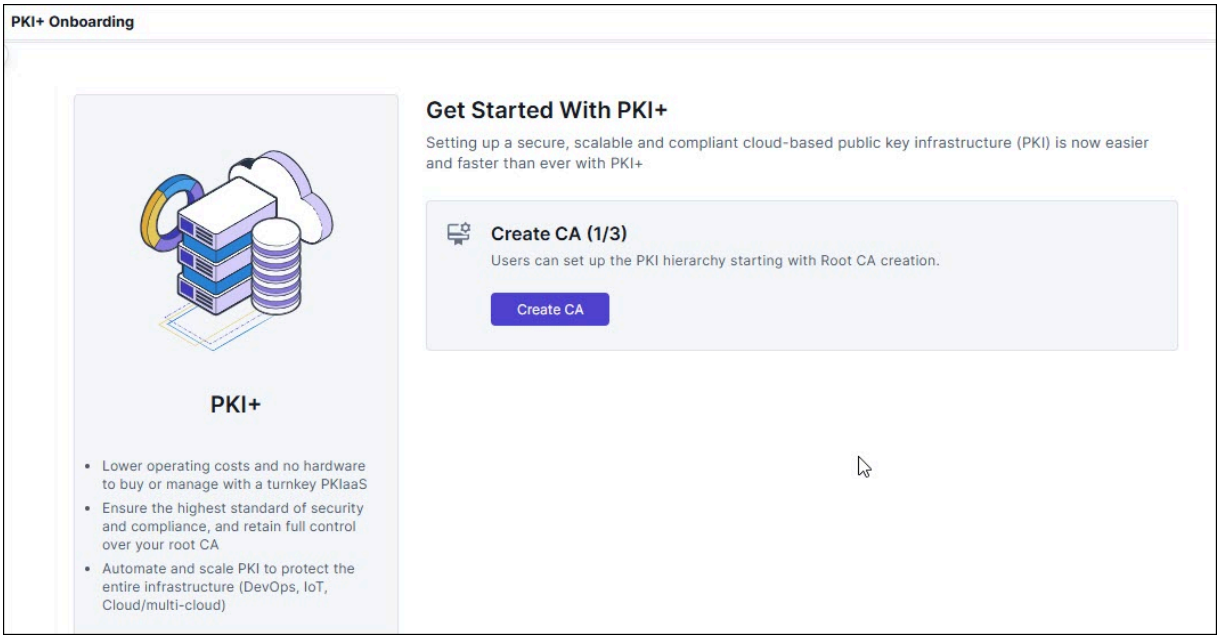
- Click **Manage PKI** on the banner page on top.

Clicking **Manage PKI** will direct customer to the PKI license subscription page.

The PKI+ trial journey involves the following steps:

1. On activating the license, your PKI+ free plan is also activated.
2. Select user group from the dropdown list, and click **Assign.**

The **Get Started with PKI+** page appears.



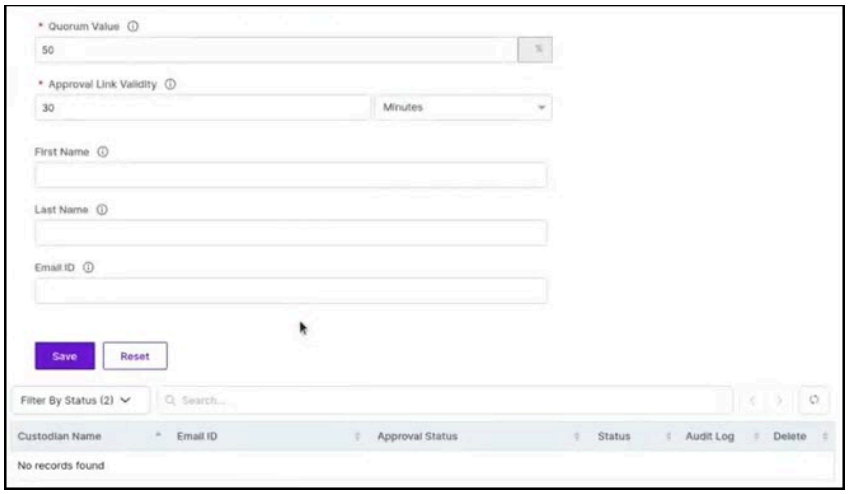
3. Click **Create CA** to create a PKI hierarchy by creating the Root Certificate Authority.

The **Create Root CA** page appears.

a. Enter the fields as described in the table.

Field Description for Create Root CA page

Field	Description
Select CA Type	
***CA Name**	By default, the CA name is populated. You can edit this.
***Template**	By default, RootCA_Default is selected. You can edit this.
***Valid for**	By default, validity is set to 10 years.
Configure CA Subject DN Details	
***CA Common Name**	Enter the root CA subject name.
***Organization**	Enter the organization name owning the CA.
Organization Unit	Enter the business unit for CA operations.
City	Enter the city name.
State	Enter the state name.

Field	Description
Country	Enter the country of the organization.
Configure CA Key Size and Algorithm	
CSR Generation	You can only select AppViewX.
***Key Size and Algorithm**	Select the CA key size and algorithm from the dropdown list.
Custodian Settings	
Custodian	By default, the freemium customer (logged in user) is added as the custodian. Custodians are responsible for approving any action performed in PKI+. Custodians are the individuals responsible for issuance of root and intermediate certificates. They approve or reject certificate requests, manage the lifecycle of certificates, and ensure auditability and compliance. To add more custodians, click **Manage.** The following screen appears. 

Field	Description
	Note: • Quorum value is set to 50%, which means that if the custodian group has two members, then only one custodian is needed to approve any CLM action. For more information, see Custodian Management. • Quorum value is an editable field and can contain values ranging from 20 through 100. You can edit this per your organizational need. • Newly added custodians appear in the Custodian text box with an increment, for example, if one custodian is added, then the text box displays the default custodian +1.

Note:

Fields marked with red asterisk (*) symbol are mandatory.

b. Click **Create.**

A window with the summary of values entered appears.

c. Click **Proceed.**

The freemium customer gets an email for approving the CA creation.

Note:

The approval link in the email is valid only for 48 hours.

Action Status Description and Required Action

Action Status	Status	Description	Required Action
Email Verification - Pending	Inactive	The custodian's email verification is pending approval and is not active.	The requester receives a notification email. Click the **here** hyperlink to be directed to the AppViewX login page

Action Status	Status	Description	Required Action
		Note: If you want to abort the action, click **Abort.** Any workflow that is triggered and is in progress is killed from the Request page prior to triggering any further actions.	and approve the request by going to **Menu > Requests > All requests.**
Add - Approval Pending	Inactive	The custodian has been added but is awaiting approval from active custodians.	Active custodians must click the **here** hyperlink in the email to be redirected to the AppViewX login page.
Add - Approved	Active	The custodian has been approved and added successfully.	On successfully logging in, go to **Menu > Requests > All requests**. Click **Approve**.
Email Verification - Rejected	Inactive	The custodian has been rejected.	On rejecting a request, a confirmation popup window appears if the requester wants to submit the request. Click **OK** to resubmit.

On approval, a message, *CA Creation in Progress,* appears. Wait until it changes to *CA Successfully Created.*

4. Click **Create subordinate CA.**

   The **Create Subordinate CA** page appears.

a. Enter the fields as described in the table.

Field Description for Create Subordinate CA page

Field	Description
Select CA Type	
***CA Name**	By default, the CA name is populated. You can edit this.
***Issuer Name**	By default, the issuer name is populated. You can edit this.
***Template**	By default, RootCA_Default is selected.
***Valid for**	By default, validity is set to 5 years.
Configure CA Subject DN Details	
***CA Common Name**	Enter the root CA subject name.
***Organization**	Enter the organization name owning the CA.
Organization Unit	Enter the business unit for CA operations.
City	Enter the city name.
State	Enter the state name.
Country	Enter the country of the organization.
Configure CA Key Size and Algorithm	
CSR Generation	You can only select AppViewX.
***Key Size and Algorithm**	Select the CA key size and algorithm from the dropdown list.
Custodian Settings	
Custodian	By default, the freemium customer (logged in user) is added as the custodian. He/she will get the approval links via email for all the actions performed in the PKI hierarchy creation.

Note:

Fields marked with red asterisk (*) symbol are mandatory.

b. Click **Create.**

A window with the summary of values entered appears.

c. Click **Proceed.**

The freemium customer gets an email for approving the CA creation.

> **Note:**
>
> The approval link in the email is valid only for 48 hours.

Action Status Description and Required Action

Action Status	Status	Description	Required Action
Email Verification - Pending	Inactive	The custodian's email verification is pending approval and is not active. **Note:** If you want to abort the action, click **Abort.** Any workflow that is triggered and is in progress is killed from the Request page prior to triggering any further actions.	The requester receives a notification email. Click the **here** hyperlink to be directed to the AppViewX login page and approve the request by going to **Menu > Requests > All requests.**
Add - Approval Pending	Inactive	The custodian has been added but is awaiting approval from active custodians.	Active custodians must click the **here** hyperlink in the email to be redirected to the AppViewX login page.
Add - Approved	Active	The custodian has been approved and added successfully.	On successfully logging in, go to **Menu > Requests > All requests**. Click **Approve**.
Email Verification - Rejected	Inactive	The custodian has been rejected.	On rejecting a request, a confirmation popup window appears if the requester

Action Status	Status	Description	Required Action
			wants to submit the request. Click **OK** to resubmit.

On approval, a message, *Subordinate CA Creation in Progress,* appears. Wait until it changes to *Subordinate CA Successfully Created.*

5. Click **Issue Certificate.**

The **Issue Certificate** page appears.

a. Enter the fields as described in the table.

Field Description for Issue Certificate page

Field	Description
***CA Name**	Select a CA name from the dropdown list.
Certificate Type	By default, End Certificate is selected.
***Template**	Select a template from the dropdown list.
Validity	By default, the validity is set to 1 year.
***Upload CSR**	Browse and upload CSR.
***Certificate Download Format**	By default, .PEM is selected.

Note:

Fields marked with red asterisk (*) symbol are mandatory.

b. Click **Issue Certificate**.

A message, *Certificate generated successfully,* appears on the top of the page.

You can view server/client/code signing certificate or enroll server/client/code signing certificate by clicking the links on the RHS of the page.

Navigating PKI Menu

From the PKI menu, you can access:

- **Get Started:** Use this page to initialize PKI, configure SMTP server, onboard custodians, create PKI CA, and set up Cloud Connector to enable connectivity to the Enterprise's private network.
- **Dashboard:** This page gives a quick summary of all the root and subordinate CAs created via AppViewX PKIaaS certificate authority. See Dashboard.
- **CA Inventory:** Create root CAs and subordinate CAs and enroll them to the AppViewX PKIaaS certificate authority. See sections under CA Inventory.
- **Custodian Management:** Custodians are responsible for approving any action performed in PKI. You can add or delete custodians from this page. See Onboarding Custodians.

  On completing custodian onboarding, you can add your root CAs and subordinate CAs to PKI.
- **Settings:** Use this page to configure PKI settings. See Settings.
- **Validation Authority:** Certificate authorities use Certificate Revocation List (CRL) or Online Certificate Status Protocol (OCSP) to obtain the revocation status of x.509 digital certificates. See Validation Authority.
- **Templates:** This module is available only for AppViewX PKIaaS Native CA users. Use this page to select any of the listed templates or to create your own template to specify certificate parameters. See Templates.
- **Issue Certificate:** This module is available only for AppViewX PKIaaS Native CA users. Use this page to issue certificates. See Issue Certificates.

PKI Modules

Prerequisites

On-premise deployments using AppViewX PKIaaS Native CA

1. Ensure these plugins are available:
   - avx_pkiaas_ca_server
   - avx_pkiaas_cert_ocsp_server
   - avx_pkiaas_cert_ocsp_generator
   - avx_platform_gateway_external
   - avx_vendor_cert_scep_agent
2. Add the plug-ins in ENABLED_PLUGINS in the appviewx.conf file and ensure they are enabled and are up and running. In PKIaaS native setup, enable these plugins first and then ensure they are available.
3. OCSP HTTP Response Verification
   - Use the following command to verify the presence of the required service port:

     ```
     bash kubectl get svc -A | grep "avx-platform-gateway-scep"
     ```

   - Ensure that the 30022 port is listed. This port is critical for serving OCSP HTTP responses, which are used to check certificate statuses.
4. Configure SMTP server, which is tested successfully, to send test emails to the custodian email ID addresses.
5. Provide a CA name for reference and activate by going to (**Menu**) icon **> CERT+ > Administration > Certificate Authority**.
6. Onboard at least two custodians before creating CA hierarchy. You can complete the addition of custodians by going to (**Menu**) icon **PKI+ > Custodian Management** with the following privileges under RBAC roles and resources.
   a. Roles automation > service request full
   b. PKI > view all (optional)
   c. Resources > workflow studio, workflow request > PKI+, approval_request

   Note:

   No CA action is possible until at least two active custodians are in the system.

7. **Network Prerequisites**

- All infrastructure network devices must be able to connect to the AppViewX nodes on 31443 (for Web, API calls, CRL).
- All infrastructure devices must be able to connect to the AppViewX nodes on 30022 (for OCSP and SCEP).
- AppViewX must be able to connect to the SMTP server to send test emails to the custodian email ID addresses.

8. (Optional) Loadbalancer configuration for OCSP and CRL:
   - **To publish CRL for users/devices that do not have external (internet) access in SaaS deployment:**
     - Edit the certificate issuance template to include a custom CRLDP with the LB URL.
     - The LB URL must act as a load balancing point for the number of CC URLs.
     - **URL to be configured in templates:** https://[LB Host]:[httpsport]/avxapi/download-crl/[CA_Name]/crl.crl
     - **LB to be balanced:** https://[CC1 Host]:30020/avxapi/download-crl/[CA_Name]/crl.crl , https://[CC2 Host]:30020/avxapi/download-crl/[CA_Name]/crl.crl
   - **To publish the OCSP URL for the users/devices that do not have access to the SaaS tenant:**
     - Create a certificate issuance template for those endpoints to enroll certificates from, and include the custom OCSP URL as the LB URL.
     - The LB URL can be load balanced between cloud connectors.
     - **URL to be configured in templates:** https://[LB Host]:[httpsport]/ocsp
     - **LB to be balanced:** http://[CC1 Host]:30022/ocsp, http://[CC2 Host]:30022/ocsp

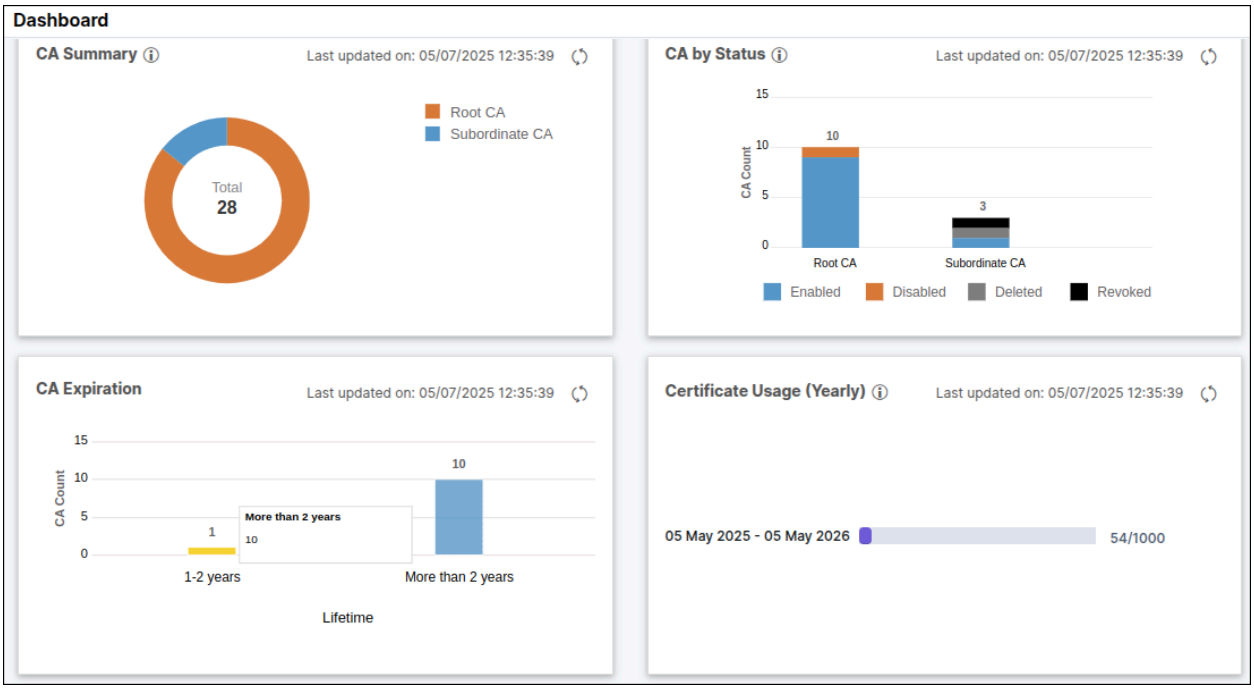
Dashboard

This page gives a quick summary of all the root and the intermediate CAs created via AppViewX PKIaaS.

- For Standard Initialization
- For PKIaaS Native Initialization

For Standard Initialization

CA Insight



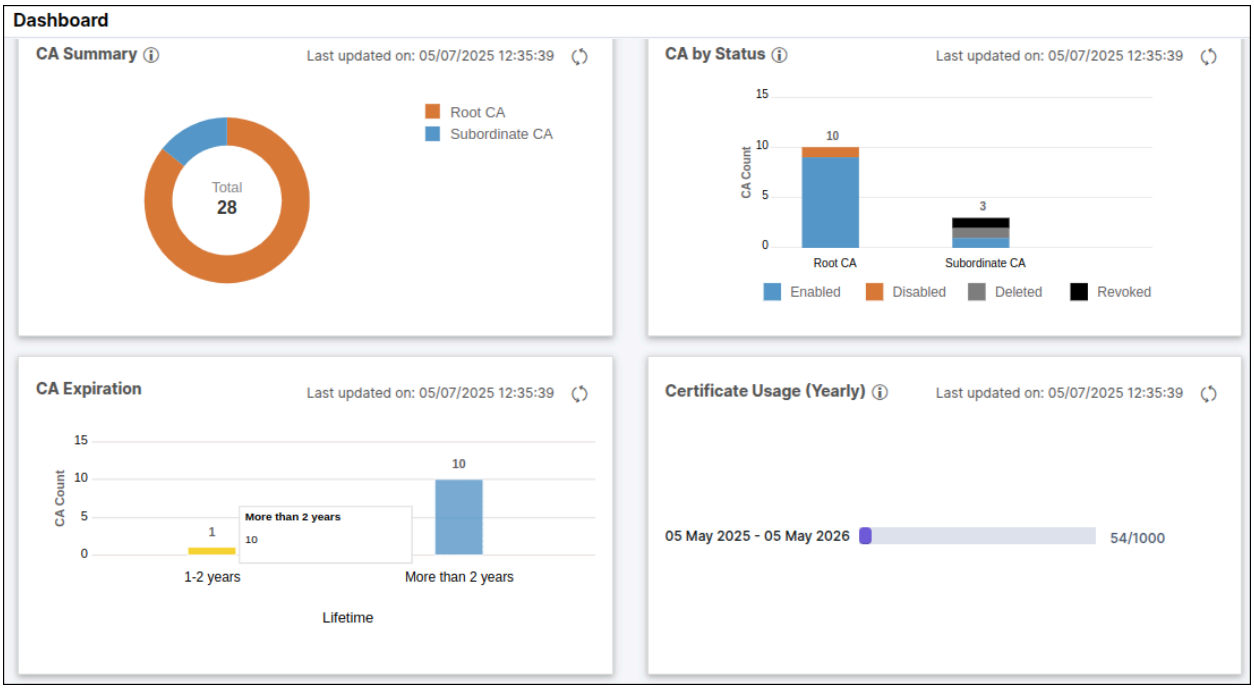
The following widgets are displayed on this page:

- **CA Summary:** This widget displays the number of CAs created via AppViewX PKIaaS. This contains the root and intermediate CAs. Click the graph to redirect you to the CA inventory for the selected CAs.
- **CA by Status:** This widget displays the CA count based on the status in the CA inventory. Click the graph to redirect you to the CA inventory for the selected CAs.
- **CA Expiration:** This widget displays the CA count based on the expiry date. Click the graph to redirect you to the CA inventory for the selected CAs.
- **Certificate Usage (Yearly):** This widget displays the yearly certificates count starting from the initialization date showing the certificate issuance/license count by year for the reset date.

For PKIaaS Native Initialization

The following dashboards are available:

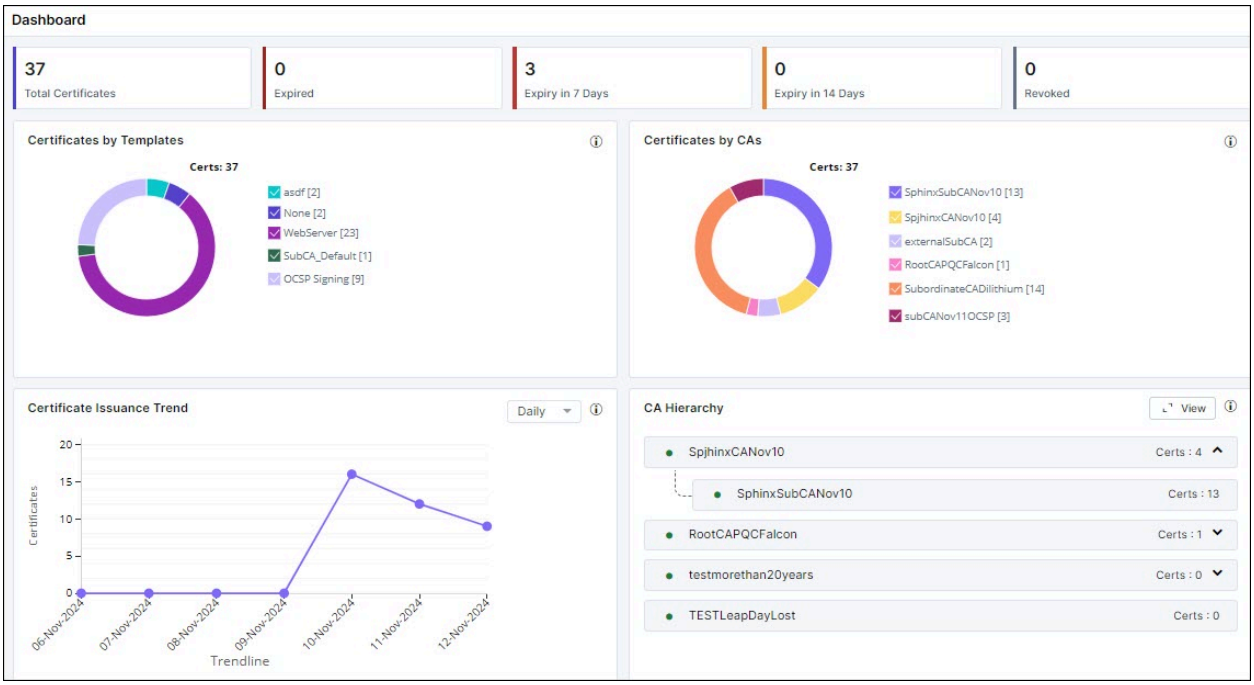
CA Insight



The following widgets are displayed on this page:

- **CA Summary:** This widget displays the number of CAs created via AppViewX PKIaaS. This contains the root and intermediate CAs. Click the graph to redirect you to the CA inventory for the selected CAs.
- **CA by Status:** This widget displays the CA count based on the status in the CA inventory. Click the graph to redirect you to the CA inventory for the selected CAs.
- **CA Expiration:** This widget displays the CA count based on the expiry date. Click the graph to redirect you to the CA inventory for the selected CAs.
- **Certificate Usage (Yearly):** This widget displays the yearly certificates count starting from the initialization date showing the certificate issuance/license count by year for the reset date.

End certificates count by CA



The following widgets are displayed on this page:

- **Certificates by Templates:** This widget displays all the certificates created via the various templates. By default, all the templates are selected.
- **Certificates by CAs:** This widget displays all the certificates created via the various CAs. By default, all the CAs are selected.
- **Certificate Issuance Trend:** This widget displays all the certificates based on their issuance trend such as daily, weekly, monthly, yearly, and custom. By default, daily is selected.
- **CA Hierarchy:** This widget displays the number of CA hierarchies. Click **View** to display the CA hierarchies created and the certificates under each of the CAs listed along with their count.

By default, the widgets display the certificates for all CAs. You can filter data by selecting a particular CA from the dropdown list on the top right corner of the page.

Custodian Management

Custodians are responsible for approving any action performed in PKI. Custodians are the individuals responsible for issuance of root and intermediate certificates. They approve or reject any action performed on the CA certificates. Custodians typically work in a M-of-N model (or M/N model) to ensure high levels of security and prevent unauthorized issuance of certificates.

Onboard at least two custodians before creating CA hierarchy. You can complete the addition of custodians by going to (**Menu**) icon **> PKI+ > Custodian Management** with the following privileges under RBAC roles and resources.

1. Roles automation > service request full
2. PKI+ > view all (optional)
3. Resources > workflow studio, workflow request > PKIaaS, approval_request

Note:

No CA action is possible until at least two active custodians are in the system.

Any administrator can add custodians from the **Custodian Management** page if the key ceremony admins are not configured. Key ceremony admins are an additional layer of control delegation on who can have the authority to add or modify custodians. This is an optional field. Key ceremony admins cannot be added as custodians.

Note:

Only two key ceremony admins can be added.

Key Ceremony Process

Virtual key ceremony in AppViewX PKI is where customers can set a closed group of CA administrators (custodians).

The approvals are based on a M(N) method with a user-defined quorum value, where **M** is the minimum number of custodians required to approve an action, and **N** is the total number of custodians available. A **Quorum** value is the minimum percentage of the number of custodians that must agree or participate to authorize an action or to make a decision regarding the lifecycle of CA Certificates. The default quorum is set to 51%, for example, if the custodian group has three members, then at least two custodians must approve any action to achieve 51% of quorum.

The first custodian is auto-approved and the approval flow gets triggered after adding the second custodian. On adding the second custodian, the individual receives a notification stating *Email Verification - Pending*. Once the email verification is completed, an approval link is sent to the first custodian. Upon approval, the second custodian transitions to the active state.

Onboarding Custodians

Prerequisite

On-prem users need to configure the SMTP server for Custodian Management by clicking the link provided on the **Getting Started with PKI+** Web page for instructions.

To onboard custodians:

1. Go to (**Menu**) icon **> PKI+ > Custodian Management.**
2. Enter the following fields:

Field Description for Custodian Management page

Field	Description
***Quorum Value**	By default, the quorum value is configured to 51%. This value signifies the minimum number of approvals needed for tasks such as adding or removing custodians and approving the creation of a certificate authority (CA). For instance, if there are three custodians, the minimum approval required is rounded off to two. In case of six custodians, the minimum approval required is four.
***Approval Link Validity**	By default, the approval link is valid for 30 minutes. Minimum value is 10 minutes while maximum value is 7 days.

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

3. Click **Save.**
4. Add custodians by entering the following information:

Field Description for Custodian Management page

Field	Description
***Username**	Select from the list of usernames. **Important:** SSO users must log in to AppViewX at least once for their names to appear in the dropdown list.
***Email ID**	This field is auto-populated on selection of a username. This email address of the custodian is where the approval link and notification messages are sent.
***First Name**	The first name of the custodian being added. If this is not auto-populated, then type the first name. **Important:** Custodian must have login access to AppViewX.
***Last Name**	The last name of the custodian being added. If this is not auto-populated, then type the last name.
Note: Fields marked with red asterisk (*) are mandatory.	

5. Click **Add.**

Note:
If the custodian being added is not part of the AppViewX users, then the following confirmation screen appears. Click **Save and Continue** to proceed as an SSO user.

Important:

- If any of the approvals is in the pending state, then no new actions on the CA or the Custodian Management pages are allowed until the current one is either approved, rejected, or aborted.
- At least two custodians must be added to perform the M(N) approvals in PKI.

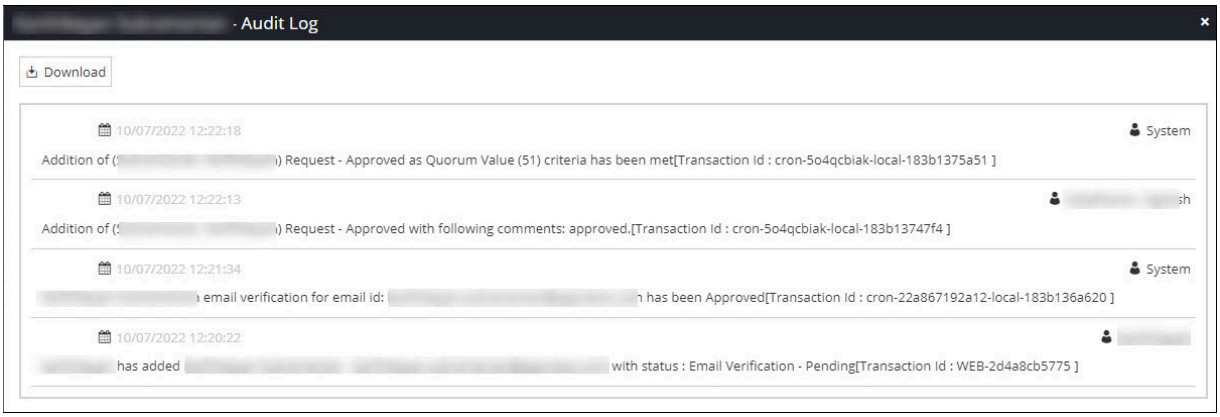
Action Status Description and Required Action

Action Status	Status	Description	Required Action
Email Verification - Pending	Approval Pending	The custodian's email verification is pending approval and is not active. **Note:** If you want to abort the action, click **Abort.** Any workflow that is triggered and is in progress is killed from the Request page prior to triggering any further actions.	The newly added custodian receives a notification email. Click the **here** hyperlink to be directed to the AppViewX login page. On successful login, users are directed to the approval page. Users can also approve the request by going to **Menu > Requests > All requests.** **Tip:** You can click the **(Notification Center)** on the top right-hand-corner of the page to verify the email address.
Create - Approval Pending	Approval Pending	The custodian has been added but is awaiting approval from active custodians.	Active custodians must click the **here** hyperlink in the email to be redirected to the AppViewX login page. On successful login, users are directed to the approval page. User can also approve the

Action Status	Status	Description	Required Action
			request by going to **Menu > Requests > All requests.** **Tip:** You can click the  **(Notification Center)** on the top right-hand-corner of the page to add the custodian to the custodian group.
Create - Approved	Active	The custodian has been approved and added successfully.	-
Email Verification - Rejected	Inactive	The custodian has been rejected.	On rejecting a request, a confirmation popup window appears if the requester wants to submit the request. Click **OK** to resubmit.

6. To add consecutive custodians, follow the aforesaid steps. Successful addition of custodians depends on the approval of active custodians per the quorum value set.
7. [Optional] Click **Audit Log** against each custodian for more information about the request and the response count along with comments, if any, from other approvers. You can also download the audit log by clicking the **Download** button on the Audit Log view page and exporting it in the .xls format.

Once the audit log is fully loaded, the **Loading** button will turn to **View.** Refresh the page to see the **View** button.



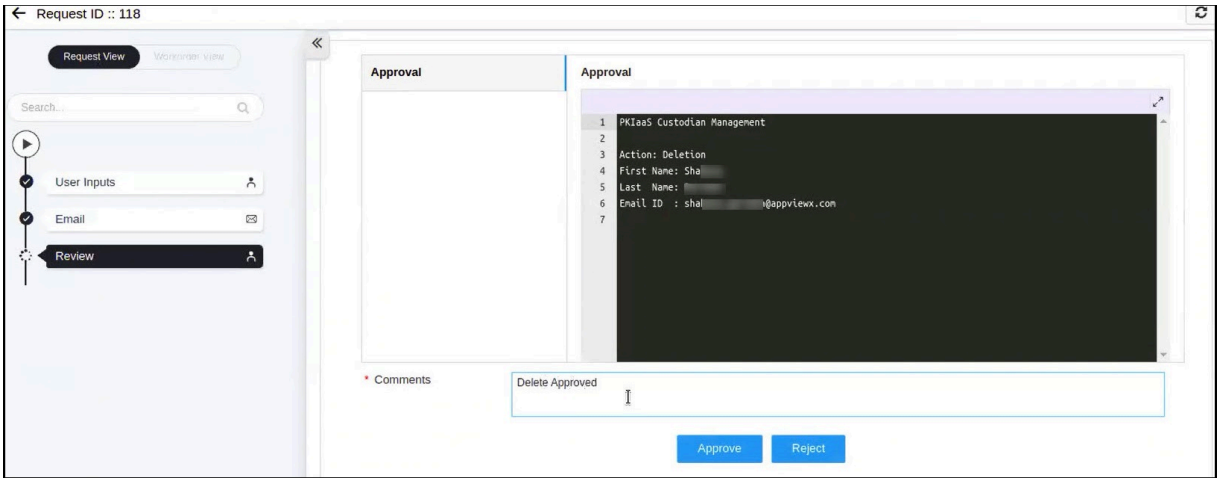
Deleting Custodians

Attention:

Any administrator or key ceremony administrator, if configured, can delete custodians. The quorum value must be met for m(n) approval.

To delete custodians:

1. Go to (**Menu**) icon **> PKI+ > Custodian Management.**
   The **Custodian Management** page appears.
2. Click the (**Delete**) icon against the custodian you want to delete.
   A deletion mail is sent to all active custodians. The approval status changes to *Delete - Approval Pending.*
3. Click **Approve** to delete the custodian.



A confirmation popup window appears.

4. Click **OK** to confirm.

   Once the approval count meet the minimum approval required by the quorum number, the custodian is deleted from the table. On successful approval, the approval status changes to *Delete - Approved* and the status changes to *Inactive.* If the deletion request is rejected, then the approval status changes to *Delete - Rejected* and the status remains as *Active.*

Filtering Custodians

You can apply filters on custodians based on their status using the **Filter By Status** option. The default filter includes both active and inactive selections. Clearing the filter allows you to see all entries.

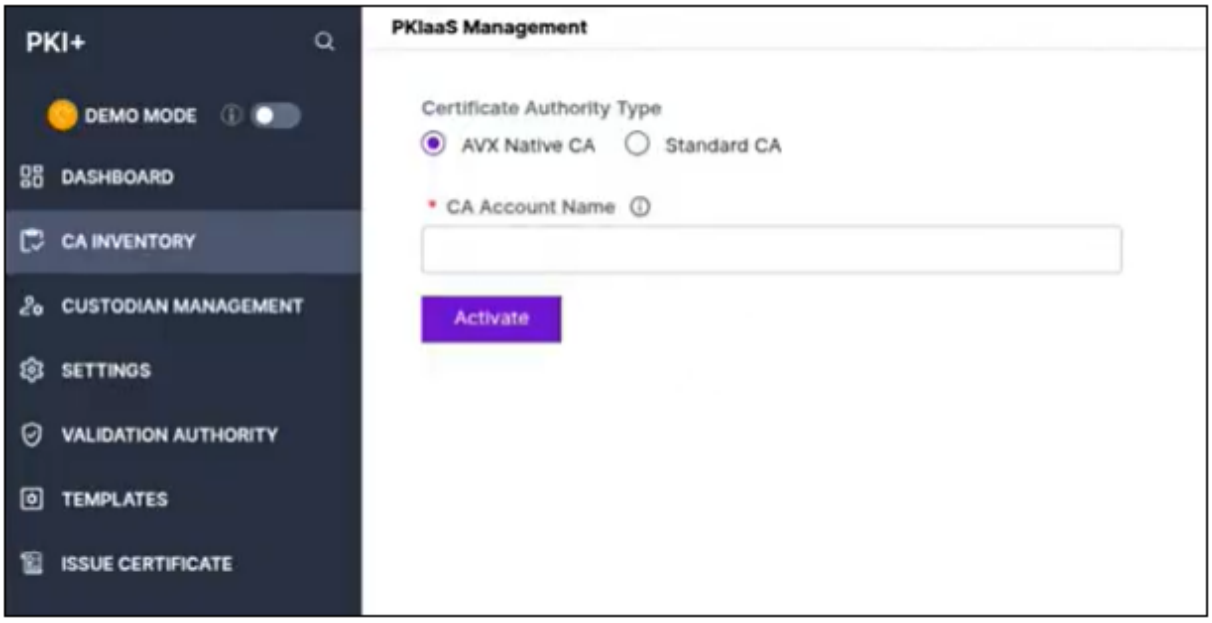
CA Inventory

You can use this page for:

- Creating Certificate Authority
- Complimentary CA
- Creating Root CA
- Creating Subordinate CA from PKIaaS Root CA
- Creating Subordinate CA from External Root CA
- Actions
- Expired CA

Creating Certificate Authority

1. Go to (**Menu**) icon **> PKI+ > CA Inventory.**
2. From the CA Inventory page, you can choose any of the certificate authority type:



- **AVX Native CA:** By default, this is selected. This CA is a PQC-ready PKI owned by AppViewX for issuing digital certificates with both traditional and PQC algorithms. These certificates are used to verify the authenticity and identity of parties in secure communications, such as SSL/TLS for websites, email encryption and others.

> **Note:**
>
> AVX Native CA once activated cannot be reversed.

- **Standard CA:** This is a private CA with a cloud CA backend to issue digital certificates with only traditional algorithms.

  To use this CA, you need to first initialize PKI. Reach out to saashelp@appviewx.com.

Once initialized, go to (**Menu**) icon **> CERT+ > Administration > Certificate Authority > AppViewX PKIaaS** to create the CA account.

Note:

You must create a CA account before adding custodians or CAs.

3. Enter a unique name in the **CA Account Name** field and click **Activate.**

Note:

Alternatively, you can create an AVX Native CA by going to **(Menu) icon > CERT+ > Administration > Certificate Authority.**

What to do Next:

- Create a root CA. See Creating Root CA.
- Create a subordinate CA from AppViewX PKIaaS root CA. See Creating Subordinate CA from PKIaaS Root CA.

-OR-

- Create a subordinate CA from an external root CA. See Creating Subordinate CA from External Root CA.

Complimentary CA

A complimentary CA is provided to all CERT+ customers. Customers using this CA cannot create a root CA but can create a subordinate CA that is signed by the AppViewX root CA as explained in Creating Subordinate CA from External Root CA. Once the CSR is downloaded, reach out to saashelp@appviewx.com. The complimentary CA can be deleted and re-created as required.

Creating Root CA

Note:

Customers using the complimentary CA can click the **+Create CA** button to directly create subordinate CA for external CA as explained in the Section, Creating Subordinate CA from External Root CA. The complimentary root CA is considered as an external CA in this case. The complimentary CA can be deleted and re-created as required.

To create root CA:

1. Go to (**Menu**) icon **> PKI+ > CA Inventory.**
   The **CA Inventory** page appears.
2. Click **+Create CA** on the top-right corner of the page.

   The **Create CA** page is displayed.
3. Enter the fields as described in the table.

Field Description for PKIaaS Management page

Field	Description
***CA Name**	Provide a friendly name for reference with no special characters except dash (-) and underscore (_).
Description	Provide a description for the CA. The maximum character limit is 500. Special characters that are not supported include ', ", ;, <, >, &, $, |, #, \, `.
Tier	This is a ready-only field. In case of standard initialization, **Standard** is selected; else **AppViewX PKIaaS Native** if it was used for PKI initialization.
Certificate Authority Type	Select **Root CA.**
***Template**	This field appears only if **Tier = AppViewX PKIaaS Native.** Select a template from the dropdown list.
***Valid for**	Select the number of years to CA expiry.
Configure CA Subject DN Details	

Field	Description
***CA Common Name**	Enter the root CA subject name.
***Organization**	Enter the organization name owning the CA.
Organization Unit	Enter the business unit for CA operations.
City	Enter the city name.
State	Enter the state name.
Country	Enter the country of the organization.
Configure CA Key Size and Algorithm	
CSR Generation	Select AppViewX if you are generating keys using HashiCorp Vault; else, select HSM.
Use Existing Key	Select this option if you want to use an existing key from HSM.
***Device**	This field is displayed only when **CSR Generation = HSM.** Select a configured device from the dropdown list. This list is retrieved from the HSM configured under the platform.
***Key Handler Name**	This field is displayed only when **CSR Generation = HSM.** You can either create the new key in HSM by providing the reference name or use an existing key handler name (alias/label name) in HSM by running the following command: `pkcs11-tool --module /path/to/pkcs11.so --list-objects` Click **Validate** button: • If validation is successful, then a message, *Key is available in the HSM,* is displayed. • If validation is unsuccessful, then a message, *Key is not available in the HSM,* is displayed. • If the key provided is not supported by the CA being created, then a message, *The algorithm for this key is not supported for CA creation,* is displayed.
***Key Size and Algorithm**	Select the CA key size and algorithm from the dropdown list. By default, RSA_PKCS1_4096_SHA256 is selected.

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

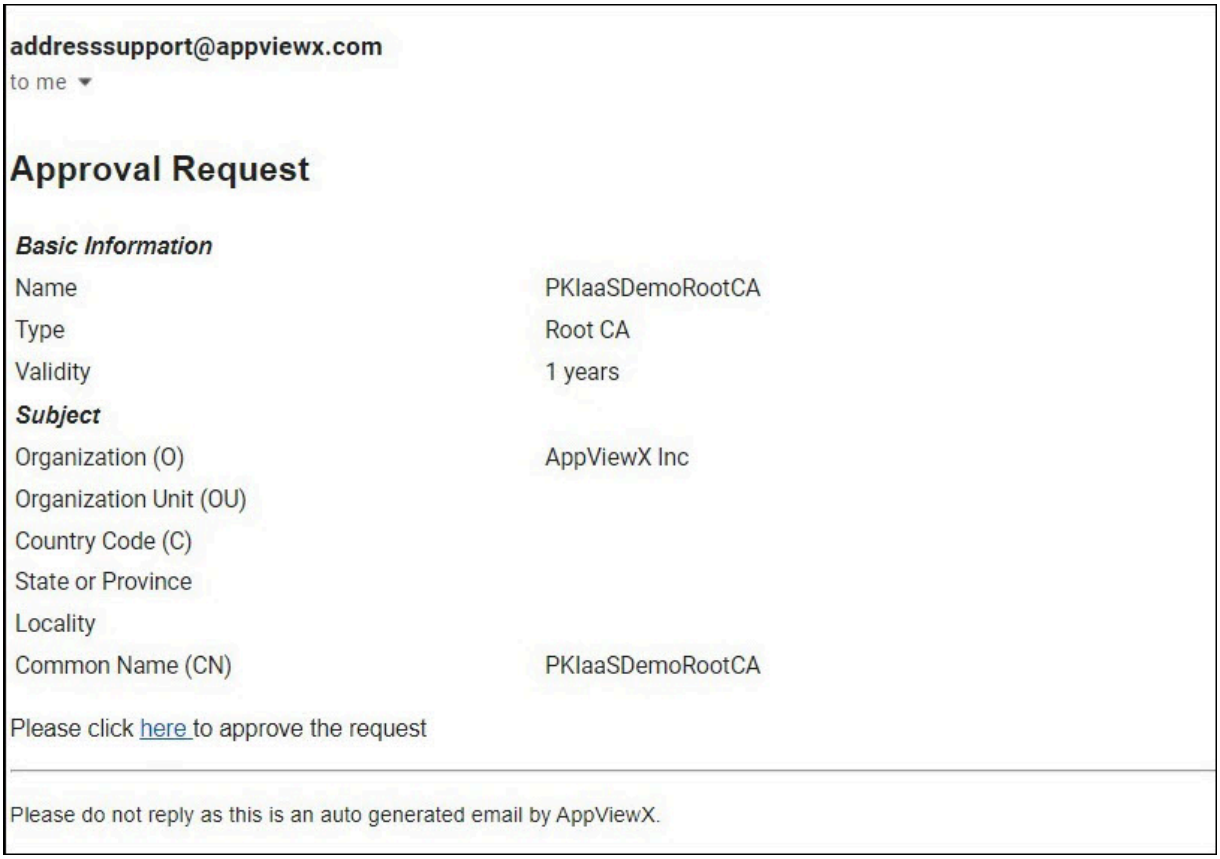
4. Click **Save.**

   A window with the summary of values entered appears.

5. Click **Proceed** to trigger the approval flow.

   The newly created CA appears in the table with the approval status as *Create - Approval Pending* and the status as *Awaiting Approval* until all the necessary approvals are completed. If you want to abort the action, then click **Abort.**

   An email from AppViewX is sent to all the active custodians for approving the CA.



addresssupport@appviewx.com
to me

Approval Request

Basic Information

Name	PKIaaSDemoRootCA
Type	Root CA
Validity	1 years

Subject

Organization (O)	AppViewX Inc
Organization Unit (OU)	
Country Code (C)	
State or Province	
Locality	
Common Name (CN)	PKIaaSDemoRootCA

Please click here to approve the request

Please do not reply as this is an auto generated email by AppViewX.

6. Click the **here** hyperlink in the email to be redirected to the AppViewX login page.

On successfully logging in, the approval request is displayed with the **Approve** and **Reject** buttons.

> **Tip:**
>
> You can also approve by clicking the  **(Notification Center)** on the top right-hand-corner of the page.

7. Enter the comments and click **Approve.**

   A confirmation popup window appears if you want to submit the request.

8. Click **OK.** Once the approval count reaches the minimum approval as set by the quorum number, the custodian is approved.

   The approval status changes to *Create - Approved* and the status to *In Progress* until the CA is created and is enabled.

9. Click the  (**Refresh**) icon to see the status as *Active* once the CA is activated. Click **Resubmit** if the action fails for any reason.

   Certificates can be issued from this CA. CRLs are generated for this CA.

10. [Optional] Click the **Audit Log** against the CA to view the audit log details. You can also download the audit log by clicking the **Download** button on the Audit Log view page. The audit log is exported in the .xls format.

> **Note:**
>
> Once the audit log is fully loaded, the **Loading** button will turn to **View.** Refresh the page to see the **View** button.

11. [Optional] Click the **Approval Status** column value link to check the update on approvals.

> **Note:**
>
> The PKI CA thus created cannot be modified but can be viewed from the **PKI+ > CA Inventory** page.

What to do next:

- Creating Subordinate CA from PKIaaS Root CA -OR-
- Creating Subordinate CA from External Root CA

Creating Subordinate CA from PKIaaS Root CA

To create subordinate CA from PKIaaS root CA:

1. Go to (**Menu**) icon **> PKI+ > CA Inventory.**
   The **CA Inventory** page appears.
2. Click **+Create CA** on the top-right corner of the page.
   The **Create CA** page is displayed.
3. Enter the fields as described in the table.

Field Description for PKIaaS Management page

Field	Description
Select CA Type	
***CA Name**	Provide a friendly name for reference with no special characters except dash (-) and underscore (_).
Description	Provide a description for the CA. The maximum character limit is 500. Special characters that are not supported include ', ", ;, <, >, &, $, |, #, \, `.
Tier	This is a ready-only field. In case of standard initialization, **Standard** is selected; else **AppViewX PKIaaS Native** if it was used for PKI initialization.
Certificate Authority Type	Select **Subordinate CA.** On clicking **Subordinate CA,** you see **Root CA** field with **External** and **PKIaaS** options.
Root CA	This field appears only on selecting **Subordinate CA.** Select **PKIaaS** if root CA is already in the AppViewX system. **Note:** Subordinate CAs need to activated and shows status as *Create - Approval Pending* until they are approved by the active custodians.
***Issuer Name**	This field appears only on selecting **Subordinate CA** as *PKIaaS*. Select an issuer name from the dropdown list.

Field	Description
***Template**	This field appears only if **Tier = AppViewX PKIaaS Native.** Select a template from the dropdown list.
***Valid for**	Select the number of years to CA expiry.
Configure CA Subject Name	
***CA Common Name**	Enter the root CA subject name.
***Organization**	Enter the organization name owning the CA.
Organization Unit	Enter the business unit for CA operations.
City	Enter the city name.
State	Enter the state name.
Country	Enter the country of the organization.
Configure CA Key Size and Algorithm	
CSR Generation	Select AppViewX if you are generating keys using HashiCorp Vault, else select HSM.
Use Existing Key	Select this option if you want to use an existing key from HSM.
***Device**	This field is displayed only when CSR Generation = HSM. Select a configured device from the dropdown list.
***Key Handler Name**	This field is displayed only when CSR Generation = HSM. You can either create the new key in HSM by providing the reference name or use an existing key handler name (alias/label name) in HSM by running the following command: `pkcs11-tool --module /path/to/pkcs11.so --list-objects` Click **Validate** button: • If validation is successful, then a message, *Key is available in the HSM,* is displayed. • If validation is unsuccessful, then a message, *Key is not available in the HSM,* is displayed. • If the key provided is not supported by the CA being created, then a message, *The algorithm for this key is not supported for CA creation,* is displayed.

Field	Description
***Key Size and Algorithm**	Select the CA key size and algorithm from the dropdown list. By default, RSA_PKCS1_4096_SHA256 is selected.
Configure CA Artifacts	
Path Length Constraint	This is an optional parameter in an issuing CA certificate; it defines the number of sub CA chains created under that specific issuing CA certificate holding the path constraint value. By default, the value is None. This field can have any of these values: 0, 1, 2, 3, or none. For example, if it is set to **2**, it means that only two intermediate CAs are allowed between the end-entity certificate and this CA certificate. None indicates unlimited.
Custodian Settings	
Custodian	By default, the SaaS trial customer (logged in user) is added as the custodian. He/she will get the approval links via email for all the actions performed in the PKI hierarchy creation. Click **Manage** to add more custodians.
Note: Fields marked with red asterisk (*) symbol are mandatory.	

4. Click **Create.**
   A window with the summary of values entered appears.
5. Click **Proceed** to trigger the approval flow.
   The newly created CA appears in the table with the status as *Create - Approval Pending.*

   An email from AppViewX is sent to all the active custodians for approving the CA. If you want to abort the action, then click **Abort.**
6. Click the **here** hyperlink in the email to be redirected to the AppViewX login page.
   On successfully logging in, the approval request is displayed with the **Approve** and **Reject** buttons.

> **Tip:**
>
> You can also approve by clicking the  **(Notification Center)** on the top right-hand-corner of the page.

7. Enter the comments and click **Approve.**
   A confirmation popup window appears if you want to submit the request.
8. Click **OK.** Once the approval count reaches the minimum approval as set by the quorum number, the custodian is approved.
9. Click the  (**Refresh**) icon on the **PKIaaS Management** page to see the *Active* status. Click **Resubmit** if the action fails for any reason.
   Once the PKIaaS subordinate CA is activated, the status changes to *Active*.
10. [Optional] Click the **Audit Log** against the CA to view the audit log details. You can also download the audit log by clicking the **Download** button on the Audit Log view page. The audit log is exported in the .xls format.

> **Note:**
>
> Once the audit log is fully loaded, the **Loading** button will turn to **View.** Refresh the page to see the **View** button.

11. [Optional] Click the **Approval Status** column value link to check the update on approvals.

Creating Subordinate CA from External Root CA

> **Note:**
>
> If you are using the complimentary root CA created in AppViewX, then you can create subordinate CA from external root CA as explained here.

To create subordinate CA from external root CA:

1. Go to  (**Menu**) icon **> PKI+ > CA Inventory.**
   The **CA Inventory** page appears.
2. Click **+Create CA** on the top-right corner of the page.
   The **Create CA** page is displayed.

3. Enter the fields as described in the table.

Field Description for PKIaaS Management page

Field	Description
Select CA Type	
***CA Name**	Provide a friendly name for reference with no special characters except dash (-) and underscore (_).
Description	Provide a description for the CA. The maximum character limit is 500. Special characters that are not supported include ', ", ;, <, >, &, $, |, #, \, `.
Tier	This is a ready-only field. In case of standard initialization, **Standard** is selected; else **AppViewX PKIaaS Native** if it was used for PKI initialization.
Certificate Authority Type	Select **Subordinate CA.** On clicking **Subordinate CA,** you see **Root CA** field with **External** and **PKIaaS** options.
Root CA	This field appears only on selecting **Subordinate CA.** Select **External** if root CA is outside of the AppViewX system.
***Template**	This field appears only if **Tier = AppViewX PKIaaS Native.** Select a template from the dropdown list.
***Valid for**	Select the number of years to CA expiry.
Configure CA Subject DN Details	
***CA Common Name**	Enter the root CA subject name.
***Organization**	Enter the organization name owning the CA.
Organization Unit	Enter the business unit for CA operations.
City	Enter the city name.
State	Enter the state name.
Country	Enter the country of the organization.
Configure CA Key Size and Algorithm	
CSR Generation	Select AppViewX if you are generating keys using HashiCorp Vault, else select HSM.

Field	Description
Use Existing Key	Select this option if you want to use an existing key from HSM.
***Device**	This field is displayed only when CSR Generation = HSM. Select a configured device from the dropdown list.
***Key Handler Name**	This field is displayed only when CSR Generation = HSM. You can either create the new key in HSM by providing the reference name or use an existing key handler name (alias/label name) in HSM by running the following command: `pkcs11-tool --module /path/to/pkcs11.so --list-objects` Click **Validate** button: • If validation is successful, then a message, *Key is available in the HSM,* is displayed. • If validation is unsuccessful, then a message, *Key is not available in the HSM,* is displayed. • If the key provided is not supported by the CA being created, then a message, *The algorithm for this key is not supported for CA creation,* is displayed.
***Key Size and Algorithm**	Select the CA key size and algorithm from the dropdown list. By default, RSA_PKCS1_4096_SHA256 is selected.
Configure CA Artifacts	
Path Length Constraint	This is an optional parameter in an issuing CA certificate; it defines the number of sub CA chains created under that specific issuing CA certificate holding the path constraint value. By default, the value is None. This field can have any of these values: 0, 1, 2, 3, or none. For example, if it is set to **2**, it means that only two intermediate CAs are allowed between the end-entity certificate and this CA certificate. None indicates unlimited.
Custodian Settings	
Custodian	By default, the SaaS trial customer (logged in user) is added as the custodian. He/she will get the approval links via email for all the actions performed in the PKI hierarchy creation. Click **Manage** to add more custodians.

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

4. Click **Create.**
   A window with the summary of values entered appears.
5. Click **Proceed** to trigger the approval flow.
   The newly created CA appears in the table with the status as *Create - Approval Pending.* If you want to abort the action, then click **Abort.**

   An email from AppViewX is sent to all the active custodians for approving the CA.
6. Click the **here** hyperlink in the email to be redirected to the AppViewX login page.
   On successfully logging in, the approval request is displayed with the **Approve** and **Reject** buttons.

> **Tip:**
>
> You can also approve by clicking the  **(Notification Center)** on the top right-hand-corner of the page.

7. Enter the comments and click **Approve.**
   A confirmation popup window appears if you want to submit the request.
8. Click **OK.** Once the approval count reaches the minimum approval as set by the quorum number, the custodian is approved.
9. Click the  (**Refresh**) icon.
10. Click **Activate.** Until the signed certificate is uploaded, the status of the external subordinate CA remains as *Pending Signed Certificate.*
    The **Certificate Authority Activation** window appears.
11. Click **Download CSR.**
12. Once the CSR is downloaded, sign with valid root CA and click **Upload.**

> **Note:**
>
> Copy and paste or upload the complete certificate chain, ordered from leaf to root, starting with the subordinate certificate authority being activated.

Once the external subordinate CA is activated, the status changes to *Active*. Click **Resubmit** if the action fails for any reason.

13. [Optional] Click the **Audit Log** against the CA to view the audit log details. You can also download the audit log by clicking the **Download** button on the Audit Log view page. The audit log is exported in the .xls format.

> **Note:**
>
> Once the audit log is fully loaded, the **Loading** button will turn to **View.** Refresh the page to see the **View** button.

14. [Optional] Click the **Approval Status** column value link to check the update on approvals.

Actions

Prerequisites

Prior to performing any action on the CA, ensure that you have necessary role-based access controls and workflow access pertaining to the CA request.

You can perform the following actions from the **Actions** menu of the **PKIaaS Management** page:

- Enable
- Disable
- Renew
- Revoke
- Delete

Enable

You can enable a root CA or a subordinate CA. Certificates can be issued from this CA. CRLs are generated for this CA.

To enable CA:

1. Go to (**Menu**) icon **> PKI+ > CA Inventory.**
   The **CA Inventory** page appears.
2. Select the check box against the CA Name you want to enable.

3. Click **Actions** and select **Enable** from the dropdown menu.

   The approval status of the CA changes to *Enable - Approval Pending.* If you want to abort the action, then click **Abort.**

4. An email from AppViewX PKIaaS for approval is sent to all active custodians. Approval can be done either via email or by clicking the **(Notification Center)** on the top right-hand-corner of the page. Once the approval meets the quorum value, the CA is enabled. The approval status of the CA changes to *Enable - Approved* and the status changes to *Active.* If the request is rejected, then the approval status of the CA changes to *Enable - Rejected.* Click **Resubmit** if the action fails for any reason.

   A message that the operation is performed successfully appears.

   You can follow the aforesaid steps to enable CAs.

   You can view all the enabled CAs by selecting **Enabled** option from **Filter by Status.**

Disable

You can disable a root CA or a subordinate CA. No certificates can be issued from a disabled CA. CRLs will still be generated.

To disable CA:

1. Go to (**Menu**) icon **> PKI+ > CA Inventory.**
   The **CA Inventory** page appears.
2. Select the check box against the CA Name you want to disable.
3. Click **Actions** and select **Disable** from the dropdown menu.
   The approval status of the CA changes to *Disable - Approval Pending* and the status remains as *Active.* If you want to abort the action, then click **Abort.**
4. An email from AppViewX PKIaaS for approval is sent to all active custodians. Approval can be done either via email or by clicking the **(Notification Center)** on the top right-hand-corner of the page. Once the approval meets the quorum value, the CA is disabled. The approval status of the CA changes to *Disable - Approved* and the status to *Disabled.* If the request is rejected, then the approval status changes to *Disable - Rejected* and the status remains as *Active.* Click **Resubmit** if the action fails for any reason.
   You can follow the aforesaid steps to disable CAs.

   You can view all the disabled CAs by selecting **Disabled** option from **Filter by Status.**

Renew

CA certificates are fundamental to public key infrastructure (PKI) systems. When the CA certificate approaches its expiration date, it is crucial to renew it to maintain the integrity of encrypted communications and the security of the entire ecosystem relying on it.

If you want to extend the validity of the current CA using same private key, you can renew it within the existing PKI inventory.

> **Note:**
> Renewal action is applicable only for certificates issued by AVX Native CA.

Before starting the renewal process, ensure that you:

- Check the expiration date of the existing CA certificate.
- Identify all dependent systems, certificates, and services relying on the CA to be renewed.
- Review the signature algorithms and the certificate policies to ensure they adhere to the current security standards.

To renew CA:

1. Go to (**Menu**) icon **> PKI+ > CA Inventory.**
   The **CA Inventory** page appears.
2. Select the check box against the CA Name you want to renew.
3. Click **Actions** and select **Renew** from the dropdown menu.
   The **Renew CA** page is displayed.

   > **Note:**
   > All fields are read-only except for **Template, Valid for, Configure CA Subject DN Detail,** and **Key Size and Algorithm.**

4. Enter the renewal period in the **Valid for** field.
5. Modify the **Key Size and Algorithm**, if required.
6. Click **Renew.**
   A **Confirm CA Renewal** pop-up window with the message, *CA certificate will be replaced all references to the previous, with the newly renewed CA certificate in auto-enrollment, policy and enrollment pages,* is displayed.

7. Click **Proceed** to confirm the changes.
   The custodians receive an email with the subject line, *PKIaaS CA Management: CA renewal,* in their inbox. Approval can be done either via email or by clicking the **(Notification Center)** on the top right-hand-corner of the page.

   Once the necessary custodian approvals are completed, the **Approval Status** changes from *Renewal - Approval Pending* to *Renewal Approved.*

What to do next:

- You can enroll certifications by referring to the steps detailed in the Section, Adding/Enrolling Certificate.
- You can click the **View Certificate** ( ) icon and click the Common Name to access the holistic view and download the certificate.
- You can view the audit log.

Revoke

Certificate revocation is the process of invalidating a digital certificate before its scheduled expiration date. Revocation is typically done when a CA's certificate is compromised, expired, or no longer needed. This is done to ensure the security and trustworthiness of systems that rely on certificates for authentication, encryption, and secure communication. As soon as the certificate is revoked, the certificate is no longer considered to be trusted. Revoked certificates are listed in the Certificate Revocation List (CRL) maintained by each certificate authority.

Note:

Revocation can be performed only on PKIaaS subordinate CAs.

To revoke CA:

1. Go to (**Menu**) icon **> PKI+ > CA Inventory.**
   The **CA Inventory** page appears.
2. Select the check box against the CA Name you want to renew.
3. Click **Actions** and select **Revoke** from the dropdown menu.
   The **CA Certificate Revoke** window is displayed.
4. Select the reason for revocation from the dropdown list.

By default, the reason for revocation is set to **Key compromise,** and the **Revoke All Certificates** checkbox is disabled. This action will revoke every CA certificate linked to this private key, including all the renewed versions. As a result, all related end-entity certificates will be invalidated.

On selecting a different revocation reason and unselecting the **Revoke All Certificates** checkbox, you can revoke only the currently active CA certificate linked to this private key. As a result, all related end-entity certificates will be invalidated.

5. Click **Revoke.**
   A message, *Revoking this Certificate Authority (CA) may disrupt certificate validation and affect trust for all issued certificates. Please ensure that you understand the impact before proceeding with revocation.This will affect the autoenrollment configuration. Please verify the autoenrollment settings having this CA,* is displayed.
6. Click **Proceed** to confirm the changes.
   The custodians receive an email with the subject line, *PKIaaS CA Management: CA revocation,* in their inbox. Approval can be done either via email or by clicking the  **(Notification Center)** on the top right-hand-corner of the page.

   Once the necessary custodian approvals are completed, the **Approval Status** changes from *Revocation - Approval Pending* to *Revocation Approved.*

   You can view all the revoked CAs by selecting **Revoked** option from **Filter by Status.**

Delete

Before you begin:

- Remove the CA you want to delete from any auto-enrollment settings, policies, or workflows that are used to issue or revoke certificates from that CA.
- Check for any unrevoked and unexpired certificates that may have been deleted from the AppViewX inventory by running a CA discovery to get all the valid certificates issued by that CA for revocation.

You can delete a root CA or a subordinate CA (PKIaaS or external). Once the CA has been deleted, no new certificates can be issued from this CA and no new CRLs will be generated.

To delete CA:

1. Go to (**Menu**) icon **> PKI+ > CA Inventory.**
   The **CA Inventory** page appears.
2. Select the check box against the CA you want to delete.
3. Click **Actions** and select **Delete** from the dropdown menu.

   Note:
   - If you are deleting a subordinate CA and if there are valid certificates issued by the CA, then you get a message that you must first revoke the certificates and the CA certificate before deleting the CA. The revocation of certificates is permanent and not reversible. Click **Continue** to view the certificates that will be revoked. Click **Revoke and Delete CA.**
   - If the CA has no active certificates, then the delete workflow is triggered.

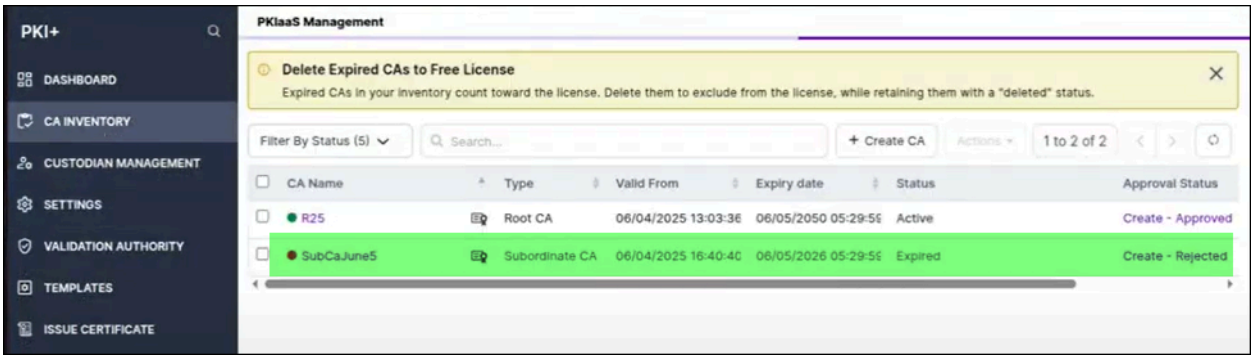
   The approval status of the CA changes to *Delete - Approval Pending.* If you want to abort the action, then click **Abort.**
4. An email from AppViewX PKIaaS for approval is sent to all active custodians. Approval can be done either via email or by clicking the **(Notification Center)** on the top right-hand-corner of the page. Once the approval meets the quorum value, the approval status of the CA changes to *Delete - Approved* and the status changes to *Deleted.* If the request is rejected, then the approval status of the CA changes to *Delete - Rejected.* Click **Resubmit** if the action fails for any reason.
   A message that the operation is performed successfully appears.

   Note:
   - If deletion fails, reach out to saashelp@appviewx.com.
   - You can view all the deleted CAs by selecting **Deleted** option from **Filter by Status.**

Expired CA

A PKI administrator can quickly identify CA certificates nearing expiry and take action before they expire, ensuring the PKI system remains secure and up-to-date.

When there are expired CA certificates in the inventory, they will be shown automatically on the **CA Inventory page** on logging in. The expired CA certificates are displayed with a red indicator along with the status as **Expired** and expiry date. The administrator can renew or delete these expired CA certificates.



You can view all the expired CAs by selecting **Expired** option from **Filter by Status.**

Validation Authority

Certificate authorities use Online Certificate Status Protocol (OCSP) to get the revocation status of x.509 digital certificates. When a user requests the validity of a certificate, an OCSP request is sent to an OCSP server for verification against a trusted certificate authority. The OCSP server then returns a response indicating whether the certificate is good, revoked, or unknown.

Prerequisites

- OCSP URL must be published in the AIA field of the certificate with the AppViewX OCSP server URL.
- **Plugins required:** OCSP Server and OCSP Generator must be deployed for OCSP to work.
- For on-premise deployment, configure OCSP as explained here.

You can select one or more certificates from the inventory and click **Actions > Revocation Check** to perform revocation validation. After successful validation, the certificate status is reflected through color-coding in the Common Name column.

- CRL Profiles
- OCSP Profiles

CRL Profiles

Note:

- This module is available starting from the Thames HF2 (2024.0.2.0) release for those using AppViewX PKIaaS Native CA for PKI initialization.
- CRL routed via CC will work only with the latest version of CC.

CRL Scheduler

The CRL scheduler ensures that CRLs for root and subordinate CAs are automatically generated and updated at regular intervals as defined by the CA's policies. The frequency of updates may depend on the CA's configuration (e.g., daily, weekly, etc.). To do it manually, click **Publish Now** in **Actions**.

Note:

The **CRL Scheduler** and **Actions** are available only for AppViewX Private CA. Ensure that you have necessary role-based access controls and workflow access to publish CRL.

Enter the following details:

Fields for CRL Scheduler

Field	Description
***Timezone**	Select a timezone from the dropdown list.
Starts on	Select a start date and time by clicking the calendar.
***Frequency**	Select the frequency as daily, weekly, or monthly.
***Days of Week**	This field appears only for root CA. Select the days of the week you want the scheduler to run.
***Overlap Period**	Select the overlap period in days or weeks.

Note:

Field marked with red asterisk (*) symbol are mandatory.

OCSP Profiles

> **Note:**
>
> OCSP routed via CC will work only with the latest version of CC.

You can create the following OCSP profile by going to **PKI+ > Validation Authority > OCSP:**

OCSP Signing: By default, an OCSP signing certificate is created along with a new CA creation. Clicking this field lists all the valid OCSP signing certificates available in the AppViewX PKIaaS inventory along with common name, serial number, issuer common name, extended key usage, and status.

> **Note:**
>
> Only one OCSP signing certificate is active at any given point of time.

- If you want to activate a selected OCSP signing certificate, you can do it from **Actions > OCSP Signing.** The OCSP configuration is updated with the selected certificate.

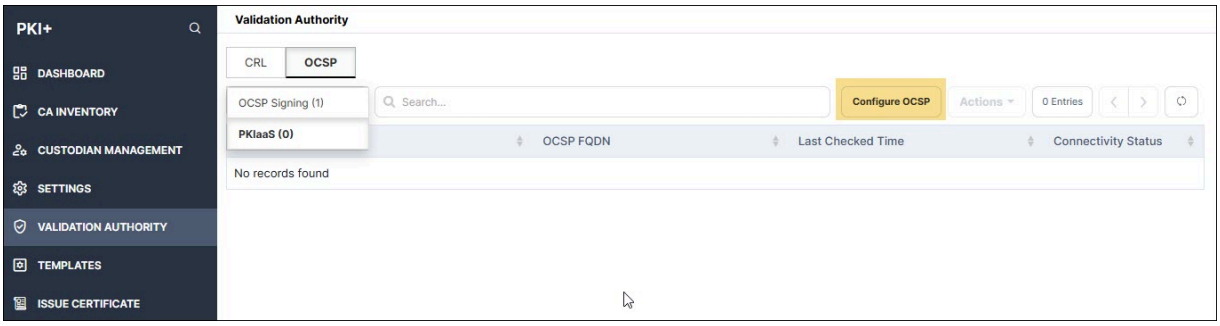
  > **Note:**
  >
  > An OCSP signing certificate can be revoked only on deleting the CA. If an OCSP signing certificate is revoked or deleted from the **CERT+ > Certificate Inventory > Server** page, then the OCSP responder will not work. To remediate this action, you can create a new OCSP signing certificate by going to **CERT+ > Certificate Action > Enroll Certificate** and following the procedure explained in the Section, Creating OCSP Signing Certificate.

- Configuring OCSP for On-Premise Deployment
- Creating OCSP Signing Certificate

Configuring OCSP for On-Premise Deployment

To configure OCSP:

1. Go to (**Menu**) icon **> PKI+ > Validation Authority.** By default, CRL is selected.
2. Click the **OCSP** tab and click **PKIaaS** from the dropdown list as shown.



3. Click **Configure OCSP.**

   The **Configure OCSP - PKIaaS** window is displayed.

4. Enter the following fields:

Field Description for Configure OCSP - PKIaaS page

Field	Description
***OCSP Name**	Provide a friendly name.
***OCSP FQDN**	Enter the node domain name where OCSP plugin is hosted.

Note:

Fields marked with red asterisk (*) are mandatory.

5. Click **Add.** The entered information is displayed in the table.

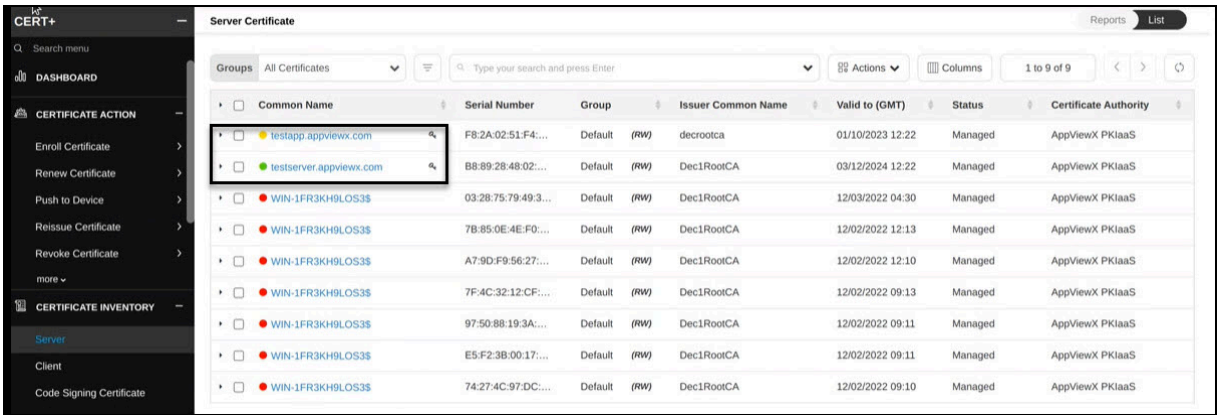
   To troubleshoot OCSP responder with openSSL, see Section, Troubleshooting OCSP Request with OpenSSL.

Creating OCSP Signing Certificate

To create an OCSP signing certificate:

1. Go to **CERT+ > Certificate Action > Enroll Certificate.**
   The **Enroll Certificate** page is displayed.
2. Select the **Certificate Authority** as *AppViewX PKIaaS.*
3. Select the **Certificate Profile** as *OcspSigning.*
4. Fill out the other fields as explained in the Section, Adding/Enrolling Certificate.

The OCSP signing certificate appears on the **CERT+ > Certificate Inventory > Server** page as shown with a key symbol beside the common name.



Settings

You can use this page to set the value for the data center, which is reflected on the AppViewX PKIaaS Certificate Authority page. You can also configure custodian administrators who can control the actions on the **Custodian Management** page.

What to do next:

Onboarding Custodians

- For Standard Initialization
- For AVX Native Initialization

For Standard Initialization

1. Go to (**Menu**) icon **> PKI+ > Settings.**
   The **Settings** page appears.
2. Enter the fields as described in the table.

Fields for General Settings section

Field	Description
***Data Center**	Select a data center to establish connection with PKIaaS.
***Default Region**	Select a default region to create a CA.

Field	Description
Email ID for PKI+ Alerts	Enter email IDs of users who can receive PKI alerts. You can add more than one user email ID using a comma (,) as a separator. The service connectivity and each CA status are monitored periodically. When there is a failure, an alert is triggered and an email is sent. Reach out to saashelp@appviewx.com for help. The maximum duration for which you can receive alerts is 30 days.
Custodian Admins	Select two custodian administrators. **Note:** • Only the default administrators can add custodian administrators. • If custodian administrators are configured, only they have the authority to add or remove custodians. However, custodian administrators cannot be designated as custodians themselves. • SSO users cannot be custodian administrators.
Select Certificate Authorities	Select from the dropdown list. Certificates issued by selected authorities will be maintained in the *Managed* status in the CA inventory while certificates from unselected authorities will be maintained in the *Monitored* status in the CA inventory.
Note: Fields marked with red asterisk (*) symbol are mandatory.	

3. Click **Save.**

For AVX Native Initialization

1. Go to (**Menu**) icon **> PKI+ > Settings.**
   The **Settings** page appears.
2. Enter the fields as described in the table.

Field Description for General Settings section

Field	Description
Custodian Admins	Select two custodian administrators. **Note:** • Only the default administrators can add custodian administrators. • If custodian administrators are configured, only they have the authority to add or remove custodians. However, custodian administrators cannot be designated as custodians themselves. • SSO users cannot be custodian administrators.
Select Certificate Authorities	Select from the dropdown list. Certificates issued by selected authorities will be maintained in the *Managed* status in the CA inventory while certificates from unselected authorities will be maintained in the *Monitored* status in the CA inventory.
Note: Fields marked with red asterisk (*) symbol are mandatory.	

3. Click **Save.**
4. Upload the CPS document.

Fields for CPS Upload section

Field	Description
***Upload CPS**	A CPS (Certification Practice Statement) is a comprehensive document that defines the practices, procedures, and responsibilities of a Certificate Authority (CA) in issuing and managing digital certificates. It offers transparency into the CA's operations, detailing how certificates are requested, validated, issued, renewed, revoked, and how the CA ensures the security and integrity of these processes. The CPS is a critical element in PKI that establishes the trust framework governing the CA's activities. It is especially important for auditors,

Field	Description
	relying parties (those who verify certificates), and relying organizations to understand the CA's operational procedures, security safeguards, and risk management strategies. CP and CPS are configurable under templates as per the customer policies. Customers can also upload their CPS document (.pdf) to PKIaaS for hosting. In this case, the CPS URL will be auto generated while template configuration. The certificate policy link present in the template will be part of the issued certificate's policy extension. **Note:** If you are using AppViewX to host, then by default the URI is generated for the template that is reflected in the certificate, so the default URI has to be retained as is. If any changes are made, then those changes will be reflected in the certificate and the CPS will not be hosted.
Note: Field marked with red asterisk (*) symbol are mandatory.	

What to do Next:

- Enable **Templates** and **Issue Certificate** functions by going to (**Menu**) icon **> Platform > Role.** Search for the created administrator role and click the link. Switch to the **Authorized functions** tab and select the **Templates** and **Issue Certificate** check boxes in the PKI module.

Templates

Note:

- This module is available starting from the Thames HF2 (2024.0.2.0) release for those using AppViewX PKIaaS Native CA for PKI initialization.
- For versions prior to Thames FP1 HF3, enable **Templates** function by going to **(Menu)** icon **> Platform > Role.** Search for the created administrator role and click the link. Switch to the **Authorized functions** tab, and select the **Templates** check box in the PKI module.

You can either use any of the existing templates or create a customized template to specify certificate parameters.

Using Existing Templates

AppViewX PKIaaS Native CA provides the following preconfigured templates:

- **Basic EFS:** Used for enabling file encryption and decryption in Windows environments. It allows users to securely encrypt their files with a public key infrastructure (PKI) certificate, ensuring that only authorized users can access the encrypted files.
- **Code Signing:** Used to issue certificates that digitally sign software applications or scripts, ensuring their authenticity and integrity. This helps users verify that the code has not been tampered with and that it originates from a trusted source, providing assurance and security in software distribution.
- **DomainController:** Used to secure domain controllers in Active Directory environments. It enables secure LDAP (LDAPS) communication and authenticates domain controllers, ensuring trusted interactions within the domain.
- **EFS recovery Agent:** Used to issue certificates to designated recovery agents who can decrypt encrypted files within the Encrypting File System (EFS). This template ensures that if a user loses access to their encryption keys, the recovery agent can restore access to encrypted data
- **Enrollment Agent:** Used to issue certificates to individuals or devices authorized to request and enroll certificates on behalf of other users. This template enables trusted entities, such as administrators, to assist in the certificate enrollment process, typically for users or devices that cannot directly request certificates themselves.
- **IPSec:** Used to issue certificates that enable secure communication over IP networks by supporting IPSec protocols. These certificates authenticate and encrypt data between devices, ensuring secure, encrypted communication for network traffic, such as virtual private networks (VPNs).

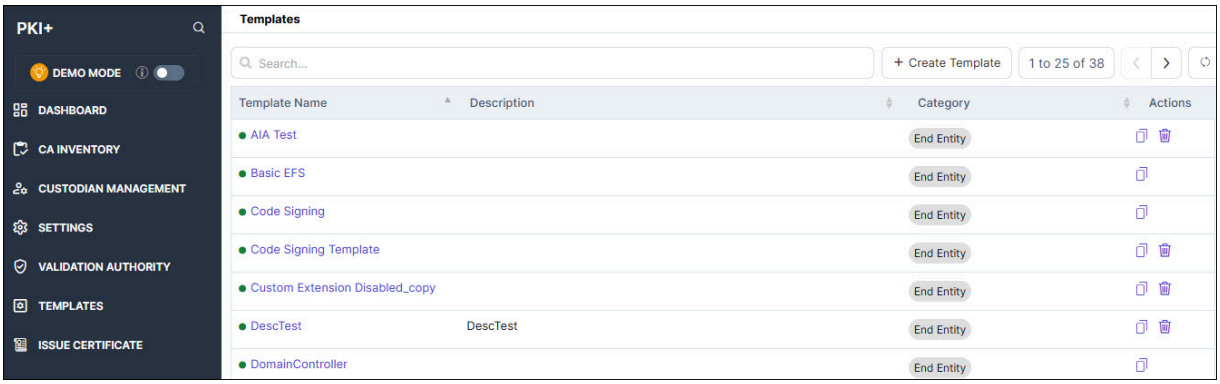
- **Kerberos Authentication:** Used to support the Kerberos authentication protocol in a networked environment. It allows for secure identity verification by issuing certificates that can be used by clients and servers to prove their identity within a Kerberos-secured domain.
- **OCSP Signing:** Used to issue certificates for Online Certificate Status Protocol (OCSP) responders, which are responsible for providing real-time status checks of digital certificates. This template ensures the authenticity and integrity of the OCSP responses, allowing clients to verify whether a certificate is revoked or valid.
- **RootCA_Default:** Used for the issuance of certificates to a Root Certificate Authority (CA) in a public key infrastructure. This template defines the settings and policies for the Root CA's certificate, which serves as the foundation of trust for all other certificates issued within the PKI hierarchy.
- **Router:** Used to issue certificates to network routers, enabling secure communication and authentication between routers in a network. These certificates help ensure that data transmitted across the network is encrypted and that the routers are trusted entities within the infrastructure.
- **Smart Card Logon:** Used to issue certificates that enable secure authentication via smart cards. This template allows users to log into a system by using a smart card, ensuring strong, two-factor authentication for enhanced security in accessing networks and applications.
- **SubCA_Default:** Used to issue certificates for subordinate Certificate Authorities (CAs) within a PKI hierarchy. This template defines the settings and policies for subordinate CAs, enabling them to issue certificates while maintaining the trust chain established by the Root CA.
- **User:** Used for generating and managing user certificates. It specifies the type of certificate, key usage, and other policies that should be applied to users, streamlining the certificate issuance process and ensuring consistency across users within an organization.
- **WebServer:** Used to secure web servers with SSL/TLS, ensuring encrypted communication, trust, and identity verification for web traffic, typically supporting HTTPS.
- **Workstation Authentication:** Used to issue certificates for computers or workstations, enabling secure authentication when they connect to a network. These certificates help verify the identity of the workstation, ensuring secure communication and access control within an organization's infrastructure.

To create new templates:

1. Go to (**Menu**) icon **> PKI+ > Templates.**

   The **Templates** page is displayed with pre-existing templates to choose from.

2. Select a template that best suits your needs and click the **(Copy)** icon in the **Action** column to create a copy of the selected template.



A copy of the selected template is displayed.

3. Edit the fields and click **Save**.

   The newly created template appears on the home page of **Templates** with a green dot beside it to indicate it is active.

Note:

- You can only delete the templates that you created. To delete a template, click the **Delete** icon against the selected template. Once deleted, the template will appear on the last page of the Templates page with a red dot beside it as shown.

  Deleted templates cannot be used for creating CAs or issuing certificates.

  You can create a copy of the deleted template.

- All changes or errors related to templates are logged and can be viewed under **Platform > Observe & Explore > Logs**.

Creating Custom Templates

You can create **custom templates** for short-lived and long-lived certificates using the **AppViewX PKIaaS Native CA** offering a wide range of benefits, including enhanced security, consistency, scalability, and ease of use. By aligning the certificate issuance process with your organization's specific requirements, you can optimize the management of digital certificates and strengthen your overall PKI environment. Custom templates help ensure compliance, reduce errors, and streamline the certificate lifecycle, making

the process more efficient and secure for your organization. You can either use any of the existing templates or create a customized template to specify certificate parameters.

To create custom templates:

1. Click **+ Create Template** on the top right corner of the screen.

   The **Templates** page is displayed.

2. Enter the following information:

Field Description of Templates Section

Field	Description
General	
***Template Name**	Provide a name for easy reference. **Note:** This field can no longer be edited once a certificate is issued using this template.
Description	Provide particulars on template creation as to who created it, when it was created, and why it was created.
Category	Select any of the options: • Root CA • Sub CA • End Entity (default value) **Note:** This field can no longer be edited once a certificate is issued using this template.
Validity Offset	This is the value provided to adjust the start date of certificate validity. By default, it is -10 minutes from the current time. Maximum is 24 hours.

Field	Description
Override Validity	This field is available only on selection of **End Entity** category. By default, this option is selected to use the validity from the enrollment request. On unselecting this option, you can see: • **Duration:** Use this option to set the validity duration (hours, days, months, years) for the certificates issued through this template. By default, the value is 1 month. • **Valid Until:** Use this option to set a pre-defined validity end date for the certificates issued through this template.
noRevAvail	This field is available only on selection of **End Entity** category. It is enabled automatically for short-lived certificates when the Override Validity duration is set in hours or days. On selecting this option, no revocation information (CRL and OCSP) is published for the certificates. This is typically used for short-lived and long-lived certificates as outlined in RFC 9608. The extension can be enabled to achieve the IoT use cases. The CA validity restrictions related to the certificate expiration and revocation are also relaxed, allowing the certificate to be issued even if the validity period is longer than the typical CA restrictions. **Note:** This field can no longer be edited once a certificate is issued using this template.
Override Subject Details in the CSR	By default, this check box is selected. Unselect the check box to consider the subject details from the CSR.
Override SAN details in the CSR	By default, this check box is selected. Unselect the check box to consider the SAN details from the CSR.
Basic Constraint Details	
Override	Unselect the check box to consider the extension provided in the CSR.
Critical	Select this option to indicate the information in an extension is important.
Key Usages	
Override	Unselect the check box to consider the extension provided in the CSR.

Field	Description
Critical	Select this option to indicate the information in an extension is important.
***Basekey Usage**	Select a value from the dropdown list that defines the functional purpose of the certificate. **Note:** This field can no longer be edited once a certificate is issued using this template.
Extended Key Usages	
Override	Unselect the check box to consider the extension provided in the CSR.
Critical	Select this option to indicate the information in an extension is important.
Extended Key Usage	Select a value from the dropdown list that defines the application usage of the certificate. **Note:** This field can no longer be edited once a certificate is issued using this template.
Enable Custom	Select this option to provide custom EKU values in the text box below. Multiple entries must be separated by a comma.
Custom Extensions	
Enable Custom Extensions	Based on your organization needs, you can add more custom extensions to a certificate using a Base64-encoded ASN.1 that will be included in every certificate issued using this template.

Field	Description
OID	OIDs are used to define specific certificate policies in a Certification Practice Statement (CPS). Each policy has a unique OID. An **OID** (Object Identifier) is a globally unique identifier used to represent specific objects,

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>attributes, or policies in systems like PKI, LDAP, SNMP, and more. It follows a dot-separated numeric format that uniquely identifies each object in a hierarchical structure.</td></tr>
<tr><td>Is Base64 Input Value</td><td>By default, this check box is not selected. Selecting this check box allows you to input your custom Base64 value in the correct format (Base64 encoded ASN.1 sequence) else it will throw an error message.

After you add and validate the custom extension, the system will automatically include it in the certificate when it is issued.</td></tr>
<tr><td>Encoding Type</td><td>Specifies the format in which the custom extension date is encoded. Select a value from the dropdown list based on your data requirements and usage context.</td></tr>
<tr><td>Value</td><td>Provide the field value. You can give any value for the provided custom OID.</td></tr>
<tr><td>Critical</td><td>Select this option to indicate the information in an extension is important.</td></tr>
<tr><td>Override</td><td>Unselect the check box to consider the extension provided in the CSR.</td></tr>
</table>
On clicking Add, the data is populated in a table.
</td></tr>
<tr><td colspan="2">Certificate Policy</td></tr>
<tr><td>Override</td><td>Unselect the check box to consider the extension provided in the CSR.</td></tr>
<tr><td>Enable Certificate Policy</td><td>Certificate Policy specifies the policy under which a certificate was issued. On enabling it, the following fields are displayed.</td></tr>
</table>

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Inherited from CA</td><td>Select this option to indicate it was inherited from CA.</td></tr>
<tr><td>Critical</td><td>Select this option to indicate the information in an extension is important.</td></tr>
<tr><td>Additional Policies</td><td>Enable this option if you want to create custom policies as CPS URI or User Notice Text. Provide OID, type, and value. Click Add for the data to be populated in a table.</td></tr>
</table>
</td></tr>
<tr><td colspan="2">Subject Alternative Names</td></tr>
<tr><td>Critical</td><td>Select this option to indicate the information in an extension is important.</td></tr>
<tr><td>Field Name</td><td>Select value as DNSName, IPAddress, Email, or URI.</td></tr>
<tr><td>Encoding Type</td><td>Select a value from the dropdown list.</td></tr>
<tr><td colspan="2">Other Extensions</td></tr>
<tr><td>Authority Key ID</td><td>Sha1 hash of the issuer public key. By default, this is enabled. Disabling this field can cause issues with certificate validation and trust chains.

Note:
This field can no longer be edited once a certificate is issued using this template.</td></tr>
<tr><td>Subject Key ID</td><td>Sha1 hash of the subject public key. Select hash value as 60 or 160 bit. By default, this is enabled. Disabling this field can cause issues with certificate validation and trust chains.

Note:
This field can no longer be edited once a certificate is issued using this template.</td></tr>
</table>

Field	Description
Validation URLs Override	By default, this is enabled. Disable it to consider the extension provided in the CSR tempate.
Enable CRLDP	Enable to add CRLDP to the certificate for status verification. By default, this is enabled only for sub CA and end entity categories in case of the free shipped templates. Disabling this field may impact the ability to check the certificate revocation status. **Note:** This field can no longer be edited once a certificate is issued using this template.
CA defined CRL Distribution Point	This field appears only for sub CA and end entity categories. This is selected when **Enable CRLDP** is enabled. **Note:** This field can no longer be edited once a certificate is issued using this template.
Custom CRL Distribution Point URI	This field appears only when **Enable CRLDP** is enabled. Provide custom CRL URLs in the text box below. Multiple entries must be separated by a comma.
Enable AIA	By default, this is enabled for sub CA and end entity categories. Disabling this field may prevent online status checks to confirm the certificate validity. **Note:** This field can no longer be edited once a certificate is issued using this template.
Issuer Certificate download link	This field appears enabled only for sub CA and end entity categories. You can disable this option to remove the issuer certificate link from the certificates issued using this link.

Field	Description
CA defined OCSP link	Enable this for issuer defined OCSP. **Note:** This field can no longer be edited once a certificate is issued using this template.
Custom OCSP URI	Enable this for user defined OCSP. Select this option to provide custom OCSP URLs in the text box below. Multiple entries must be separated by a comma.

3. Click **Save**.

The newly created template appears on the home page of **Templates** with a green dot beside it to indicate it is active.

Note:

- You can only delete the templates that you created. To delete a template, click the **Delete** icon against the selected template. Once deleted, the template will appear on the last page of the Templates page with a red dot beside it as shown.

  Deleted templates cannot be used for creating CAs or issuing certificates.

  You can create a copy of the deleted template.

- All changes or errors related to templates are logged and can be viewed under **Platform > Observe & Explore > Logs**.

Issue Certificates

Note:

- This module is available starting from the Thames HF2 (2024.0.2.0) release for those using AppViewX PKIaaS Native CA for PKI initialization.
- For versions prior to Thames FP1 HF3, enable **Issue Certificate** function by going to **(Menu)** icon **> Platform > Role.** Search for the created administrator role and click the link. Switch to the **Authorized functions** tab, and select the **Issue Certificate** check box in the PKI module.

Once you have created a template, you can issue certificates by generating a **Certificate Authority (CA)** and signing a **digital certificate** for an entity (such as a user, device, server, or application). This certificate serves as proof of identity and facilitates secure communication, authentication, and encryption in a PKI-enabled system.

To issue certificates:

1. Go to (**Menu**) icon **> PKI+ > Issue Certificate.**

   The **Issue Certificate** page is displayed.

2. Enter the following information:

Field Description for Issue Certificate section

Field	Description
***CA Name**	Select the CA name from the dropdown list.
Certificate Type	Select the certificate type as end certificate or CA certificate. By default, end certificate is selected.
***Template**	Select a template from the dropdown list.
***Validity**	Select the certificate validity in years, months, or days.
***Upload CSR**	Browse and upload the CSR.
***Certificate Download Format**	Select the format for the certificate to be downloaded.

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

3. Click **Issue Certificate.**

The certificate will be issued with the selected parameters.

PKI Standard Practices

Overview

This section outlines some of the PKI standard practices.

Offline Root CA

- The root CA should never be connected to the network or to the domain and no fingerprint of the server should ever be recorded since the root key compromise will impact the entire PKI hierarchy.
- Root CAs should always stay offline and shut down except when signing the Issuing CA certificates and during root CRL publish.
- Access to the Root CA to sign the Issuing CA request should be initiated in an agreed and controlled workflow so as to not compromise the Root CA in any means.
- Once the Issuing CA certificate has been issued and Root CRL published the Root CA should be turned off.

- Ensure to publish a reasonably short-lived Root CA CRL, the recommendations from NIST is to have the Root CA CRL published for 1 year and ensure to renew the CRL before expiry.
- We strongly recommend that all your CA keys be stored securely in a FIPS 140-2 Hardware Security Module (HSM).
- Protect the server during boot using Bitlocker or any other encryption system of choice and ensure to backup CA private key, CA registry Key, the CA database, and the CA certificate.
- Ensure to enable an audit event to track all actions performed on the Root CA.

Inline with Compliance

- Ensure to have a CP and CPS created to suit the organization's needs and ensure the PKI infrastructure meets all standards and requirements with respect to the CP and CPS.
- Any changes or addition of features ensure to capture in the CP and CPS documents.
- Ensure to renew the CA certificates (root and subordinate) within half its lifecycle.
- Enterprise key and certificate security policies should align with the latest regulatory, industry-standard recommendations, and guidelines such as key storage, secure communication protocols (TLSv1.2), cryptographic algorithms (RSA-2048), and hashing algorithms (SHA-2).
- Enterprise security architects should constantly monitor security standard recommendations and periodically update the enterprise's security policy.
- Ensure all security events are audited and a periodic security audit is performed to validate the security adherences and metrics.
- Encourage short-lived certificates for all key usages.

CSR Generation Standardization

- A process must be defined across the enterprise to generate CSR that aligns with the security standards and to store keys securely.
- Harden parameters such as Country and Organization in accordance with organizational requirements.
- Access to keys should be restricted to authorized personnel.
- Key Generation, Certificate Request, and Approval processes should be well defined.
- Archival

Archival

Signing keys do not require archival. We can always generate new keys for signing since the signed data is not encrypted. But encryption keys have to be archived so that the encrypted files during the certificate validity can be decrypted even after the certificate expiry. Also, this is recommended for security audits.

Secure Storage of Keys

- It is recommended to store private keys in HSM.
- Ensure respective certificate owners or certificate authorized administrators are granted access to private keys using the RBAC solution.
- Best practices training can be provided to certificate users and administrators to keep private keys secure.

Compromised CA/CA keys

- Ensure to discover a compromise as quickly as possible by implementing tracking and detection mechanisms and performing regular manual operational sanity checks.
- Establish well-defined communications plans for informing subjects, relying parties, and other stakeholders with sufficient details about the type of compromise so these parties can implement the appropriate remedial actions.
- If a CA system or signing key compromise occurs, the organization should perform the following steps:
  - Ensure that certificates issued to the organization’s systems or users from the compromised CA are revoked.
  - Notify all owners of the affected certificates about the CA compromise and establish a point of contact for responding to questions and providing guidance and instructions.
  - Replace all certificates from the compromised CA with new certificates from a different CA effective immediately.
  - Ensure that all relying parties have the certificate trust chains required to validate certificates from the new CA.
  - Ensure that revocation checking is enabled on all relying party systems.
  - If the compromised CA is a root CA, the root certificate must be removed from all trust stores and relying on party systems.

Compromised Certificate Handling

- Ensure to respond in a timely manner in case of a CA or end-entity certificate compromise and have a plan or workflow to replace all affected certificates or the trust chain.
- In the event of a key or certificate compromise, a fresh key pair should be generated on a secured system. The compromised item should be revoked and taken out of the service as soon as the systems are secured.
- If you are not sure of your private key possession, report it to your CA and suspend the key immediately. Once you find the key is secure, reinstate the certificate.

CA Compromise and Remediation Matrix

Issue Type	Revoke compromised/ counterfeit certificates	Revoke CA certificate	Replace all certs issued	Remove/ Revoke Root certificate
Impersonation	Yes	NA	NA	NA
RA compromise	Yes	NA	NA	NA
CA system compromise	NA	Yes	Yes	NA
CA key compromise	NA	Yes	Yes	NA
Root CA compromise	NA	NA	Yes	Yes

Managing Certificates

Short-Lived Certificates

Short-lived certificate refers to an SSL/TLS certificate that is issued with a very short validity period, typically ranging from a few days to a few months. These typically reduce the risk associated with certificates that might be compromised or misused over time. By limiting their validity, the attack surface is minimized because certificates are rotated more frequently.

Benefits of Short-Lived Certificates

- **Improved Security**: Short-lived certificates lower the impact of a potential compromise since certificates are valid only for a short period.
- **Encourages Automation**: With shorter validity periods, the use of automated tools (like **ACME protocol** for certificate management and many more MDM tools) becomes more common. This encourages the automation of certificate renewal, which reduces human errors and increases operational efficiency.
- **Faster Revocation**: If a certificate is compromised, revocation becomes more effective because the certificate will expire quickly anyway.

Long-Lived Certificates

Certificate Group

Prerequisites

Before starting **Certificate Groups** configuration:

- **Certificate Groups** are used to categorize the certificates according to various **business units.**
- In some organizations, **Certificate Groups** are also used to assign access permissions. Only privileged users (inherited from **Resource > User Group)** can view the respective **Certificate Groups**.
- Users should be assigned to a **Role** (inherited from Role > User Group) that has access to perform the below actions:
  - View a group
  - Assign a group
  - Unassign a group
- With these actions, users can assign a group during **Certificate Discovery** to avoid movement of certificates post-discovery.

- Along with the view, assign, and unassign options, administrators should be assigned to a **role** that has access to additional actions,
  - Create/ modify a group
  - Delete a group
  - Edit default group

Adding Certificate Group

To create a certificate group:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **Groups** from **Groups & Policies** on the LHS pane.
3. Click **+ Create.**

   The **Create Group** page is displayed.
4. In the **Group Details** section, enter the following details:

Field Description for Group Details section

Field	Description
***Select Group Hierarchy**	From the list of group hierarchies, select the parent group of the new group.
***Group Name**	Enter a unique name.
Application ID	Enter an ID specific to your organization.
Description	Enter detailed information regarding the group stating the purpose.

Note:

Fields marked with red asterisk (*) symbol are mandatory.

5. In the **Other Details** section, provide the following details about the certificate group:

Field Description for Other Details section

Field	Description
Contact Name	Enter the name of the person to be contacted in case of any changes.
Line of Business Name	Enter the name of the business unit.
Email	Enter the email address of the contact person.
Environment Name	Enter the name of the environment.
Phone Number	Enter the phone number of the contact person.
Inventory Number	Enter the number related to the inventory.
Cost Center/ Hierarchy	Enter the cost center code/ label.
Push Certificate Automatically	To associate the certificate automatically with its device, select the **Push Certificate Automatically** check box.
Renew Automatically	To enable automatic renewal of the certificates under this group, turn on the **Renew Automatically** toggle. **Note:** If you enable the automatic renewal, two more details have to be entered: • **Start Renewing:** Enter a number between 1 to 90 to denote the number of days. The system will renew the certificate before expiry. • **Approval required:** To enable the requirement for approval, select this check box. **Warning:** If you change the group associated with the certificate, the number of renewal days will be overwritten as per the new group.
Associated Policy	From the list of CA policies, select the required **Associated Policy.**

6. Click **Create** to add the certificate group to the system.

> **Note:**
>
> You can search for the required group and add the frequently used keywords as favorites. You can also create a certificate group for Server, Client, and Device certificates by clicking the **Group** () icon from the respective tabs under **Certificate Inventory.**

Editing Certificate Group

To modify a certificate group:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **Groups** from **Groups & Policies** on the LHS pane.

   The group inventory page appears.
3. Click the name of the certificate group you want to edit.
4. On the Modify screen that appears, make whatever changes you want to the content.
5. Click **Update** to save your edits.

Delete Certificate Group

To delete a certificate group:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **Groups** from **Groups & Policies** on the LHS pane.

   The group inventory page appears.
3. Select the group you want to delete and click **Delete.**
   A **Confirmation** popup window appears.
4. Click **Yes**.
   The group is deleted from the inventory.

Assigning or Unassigning Group to Certificate

To assign a group to a certificate from within the Inventory module:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. From **Certificate Inventory,** click **Common Name** of the certificate whose CSR you want to download and click **Assign Group.**

   -OR-

   On the certificate list, select the check box beside the certificate that you want to assign a group to. Click **Actions** and select the **Assign Group** option from the dropdown.

   The **Assign/Unassign Certificates** screen appears.
3. Select the group you want to assign to the certificate.
4. Click **Assign**.

> **Note:**
> You can follow the same steps selecting **Unassign Group** to unassign. You cannot unassign a certificate from the Default group. If you unassign a certificate from the assigned group, it is assigned to the Default group.

Certificate Authority Policy

The CA policy defines rules and templates to ensure certificate attributes comply with the organization.

- For Standard Initialization
- For PKIaaS Native Initialization

For Standard Initialization

The CA policy defines rules and templates to ensure certificate attributes comply with the organization.

To create a CA policy:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.

2. Click **CA Policy** from **Groups & Policies** on the LHS pane.
3. Click **+ Create** in the command bar to configure certificate practice standards for the business unit.

   The **Policy Details** page is displayed.

4. Enter the details as described:

Field Description for Policy Details section

Field	Description
***Policy Name**	Enter a unique name for the certificate policy.
Description	Enter the policy information.
Policy Enforcement Type	Choose any of the options: • **Strict:** While adding or updating the Certificate Authority (CA) connector, values provided as part of the Certificate Signing Request (CSR) information should match the values provided in the policy. If the values do not match the policy, you cannot save the CA connector details. • **Suggestive:** While adding or updating the Certificate Authority (CA) connector, values provided as part of the Certificate Signing Request (CSR) information do not have to be an exact match to the values provided in the policy. You can modify the values provided, but the certificate is then considered to be non-compliant.
Certificate Requests Need Approval?	Enable proper control through appropriate approvals for various actions performed on the group of certificates to which this policy is applicable.
Enable Access to Private Key?	Enable the option to allow private keys of the certificates to be exported.
Enable certificate push-bind access for read-only user	Enable the option to allow certificate push, bind and rollback operations from the holistic view for the user who got only read permission on the certificate group.
Validate issuer and root certificate for compliance?	Enable the option to check if issuer and root of the certificate are compliant to the standard defined in the policy.
Email Address mandatory for Client Certificate	Enable the option to set email address as mandatory during the client certificate enrollment.

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

5. In the **CA details** section, enter the following information:

Field Description for CA details

Field	Description
***CA Accounts**	Select the CA account name configured during initialization.
Certificate Issuance From	Select Issuer Name.
***Issuer Location**	Select a location from the dropdown list.
***Issuer Name**	Select issuer name from the dropdown list. This field appears only on selecting **Issuer Name** in the **Certificate Issuance From** field.
***Validity**	Enter a value and press Enter.
***Bit Length-Key Type**	Select a value from the dropdown list.
***Hash Function**	Select a value from the dropdown list.

6. [Optional] **Certificate parameters** section can be used later to help distinguish between multiple policies within the system.

Field Description for Certificate parameters section

Field	Description
Restrict Wild Card Certificate	Enable this option to restrict wildcard certificates.
Host Name	Enter a host name. Host name must not start or end with a period (.).
Allowed Domain Names	Type a domain name and press **Enter.**
Common Name	The fully qualified domain name (FQDN) or common name that exactly matches your web browser.

Field	Description
Organization	The name of the organization requesting the certificate.
Organization Unit	The division of the organization requesting the certificate.
Locality	The location of the organization requesting the certificate.
State	The state in which the organization is located.
Country code	The country and the country code in which the organization is located.
Email	The email contact details of the person responsible for maintaining the certificate.
Subject Alternative Name	Any additional hostnames, such as alternative websites, IP addresses and so on that have to be protected with the single SSL certificates.

7. Click **Save CA Details.**
   The added CA account is displayed in the table. You can view the CA account details, edit, or delete the CA account using the options provided.
8. Under the **Group selection** section, select the group(s) you want to include in the policy or create a new group to which the policy must be assigned.

   Note:
   You can search for the required group and add the frequently used keywords as favorites.

   Based on your selection, there will be a compliance report created under the dashboard for the list of certificates along with its non-compliant parameters relevant to this policy.
9. Click **Create Policy.**

   Note:
   If you want to make any changes to the policy in the future, you can select the policy and make the respective changes. If you want to completely reset the policy data, click **Reset** beside the CA name on the right pane.

For PKIaaS Native Initialization

The CA policy defines rules and templates to ensure certificate attributes comply with the organization.

To create a CA policy:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **CA Policy** from **Groups & Policies** on the LHS pane.
3. Click **+ Create** in the command bar to configure certificate practice standards for business unit.

   The **Policy Details** page is displayed.
4. Enter the details as described:

Field Description for Policy Details section

Field	Description
***Policy Name**	Enter a unique name for the certificate policy.
Description	Enter the policy information.
Policy Enforcement Type	Choose any of the options: • **Strict:** While adding or updating the Certificate Authority (CA) connector, values provided as part of the Certificate Signing Request (CSR) information should match the values provided in the policy. If the values do not match the policy, you cannot save the CA connector details. • **Suggestive:** While adding or updating the Certificate Authority (CA) connector, values provided as part of the Certificate Signing Request (CSR) information do not have to be an exact match to the values provided in the policy. You can modify the values provided, but the certificate is then considered to be non-compliant.
Certificate Requests Need Approval?	Enable proper control through appropriate approvals for various actions performed on the group of certificates to which this policy is applicable.
Enable Access to Private Key?	Enable the option to allow private keys of the certificates to be exported.
Enable certificate push-bind access for read-only user	Enable the option to allow certificate push, bind and rollback operations from the holistic view for the user who got only read permission on the certificate group.
Validate issuer and root certificate for compliance?	Enable the option to check if issuer and root of the certificate are compliant to the standard defined in the policy.

Field	Description
Email Address mandatory for Client Certificate	Enable the option to set email address as mandatory during the client certificate enrollment.

Note:

Fields marked with red asterisk (*) symbol are mandatory.

5. In the **CA details** section, enter the following information:

Field Description for CA details

Field	Description
***CA Accounts**	Select a CA account from the dropdown list.
Certificate Issuance From	By default, Issuer Name is selected.
***Issuer Location**	Select a location from the dropdown list.
Template Name(s)	Select the templates from the dropdown list.
***Validity**	Enter a value and press Enter.
***Bit Length-Key Type**	Select a value from the dropdown list.
***Hash Function**	Select a value from the dropdown list.
***Key Versions**	Applicable only for Sphincs+.

6. [Optional] **Certificate parameters** section can be used later to help distinguish between multiple policies within the system.

Field Description for Certificate parameters section

Field	Description
Restrict Wild Card Certificate	Enable this option to restrict wild card certificates.
Host Name	Enter a host name. Host name must not start or end with a period (.).

Field	Description
Allowed Domain Names	Type a domain name and press **Enter.**
Common Name	The fully qualified domain name (FQDN) or common name that exactly matches your web browser.
Organization	The name of the organization requesting the certificate.
Organization Unit	The division of the organization requesting the certificate.
Locality	The location of the organization requesting the certificate.
State	The state in which the organization is located.
Country code	The country and the country code in which the organization is located.
Email	The email contact details of the person responsible for maintaining the certificate.
Subject Alternative Name	Any additional hostnames, such as alternative websites, IP addresses and so on that have to be protected with the single SSL certificates.

7. Click **Save CA Details.**
   The added CA account is displayed in the table. You can view the CA account details, edit, or delete the CA account using the options provided.
8. Under the **Group selection** section, select the group(s) you want to include in the policy or create a new group to which the policy must be assigned.

> **Note:**
> You can search for the required group and add the frequently used keywords as favorites.

9. Under the **Compliance check** section, you can turn on the **Perform Compliance Check** toggle button to check the compliance for the defined rules and certificates attributes of the inventoried certificates.
10. Click **Create Policy.**

> **Note:**
> If you want to make any changes to the policy in the future, you can select the policy and make the respective changes. If you want to completely reset the policy data, click **Reset** beside the CA name on the right pane.

Adding/Enrolling Certificate

To enroll a certificate:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **Enroll Certificate** from **Certificate Action** on the LHS pane.
3. Select **Server**, **Client,** or **Code Signing Certificate** depending on the type of certificate(s) you want to enroll.

   The **Enroll Certificate** page appears.
4. In the **General Information** section of the **Enroll Server Certificate** page, select the desired **Assign Group** from the dropdown list.
5. In the **CA Details** section, enter the details as follows:

Field Description for CA Details section

Field	Description
***Certificate Authority**	Select **AppViewX PKIaaS**.
***Regenerate Automatically**	Select the toggle button to On or Off. • When the toggle is enabled, the **Start Regenerating** option is enabled. • Enter the number of days to regenerate the certificate automatically before expiry.
***CA Account**	The account to which the enrollment request is submitted. By default, it is *pkidev.*
Certificate Profile	Select the profile from the dropdown list. While enrolling server certificate, you get the option of *OcspSigning* as well in the dropdown list. For more information, see **CERT+ > Administration > Certificate Profiles.**
***Issuer Location**	Select an issuer location from the dropdown list.
***Issuer Name**	Select an issuer name to issue the certificate from the dropdown list.
***Connector Name**	Enter the friendly name for Certificate Authority connector in this field, which will be displayed in the holistic view on saving this form. By default, it is *AppViewX PKIaaS CA connector.*
Description	Enter the description in this field.

Field	Description
	Note: You can enter a maximum of 2000 words in the field.
***CSR Generation**	Select the CSR generation option as required. • **AppViewX:** Private key and CSR are created in AppViewX based on CSR parameters given. • **Upload CSR:** Uploaded CSR is taken as a source to populate CSR parameters and submit to CA.

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

6. In the **CSR Parameters** section, enter the details as follows:

Field Description for CSR Parameters section

Field	Description
***Common Name**	The common name is one of the key values of the Certificate Signing Request (CSR) to be present on the certificate. For example, <appviewx>. **Note:** No special characters allowed except period(.), hyphen (-), and underscore (_).
Subject Alternative Name	Select the subject alternative subject name from the dropdown list. You can see the count of subject alternative names (SAN) available for a certificate in the CSR parameter section, inventory grid, and CA connector page.

Field	Description
	Note: • Multiple values must be separated by a comma. • The cumulative count SANs appears in the certificate property window from the holistic view.
Organization	The organization name is one of the CSR parameters to be present in the certificate. This field will be auto-filled and editable based on the configuration in the selected group's policy.
Organization Unit	The organization unit name is one of the CSR parameters to be present in the certificate. This field will be auto-filled and editable based on the configuration in the selected group's policy.
Locality	The locality name is one of the CSR parameters to be present in the certificate. This field will be auto-filled and editable based on the configuration in the selected group's policy.
State	The state name is one of the CSR parameters to be present on the certificate. This field will be auto-filled and editable based on the configuration in the selected group's policy.
Country	Country name is one of the CSR parameters to be present in the certificate. This field will be auto-filled and editable based on configuration. It must be a 2-letter country code (for example, US, and so on).
Email Address	The email contact details of the person responsible for maintaining the certificate. Enter a valid e-mail address.
***Validity**	Enter the number in this field and select the entered validity list to be in Days, Months, and Years from the dropdown lists controlled by the group's policy.
***Hash Function**	The Hash function with which the CSR has to be signed. Any information specific to any CA or vendor has to be covered in the Note section. This field will be auto-filled and editable based on the configuration in the selected group's policy.

Field	Description
***Key Type**	The key type is used while creating a private and public key pair. This field will be auto-filled and editable based on the configuration in the selected group's policy.
***Bit Length**	The bit length is used while creating a private and public key pair. This field will be auto-filled and editable based on the configuration in the selected group's policy.

7. In the **Attachments** section, there is an optional field where the user/admin wants to keep any relevant attachment for the certificate enrollment, such as an approval email.

> **Note:**
> During certificate actions, the user can upload and maintain the additional necessary documents.

The following table describes the options available in the attachments section.

Field Description for Attachments section

Field	Description
Name	Enter the alternate name for the document to be uploaded.
Comments	Enter the comments in this field. **Note:** You can enter a maximum of 2000 words in the field.
Upload File	Click to upload a file.

8. Other than the CSR fields, you can add organization-specific values along with CSR. These values will not be part of the certificate but will be available in the AppViewX inventory. For example: cost center. Inventory can be filtered based on these attributes as well. If the Certificate Attributes are added under **Administration > Certificate Attributes,** it is reflected in the enrolment page.
9. In the **Generic Fields** section, enter the **Device Name** and the **Application IP Address**.
10. In the **Vendor specific details** section, the **Certificate ID** is auto-populated based on the value entered in the **Common Name** field.

11. Click **Add.** Once the details are added, it will redirect you to the page where you can see the respective CSR and CA details added as a connector. This page is called holistic view and from here any action on the certificate can be performed including provisioning the certificate to a server.
12. Click the **Submit** button to trigger the request.
    Once the submit action is triggered, the Submit popup window appears. Add comments if needed, and then click **Yes.** If the approved option is enabled in CA Policy, the request goes to the Approve and Implementation stages.
13. Click **Approve**.
14. The **Approve** pop-up window appears. Click the **Schedule later** button if the workflow request has to be approved automatically in the future.
15. Enter the comments in the field.
16. Click **Yes**.
    Once approved, you can see the Implement option in a holistic view.
17. Click **Implement.**
18. The **Implement** pop-up window appears. Click the **Schedule later** button if the workflow request has to be implemented automatically in the future.
19. Enter the comments in the field.
20. Click **Yes**.

What to do next

CSR Submission to CA is in progress.

Once the CSR submission is successful, the request state will be changed to *Submit certificate - retrieval in progress state.*

If the enrollment request is compliant with conditions defined and auto-approval enabled in the targeted CA, the certificate is fetched in a few seconds.

If auto-approval is disabled in the targeted CA, the user has to be logged into CA and approve the request.

Once the certificate is issued successfully, the certificate is retrieved to AppViewX.

Uploading Key

To upload a certificate key for the CSRs and certificates generated outside AppViewX:

1. Go to  (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.

2. Select the type of certificate you want to upload key for from the **Certificate Inventory.**
3. In the list of certificates, click the common name of the certificate for which you want to upload a certificate key.
   The certificate topology appears.
4. Hover the mouse over  (**More**) icon on the server certificate and click **Upload Key.**
5. If the key you want to upload is password-protected, a popup screen appears asking you to enter the associated password.
6. Click **Submit.**
7. On the screen that pops up, navigate to the key you want to upload and click **Open.**

> **Note:**
> If the key you are trying to upload does not match the certificate, an error message that the *Certificate and key do not match* appears.

If everything is correct, the key is uploaded to the certificate.

Post-Enrollment Usage of Certificates

Once a requester obtains a digital certificate signed by a CA, they can install this certificate onto an endpoint, which becomes a trusted network entity (it is assumed that the third party possesses the CA's public key in order to do this – the root CAs of leading CAs are installed on all major browsers).

As part of the standard TLS handshake process, any third party that interacts with the certificate owner will proceed to review the validity of the issued certificate by decrypting the digital signature provided by the CA.

The third party contrasts the decrypted hash function against the hash obtained by hashing the digital certificate. A match indicates integrity of the certificate. The communicating third party can then retrieve the public key from the digital certificate and proceed to establish a secure encrypted connection.

Adding Application Connector to Certificate

To add an application connector to a certificate:

1. Go to  (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **Server** or **Client** from **Certificate Inventory.**

3. In the Certificate list view page, click the **Common Name** of a certificate to add an application connector.
4. In the Certificate topology page, click **Add connector** or click **Connector actions > +Add App Connector.**

   The **Add Connector** is displayed.

5. In the **General Information** screen:
   - Select the device type from the **Category** dropdown list.
   - Select the device vendor from the **Vendor** dropdown list.
   - In the **Connector Name** field, enter a name for the connector that is descriptive enough when viewed within the Certificate topology.
   - Enter a description for the connector. This description shows up when you hover the mouse over the connector within the Certificate topology.

   > **Note:**
   >
   > Applicable only for Citrix application type] The SNI-enabled virtual server option is displayed. When this check box is selected, the virtual servers whose SNI are enabled are listed. You can also enable SNI for the virtual server by selecting Enable SNI push for Certificate and Enable SNI in Virtual Server.

6. From the list of available devices, click **Add to List** () button beside each device you want to select.
7. In the **Certificate Details** section:
   - From the **Certificate Type** dropdown, click the type of certificate to be used with the connector.
   - From the **Certificate File Name** field, enter the name of the certificate. The file format of the selected certificate type is automatically displayed.
   - In the **Key File Name** field, enter a name for the key file.
   - Select the **Push Root and Intermediate Certificates** to be pushed to the device.
8. In the **Push Details** section:
   - In the **Script location** field, specify whether the **Pre - Push** script and **Post - Push** script file is in AppViewX or target device.
   - Enter the script location that must be executed before and after the push in the Pre – Push script and Post - Push script fields.
   - Select the **Overwrite** check box to overwrite existing certificates with the new certificate.
   - Select **Push automatically** check box to push certificates to the device automatically.

> **Note:**
>
> [Applicable for F5 application type] The Secure push checkbox is selected by default. This option encrypts certificates while pushing them to a device. You can uncheck this option if you have the necessary permissions.

9. Click **Save** to add the application connector to the certificate topology.

Pushing Certificate to Device

The push to device option allows you to push the certificate to the load balancer or server device and associate it to a profile, template, or virtual server.

If the **Push automatically** field is selected while adding application connectors to a new certificate, then the certificate is automatically pushed to the device when it is retrieved. In such cases, you need not complete the process manually.

Prerequisites

Prior to pushing the certificate to a device, ensure that you have necessary role-based access controls and workflow access pertaining to the template and request.

To push a certificate to a device:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Select **Push to Device** from **Certificate Action.**
   The **Server Certificate** page appears.
3. Search for the certificate in the inventory and click the **Common Name** of the certificate to view the holistic view.
4. Click **Push to device.**
5. In the **Confirmation** popup window, enter comments and click **OK.**
   A request ID and work order ID are generated automatically and the work order status is displayed beside the connector in the topological view.
6. Click **Approve.** The work order status displayed beside the connector updates to *Push-Review In Progress.*

   On the **Approve** screen that pops up:

- Click **Now** or **Schedule Later** button in the **Implement** field.
- If you select **Schedule Later**, set the date and time that you want the certificate push to occur.
- Enter comments and click **OK**.

7. Click **Implement.**
8. On the **Implement** screen that pops up:
   - Click **Now** or **Schedule Later** button in the **Implement** field.
   - If you select **Schedule Later**, set the date and time that you want the certificate push to occur.
   - Enter comments and click **OK**.
9. Click  (**Refresh**) at the top of the page until the topology updates.

   After the push action is completed, the status is updated to *Completed.*

   The topological view follows a color-coding scheme to identify certificate statuses.

Color Coding for Certificate Statuses

Color	Certificate Status
Green	Certificate is valid.
Red	Certificate has expired.
Gray	Certificate is new.
Blue	Certificate will expire in 90 days.
Yellow	Certificate will expire in 30 days.
Orange	Certificate will expire in 10 days.
Black	Certificate is revoked.

Auto-Enrollment Protocols

AppViewX CERT+ enables certificate auto-enrollment by automating all the steps involved, including CSR generation, domain ownership verification, certificate download, and provisioning, making the process efficient, scalable, and secure. AppViewX CERT+ supports all major auto-enrollment protocols including – ACME, EST, SCEP, CMP, WAEP, and Microsoft Intune. Automating certificate enrollment reduces human error, outages, and security compromises, while improving productivity.

Auto-enrollment protocols are standardized enrollment mechanisms accepted across a wide range of enterprise systems for device and application certificate enrollment. Systems leveraging Auto-enrollment protocols typically expect minimum to no admin intervention. Network devices such as routers-switches,

DevOps tools, and Enterprise Mobility Management platforms are typical examples of such systems.If the deployment mode is

- SaaS, deploying a cloud connector enables auto-enrollment.
- On-prem installations without cloud connectors, users should provide the AppViewX host information, which includes the IP address and port of the URL or endpoint. If their devices support auto-enrollment to a public URL, auto-enrollment is available as part of the tenant, and configuration details are provided in the documentation.

The cloud connector is advised for DMZ-based deployments or for enrollment through your cloud connector. This is especially useful in scenarios where endpoints cannot communicate with a public URL for auto-enrollment through a private channel, necessitating the use of the cloud connector.

- EST
- ACME
- SCEP
- MS Intune
- CMP
- WAEP

For more information, refer to the CERT Guide.

Service Catalogs

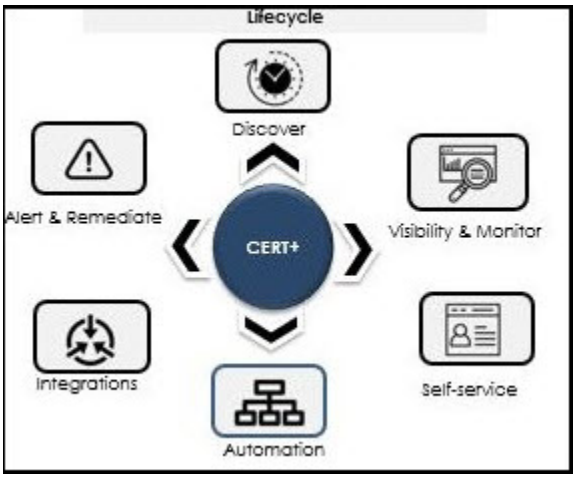
AppViewX's Page builder tool gives you step-by-step instructions on creating custom pages, incorporating various page elements like HTML pages, reports, workflow catalogs, forms, tables etc to suit user-specific requirements. or more informaton, refer to the Service Catalog User Guide.

Certificate Lifecycle Management

What is Certificate Lifecycle Management (CLM)?

AppViewX's CERT+ provides an end–to–end lifecycle management of x.509 digital certificates across complex networks to secure your business. With CERT+, security teams can manage the certificate lifecycle from an intuitive single-pane management Interface. It enables the Certificate Lifecycle Management and Automation solution which helps enterprise IT manage and automate the entire

lifecycle of their internal and external PKI. The key stages of the certificate lifecycle can be broken into the following stages:



- **Certificate Discovery & Inventory Management**: Allows users to discover certificates across the network and manage inventory of all certificates in one place.
- **Visibility and Monitoring**: Enables the user to monitor certificate expiry and usage. The monitored data is represented as a detailed report on the web portal along with options to trigger email alerts. Allows users to gain insights into certificates; monitor and take remedial action.
- **Certificate Enrollment**: Allows users to request certificates from a certificate authority (CA) that confirms their identity and generates a certificate.
- **Certificate Renewal**: Allows users to either manually or automatically renew a certificate before the expiry date by retaining the old private key.
- **Certificate Regeneration**: Allows users to enroll new certificates with similar parameters to an old certificate. When a user generates a new private key, the user can modify the parameters if required.
- **Certificate Revocation**: Allows users to revoke a certificate in the event of certificate loss, compromise, or any other reason when the certificate is no longer necessary for business.
- **Certificate Audit**: Track and audit the usage, creation, expiration, and revocation of certificates. Track user interaction with the platform.

Inventoried Certificate Actions

Important:

Configure policy first before performing any of the certificate actions.

The following actions can be performed on certificates:

Downloading Certificate

Note:

This functionality is available only for server, client, device, code signing, intermediate, and root certificates.

You can download a certificate from the Certificate page and the topology page within AppViewX.

Download from Certificate Inventory

To download a certificate as a .PEM file that is designed to be safe for inclusion in ASCII or rich-text documents such as emails:

1. Go to (**Menu**) icon **> CERT+.**

   The **CERT+** left navigation pane appears.
2. Click **Download** from the **Certificate Inventory** after selecting the type of certificate you want to download.
3. Switch to the **List** toggle button on the top right corner of the certificate page.
4. Select the check box for the certificate that you want to export.

> **Note:**
>
> Client certificates cannot be downloaded directly from the Certificate page; they can only be downloaded from the certificate topology screen. For more details, see the Section, *Download from Certificate Topology*.

5. Click **Actions,** and select **Download Certificates.**
6. In the **Download Certificate** popup window, select **Certificates Only.**
7. You can also enable/disable the **Download Trust Store Certificates** option.

> **Note:**
>
> If you have permission to view the restricted content mentioned in Step 6, the certificate details are then downloaded inside a zip file. If you do not have the necessary permissions, the system creates and downloads an empty zip file to the destination you specify.

8. Click **Download.**
9. To view details of the certificate, unzip the file, and open the security certificate file. Click **Details.**

Download from the Certificate Topology

1. Go to (**Menu**) icon **> CERT+.**

   The **CERT+** left navigation pane appears.
2. Click **Download** from the **Certificate Inventory** after selecting the type of certificate you want to download.
3. Switch to the **List** toggle button on the top right corner of the certificate page.
4. From the **Common Name** certificate list, select the certificate that you want to download.
5. Hover the mouse over on the certificate and click **Download Certificate.**
6. In the **Download certificate** pop-up window, select the file format.
   - For PEM and DER certificate types, you can enable/disable the **Download Trust Store Certificates** option along with the end certificates.
   - For PEM and DER certificate types, you can enable/disable the **Download Trust Store Certificates** option along with the end certificates.
7. Click **Yes.**

Uploading Certificate

To upload a certificate:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **Upload** from **Certificate Inventory.**

   The **Upload Certificate** screen is displayed.
3. Select the **Certificate Group** into which the uploaded file must be mapped in CLM.
4. Choose the certificate file and click **Open.**
5. Click **Upload.**
   Once uploaded, go to the selected certificate group in inventory to see the uploaded certificate-keys.

Exporting Certificate

You can export all the certificates in the inventory or select only specific certificates and export. You export certificate details in the form of columns and values. The output can be exported in <.xls> or <.csv> format. This can be used for reporting or making another inventory.

To export the server certificate:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click the **Certificate Inventory** and select the type of certificate you want to export.
   The **Certificate** screen is displayed.
3. Switch to the **List** toggle button on the top right corner of the certificate page.
4. In the **Common Name** column certificate list, select the check box against the certificate that you want to export certificate to.
5. Click **Actions**, and then select **Export Certificates** from the list.
   The **Export** popup window appears.
6. Select the desired **Options** and **Format** in the **Export** pop-up window.
   The selected certificate is exported to your local machine.

Renewing Certificate

Note:

Only certificates having CSR/private keys can be renewed. Click **Renew Certificate** to renew certificates with existing keys; click **Regenerate Certificate** to renew certificates with new keys. Enable **Renew Automatically** to avoid doing it manually. It is recommended to renew certificates with new keys.

From Holistic View

To renew a certificate from the holistic view:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **Renew Certificate** from **Certificate Action.**
3. Click **Server, Client,** or **Process Explorer** depending on the type of certificate you want to renew.
4. Switch to the **List** toggle button on the top right corner of the page.
5. In the **Common Name** column certificate list, select the certificate that you want to renew.
6. Hover the mouse over (**More**) icon and click **Renew.**
   You are redirected to the **Certificate** page.
7. In the **Vendor Specific Details** section, enter a new **Certificate ID** and click **Renew.**
   In the Renew popup window, enter comments and click Yes. A request ID and work order ID are then generated automatically and the work order status is displayed beside the certificate in the topological view. The work order status displayed beside the connector updates to *Renew Certificate renewal request In Progress.*
8. Click **Approve.**
9.
10. On the **Approve** screen that pops up:
    - Click **Now** or **Schedule Later** button in the **Implement** field.
    - If you select **Schedule Later**, set the date and time that you want the certificate push to occur.
    - Enter comments and click **OK**.

    The work order status displayed beside the connector updates to *Push-Review In Progress.*
11. Click **Implement.**
12. On the **Implement** screen that pops up:

- Click **Now** or **Schedule Later** button in the **Implement** field.
- If you select **Schedule Later**, set the date and time that you want the certificate push to occur.
- Enter comments and click **OK**.

13. Click  (**Refresh**) icon on the top of the page until the topology updates.
    After the renewal action is completed, the status is updated to *Completed.*
14. On the **Renew Certificate** popup window, select the type of certificate renewal as **Now** or **Set auto-renew**.
15. Select **Submit**.
    The status of the trigger can now be monitored under process explorer.

> **Note:**
>
> Alternatively, you can go to **Certificate Inventory** and select the check box against the certificate name you want to renew and click **Actions > Renew Certificate** from the command bar.

Regenerating Certificate

> **Note:**
>
> The regenerate option allows you to create a new certificate with a new key and with similar parameters to an existing certificate so that you can host it on a different type of web or application.

To regenerate a certificate:

1. Go to  (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Switch to the **List** toggle button on the top right corner of the page.
3. In the **Common Name** column certificate list, select the certificate that you want to regenerate.
   The Certificate page is displayed.
4. Hover the mouse over  (**More**) icon on the certificate, and click **Regenerate.**
5. In the **Vendor Specific Details** section, enter a new **Certificate ID** and click **Regenerate.**
6. Click **Approve**.
7. On the **Approve** screen that pops up:

- Click **Now** or **Schedule Later** button in the **Implement** field.
- If you select **Schedule Later**, set the date and time that you want the certificate push to occur.
- Enter comments and click **OK**.

8. Click **Implement.**
9. On the **Implement** screen that pops up:
   - Click **Now** or **Schedule Later** button in the **Manual Implementation** field to choose the mode of implementation.
   - If you select **Schedule Later**, set the date and time that you want the certificate implementation to occur.
   - Enter comments and click **Yes**.

   A request ID and work order ID are generated automatically. The work order status is displayed beside the certificate on the topological view.
10. Click (**Refresh**). The work order status is displayed beside the certificate.
    After the regenerating action is completed, the status is updated to *Completed.*

Revoking Certificate

If you have the necessary permission, you can submit a request to the issuer of a certificate to revoke it. As soon as the certificate is revoked, the certificate is no longer considered to be trusted. Revoked certificates are listed in the Certificate Revocation List (CRL) maintained by each certificate authority.

Note:

Revoke old certificates after renewing and provisioning new keys.

To revoke a certificate:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Switch to the **List** toggle button on the top right corner of the page.
3. In the **Common Name** column certificate list, select the certificate that you want to revoke.
4. Hover the mouse over (**More**) icon on the certificate, and click the **Revoke** option.
5. Select a reason for revoking the certificate.
6. Click **Yes.**
   A request ID and work order ID are generated automatically and the work order status is displayed beside the certificate on the topological view.

7. Click **Approve**.
8. On the **Approve** screen that pops up:
   - Click **Now** or **Schedule Later** button in the **Implement** field.
   - If you select **Schedule Later**, set the date and time that you want the certificate push to occur.
   - Enter comments and click **OK**.
9. Click **Implement.**
10. On the **Implement** screen that pops up:
    - Click **Now** or **Schedule Later** button in the **Implement** field.
    - If you select **Schedule Later**, set the date and time that you want the certificate push to occur.
    - Enter comments and click **OK**.
11. Click  (**Refresh**). The work order status is displayed beside the certificate.

> **Note:**
>
> Alternatively, you can go to **Certificate Inventory** and select the check box against the certificate name you want to revoke and click **Actions > Revoke Certificate** from the command bar.

After the regenerate action is completed, the status is updated to *Completed.*

• Performing Revocation Check

Performing Revocation Check

For CAs (both external and AppViewX), you can check the most recent status of the certificate even if it is moved to the inventory for the first time. This check is performed automatically twice a day and the user can check for the revoked certificates anytime.

To perform a revocation check:

1. Go to  (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **Server, Client, Device, or Code Signing** depending on the type of revoked certificates you want to view.
3. In the certificate list, select certificates for which you want to view the status.
4. Click **Actions,** and select **Revocation check** option from the dropdown.

   The **Revocation Check** dialog box appears.

5. Click **OK.**
   Once validated, the status certificate is updated in the color code of the **Common Name** column.

Generating CSR for Certificate

To generate a manual CSR for the certificate:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **Generate CSR** from **Certificate Action.**
3. Click **Server** or **Code Signing Certificate.**

   The **Generate CSR** page appears.
4. In the **Group details** section, select the **Assign Group** from the dropdown list where you want to assign a CSR to the desired group of certificates.

Field Description for Group details section

Field	Description
***CSR Selection**	Select an option.
***Common Name**	Common name is one of the key values of the Certificate Signing Request (CSR) to be present on the certificate. For example, <appviewx>. **Note:** No special characters are allowed except period (.), hyphen (-), and underscore (_).
Subject Alternative Name	Select the alternative subject name from the dropdown list. You can see the count of subject alternative names (SAN) available for a certificate in the CSR parameter section, inventory grid, and CA connector page. **Note:** • Multiple values must be separated by a comma. • The cumulative count SANs appears in the certificate property window from the holistic view.

Field	Description
Organization	The organization name is one of the CSR parameters to be present in the certificate. This field will be auto-filled and editable based on the configuration in the selected group's policy.
Organization Unit	Organization Unit name is one of the CSR parameters to be present in the certificate. This field will be auto-filled and editable based on the configuration in the selected group's policy.
Locality	The locality name is one of the CSR parameters to be present in the certificate. This field will be auto-filled and editable based on the configuration in the selected group's policy.
State	The state name is one of the CSR parameters to be present on the certificate. This field will be auto-filled and editable based on the configuration in the selected group's policy.
Country	Country name is one of the CSR parameters to be present in the certificate. This field will be auto-filled and editable based on configuration. It must be a 2-letter country code (for example, US, and so on).
Email Address	The email contact details of the person responsible for maintaining the certificate. Enter the valid e-mail address.
Challenge Password	The challenge password for the certificate. Enter if it is applicable. Password must contain at least one alphabet (uppercase and lowercase), one number, and one special character.
Confirm Password	The password to confirm the Challenge Password entered matches with the Challenge Password.
***Hash Function**	The hash function with which the CSR has to be signed. Any information specific to any CA or vendor has to be covered in the Note section. This field will be auto-filled and editable based on the configuration in the selected group's policy.
***Key Type**	The key type is used while creating a private and public key pair. This field will be auto-filled and editable based on the configuration in the selected group's policy.
***Bit Length**	The bit length is used while creating a private and public key pair. This field will be auto-filled and editable based on the configuration in the selected group's policy.

Note:

Fields marked with red asterisk (*) symbol are mandatory.

5. In the **Attachments** section, enter the details as follows:

Field Description for Attachments section

Field	Description
Name	Enter the alternate name for the document to be uploaded.
Comments	Enter the comments in this field. **Note:** You can enter a maximum of 2000 words in the field.
Upload File	Click to upload a file.

6. Click **Add** to generate the CSR and add it to the intended group.

Submitting CSR to Certificate Authority

After you have generated a CSR, you must submit it to the respective certificate authority (CA) for signing.

To submit CSR to CA:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. On the Certificate list view, locate the CSR you generated and click the Common Name of the certificate.

   The certificate topology screen opens.
3. Add a CA connector to the certificate topology as explained in the Section, Add Certificate Authority Connector to Certificate.
4. Click **Submit** to trigger the request.

Once the submit action is triggered, the Submit popup window appears. Add comments if needed, and then click **Yes.** If the approval required option is enabled in CA Policy, the request goes to Approve and Implementation stages.

5. Click **Approve**.
6. Click the **Schedule later** button if the workflow request has to be approved automatically in the future.
7. Enter the comments in the field.
8. Click **Yes**.
   Once approved, you can see the Implement option in the holistic view.
9. Click **Implement.**

   The **Implement** pop-up window appears.

   - Click the **Schedule later** button if the workflow request has to be implemented automatically in the future.
10. Enter the comments in the field.
11. Click **Yes**.
    CSR Submission to CA is in progress.
12. Once the CSR submission is successful, the request state will be changed to **Submit** certificate - retrieval in progress state.

    If the enrollment request is compliant with conditions defined and auto-approval enabled in the targeted CA, the certificate is fetched in a few seconds.

    If auto-approval disabled in the targeted CA, the user has to be logged into CA and approve the request.

    Once the certificate is issued successfully, the certificate is retrieved into AppViewX.

Downloading CSR

To download a certificate signing request (CSR) for a certificate:

From holistic view:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. From **Certificate Inventory,** click **Server** or **Code Signing Certificate.**
3. On the certificate list view, click the **Common Name** of the certificate to view the topology.

4. Hover over  (**More**) icon on the certificate and click **Download CSR**.

> **Note:**
>
> Alternatively, you can go to **Certificate Inventory** and select the check box against the certificate name you want to download CSR and click **Actions > Download CSR** from the command bar.

Suspending Certificate

If you have the necessary permission, you can suspend a certificate. As soon as the certificate is suspended, it is revoked. The suspended certificates are listed on the Certificate Revocation List (CRL) maintained by each certificate authority.

To suspend a certificate:

1. Go to  (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Switch to the **List** toggle button on the top right corner of the page.
3. Click **Server, Client,** or **Device** tab depending on the type of certificate you want to suspend.
4. In the **Common Name** column certificate list, select the certificate that you want to suspend.
   The certificate topology appears on the screen.
5. In the **Comments** field, enter the reason for suspending the certificate.
6. Click **Yes.**

Changing Status of Certificate

Before changing the status of a certificate, the user should plan for the impact that might have on existing work orders.

To change the status of a certificate:

1. Go to  (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click **CA Switch** from **Certificate Action** and select the type of certificate for which you want to change status.

3. On the **Change Status** pop-up screen that appears, select **Managed** (to create, renew, or revoke actions on those certificates) or **Monitored** (to only alert) from the Change status to dropdown.
4. [Recommended] In the **Comments** field, enter the reason for changing the status.
5. Click **Yes.**

What to do next:

> **Note:**
>
> Alternatively, you can go to **Certificate Inventory** and select the check box against the certificate name you want to renew and click **Actions > Change Status** from the command bar.

Deleting Certificate

To delete a certificate or policy:

1. Go to (**Menu**) icon **> CERT+.**
   The CERT+ left navigation pane appears.
2. Click the type of certificate you want to delete from **Certificate Inventory** list.
3. From the certificates inventory, select the check box beside the certificate or policy you want to delete.
4. Click **Actions,** and select **Delete** from the dropdown list.

   > **Note:**
   >
   > This functionality is available only for server certificates and policy.

5. Click **Yes** to confirm.
   The certificate or policy is then removed from the list and deleted from the AppViewX system.

Revocation Check - OCSP

Certificate authorities use Online Certificate Status Protocol (OCSP) to obtain the revocation status of x.509 digital certificates. When a user requests the validity of a certificate, an OCSP request is sent to an OCSP server to check the specific certificate with a trusted certificate authority. The OCSP server then sends a *good, revoked, or unknown* response.

Prerequisites

- OCSP URL must be published in the AIA field of the certificate with the AppViewX OCSP server URL.
- **Plugins required:** OCSP Server and OCSP Generator must be deployed for OCSP to work.

You can then proceed to select one or more certificates from the inventory and click **Actions > Revocation Check** to perform revocation validation. Once validated, the certificate status is updated in the color code of the Common Name column.

Business Continuity and Key Security Mechanism

Backup and Recovery and Business Continuity

AppViewX Backup and Recovery and Business Continuity are key features designed to ensure that AppViewX PKI (Public Key Infrastructure) and other AppViewX services remain operational and secure, even in the event of system failures or disasters. These features help maintain the availability, integrity, and reliability of critical cryptographic and certificate management processes in on-premises environments.

Here’s an overview of Backup and Recovery and Business Continuity:

Key Aspects of AppViewX Backup and Recovery

1. Configuration Backup
   - AppViewX backs up all configuration settings, including user roles, policies, certificate templates, CA (Certificate Authority) configurations, and other operational settings.
   - This ensures that even if the system crashes or the configuration is lost, you can restore it quickly without manual reconfiguration.
2. Certificate & Key Store Backup
   - AppViewX can securely back up the certificate inventory and cryptographic keys (both public and private keys) used in the PKI infrastructure.
   - Backups ensure that certificates can be reissued, renewed, and redeployed without the risk of losing access to critical certificates or keys.
3. Database Backup
   - AppViewX relies on a centralized database to store critical data such as logs, audit trails, certificate information, and configuration details.
   - Regular database backups ensure that if there’s a database corruption or failure, the data can be restored to its most recent state without loss.
4. Automated and Manual Backups

- The system can be configured for automated backups at defined intervals (e.g., daily, weekly, monthly) to reduce the risk of data loss.
- Administrators can also trigger manual backups based on specific needs, such as before a major system update or change.

5. Offsite Backup and Disaster Recovery
   - Backup files can be stored offsite or replicated to secondary locations to ensure data availability in case of site-specific failures (e.g., a data center outage).
   - Offsite backups provide an additional layer of protection for disaster recovery.
6. Key Backup
   - Private keys used in the AppViewX-managed PKI (including internal CAs, certificates, etc.) can be backed up securely to ensure that even in the case of a failure, private keys remain recoverable. It ensures continuity in operations like signing certificates and key management.
7. Restoration Process
   - The system provides easy-to-follow procedures for restoring from backups, ensuring minimal downtime.
   - AppViewX's restoration tools enable quick recovery of the platform, including full system recovery or targeted recovery of specific data, configurations, or keys.
8. Granular Recovery Options
   - Users can choose to restore entire systems, specific configurations, or individual certificates and keys.
   - This granular control enables efficient recovery and reduces the time needed to get the system back to full functionality.

For more information, refer to these sections.

Key Security Mechanism

In AppViewX , **Key Security Mechanism** refers to the set of processes and technologies used to protect the cryptographic keys (especially private keys) that are central to the operation of a Public Key Infrastructure (PKI). These mechanisms ensure that keys, including private keys used by Certificate Authorities (CAs), servers, and end-user devices, are stored securely and managed in compliance with industry standards and best practices.

AppViewX PKI offers several key security mechanisms that help to safeguard sensitive cryptographic data, ensure the integrity of digital certificates, and maintain the confidentiality of key material.

AppViewX PKI is a powerful, flexible, and automated platform designed to streamline the management of PKI infrastructure, ensuring security, compliance, and operational efficiency across an organization's

certificate and key management ecosystem. It is ideal for large enterprises, cloud environments, and organizations that require seamless certificate management across diverse infrastructure components.

Reporting and Monitoring

Reporting and Monitoring

Once the certificates in the infrastructure are discovered in AppViewX, they can be monitored as the reports in the Dashboards. In the dashboards, the user can track the certificates expiry, compliance, security details as the reports in the dashboard.

Reporting and monitoring the certificates are essential for an administrator to get complete visibility of all the certificates across multiple vendors and data centers in one single window pane. Certificates have a finite life span and are set to expire at different dates and times. Due to advancements in cryptography, there are high chances that the infrastructure will carry the weaker algorithm certificates which will be vulnerable to several attacks which will cause business outages.

Using the dashboards and reports, the administrator can continuously monitor the status of the certificates in terms of expiry, security, compliance and so on.

Certificate Reporting

For more information, refer to the **Certificate Reporting** section in the CERT User Guide.

Dashboard Actions

This section explains how to create, export, import, and delete dashboards.

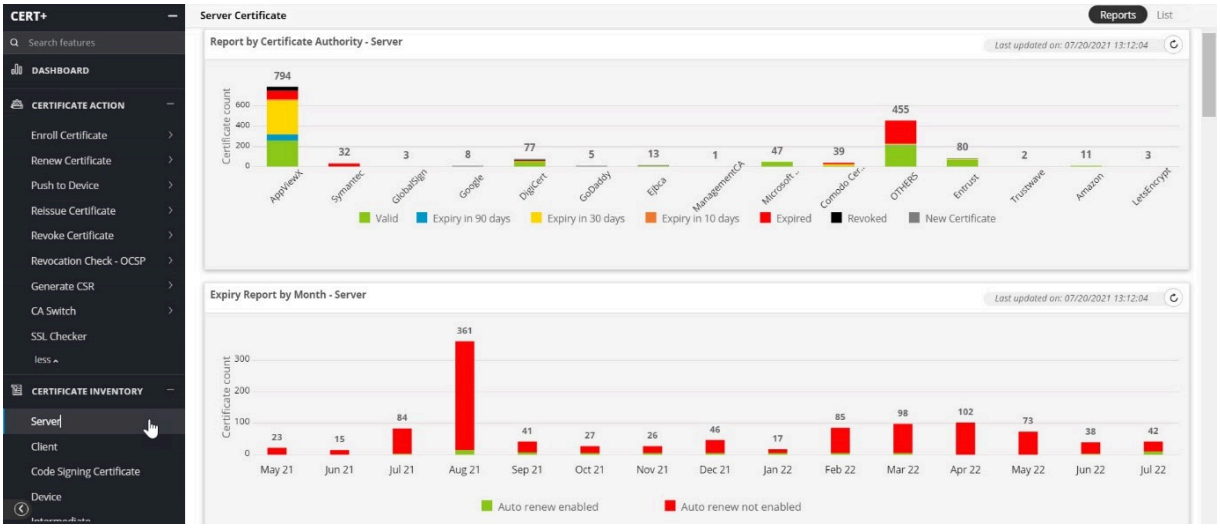
- Importing Dashboard
- Deleting Dashboard

Viewing Certificate Reports

To view certificate reports:

1. Click **Certificate Inventory** and click the type of certificate for which you want to view the report.

   The Reports page is selected.



   Although each certificate report displays the data differently, the same set of data is used to generate each report.

2. The following reports are segregated and displayed as widgets on the **Client Certificate** screen:
   - **Report by Certificate Authority:** A bar chart that shows the total certificate count for each Certificate Authority (CA), made up of colored bars representing the following statuses:
     - Green - Valid certificates
     - Blue - Certificates with an expiry in 90 days
     - Yellow - Certificates with expiry in 30 days
     - Orange - Certificates with expiry in 10 days
     - Red - Expired certificates
     - Black - Revoked certificates
     - Gray - New certificates
   - **Expiry Report by Month:** A bar chart that shows the total number of certificates expiring each month.

- **Policy Compliance:** A pie chart that shows the number of compliant and non-compliant certificates in the system, with each sector in the chart representing a different kind of policy such as Strict or Suggestive. You can also export the report details from the Policy Compliance Report widget.
- **Stale Certificate:** A pie chart that shows the number of expired and revoked certificates.
- **Certificate Summary:** A doughnut chart that categorizes the certificates based on expiration, with the total count of certificates made up of colored bars representing the same statuses listed for the Report by Certificate Authority widget. You can also configure the report settings from the Certificate Summary Report widget.
- **Count by Issuer:** A doughnut chart that shows the total number of certificates managed by the issuer such as Root CA or the Intermediate CA. You can also configure the report settings from the Count by Issuer widget.

Creating Dashboard

To create a dashboard:

1. Go to (**Menu**) icon **> CERT+**.

   The **CERT+** left navigation pane appears.

2. Click **Dashboard** in the left navigation pane.
3. Click the **Create (+)** icon in the command bar.

   The **Create dashboard/widget** window appears.

4. Enter the field information in the **Create dashboard/widget** window.

   The following table provides the field description to create a dashboard:

Field Description for Create dashboard/widget section

Field	Description
***Dashboard name**	Name of the dashboard.
***Select solution**	ADC is the select solution.
***Widget type**	Type of the widget. Options are: • **Custom:** Choose this option to create a customized widget. By default, this option is selected. • **Default:** Choose this option to select the default widget. When you choose this option, the **Choose widgets** option appears, which allows you to select the widgets.

Field	Description
***Select widget**	Customized widgets appear in the drop-down menu. Select the appropriate widget.
***Widget name**	Name of the widget.

Note:

Fields marked with red asterisk (*) symbol are mandatory.

5. To create a dashboard/widget, click **Create**.

Exporting Dashboard

For more information, refer to the **Exporting Dashboard Information** section in the CERT User Guide.

Importing Dashboard

For more information, refer to the **Importing Dashboard** section in the CERT User Guide.

Deleting Dashboard

For more information, refer to the **Deleting Dashboard** section in the CERT User Guide.

Alerting and Logging

CERT+ allows you to monitor the AppViewX component level and certificate-related alerts in a dashboard with predefined filters. Also, you can configure alerts based on your business needs. With these alerts, you can trigger an email with the necessary information. To run a custom logic based on the alert condition, you can configure it through a visual workflow in AppViewX. Alerts and logs help you to ensure the system performance is monitored.

You can view logs and receive certificate alerts through:

- Certificate Logs
- Certificate Alerts

For more information, refer to the **Alerts and Logs** section in the CERT User Guide.

Steps for Migration

Following are the steps to migrate:

- CA policy must have only issuer-based configuration.
- Reconfigure the RBAC configuration for PKI+.
- Ensure that there is no custodian or CA in the *in-progress* state.
- For on-premise deployments, the settings have to be configured. See Settings.

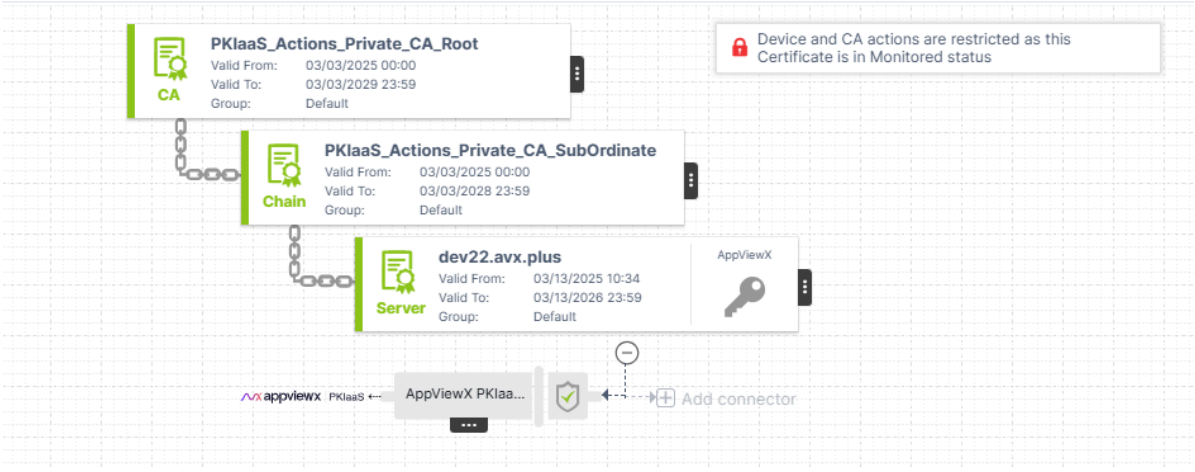
Troubleshooting

For more information to troubleshoot some common problems that can occur while executing CLM actions, refer to the CERT Troubleshooting Guide.

Troubleshooting OCSP Request with OpenSSL

Follow these steps to test the OCSP service via CC URL: http://{CCURL}:{Scep port}/ocsp:

1. Download the immediate issuer and end certificates from the AppViewX UI with the following names:
   - PKIaaS_Actions_Private_CA_SubOrdinate - issuer.crt
   - dev22.avx.plus - cert.crt



2. Configure openSSL in the end device.
3. Trigger the following OpenSSL command with issuer.crt and cert.crt:

```
openssl ocsp -issuer issuer.crt -cert cert.crt -text -url http://pe-pltf-node66.lab.appviewx.net:30022/ocsp -noverify
```

If the certificate is revoked, the revoke status will be received in the OCSP response as shown:

```
        73:c3:2e:b4:ba:86:8f:51
-----BEGIN CERTIFICATE-----
MIIE8DCCAtqgAwIBAgIQGfAd1HQ3U6lEDbdrBXSiNzALBgkqhkiG9w0BAQswJDER
MA8GA1UECgwIQXBwVmlld1gxDzANBgNVBAMMBnN1Yl9jYTAeFw0yNTAyMjQwNTQx
MTdaFw0yNjAyMjQyMzU5NTlaMCkxETAPBgNVBAoMCEFwcFZpZXdYMRQwEgYDVQQD
DAtzdWJfY2Eub2NzcDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAJ+v
tt6sUEonDaqMN1ylsEXbsRS8uCiKdexoDWKTKqkkvtXdDrnjrUht694p1yX/quSY
rKXe5WD7zMqsrOsR9PLDui9LISYLiIKL424NBcm7HHUcRuKM5uEqcaBvCXhpChJS
XSxXHqsVBf23GofB0hwIerZ1wB6ITTqaUjHWw1qiOK5/rcrQexiUOeAeVHjstE7q
vMPwY7TX7EWR/WCFNiD5JpnnNL0gZB8GcVDPgUfyImhuvqGNBp9HgoqpwPsOKfTs
15Pi4uH4mxryyICy/Me7vtJutomHhyafKVM4NApEKQk2VjNZYlDSvI59hShnHV01
Q5fBdB+NGAcxNp5/9lsCAwEAAaOCARswggEXMB0GA1UdDgQWBBSrFsz60PxDIfqw
TakroVXL1gEtrDAfBgNVHSMEGDAWgBTq2vj0fYYIEsyaoBOt7LUV8LxkkDAOBgNV
HQ8BAf8EBAMCB4AwDAYDVR0TAQH/BAIwADATBgNVHSUEDDAKBggrBgEFBQcDCTBN
BgNVHR8ERjBEMEKgQKA+hjxodHRwczovLzQuMjI0LjExMS4yMzM6NDQzL2F2eGFw
aS9kb3dubG9hZC1jcmwvc3ViX2NhL2NybC5jcmwwUwYIKwYBBQUHAQEERzBFMEMG
CCsGAQUFBzAChjdodHRwczovLzQuMjI0LjExMS4yMzM6NDQzL2F2eGFwaS9kb3du
bG9hZC1pc3N1ZXIvc3ViX2NhMAsGCSqGSIb3DQEBCwOCAgEAcAm7kOqekT6/wN0R
kiHbpa4dinbemrg+3LnpSaAh/67hmNfDn1otjWurLrvkQNKS4e6yyn96Op0soU6y
2dPruL7Z5bEItLuOhfiY4+/DtvylvMrAuKv63j1fW452oOyHkM0ZD8HwPZ51yM3y
B7WdPNJGYFsM5x/Ta2glu6HFC7KcEKK7FSrZ7opx+WJ87fudt0/fEU3ei/oiKIPs
MV8HD7FXKNOQd2i5qFuLctpkkM/I813C+v/wPdoKzgwOzRZaH5hGYVpJFyJowyvM
PuE438DcGoGImEhWQH6CS8G8vPAFQ5AeD6i1Xl5EsjQ/TfQuCWlEyEdwvoFwAdqm
pSoT4qjzRfpBPdXUIuEW5qOz14Jj1tMMNIPwrkUEZX7z/8oqbc8ltJprNp9gRUxM
D8TqJoOf9rFPDIIz5Rkt5nRgW5k7bpF8N+9xTl7n18sJ8f6wQef0/KoO2DWPIHB1
ybf8FfA7yXCkjZY5M1X+MCXrg7j77gc4UmQ1p2of60AotkVozj8LDVX4XkG352bu
wMOeXff56SuUjPeyVoT9744iHQRRxPG6y5d/8OVPPFeOjcxXVesDrgUiO8aUuiuD
0bhlXzZWlZQNpO5m7rLB5quLy0++fYn3iZpE5xS5tI7t8bdmd5Jf57oARB623a3m
pkIwU3pBSt6E7Dx1c8MutLqGj1E=
-----END CERTIFICATE-----
ocsp/PKIOCSPServerSHA256RSA4096.appviewx.info.crt: revoked
        This Update: Feb 25 10:19:10 2025 GMT
        Next Update: Feb 25 10:19:10 2025 GMT
        Reason: keyCompromise
        Revocation Time: Feb 25 10:19:10 2025 GMT
```

4. Once the CC URL is accessible from OCSP, test the same with CC's load balancer.

Chapter 2: PKI+ API Guide

This guide provides information on the APIs to use for various actions.

Best Practices for Working with the AppViewX API

- **Use appropriate HTTP methods.**

  Ensure that the correct HTTP method is used for each operation (e.g., GET for retrieval, POST for creation).

- **Handle errors gracefully.**

  Implement proper error handling in your application to manage API responses.

- **Use secure storage.**

  Store access tokens securely and avoid hardcoding them in your application code.

- **Implement pagination.**

  For endpoints that return large datasets, implement pagination using limit and offset parameters.

Understanding the AppViewX PKI+ API

The AppViewX PKI+ API provides a set of RESTful endpoints for managing certificates across your infrastructure. This section covers how to make requests, handle responses, and understand the structure of the API.

RESTful HTTPS Requests

The PKI+ API uses RESTful principles, leveraging standard HTTP methods to interact with resources. All requests must be made over HTTPS to ensure security.

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

All API endpoints are accessed via the following base URL. The base URL is built in the same way by the following structure:

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

Explanation

- **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?gwkey=f000ca01&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).

HTTP Code	Response Message
400 Bad Request	The request is not understood or required parameters are missing.
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 a User Account

For accessing APIs using a User account:

- **User Account**

  A **user account** represents an individual person interacting with the application or the system. User accounts are used for accessing the system on behalf of a user.

  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	(Mandatory) Use login name of the user.
Header	**Type:** String
	Example: "admin"
password	(Mandatory) Password for the username.
Header	**Type:** String

Input Parameters (continued)

	Description
	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 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.
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

Log in to the application with a username and password.

Sample 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
}
```


What's next?

After the sessionID is retrieved using the login API, you can now use the session ID for API calls.

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.

In this section, as an example, we are using the session ID with the API call for reissuing a certificate in the async mode.

Before you begin

Ensure the following before attempting to renew certificate from any CA through AppViewX:

- The CA setting must be configured in AppViewX for the CA.
- Connectivity to the CA via the chosen setting is working fine.
- **Approval is not required:** Enable this mode by setting the 'Certificate Requests Need Approval?' flag to false in the Certificate Policy.
- **Approval is required**: If the approval setting in the policy cannot be changed, users can approve specific requests by following the After you are done section.

Request Structure

Endpoint:	/certificate/action
Type:	PUT
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/certificate/action?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) Session Id received after login. **Type**: String **Constraint**: Required if username and password are not provided.
userName *Header*	(Mandatory only if **sessionID** is not provided) Username that is configured in AppViewX. **Type**: String
password *Header*	(Mandatory only if **sessionID** is not provided) Password of that user. **Type**: String
content-type *Header*	(Mandatory) Payload content-type with application/json value. **Type**: String **Constraint**: The value must be **application/json**.
gwkey *Query*	(Mandatory) Tenant Key. This is required only for multi-tenant installations and can be disregarded for other types of installations. **Type**: String
gwsource *Query*	(Mandatory) The source from which the request is triggered, e.g., external. **Type**: String
Payload	Contains all the parameters to be sent in the request body for the put request.

Input Parameters (continued)

Name	Description
Body	**Type**: Payload

Payload

Payload

Name	Description
resourceId	(Mandatory) Unique Id of the certificate. **Type**: String **Constraint**: Required if the commonName and serialNumber are not specified.
commonName	(Mandatory) Common name of the certificate. **Type**: String **Constraint**: Required if resourceId is not specified.
serialNumber	(Mandatory) Serial number of the certificate. **Type**: String **Constraint**: Required if resourceId is not specified.
action	(Mandatory) Action name for the reissue. **Type**: String **Possible values**: Reissue
reason	(Mandatory) Reason for reissue request. **Type**: String

Response Structure

Response returns string of type application/json with the following body parameters:

Parameters

Name	Description
response	Contains the response attributes for the reissue request.
resourceId response	Identifier of the certificate record that has been created. **Type**: String
requestId response	Work order request Id. **Type**: String
message response	Success message - Reissue action triggered successfully. **Type**: String
message	Success message or failure description in case of error. **Type**: String
appStatusCode	Application specific status code for the response. It is a non-null value for a failure response. **Type**: String
tags	Additional information in case of failure response.

Status codes

HTTP Code	appStatusCode	Response Message
202 Accepted	null	Reissue action has been triggered successfully
401 Unauthorized	AVX_GW_003	Authentication failed, reason - Invalid Credentials. **Remediation**: Ensure that valid username and password or a valid sessionId is provided as header parameters.
404 Not Found	NO_RECORDS_FOUND	No matching records found.

HTTP Code	appStatusCode	Response Message
		Remediation: Check and ensure that the values provided for commonName / serialNumber / resourceId are correct.
400 Bad Request	INVALID_REQUEST	Please give valid common name and serial number or resourceId. **Remediation**: Provide a valid commonName and serialNumber or resourceId.
400 Bad Request	INVALID_REQUEST	Please provide a valid action. **Remediation**: Provide a valid action.
400 Bad Request	MANDATORY_FIELD_MISSING	Mandatory field is missing or invalid - action. **Remediation**: Ensure that the **action** field is available in the request payload.
400 Bad Request	MANDATORY_FIELD_MISSING	Mandatory field is missing or invalid - reason. **Remediation**: Ensure that the **reason** field is available in the request payload.
417 Expectation Failed	OPEN_WORK_ORDERS_FOUND	Since requested certificate's work order is in progress, cannot initiate another action. **Remediation**: Trigger the request once the open workOrder for the certificate is completed.
406 Not Acceptable	CERT-VWF-0006	Life cycle action is unsupported by CA or another work order is in progress or certificate belongs to read group or is in Monitored status. **Remediation**: Ensure the following: • The CA supports the reissue action. • There is no workOrder in progress for the specified certificate.

HTTP Code	appStatusCode	Response Message
		• Certificate does not belong to read only group. • Certificate is not in the monitored status.

Sample Request/Response

Request Payload


```
{
"commonName":"testcert8g.appviewx.plus",
 "serialNumber":"08:BD:54:35:FD:81:AC:A4:45:2B:FC:9F:77:56:FE:2C",
 "action":"Reissue",
 "reason":"Superseded"
 }
```


Response


```
{
"response": {
"resourceId": "5f51f7312fa95059a12b0aee",
"message": "Reissue action triggered successfully.",
"requestId": "238"
},
"message": "Reissue action has been triggered successfully",
"appStatusCode": null,
"tags": {},
"headers": null
}
```


Authentication using a Service Account

For accessing APIs using a Service account:

- **Service Account**

  A **service account** represents a non-human entity such as an application or a service. It is used for automated processes or system-to-system interactions without human intervention.

For accessing APIs with a service account, you need to get the Access Token by providing Client ID and Client Secret in get-service-token API. This Access Token can then be used for accessing other APIs.

Note:

Access Token Validity is 30 minutes by default and it can be configured in **Settings** > **Authentication** > **oAuth Settings**.

- 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"

Input Parameters (continued)

	Description
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.
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

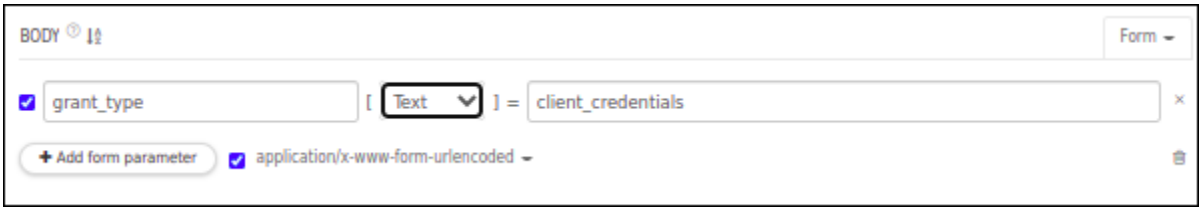
```
https://<IP/HostName/TenantName>:<GWPORT>/avxapi/acctmgmt-get-service-token?gwsource=external
```

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

Authorization: Basic

```
NTIxYzNhZDItZWE0ZS00NDdiLWE1MWItOTYyMWJiN2VhMTI2OjU1QVUjTk84JSpaaGd2TmZhWVVtdHZYMGRrRWhvZGJpCg==
```

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.

In this section, as an example, we are using the access token with the API call for reissuing a certificate in the async mode.

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:	`/certificate/create`
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".
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
 }
```

PKI API

Following are the PKI API commands:

- CA Inventory View
- Create Root CA
- Create Subordinate CA
- Enable CA
- Disable CA
- Delete CA
- Template View
- Issue Certificate

CA Inventory View

The API initiates a request to view the CA inventory.

Before you begin

Ensure the following before attempting to view the CA inventory:

- Refer to Prerequisites in the PKI User Guide.

Request Structure

Endpoint:	v1/pki/ca/ca-list
Type:	GET
Sample URL:	https://<IP/HostName/TenantName>:>GWPORT>/avxapi/v1/pki/ca/ca-list?gwsource=external To understand the elements of the sample URL, click here.
Headers	
Content-Type:	application/json

Input Parameters

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

Response Structure

Response returns string of type application/json with the following body parameters:

Parameters

Name	Description
response	Contains the response attributes for the view CA inventory request.
message	Success message or failure description in case of error. **Type**: String
appStatusCode	Application specific status code for the response. It is a non-null value for a failure response. **Type**: String
tags	Additional information in case of failure response.
name	Name of the CA or certificate. **Type**: String
description	Description field; `null` in this example. **Type**: String
certificateType	Type of certificate (e.g., Root CA, Intermediate CA). **Type**: String
createdTime	Timestamp when the entry was created. **Type**: String
expiryTime	Timestamp for expiry; `null` likely means it has not been issued yet. **Type**: String
status	Status of the certificate (e.g., `NEW`, `ISSUED`, etc.). **Type**: String
approverStatus	Approval status (e.g., `CREATE_REJECTED`, or `null` if not yet reviewed). **Type**: String
iTotalDisplayRecords	Indicates the total number of records after applying the search filter.

Parameters (continued)

Name	Description
	Type: String
searchQuery	Specifies the search term used to filter results. **Type**: String

Status Codes

HTTP Code	appStatusCode	Response Message
200 Accepted	null	View CA inventory action has been triggered successfully.
401 Unauthorized	AVX_GW_003	Authentication failed, reason - Invalid Credentials. **Remediation**: Ensure that valid username and password or a valid sessionId is provided as header parameters.

Sample Response

Response


```
{
  "response": {
    "data": [
      {
        "name": "Test1",
        "description": null,
        "certificateType": "Root CA",
        "createdTime": 1746781702035,
        "expiryTime": null,
        "status": "NEW",
        "approverStatus": null
      },
      {
        "name": "asd",
        "description": null,
```

```
            "certificateType": "Root CA",
            "createdTime": 1746781552135,
            "expiryTime": null,
            "status": "NEW",
            "approverStatus": "CREATE_REJECTED"
        },
        {
            "name": "qwe",
            "description": null,
            "certificateType": "Root CA",
            "createdTime": 1746781536971,
            "expiryTime": null,
            "status": "NEW",
            "approverStatus": "CREATE_REJECTED"
        },
        {
            "name": "Test",
            "description": null,
            "certificateType": "Root CA",
            "createdTime": 1746781524420,
            "expiryTime": null,
            "status": "NEW",
            "approverStatus": null
        }
    ],
    "iTotalDisplayRecords": 4,
    "searchQuery": null
  },
  "message": null,
  "appStatusCode": null,
  "tags": {},
  "headers": null
}
```


References

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**.

Create Root CA

The API initiates a request to create a root CA.

Before you begin

Refer to the Ensure the following before attempting to renew certificate from any CA through AppViewX:

- Refer to Prerequisites in the PKI User Guide.

Request Structure

Endpoint:	v1/pki/ca/create-ca
Type:	POST
Sample URL:	https://<IP/HostName/TenantName>:>GWPORT>/avxapi/v1/pki/ca/create-ca?gwsource=external

To understand the elements of the sample URL, click here.
Headers
Content-Type: application/json

Input Parameters

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

Payload

Payload

Name	Description
name	(Mandatory) Name of the root CA. **Type**: String
description	(Optional) Description for the root CA. **Type**: String
validityUnitValue	(Mandatory) Duration of certificate validity. **Type**: String

Payload (continued)

Name	Description
validityUnit	(Mandatory) Unit of validity (e.g., years, months). **Type**: String
policyId	(Mandatory) OID (Object Identifier) of the certificate policy. **Type**: String
maxIssuerPathLength	(Mandatory) Maximum number of intermediate CAs that can follow this CA (or `"NONE"`). **Type**: String
templateName	(Mandatory) Name of the certificate template to apply. **Type**: String
keySizeAlgorithm	(Mandatory) Key algorithm and size (e.g., RSA, ECC, with hash type). **Type**: String
issuerMaxLength	(Mandatory) Max length of issuer path, possibly for constraint enforcement. **Type**: String
caType	(Mandatory) Type of CA being created (*Root CA* in this case). **Type**: String

pkiKeySpec

Name	Description
keyGenerationSource	(Mandatory) Source of the key (e.g., "AppViewX", "HSM") **Type**: String
keySizeAlgorithm	(Mandatory) Key algorithm and size **Type**: String

csrParameters

Name	Description
organization	(Mandatory) Organization name **Type**: String
organizationUnit	(Optional) Department or unit **Type**: String
locality	(Optional) City or location **Type**: String
state	(Optional) State or province **Type**: String
country	(Optional) Country code **Type**: String
commonName	(Mandatory) Common name for the root CA **Type**: String

Response Structure

Response returns string of type application/json with the following body parameters:

Parameters

Name	Description
response	Indicates status of the API operation.
message	Success message - PKIaaS CA configuration added successfully **Type**: String
appStatusCode	Application specific status code for the response. It is a non-null value for a failure response.

Parameters (continued)

Name	Description
	Type: String
tags	Additional information in case of failure response.

Status Codes

HTTP Code	appStatusCode	Response Message
200 Accepted	null	Create root CA action has been triggered successfully.
401 Unauthorized	AVX_GW_003	Authentication failed, reason - Invalid Credentials. **Remediation**: Ensure that valid username and password or a valid sessionId is provided as header parameters.
400 Bad Request	INVALID_TEMPLATE	Occurs when specified template is not present. **Remediation**: Ensure that specified template is selected.
400 Bad Request	VALIDATION_ERROR_0004	Occurs when type of value in invalid. **Remediation**: Provide valid input value. Example: validityUnit is string but integer is given.
400 Bad Request	HSM_DETAIL_NOT_FOUND_IN_KEYSPEC	Occurs when keyGenerationSource is HSM, but keyHandlerName or hsmDeviceName is not present. **Remediation**: Provide all fields required for HSM.

HTTP Code	appStatusCode	Response Message
417 Expectation Failed	CA_CONFIG_NAME_EXIST	Occurs when CA name already exists. **Remediation**: Provide a unique CA name.

Sample Request/Response

Request Payload

```
{
    "name": "RootCA",
    "description": "sample description",
    "validityUnitValue": "1",
    "validityUnit": "years",
    "policyId": "2.5.29.32.0",
    "maxIssuerPathLength": "NONE",
    "templateName": "RootCA_Default",
    "keySizeAlgorithm": "RSA_PKCS1_4096_SHA256",
    "issuerMaxLength": "NONE",
    "caType": "Root CA",
    "pkiKeySpec": {
        "keyGenerationSource": "AppViewX",
        "keySizeAlgorithm": "RSA_PKCS1_4096_SHA256"
    },
    "csrParameters": {
        "organization": "AppViewX Inc.",
        "organizationUnit": "PE",
        "locality": "CBE",
        "state": "TN",
        "country": "IN",
        "commonName": "RootCA"
    }
}
```

Response

```
{
    "response": "Success",
    "message": "PKIaaS CA configuration added successfully.",
    "appStatusCode": null,
    "tags": {},
    "headers": null
}
```


References

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**.

Create Subordinate CA

The API initiates a request to create a subordinate CA.

Before you begin

Ensure the following before attempting to renew certificate from any CA through AppViewX:

- Refer to Prerequisites in the PKI User Guide.

Request Structure

Endpoint:	v1/pki/ca/create-ca
Type:	POST
Sample URL:	https://<IP/HostName/TenantName>:>GWPORT>/avxapi/v1/pki/ca/create-ca?gwsource=external To understand the elements of the sample URL, click here.
Headers	
Content-Type:	application/json

Input Parameters

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

Input Parameters (continued)

Name	Description
	Constraint: Value of the parameter should be **'application/json'**
gwsource *Query*	(Mandatory) Source from which the request is triggered **Type**: String
Payload *Body*	Contains all the parameters to be sent in the request body for the put request. **Type**: Payload

Payload

Payload

Name	Description
name	(Mandatory) Name of the sub CA. **Type**: String
description	(Optional) Description for the sub CA. **Type**: String
rootCAName	(Mandatory) Name of the root CA under which this subordinate CA will be created. **Type**: String
validityUnitValue	(Mandatory) Duration of certificate validity. **Type**: String
validityUnit	(Mandatory) Unit of validity (e.g., years, months). **Type**: String
policyId	(Mandatory) OID (Object Identifier) of the certificate policy. **Type**: String

Payload (continued)

Name	Description
maxIssuerPathLength	(Mandatory) Maximum number of intermediate CAs that can follow this CA (or "NONE"). **Type**: String
templateName	(Mandatory) Name of the certificate template to apply. **Type**: String
keySizeAlgorithm	(Mandatory) Key algorithm and size (e.g., RSA, ECC, with hash type). **Type**: String
issuerMaxLength	(Mandatory) Max length of issuer path, possibly for constraint enforcement. **Type**: String
caType	(Mandatory) Type of CA being created (*Root CA* in this case). **Type**: String

pkiKeySpec

Name	Description
keyGenerationSource	(Mandatory) Source of the key (e.g., "AppViewX", "HSM") **Type**: String
keySizeAlgorithm	(Mandatory) Key algorithm and size **Type**: String

csrParameters

Name	Description
organization	(Mandatory) Organization name **Type**: String
organizationUnit	(Optional) Department or unit

csrParameters (continued)

Name	Description
	Type: String
locality	(Optional) City or location **Type**: String
state	(Optional) State or province **Type**: String
country	(Optional) Country code **Type**: String
commonName	(Mandatory) Common name for the sub CA **Type**: String

Response Structure

Response returns string of type application/json with the following body parameters:

Parameters

Name	Description
response	Indicates status of the API operation.
message	Success message - PKIaaS CA configuration added successfully **Type**: String
appStatusCode	Application specific status code for the response. It is a non-null value for a failure response. **Type**: String
tags	Additional information in case of failure response.

Status Codes

HTTP Code	appStatusCode	Response Message
200 Accepted	null	Create sub root CA action has been triggered successfully.
401 Unauthorized	AVX_GW_003	Authentication failed, reason - Invalid Credentials. **Remediation**: Ensure that valid username and password or a valid sessionId is provided as header parameters.
400 Bad Request	INVALID_TEMPLATE	Occurs when specified template is not present. **Remediation**: Ensure that specified template is selected.
400 Bad Request	VALIDATION_ERROR_0004	Occurs when type of value in invalid. **Remediation**: Provide valid input value. Example: validityUnit is string but integer is given.
400 Bad Request	ISSUING_CA_CONFIG_NOT_FOUND	Occurs when the rootCA name is not found for sub CA creation. **Remediation**: Ensure rootCA name is present.
400 Bad Request	HSM_DETAIL_NOT_FOUND_IN_KEYSPEC	Occurs when keyGenerationSource is HSM, but keyHandlerName or hsmDeviceName is not present. **Remediation**: Provide all fields required for HSM.

HTTP Code	appStatusCode	Response Message
417 Expectation Failed	CA_CONFIG_NAME_EXIST	Occurs when CA name already exists. **Remediation**: Provide a unique CA name.
400 Bad Request	ROOT_CA_NAME_MANDATORY_FOR_SUB_CA	Occurs when the rootCAName field in payload for sub ca creation is not given. **Remediation**: Ensure that the rootCAName field in payload for sub ca creation is provided.

Sample Request/Response

Request Payload

```
{
    "name": "SubCA",
    "description": "sample description",
    "rootCAName":"RootCaMay8",
    "validityUnitValue": "6",
    "validityUnit": "months",
    "policyId": "2.5.29.32.0",
    "maxIssuerPathLength": "NONE",
    "templateName": "SubCA_Default",
    "keySizeAlgorithm": "RSA_PKCS1_4096_SHA256",
    "issuerMaxLength": "NONE",
    "caType": "Subordinate CA",
    "pkiKeySpec" :{
       "keyGenerationSource" : "AppViewX",
       "keySizeAlgorithm" :"RSA_PKCS1_4096_SHA256"
    },
    "csrParameters": {
       "organization": "AppViewX Inc.",
       "organizationUnit": "PE",
```

```
            "locality": "CBE",
            "state": "TN",
            "country": "IN",
            "commonName": "SubCA"
        }
}
```

Response

```
{
    "response": "Success",
    "message": "PKIaaS CA configuration added successfully.",
    "appStatusCode": null,
    "tags": {},
    "headers": null
}
```

References

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**.

Enable CA

The API initiates a request to enable the CA.

Before you begin

Ensure the following before attempting to renew certificate from any CA through AppViewX:

- Refer to Prerequisites in the PKI User Guide.

Request Structure

Endpoint:	v1/pki/ca/<CAName>/status/enable
Type:	POST
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/v1/pki/ca/<CAName>/status/enable?gwsource=external` To understand the elements of the sample URL, click here.
Headers	
Content-Type:	application/json

Input Parameters

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

Input Parameters (continued)

Name	Description
password	(Mandatory) AppViewX login password.
Header	**Type**: String **Constraint**: Required if sessionId is not provided.

Response Structure

Response returns string of type application/json with the following body parameters:

Parameters

Name	Description
response	Indicates status of the API operation.
message	Success message - PKIaaS approval request initiated successfully. **Type**: String
appStatusCode	Application specific status code for the response. It is a non-null value for a failure response. **Type**: String
tags	Additional information in case of failure response.

Status Codes

HTTP Code	appStatusCode	Response Message
200 Accepted	null	Enable CA action has been triggered successfully.
401 Unauthorized	AVX_GW_003	Authentication failed, reason - Invalid Credentials. **Remediation**: Ensure that valid username and password or a valid

HTTP Code	appStatusCode	Response Message
		sessionId is provided as header parameters.
400 Bad Request	INVALID_OPERATION_APPROVAL_IN_PROGRESS	Requested operation not allowed. Approval process in progress. **Remediation**: Ensure that approval request is either approved, rejected, or aborted and try again.
417 Expectation Failed	ACCESS_NOT_PRESENT_TO_WRITE	When user does not have resource access to CA. **Remediation**: Ensure you have resource access to CA.

Sample Response

Response


```
{
   "response": "Success",
   "message": "PKIaaS approval request initiated successfully.",
   "appStatusCode": null,
   "tags": {},
   "headers": null
}
```


References

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**.

Disable CA

The API initiates a request to disable a CA.

Before you begin

Ensure the following before attempting to renew certificate from any CA through AppViewX:

- Refer to Prerequisites in the PKI User Guide.

Request Structure

Endpoint:	v1/pki/ca/<CAName>/status/disable
Type:	POST
Sample URL:	https://<IP/HostName/TenantName>:<GWPORT>/avxapi/v1/pki/ca/<CAName>/status/disable?gwsource=external

To understand the elements of the sample URL, click here.
Headers
Content-Type: application/json

Input Parameters

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

Response Structure

Response returns string of type application/json with the following body parameters:

Parameters

Name	Description
response	Indicates status of the API operation.
message	Success message - PKIaaS approval request initiated successfully. **Type**: String
appStatusCode	Application specific status code for the response. It is a non-null value for a failure response. **Type**: String

Parameters (continued)

Name	Description
tags	Additional information in case of failure response.

Status Codes

HTTP Code	appStatusCode	Response Message
200 Accepted	null	Disable CA action has been triggered successfully.
401 Unauthorized	AVX_GW_003	Authentication failed, reason - Invalid Credentials. **Remediation**: Ensure that valid username and password or a valid sessionId is provided as header parameters.
400 Bad Request	INVALID_OPERATION_APPROVAL_IN_PROGRESS	Requested operation not allowed. Approval process in progress. **Remediation**: Ensure that approval request is either approved, rejected, or aborted and try again.
417 Expectation Failed	ACCESS_NOT_PRESENT_TO_WRITE	When user does not have resource access to CA. **Remediation**: Ensure that valid

Sample Response

Response

```
{
    "response": "Success",
    "message": "PKIaaS approval request initiated successfully.",
    "appStatusCode": null,
    "tags": {},
    "headers": null
}
```


References

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**.

Delete CA

The API initiates a request to delete a CA.

Before you begin

Ensure the following before attempting to renew certificate from any CA through AppViewX:

- Refer to Prerequisites in the PKI User Guide.

Request Structure

Endpoint:	v1/pki/ca/delete/<CAName>
Type:	DELETE
Sample URL:	https://<IP/HostName/TenantName>:<GWPORT>/avxapi/v1/pki/ca/delete/<CAName>?gwsource=external here.
Headers	
Content-Type:	application/json

Input Parameters

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

Response Structure

Response returns string of type application/json with the following body parameters:

Parameters

Name	Description
response	Indicates status of the API operation.
message	Success message - PKIaaS approval request initiated successfully. **Type**: String
appStatusCode	Application specific status code for the response. It is a non-null value for a failure response. **Type**: String
tags	Additional information in case of failure response.

Status Codes

HTTP Code	appStatusCode	Response Message
200 Accepted	null	Delete CA action has been triggered successfully.
401 Unauthorized	AVX_GW_003	Authentication failed, reason - Invalid Credentials. **Remediation**: Ensure that valid username and password or a valid sessionId is provided as header parameters.
400 Bad Request	DELETE_SUB_CA_FIRST	Occurs when trying to delete Root CA when Sub CA is present. **Remediation**: Ensure that sub CA is deleted.

HTTP Code	appStatusCode	Response Message
400 Bad Request	INVALID_OPERATION_APPROVAL_IN_PROGRESS	Requested operation not allowed. Approval process in progress. **Remediation**: Ensure that approval request is either approved, rejected, or aborted and try again.
417 Expectation Failed	ACCESS_NOT_PRESENT_TO_WRITE	When user does not have resource access to CA. **Remediation**: Ensure that valid

Sample Response

Response


```
{
   "response": "Success",
   "message": "PKIaaS approval request initiated successfully.",
   "appStatusCode": null,
   "tags": {},
   "headers": null
}
```


References

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**.

Template View

The API initiates a request to view all the templates.

Before you begin

Ensure the following before attempting to renew certificate from any CA through AppViewX:

- Refer to Prerequisites in the PKI User Guide.

Request Structure

Endpoint:	v1/pki/ca/template
Type:	GET
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/v1/pki/ca/template?gwsource=external` To understand the elements of the sample URL, click here.
Headers	
Content-Type:	application/json

Input Parameters

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

Response Structure

Response returns string of type application/json with the following body parameters:

Parameters

Name	Description
name	Template name (e.g., `"RootCA_Default"`).
	Type: String
description	Template description.
	Type: String
category	Type of CA the template is for (e.g., `"Root CA"`).
	Type: String
allowCsrPassThrough	Whether CSR fields can pass through as-is.
	Type: Boolean

Parameters (continued)

Name	Description
preShipped	Whether this template is preloaded in the system. **Type**: Any
subjectAltName	Placeholder for SAN config. **Type**: Any
sanFieldDescriptorList	Descriptor for SAN fields. **Type**: String

Authority and Subject Key Identifiers (`authorityAndSubjectKey`)

Name	Description
allowAuthorityKeyId	Includes Authority Key Identifier. **Type**: Boolean
allowSubjectKeyId	Includes Subject Key Identifier. **Type**: Boolean
subjectKeyHashBit	Bit length of Subject Key hash. **Type**: Integer
inheritAuthorityKeyIdFromCA	Whether to inherit AKI from issuing CA. **Type**: Boolean

Extended Key Usage (`extendedKeyUsage`)

Name	Description
keyUsageCritical	Marks base key usage as critical. **Type**: Boolean
extendedKeyUsageCritical	Marks extended key usage as critical.

Extended Key Usage (`extendedKeyUsage`) (continued)

Name	Description
	Type: Boolean
customExtendedKeyUsageExtensionsEnabled	Enable custom EKUs. **Type**: Integer
customExtendedKeyUsageExtensionsList	List of custom EKUs. **Type**: Any

CA Options (`caOptions`)

Name	Description
isCA	Indicates if this is a CA certificate **Type**: Boolean
critical	Marks the BasicConstraints extension as critical. **Type**: Boolean
maxIssuerPathLength	Max depth for intermediate CAs ("NONE" = unlimited). **Type**: String

Other Fields

Name	Description
validityOffsetUnitValue	Offset duration before cert validity starts. **Type**: Integer
validityOffsetUnitType	Offset unit (e.g., `"MINUTES"`). **Type**: String
keywords	Tags/labels for the template. **Type**: Array

Other Fields (continued)

Name	Description
noRevAvail	Indicates if the certificate should be non-revocable. **Type**: Boolean
allowTemplateValidity	If validity can be set via template. **Type**: Boolean
allowCSRKeyUsage	Flags that control if CSR input is allowed for each field. **Type**: Boolean
_id	Internal unique ID of the template. **Type**: String

Status Codes

HTTP Code	appStatusCode	Response Message
202 Accepted	null	Template view action has been triggered successfully.
401 Unauthorized	AVX_GW_003	Authentication failed, reason - Invalid Credentials. **Remediation**: Ensure that valid username and password or a valid sessionId is provided as header parameters.

Sample Response

Response


```
{
   "response": {
      "data": [
         {
            "name": "RootCA_Default",
            "description": null,
            "category": "Root CA",
```

```
    "allowCsrPassThrough": false,
    "preShipped": true,
    "subjectAltName": null,
    "sanFieldDescriptorList": null,
    "authorityAndSubjectKey": {
        "allowAuthorityKeyId": true,
        "allowSubjectKeyId": true,
        "subjectKeyHashBit": 160,
        "inheritAuthorityKeyIdFromCA": false
    },
    "crlConfig": null,
    "aiaConfig": null,
    "templateSubjectDetails": null,
    "templateCertificatePolicy": null,
    "keyUsages": {
        "baseKeyUsage": {
            "digitalSignature": true,
            "contentCommitment": false,
            "keyEncipherment": false,
            "dataEncipherment": false,
            "keyAgreement": false,
            "certSign": true,
            "crlSign": true,
            "encipherOnly": false,
            "decipherOnly": false
        },
        "keyUsageCritical": true,
        "extendedKeyUsage": {
            "serverAuth": false,
            "clientAuth": false,
            "codeSigning": false,
            "emailProtection": false,
            "timeStamping": false,
            "ocspSigning": false,
            "ipsecEndSystem": false,
            "ipsecTunnel": false,
            "ipsecUser": false,
```

```
            "dvcs": false,
            "sbgpCertAAServerAuth": false,
            "scvp_responder": false,
            "eapOverPPP": false,
            "eapOverLAN": false,
            "scvpServer": false,
            "scvpClient": false,
            "ipsecIKE": false,
            "capwapAC": false,
            "capwapWTP": false,
            "smartcardlogon": false,
            "macAddress": false,
            "msSGC": false,
            "nsSGC": false,
            "anyExtendedKeyUsage": false,
            "kdcAuthentication": false,
            "fileRecovery": false,
            "certificateRequestAgent": false,
            "encryptionFileSystem": false
        },
        "extendedKeyUsageCritical": false,
        "unknownExtension": null,
        "customExtendedKeyUsageExtensionsEnabled": false,
        "customExtendedKeyUsageExtensionsList": null
    },
    "caOptions": {
        "isCA": true,
        "critical": true,
        "maxIssuerPathLength": "NONE"
    },
    "policyId": null,
    "additionalCustomExtensionEnabled": false,
    "validityOffsetUnitValue": 10,
    "validityOffsetUnitType": "MINUTES",
    "keywords": [
        "RootCA_Default",
        "Root CA"
```

```
            ],
            "additionalExtensions": null,
            "allowTemplateValidity": false,
            "noRevAvail": false,
            "allowValidityEndDateOverride": false,
            "validityEndDateOverride": null,
            "allowCSRBasicConstraints": false,
            "allowCSRKeyUsage": false,
            "allowCSRExtendedKeyUsage": false,
            "allowCSRSubject": false,
            "allowCSRSubjectAltName": false,
            "allowCSRValidationURLs": false,
            "allowCSRCertificatePolicy": false,
            "_id": "681c2bb2ec3ca41eb5341589"
        }
    ],
    "iTotalDisplayRecords": 1,
    "searchQuery": null
  },
  "message": null,
  "appStatusCode": null,
  "tags": {},
  "headers": null
}
```


References

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**.

Issue Certificate

The API initiates a request to issue certificate.

Before you begin

Ensure the following before attempting to renew certificate from any CA through AppViewX:

- Refer to Prerequisites in the PKI User Guide.

Request Structure

Endpoint:	v1/pki/ca/issue/cert
Type:	POST
Sample URL:	`https://<IP/HostName/TenantName>:<GWPORT>/avxapi/v1/pki/ca/issue/cert?gwsource=external` To understand the elements of the sample URL, click here.
Headers	
Content-Type:	application/json

Input Parameters

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

Payload

Payload

Name	Description
caName	(Mandatory) Name of the Certificate Authority to issue the certificate from. **Type**: String
templateName	(Mandatory) The certificate template to use. **Type**: String
validityUnit	(Mandatory) Unit for the certificate's validity (e.g., months, years). **Type**: String
validityUnitValue	(Mandatory) Action to triggered with the request **Type**: Integer
csrContent	(Mandatory) Base64-encoded CSR (Certificate Signing Request)

Payload (continued)

Name	Description
	Type: String
certificateType	(Mandatory) Type of certificate to be issued (e.g., End Certificate, Client, etc.) **Type**: String

Response Structure

Response returns string of type application/json with the following body parameters:

Parameters

Name	Description
response	Contains the response attributes for the issue certificate request.
message	Success message - Issue certificate action triggered successfully. **Type**: String
appStatusCode	Application specific status code for the response. It is a non-null value for a failure response. **Type**: String
tags	Additional information in case of failure response.

Status Codes

HTTP Code	appStatusCode	Response Message
200 Accepted	null	Issue certificate action has been triggered successfully.
401 Unauthorized	AVX_GW_003	Authentication failed, reason - Invalid Credentials. **Remediation**: Ensure that valid username and password or a valid

HTTP Code	appStatusCode	Response Message
		sessionId is provided as header parameters.
417 Expectation Failed	CA_CONFIG_NOT_FOUND	Occurs when specified CA is not present. **Remediation**: Ensure that specified CA is present.
500 Internal Server Error	TEMPLATE_NAME_NOT_AVAILABLE_IN_DB	Occurs when specified Template is not present. **Remediation**: Ensure that specified template is present.
400 Bad Request	VALIDATION_ERROR_0004	Occurs when csrContent field is not present. **Remediation**: Ensure that csrContent field is present.

Sample Request/Response

Request Payload


```
{
   "caName": "RootCaMay8",
   "templateName": "WebServer",
   "validityUnit": "months",
   "validityUnitValue": 1,
   "csrContent":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0KTUlJQ3h6Q0NBYThDQVFBd0dURVhNQlVHQTFVRUF3d09kM2QzTG1kdmIyZHNaUzVqYjI
wd2dnRWlNQTBHQ1NxRwpTSWIzRFFFQkFRVUFBNElCRHdBd2dnRUtBb0lCQVFDVUhxWnI1bVhlWCtjOXk0bTg1R2VIUXR2S3EyQVpjVHRxCi9UM1lzMlE3M
3k0YWEvN1pncW5IM2lTM0hTWCs4RjlYOHFuSVphUllvTzJ6Z1A1MVc1dXE5QkNGTEFMZ0pRclEKenBhTEhQekorOEtUMmJSblczRmtVTitUeG1MZkhXSnFYR
y9iRUZWM3FOdUVmTGtQcXlWZU5wWE95YlJCWkNLbwpQNkViY1g0THhUN241NzAwVTRQQ242aDE4WmVFOWtIV2NIb0FVejJpV1NOVVF3U3NmTnljSnZib
09GdUxmN1hZCkFPY0xLTWltSHd1OWdaVkZtaTZDNm5TVFl4WmhMV3VhUFFtcDdzdTUzUkJ1WjRsWTlldE11Z0VWMEo1cks4TXkKZmRMbVIvRHZWUXhxM
kI2djZDOGg2ckU5QVdxOVJYQ1dKR3lXdWVSSlJNRG5wVUJhTU5wTkFnTUJBQUdnYVRCbgpCZ2txaGtpRzl3MEJDUTR4V2pCWU1Bd0dBMVVkRXdFQi93UU
NNQUF3RGdZRFZSMFBBUUgvQkFREFnV2dNQmtHCkExVWRFUVFTTUJDQ0RuZDNkeTVuYjl5bmJHVXVZMjl0TUIwR0ExVWRKUVFXTUJRR0NDc0dBUV
VGQndNQkJnZ3lKQmdFRkJRY0RBakFOQmdrcWhraUc5dzBCQVFzRkFBT0NBUUVBUGh2aXcvc0hYR1BEdzAwaURRcUtTdjRrT1VncApINmw3SC9aOTF0eE
```

hvVEZteGlRRGxXYlJJYU9KeGlqbElvMlowQ1NHK0x3SC9yUFpBbDk1a0I1NFdYLzQ3aklICnBGVE1KczhyMFFRZ25uRE5TdFQyc1BGZ3g1R0ZnMWlkSmw3Zk1
GUmhxb0QyejR2QnNSUUtPS25yR2Jhb0dLZ2YKbzFXN2NXR2REZVBoOHJmZHB4RjlBMjRUcHMwekpPcDNLUnN4V0pzeTNXaE9Qd0IxRWx6RDloS040Szh6
Sis5egpuc2hUL0tORE5VNVYwakRxbkhOdS9vdmlSZDA4TVl0K2xzano2cUk0dXNGZU5rVmZBcUpKNUl0cmZsYmN0WmpsCmEvdnVZdHhSU2hOdVcrc1FIOS9
GcFpHR3o5T3o3aVJOcjRLNGFXTFFFZndnUTI1WmdPekQ5M1lDS2c9PQotLS0tLUVORCBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0=",
"certificateType": "End Certificate"
}

Response

{
"response":
"LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUZsakNDQTRDZ0F3SUJBZ0lRYI
RRcm1ySFpsQllBZUw2NFprYXdsREFMQmdrcWhraUc5dzBCQVFzd1hqRUxNQWtHQTFVRUJoTUNTVTR4
Q3pBSkJnTlZCQWdNQWxST01Rd3dDZ1lEVlFRSERBTkRRa1V4RmpBVUJnTlZCQW9NRFVGd2NGWnBaWGRZSUVsdVl5
NHhDekFKQmdOVkJBc01BbEJGTVE4d0RRWURWUVFEREFaU2lyOTBRMEV3SGhjTk1qVXdOVEV6TURrMU5UVTJXaGNOTWp
Vd05qRXpNak0xT1RVNVdqQVpNUmN3RlFZRFZRUUREQTUzZDNjdVoyOXZaMnhsTG1OdmJUQ0NBU0l3RFFZSktvWklod
mNOQVFFQkJRQURnZ0VQQURDQ0FRb0NnZ0VCQUpRZXBtdm1aZDVmNXozTGliemtaNGRDMjhxclCbHhPMnI5UGRpelpEdmZMa
HByL3RtQ3FjZmVKTGNkSmY3d1gxZnlxY2hscEZlODdiT0EvblZibTZyMEVJVXNBdUFsQ3RET2xvc2MvTW43d3BQWnRHZGJjV1
JRMzVQR1l0OGRZbXBjYjlzUVZYZW8yNFI4dVErckpWNDJsYzdKdEVGa0lxZy9vUnR4Zmd2RlB1Zm52VFJUZzhLZnFIWHhsNFQy
UWRad2VnQlRQYUpaSTFSREJLeDgzSndtOXVnNFc0dC90ZGdBNXdzb3lLWWZDNzJCbFVXYUxvTHFkSk5qRm1FdGE1bzlDYW51e
TduZEVHNW5pVmoxNjB5NkFSWFFubXNyd3pKOTB1Wkg4TzlWREdyWUhxL29MeUhxc1QwQmFyMUZjSllrYkphNTVFbEV3T2VsUUZ
vdzJrMENBd0VBQWFPQ0FaY3dnZ0dUTUIwR0ExVWREZ1FXQkJTQ1E5UldNZTU2elBVd0x0UTIxRThpT0NaYmtUQWZCZ05WSFNNRUd
EQVdnQlRFUUF6SHM3SktveHdKalpmb2NHZXZicXVXcnpBT0JnTlZIUThCQWY4RUJBTUNCYUF3REFZRFZSMFRBUUgvQkFJd0FEQVR
CZ05WSFNVRUREQUtCZ2dyQmdFRkJRY0RBVEJSQmdOVkhSOEVTakJJTUVhZ1JLQkNoa0JvZEhSd2N6b3ZMekU1TWk0eE5qZ3VNV
FExTGpnMk9qTXhORFF6TDJGMmVHRndhUzlrYjNkdWJHOWhaQzFqY213dlVtOXZkRU5CTVM5amNtd3VZM0pzTUlIS0JnZ3JCZ0V
GQlFjQkFRU0J2VENCdWpBNkJnZ3JCZ0VGQlFjd0FZWXVhSFIwY0RvdkwzQmxMV055ZVhCMGJ5MWhjSFo0TFc0MkxteGhZaTVoY0h
CMmFXVjNlQzV1WlhRdmIyTnpjREI4QmdnckJnRUZCUWN3QW9ad2FIUjBjSE02THk4eE9USXVNVFk0TGpFME5TNDROam96TVRRME1
5OWhkbmhoY0drdlpHOTNibXh2WVdRdGFYTnpkV1Z5TDFKdmIzUkRRVEUvYzJWeWFXRnNUblZ0WW1WeVBURXpNakkzTVRneU5qVT
JORFUzTWpNd01EZ3hOREV6TVRRMk5qTTRPVFEzTWpVNU9EWTFPREFMQmdrcWhraUc5dzBCQVFzRGdnSUJBR0ZJN24xRnY4SkN3RlRMN
Hc3OEZTcDZDZmdTZXVaWTNsaysweVBSOWlSczdINW9yOGFVcWJvTUJ6dVlYYkd1R2wvU0kzRnEvYUF3MDRtZDNsRGdYQkt0OWZYMWg1
SFZUTE9IeFNGMkQ0K20zOW9OVEx5RlJ3cy9PRENOcTZ3U25BbXk5WHRMTFFBWWgzc1pQR2tPUVVla1lMN1Y2VXRKeW8rT3RYTTE0Nn
VGakhKTHA2eDBTQjNNeDhJR3B5ZEw2ME4zMDF3eXNXYloxSHJpdWJLWjdNb0hCVDBhSERBTitXRWNVQU44ajRoSTNFTFdpekFme
DFBQ013V25iMjJVUFdhQnR6elVHOUhvbWZ5VGNYWVlJNW1RMkhoWExZazlqZzFzRTZjTmdNOUpDUHdTOENXdXVqSk9pYTg0Z2pmbzkr
SnA3WXVFQUxBQi9oR2hlYlpXMmNDclBpS2U1UmpaSGJiQ2RyVGY2bmtDUGNod0VwNTB2VnhZalQ4YzFadnk4Z0kxVHJKNVpBdGg0dHdXY
3ZiTUttbHduZldtU2x2cFhvcnZpRzR2YWVPbHA4VnZ3YkNHMFFsazdFU1N3WjJROUJ0cUtjb2grbk9SQ1NZd1IxTGtBNGtqMG5lWGhpb
3BsMGJ3YmhFdTh6eEFmZWZLRFQrcGtFNHVOQThNNHpETjh1NnVVZXFMUGNDbFdnMmVlV0hPR0x6bzhqRjVzZWZKelRJR3gwVUE5ME0wYWs
3ZDRBVEU3ZUNRWTBYMG9ERHZQWGtUSEY5YlpRRTM3S0h5Qm41U3g5V2crZUQwYmZFN1p6Mm5SOUJYWjk1V2VGRm9vZGxmUkdRWmNWMm90
MkIvTTJqRkdKSWJneEZtblJMTkdpWDc0M0ZmVW5IbTFORU5RK2krMHhNTzZNazUKLS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQo=",

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


References

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**.