appviewx

Install, Upgrade, and Maintenance Guide

Version: 2022.1.0

Copyright AppViewX, Inc.

Trademarks

The trademarks, logos, and service marks displayed in this manual are the property of AppViewX or other third parties. Users are not permitted to use these marks without the prior written consent of AppViewX or such third party which may own the mark.

External Reference Links

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

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

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

All other trademarks mentioned in this document are the property of their respective owners.

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	Initial release of document for Release 2022.1.0	June 2022
1.1	Release 2022.1.0 FP1	September 2022

About this Guide

This document covers the installation, upgrade, and maintenance activities for AppViewX. The document is divided into the following sections:

- **Overview** - provides an introduction to the product and explains the basic features.
- **Working with Prerequisites** - explains the prerequisites needed to install AppViewX.
- **Deploying the AppViewX Virtual Appliance** - describes the procedure to install AppViewX using the default settings.
- **Installing AppViewX** - describes the procedure to install AppViewX using the command line interface.
- **Monitoring and Maintaining AppViewX** - explains the functions available to maintain and monitor AppViewX.
- **Uninstalling AppViewX** - describes the procedure to safely remove AppViewX from the system.
- **Troubleshooting Installation** - explains the steps to be performed in case of issues encountered during the installation of AppViewX.

Audience

This guide is intended for the following audience:

- Network Engineers
- Service Engineers
- Customer Support Executives
- System Administrators

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: Overview

Introduction

AppViewX’s offering is a modular, low-code software application that enables the automation and orchestration of network infrastructure using an intuitive, context-aware, visual workflow. Leveraging a vast library of pre-built tasks and workflows, AppViewX enables the operations teams to quickly and easily translate business requirements into automation workflows that improve agility, enforce compliance, eliminate errors, and reduce cost. AppViewX is closed-loop and state-aware, capable of verifying that intent has been achieved and providing actionable insights and automated remediation.

AppViewX is a web based application that helps users:

- Manage ADC devices
- Manage certificates

In order to perform the above functions, AppViewx provides the following modules:

- ADC
- CERT+
- Platform
- Security
- Automation

AppViewX is built on the microservice architecture. A microservice is a program that runs on a server or a virtual computing instance. The main task of this program is to respond to network requests.

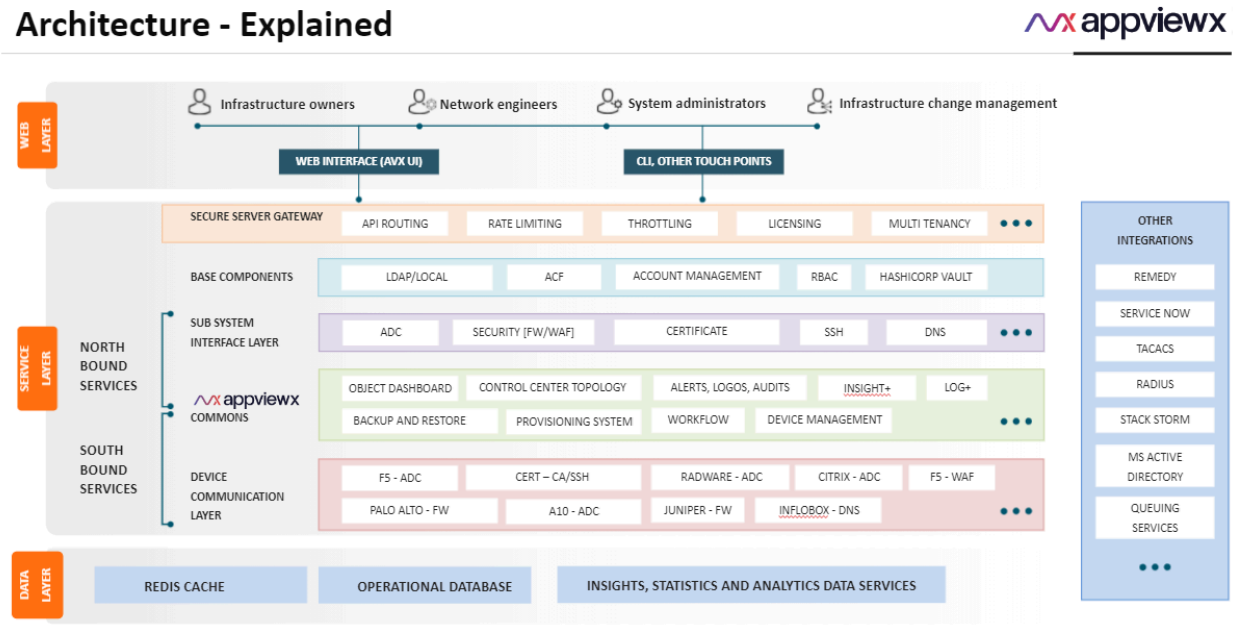
What's New

This section provides information about the features and the enhancements in 22.0. The features are described in detail in the Release Notes. For more information, refer to the Release Notes.

AppViewX Architecture

AppViewX is built on Kubernetes, an open-source platform for deploying and managing containers. It provides a container runtime, container orchestration, self-healing mechanisms, service discovery and load balancing. It's used for the deployment, scaling, management, and composition of application containers across clusters of hosts.

AppViewX is designed based on microservice architecture making it easier to move to containerized workloads and the containers being orchestrated using Kubernetes. The following diagram depicts the deployment architecture:



In the diagram:

- **Presentation/ Web Layer** - houses the AppViewX user interface related files and interacts with the service layer
- **Service Layer** - contains the Northbound & Southbound services that can be further classified into:

- **Business Layer**:
  - Houses AppViewX specific business logic
  - Interacts with the Data layer for persisting the input data
- **Device Communication Layer**:
  - Low code
  - Stateless layer
  - Routes communication to the respective vendor through APIs or SSH
  - Houses vendor specific business logic
- **Data Layer**:
  - Houses data persistence and retrieval logic
  - Redis caching is available

Benefits of AppViewX

In order to optimally utilize the resources, AppViewX has adopted Kubernetes to achieve higher security by adopting a zero trust network model. The features of AppViewX coupled with Kubernetes are given below.

- **Auto scaling**

AppViewX services can have a custom throttling capability based on pre-configured memory configuration per API. This will enable AppViewX services to utilize (scale up) resources optimally as the demand surges and scale down when not in use. This will help to horizontally scale the vendor components on demand and optimize the resource usage.

- **Resiliency**

There is no guarantee that the services will run without any interruption and they are bound to failure. Kubernetes keeps deployments healthy by restarting containers that have failed, killing and replacing unresponsive containers based on health checks. This helps to mitigate the common pain point of the application upkeep process.

- **Security**

AppViewX architecture is designed around the concept of zero trust network model to enforce tighter security within the Kubernetes cluster. This means no one is trusted by default and required verification to gain access to the services.

Supported Deployment Methods and Types

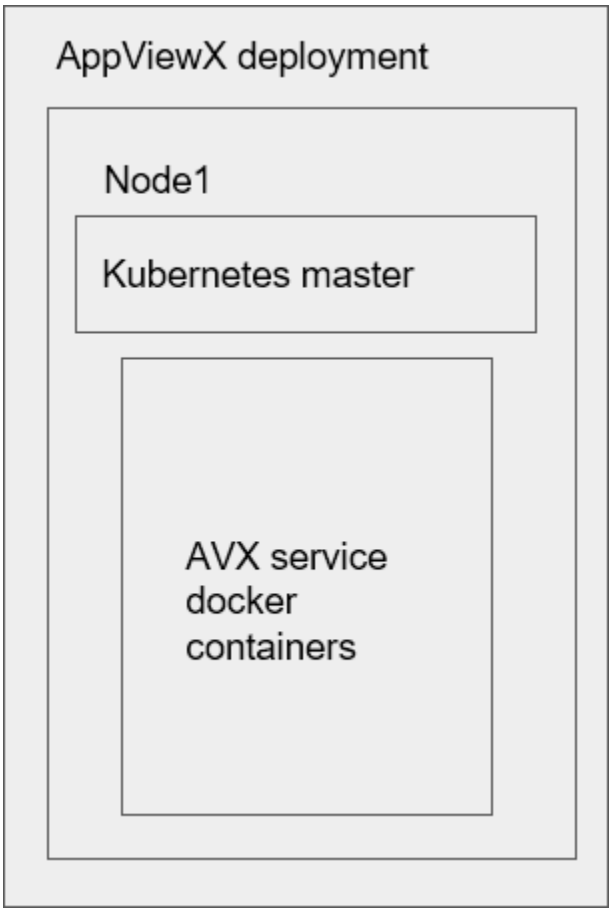
This section explains the types and methods in which you can deploy AppViewX.

> ⚠ **Warning:** Hybrid cloud management deployment is not supported in AppViewX.

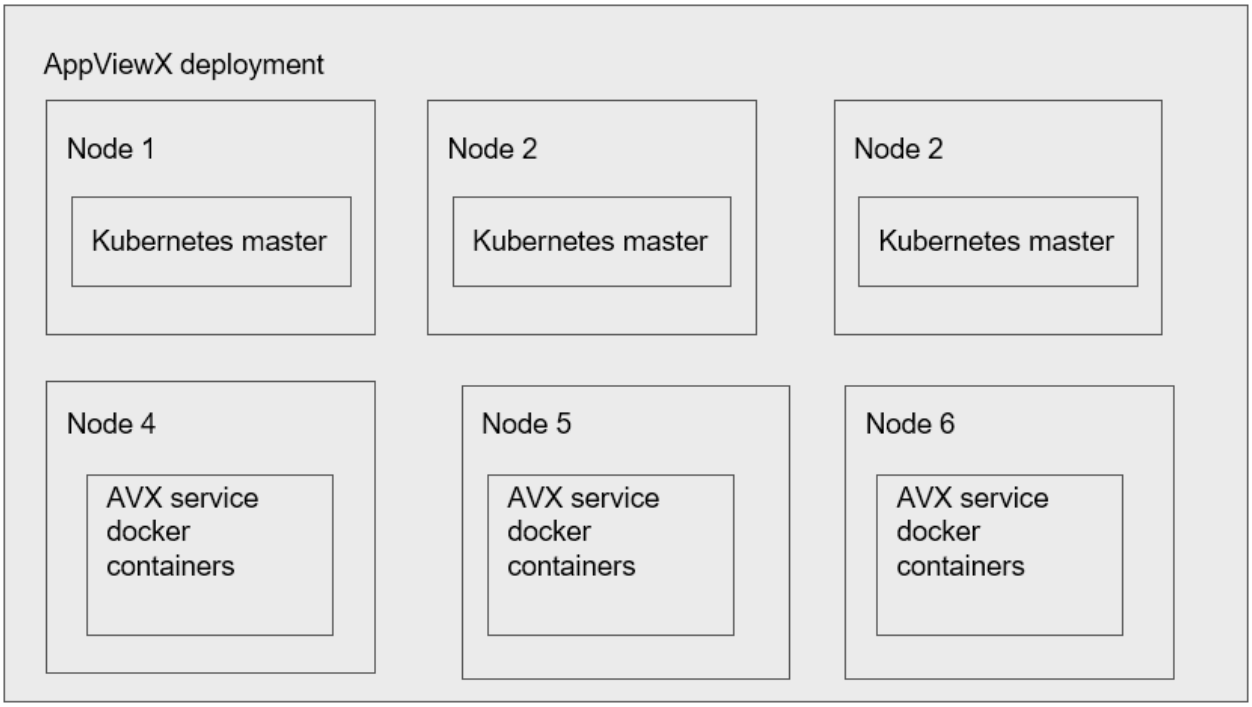
First, AppViewX can be deployed in the following modes:

- **Single Node** - is used to host all the services on a single setup.
  - Single-node setups may have lower performance because of a lack of resources.
  - Node resiliency and HA are not supported in single-node deployment.
- **Multi node** - is used to host the services across multiple nodes to ensure high availability.

The following diagrams depict AppViewX deployment on a single node and a multi node mode:



Single Node Deployment



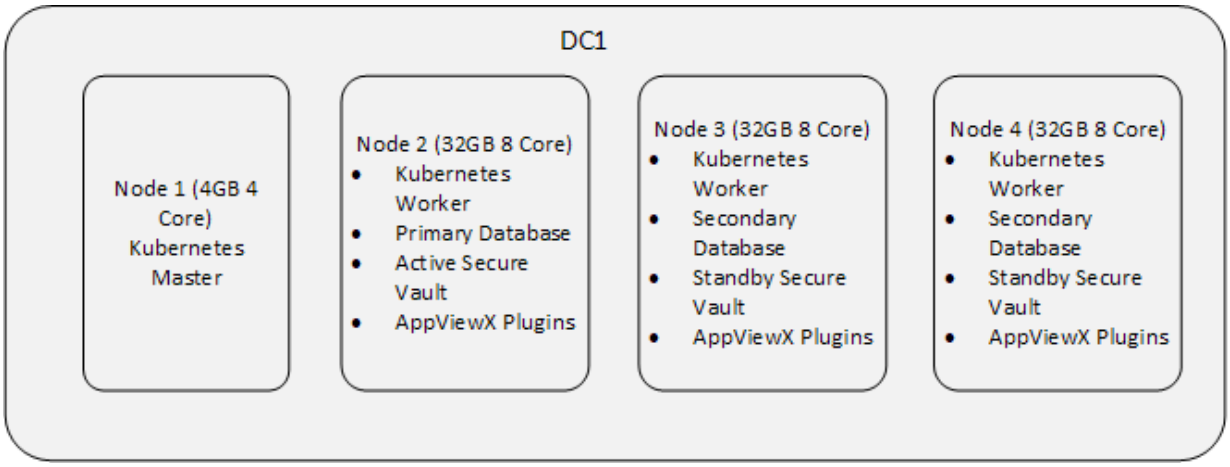
Multi Node Deployment

Once the deployment mode is finalized, AppViewx can be installed using any one of the following methods:

- **OVA Installation** - stands for Open Virtual Appliance that contains a compressed and installable version of a virtual machine. When you use an OVA-based installer, the installation-related artifacts are pre-bundled as part of the OVA.
- **Native Installation** - uses the standard command line interface to execute installation commands.

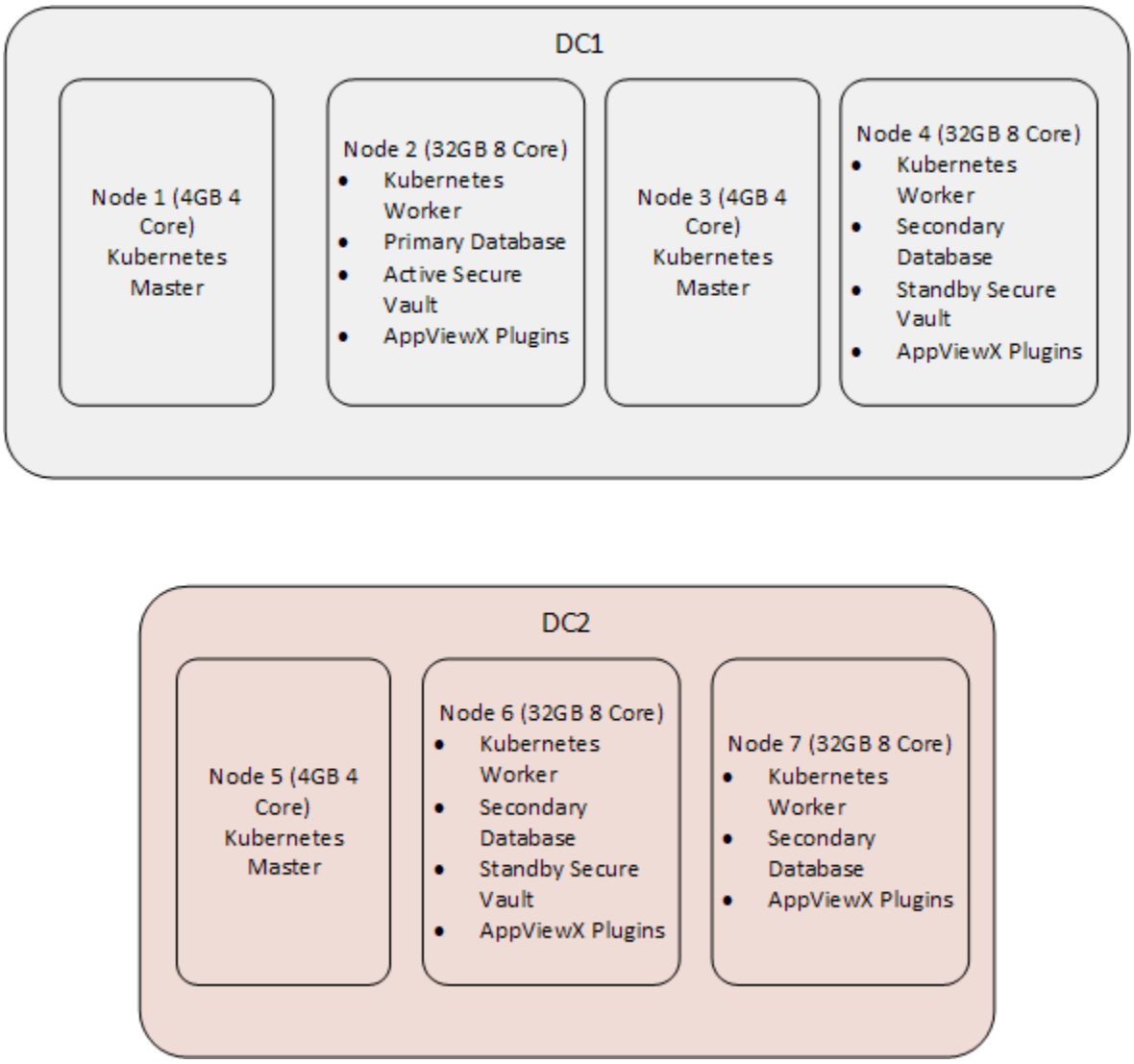
AppViewX supports the following deployment types/scenarios:

• One Data Center and Four Nodes



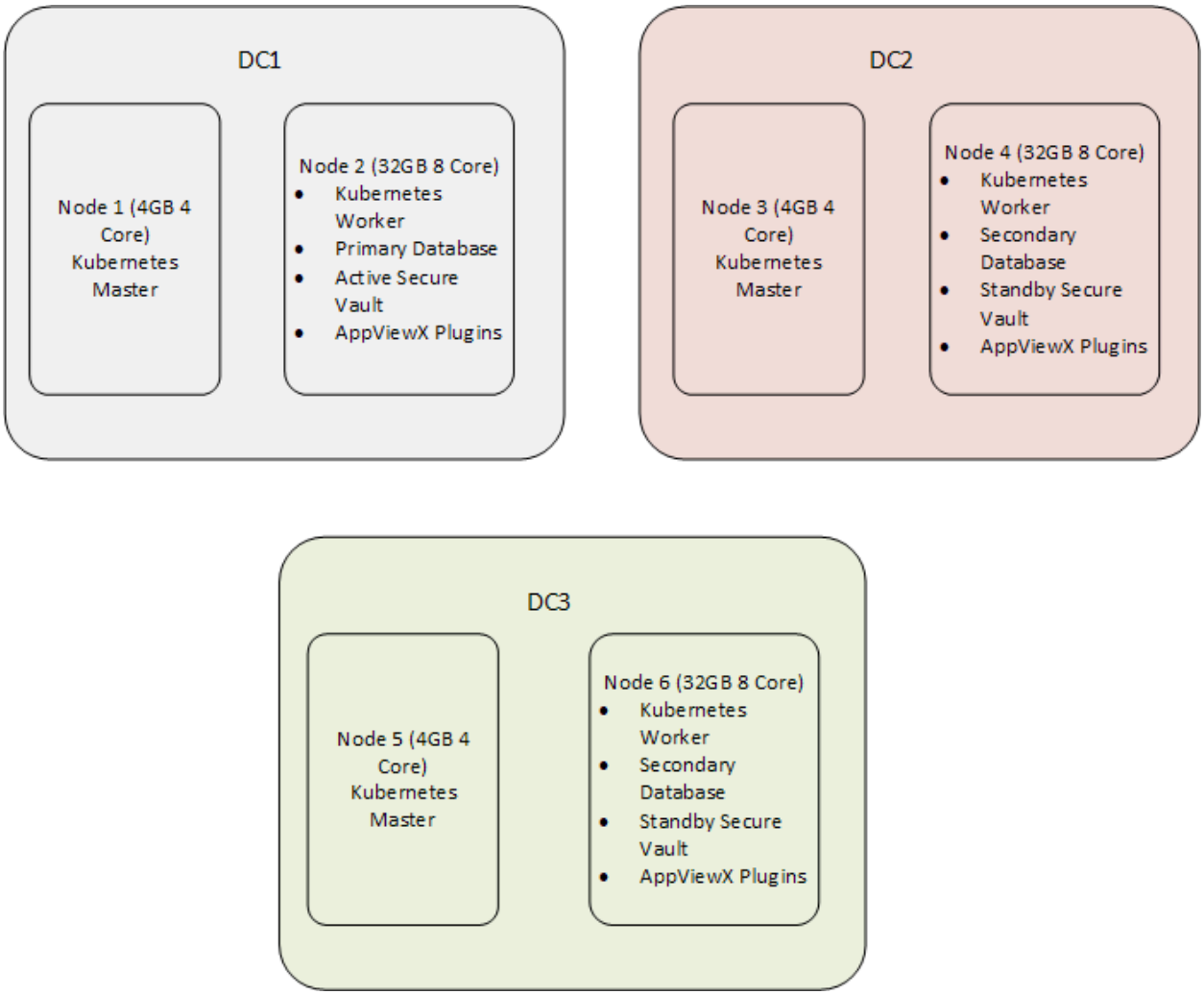
This deployment is recommended for customers who need only HA at the application level. This deployment does not support HA; neither for the Kube nor for the DC. This deployment is best suited for less than 50 ADC devices having a total of 100,000 objects and 10,000 certificates.

• Two Data Centers and Seven Nodes



This deployment is recommended for customers who require HA at the Application, Kube, and DC level. This deployment supports HA for the Kube, Application as well as the DC. This deployment is best suited for 50 to 100 ADC devices having a total of 300,000 objects and 10,000 certificates.

• Three Data Centers and Six Nodes



This deployment is recommended for customers who require HA at the Application, Kube, and DC level. This deployment supports HA for the Kube, Application as well as the DC. This deployment is best suited for 50 to 100 ADC devices having a total of 300,000 objects and 10,000 certificates.

The table below summarizes the different deployments supported by AppViewX.

Model	**Load**	**HA**		
		Kube	**DC**	**Application**
1 DC 4N	Less than 50 ADC devices having a total of 100,000 objects and 10,000 certificates.	No	No	Yes
2 DC 7N	50 to 100 ADC devices having a total of 300,000 objects and 10,000 certificates.	Yes	Yes	Yes

3 DC 6N	50 to 100 ADC devices having a total of 300,000 objects and 10,000 certificates.	Yes	Yes	Yes

Note: Apart from the deployments mentioned here, AppViewX can customize the deployment based on the needs and requirements.

Understanding the Installation Steps

This section outlines the various mandatory and optional steps in the process of installing AppViewX.

Step No	Step Name	Mandatory	Optional
1	Working with Prerequisites	Yes	No
2	Configuring the Firewall Ports	Yes	No
3	Configuring Elevated Access	Yes	No
4	Downloading Linux Packages	Yes	No
5	Downloading AppViewX Packages	Yes	No
6	Running the Prerequisite Tool	Yes	No
7	Deploying the AppViewX Virtual Appliance	No	Yes
8	Performing a Single Node or Standalone Installation	No	Yes
9	Performing a Multi-node or High Availability Installation	No	Yes
10	Configuring the Cluster	No	Yes
11	Configuring the POD and Service CIDR	No	Yes
12	Verifying the Installation	Yes	No
13	Uploading the License Key	Yes	No
14	Accessing the AppViewX Graphical User Interface	Yes	No
15	Adding Third-party Libraries	No	Yes

Chapter 2: Working with Prerequisites

Understanding Requirements

Understanding Hardware Requirements

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

• Single Node Requirements

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

• Multi Node Requirements

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

Note: For more information on the nodes, refer to the Supported Deployment Methods and Types section.

For deploying the OVA, ensure that you have all the prerequisites as mentioned below.

• Platform Bare Minimum Requirements

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

Understanding Software Requirements

- CentOS 7.X
- RHEL 7.X
- RHEL 8.5
- RHEL 8.8
- Ubuntu 20.04

Configuring Elevated Access

AppViewX is installed on top of a kubernetes engine and to install the underlying Kubernetes engine and other dependent packages like docker, we would require the user to have sudo access and executable permission for the tmp folder. Refer to the Understanding Commands Executed during Installation section to get the details on the commands that the sudo user needs access to.

Note: If you are using an OVA-based installer, a user named appviewx is already available with Super user privileges.

Configuring Firewall Ports

The following ports must be opened between the nodes to install AppViewX. Users can configure it in a firewall device, firewalld, or using iptables.

S.No	Source		Destination		Protocol Used	TCP/ UDP	Type of Information Communicated
	IP	Port	IP	Port			
1	All Nodes	Any	All Nodes*	22	SSH	TCP	Required for AppViewX installation and prerequisite checks.
2	All Nodes	Any	All Nodes*	179	BGP	TCP	To establish a common routing table for the overlay network.
3	All Nodes	Any	All Nodes*	6443	HTTPS	TCP	Kubernetes API server for communication between Kubernetes master and worker nodes.
4	All Nodes	Any	All Nodes*	10250	HTTPS	TCP	Used by Kubelet Agent which exposes Rest endpoints for the Kubernetes API Server.

S.No	Source		Destination		Protocol Used	TCP/ UDP	Type of Information Communicated
5	All Nodes	Any	All Nodes*	4243	HTTP	TCP	Required during installation and scaling up. Used to load docker images when spinning up a new container. Triggered by the node where the install process is started.No sensitive data is being transferred through this port.
6	Load Balancer (for ex, F5, GCP, etc.)	Any	ISTIO Ingress Proxy IP (Kube Worker)	31443	HTTPS	TCP	To access the AppViewX web user interface.
7	Load Balancer (for ex, F5, GCP, etc.)	Any	Kube Master IP	6443		TCP	To allow communication between the F5 load balancer and the pool members (master nodes).
8	All Nodes	Any	F5 VIP	6443		TCP	To allow all the nodes to communicate with the Kube Master for Kubernetes Control plane traffic.
9	AppViewX Admin network #	Any	ISTIO Ingress Proxy IP (Kube Worker)	30190	HTTPS	TCP	To access the AppViewX management console.
10	All Nodes	-	All Nodes*	-	IP-IP	NA	Overlay network established with IP-IP tunnels. Information over this tunnel is encrypted using mTLS.

S.No	Source		Destination		Protocol Used	TCP/ UDP	Type of Information Communicated
					IP Protocol 4		
11	Master	Any	Kube Master	2379	HTTPS	TCP	Required for etcd server communication in a multi-master setup.
12	Master	Any	Kube Master	2380	HTTPS	TCP	Required for etcd server communication in a multi-master setup.
13	All Nodes	Any	All Nodes*	9100	HTTP	TCP	Required for monitoring the node metrics.

* - indicates all the nodes present in the cluster ie. master nodes, secondary master nodes, and worker nodes.

- indicates the network/machines/nodes of users who want to manage AppViewX Infra using the management console (actions include create, delete pods, and/or services.

Note:

- IPs required - The system will require 1 IP per node.
- The externally exposed services will all use the nodes IP address to communicate within the network.
- Port 22 is used for administration of the node for example to log into the linux CLI. Need SSH access the nodes to other nodes.
- We would need an external Load Balancer to distribute user/API traffic to all Kube master nodes. We can open firewall ports depending on the network setup.

Note: Ensure that the external endpoints that you want to access from the AppViewX worker nodes are accessible. For example, Microsoft CA. Ensure that the corresponding ports and URLs are opened for communication.

Configuring Firewall Ports for External Integrations

S.No	Source		Destination		Protocol Used	TCP/ UDP	Type of Information Communicated
	IP	**Port**	**IP**	**Port**			
1	AppViewX Worker Nodes	Any	ADC		SSH		
2	AppViewX Worker Nodes	Any	ADC		HTTPS		To execute REST APIs
3	AppViewX Worker Nodes	Any	MSCA Agent		HTTPS		AppViewX to MSCA agent communication
4	AppViewX Worker Nodes	Any	CA		HTTPS		To execute REST APIs

Configuring YUM

This section guides users to configure AppViewX (CENTOS 7.x) nodes to the YUM repository hosted by AppViewX. Yum will sync only AppViewX repositories to get the OS package updates. This task is required to update the OS security patching on Appviewx supplied OVAs.

Note: For information regarding the best practices on rebooting the operating system after security patching, refer to the Understanding the Best Practices on Reboot Sequence.

Warning: This will remove all the other repositories configured in the system.

Before you configure yum, ensure that:

1. AppViewX nodes have access to the following URL https://repos.appviewx.com
2. The user has root/sudo access to configure yum.

To configure YUM:

1. Download the `appviewx.repo` file from the release portal.
2. Login as a root user.
3. To take a backup of existing yum repositories, execute the following command:

   ```
   mv /etc/yum.repos.d /etc/yum.repos.d_backup
   ```

   This is to ensure that we have a backup of the existing yum repository configurations.
4. To create a yum repository, execute the following command:

   ```
   mkdir -p /etc/yum.repos.d
   ```

5. To copy the appviewx.repo to yum.repos.d, execute the following command:

   ```
   cp appviewx.repo /etc/yum.repos.d/
   ```

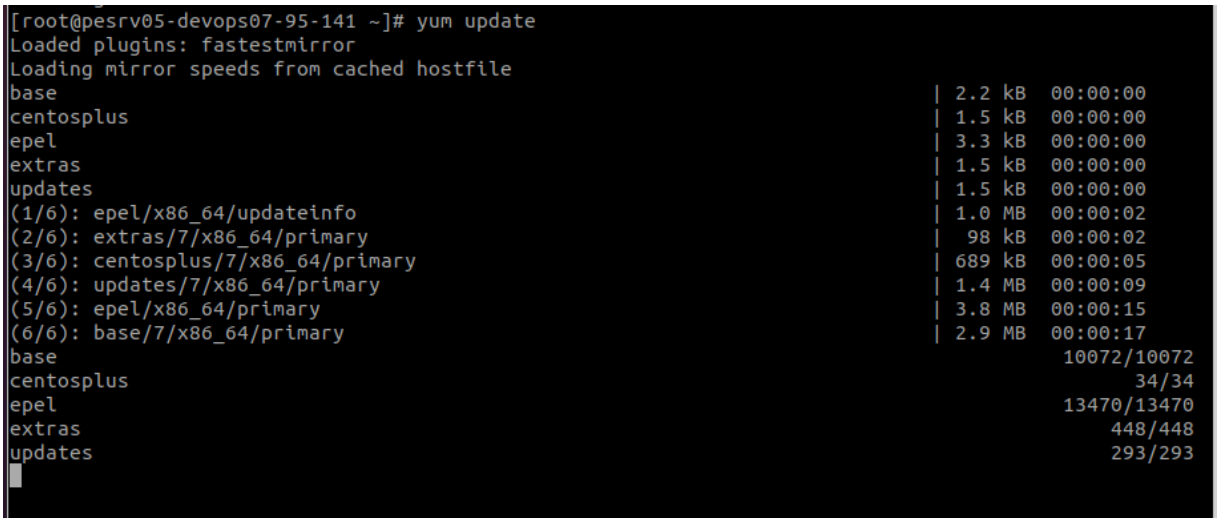
6. To clean the yum repository, execute the following command:

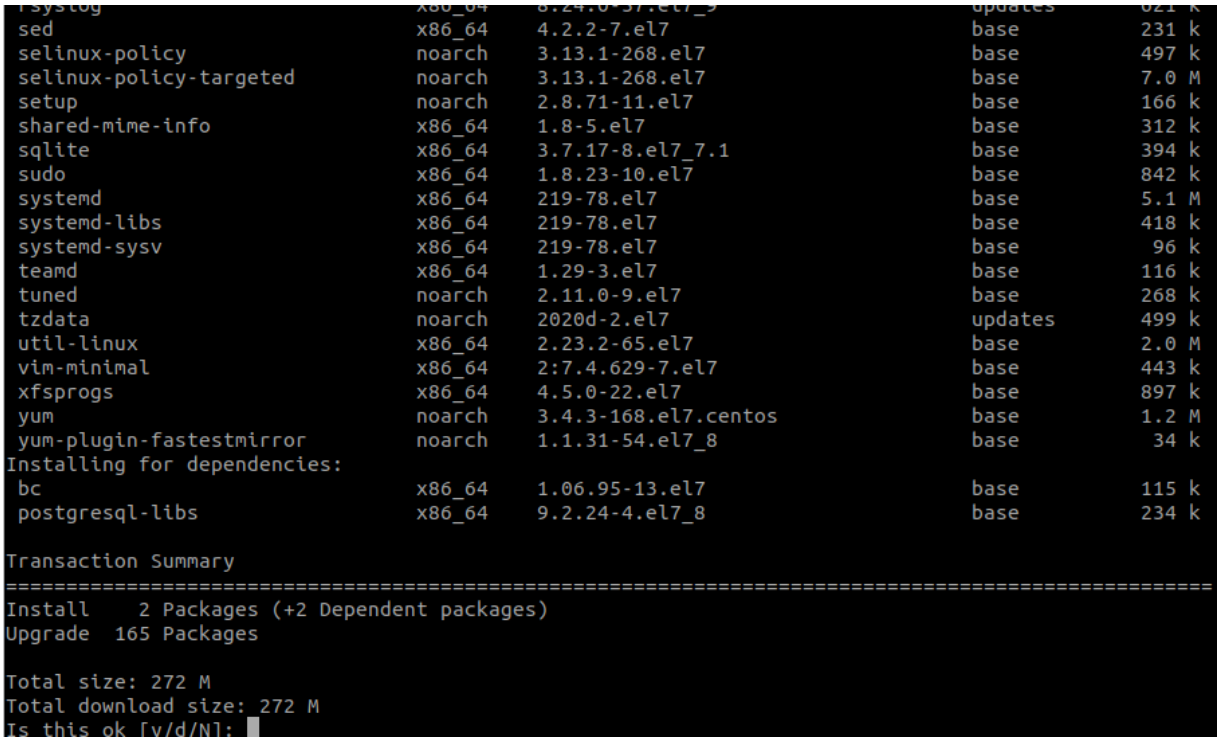
   ```
   yum clean all
   ```

7. To get the latest updates from repos.appviewx.com, execute the following command:

   ```
   yum update
   ```

   The command will connect to the AppViewX repository and update the packages. Reference images are given below:





Configuring Calico before Deployment

This section provides instructions on configuring calico before deploying AppViewX on Azure.

Warning: Follow these instructions ONLY if you are deploying AppViewX on Azure.

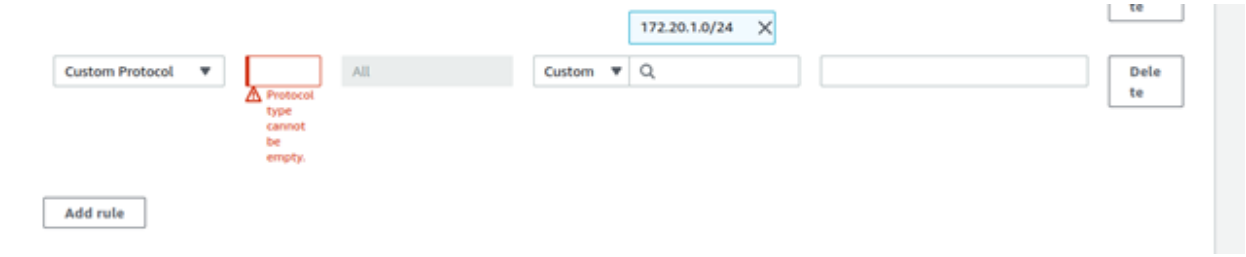
1. Navigate to the `/home/appviewx/appviewx_kubernetes/configs/kube` directory.
2. Open the `calico.yaml` file in edit mode.
3. Change the value of the `CALICO_IPV4POOL_VXLAN` parameter from `CrossSubnet` to `Always`.
4. Change the value of the `CALICO_IPV4POOL_IPIP` parameter from `Always` to `Never`.
5. Save the changes to the `calico.yaml` file.
6. Close the editor.

Enabling the IP in IP Protocol

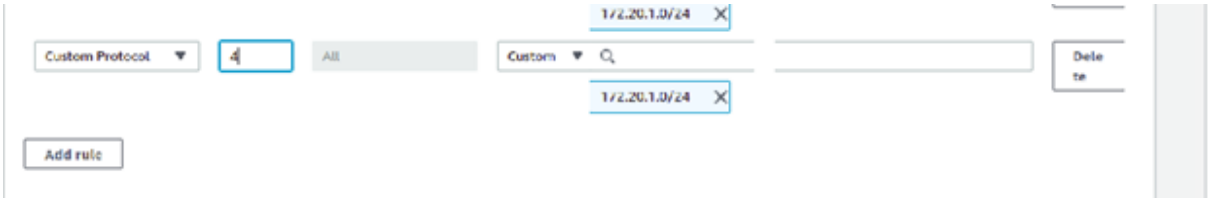
> **Warning:** Follow these steps ONLY if you want to deploy AppViewX on AWS.

You must enable the IP in IP protocol between the nodes in the AWS security group before deploying AppViewX.

1. Log in to the AWS console.
2. Navigate to the security group that needs to be modified.
3. Click **Edit inbound rules**.
4. Click **Add rule**.
5. From the **Add rule** list, select **Custom Protocol**.

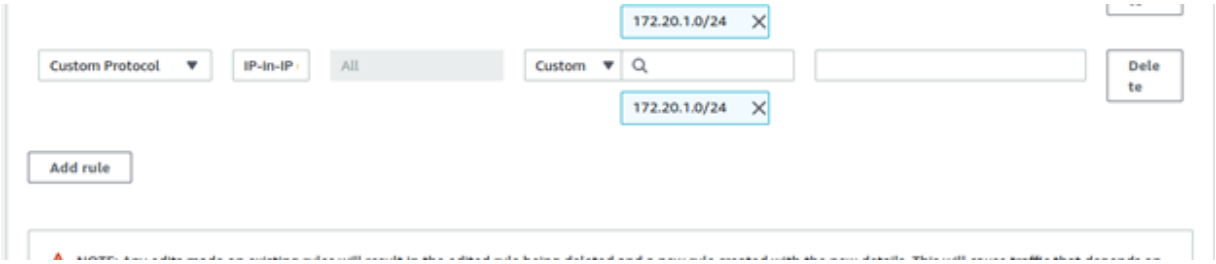


6. Enter the protocol value as **4**.



7. Enter the subnet across which IP in IP needs to be enabled.
8. Click **Save rule**.

The protocol automatically changes to IP in IP.



Downloading AppViewX Packages

To install AppViewX, download the following packages from the AppViewX Release Portal.

Note: To get the release portal credentials, contact help@appviewx.com.

File Name	Mandatory	Description	Purpose
appviewx_kubernetes_2022.1.0.tar.gz	Yes	AppViewX core installer	Core installer that has the AppViewX package from which the installation is triggered.
appviewx_kubernetes_addons_2022.1.0.tar.gz	Yes	To install AppViewX addons	Additional software to support the functionalities of AppViewX. This is mandatory for the installation.
appviewx_kubernetes_elk_2022.1.0.tar.gz	Optional	ELK stack to monitor logs	Additional package to install a GUI-based log collector to troubleshoot and Graffana-based UI to monitor the application performance.
appviewx_kubernetes_insight_2022.1.0.tar.gz	Optional	Insight for AppViewX Insight module	The insight package is an additional package to enable AppViewX to collect the statistical information of devices managed by

File Name	Mandatory	Description	Purpose
			AppViewX and generate it as a report.
upgrade.tar.gz		To upgrade from the existing version	This package is required to upgrade from older versions of AppViewX to 2021.1.0.
prerequisite_utils.tar.gz		To check whether all the components are available.	The tool checks whether all the required prerequisites are present on the system.

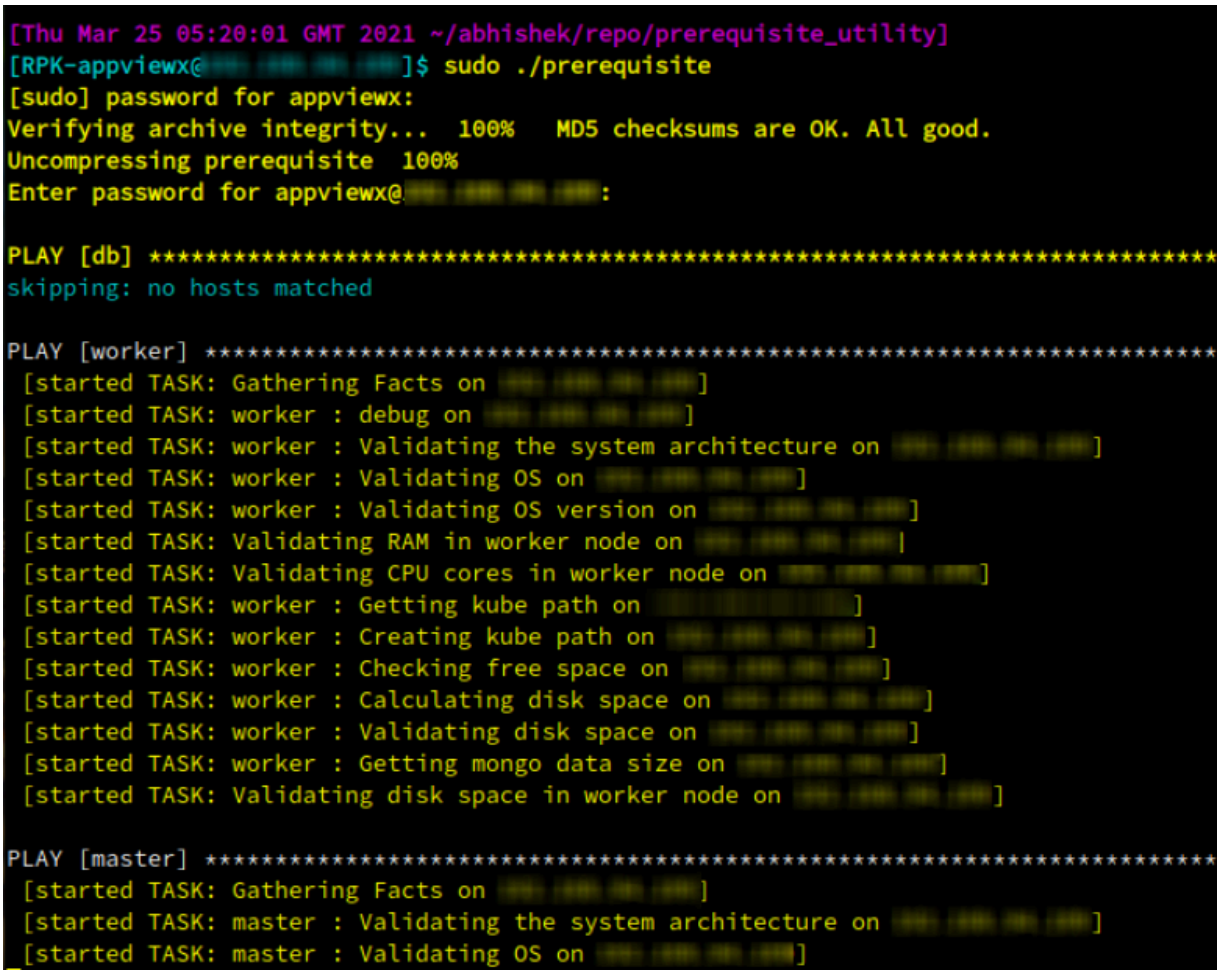
Note: All OVA related updates are maintained by AppViewX and are available on the release site.

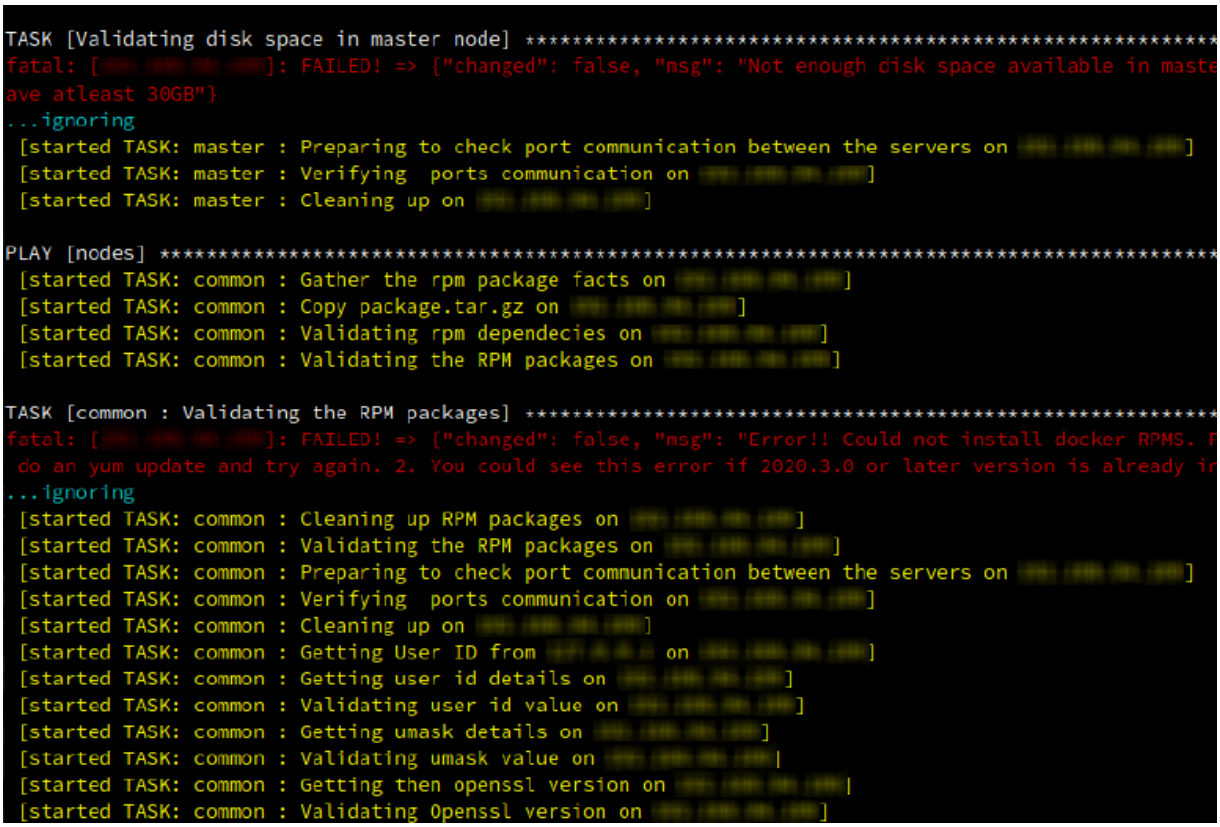
Running the Prerequisite Tool

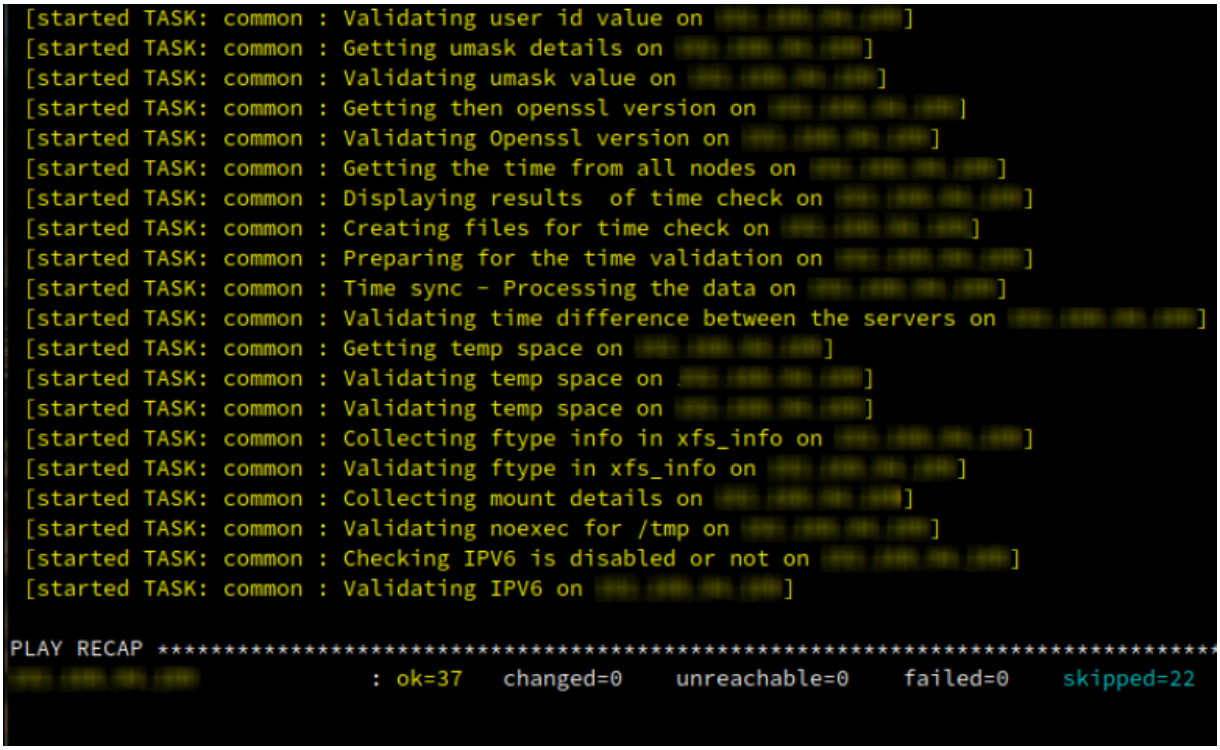
The prerequisite tool checks whether all the required prerequisites are present on the system. Sudo permissions are required to execute the tool. This utility can be executed from any of the nodes; either worker or master. The prerequisites are available at https://github.com/AppViewX/prerequisite_utility/.

To run the prerequisite tool:

1. Download and extract the prerequisite_utils.tar.gz file.
2. Copy the updated appviewx.conf file to the location where you have extracted the contents of the prerequisite.tar.gz file.
3. Specify the appviewx IP address of the VMs (master and worker nodes), DNS servers and gateway address, and users in the hosts_template file.
4. Execute the following command: `sudo ./prerequisite`







The command displays the failures in red text and at the end displays a summary of the tasks.

Chapter 3: Deploying the AppViewX Virtual Appliance

Download the Release Package

This section covers the procedures for downloading the release package.

To download the release package,

1. Visit the AppViewX download URL at https://release.appviewx.com.
2. Download the release package in <.ova> format into the Downloads folder or the Desktop in your environment.
3. Validate the md5sum of the downloaded file
4. Open a terminal window.
5. To display the md5sum value of the downloaded file, execute the command:
   a. To display the md5sum value of the downloaded file, execute the following command: `md5sum <filename>`

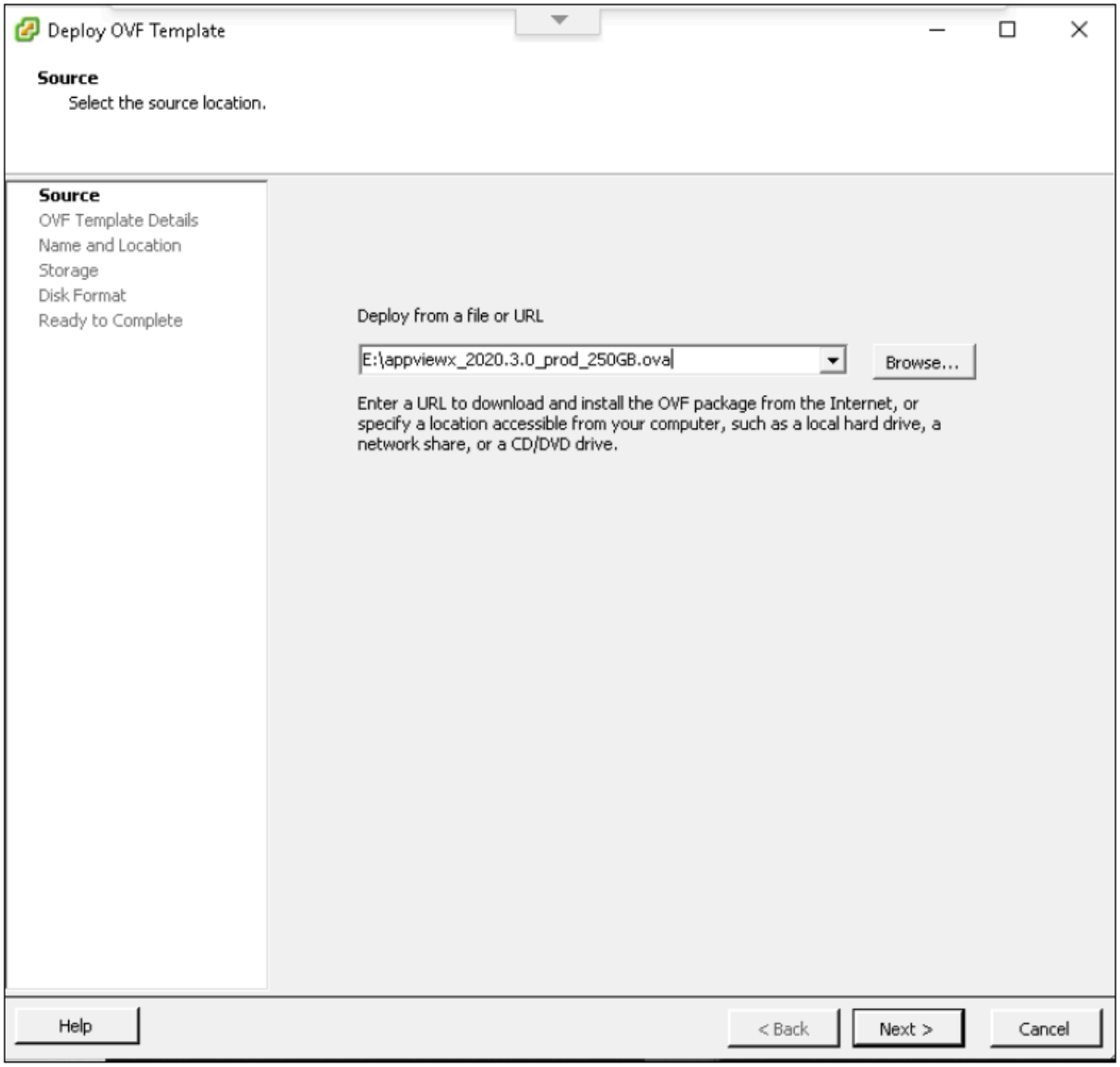
   b. Match the displayed value against the original value from the release portal.

Install the AppViewX OVA

This section covers the procedures for installing the AppViewX OVA.

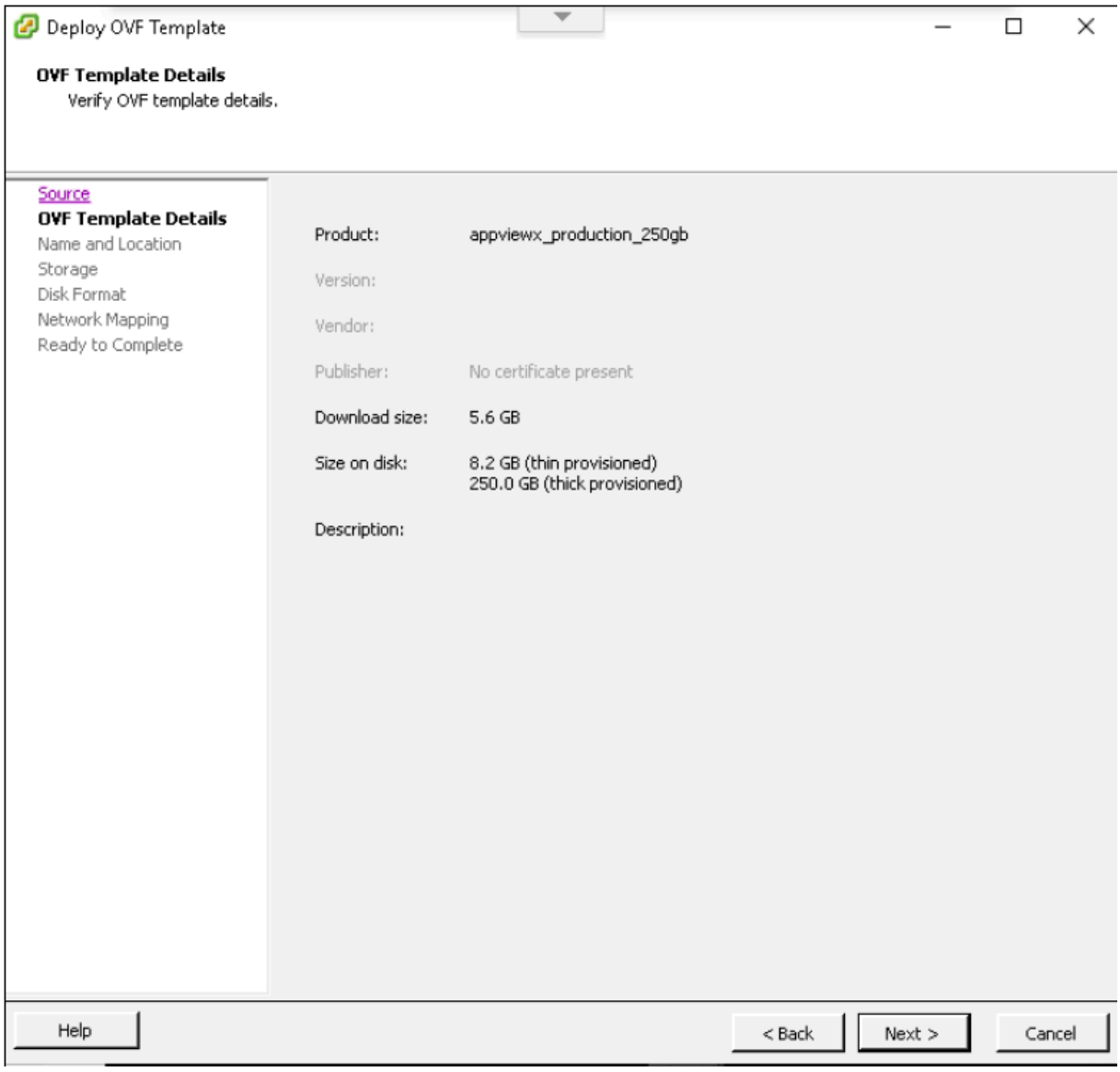
To install the AppViewX OVA,

1. Log in to the vSphere Client.
2. Select **File > Deploy OVF Template**.
   The **Name and Location** screen is displayed.



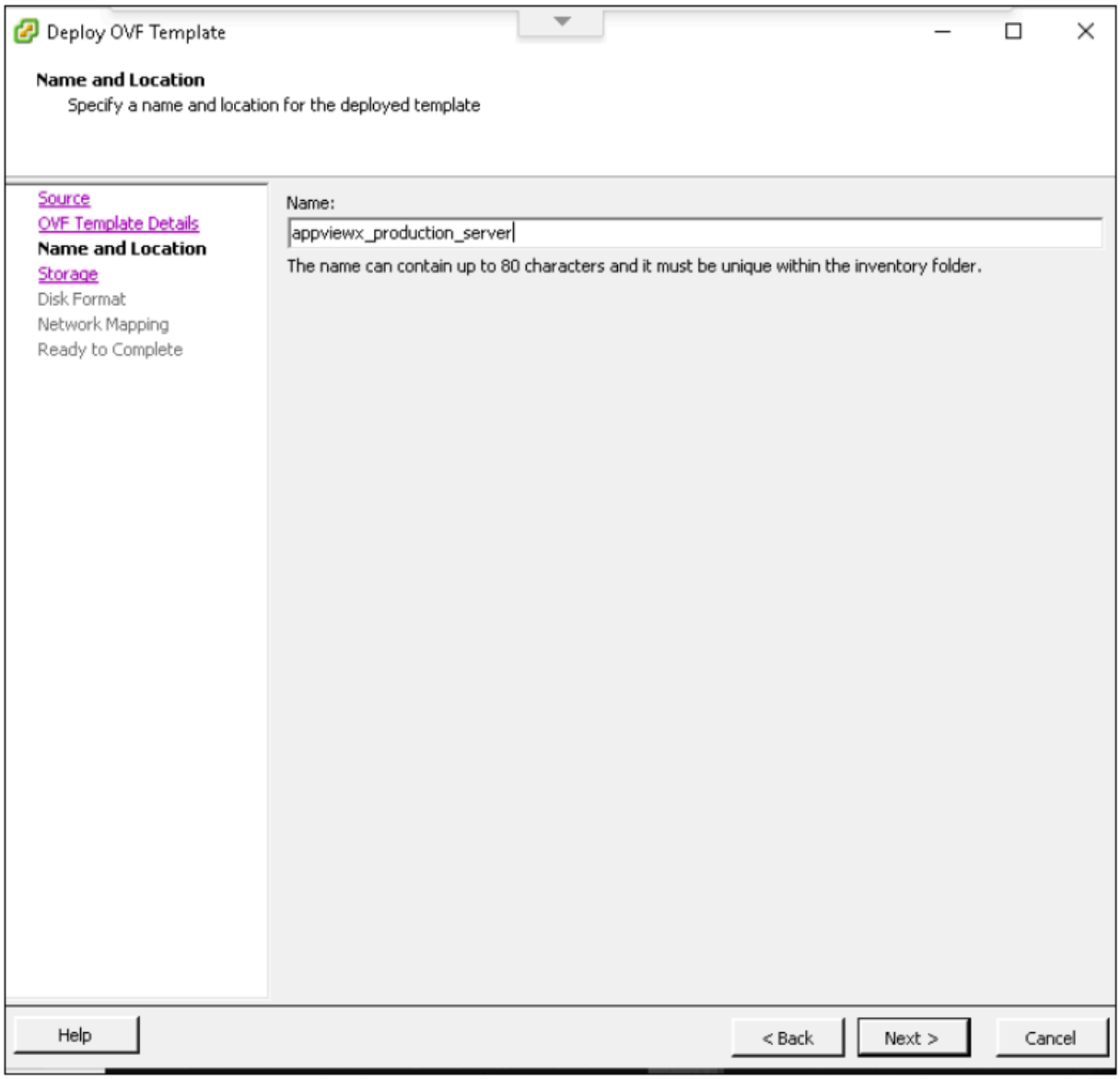
3. Click **Next**.

   The **Source** screen of the Deploy OVF Template wizard is displayed.



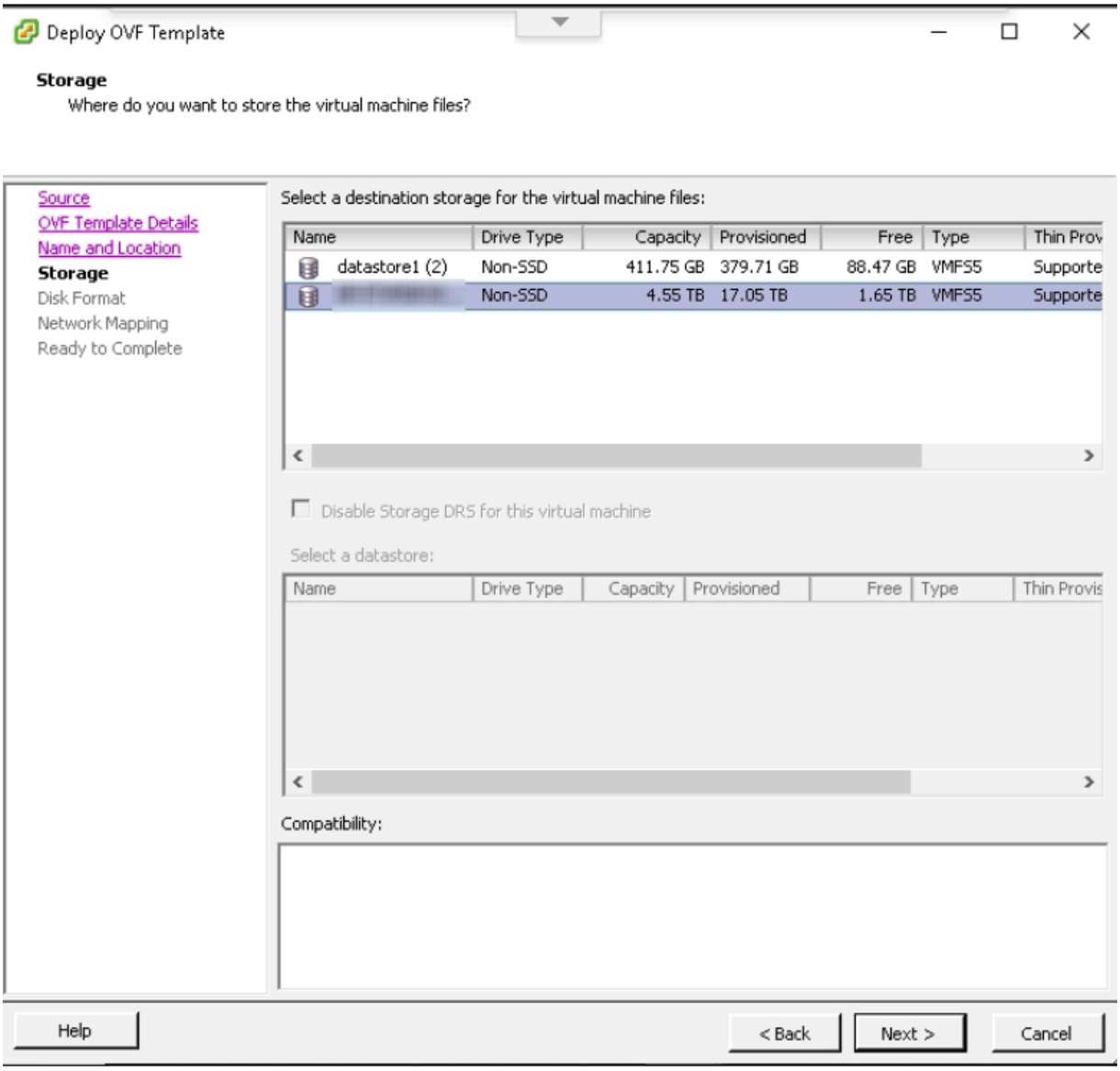
4. Click **Next**.

   Name and Location screen is displayed.



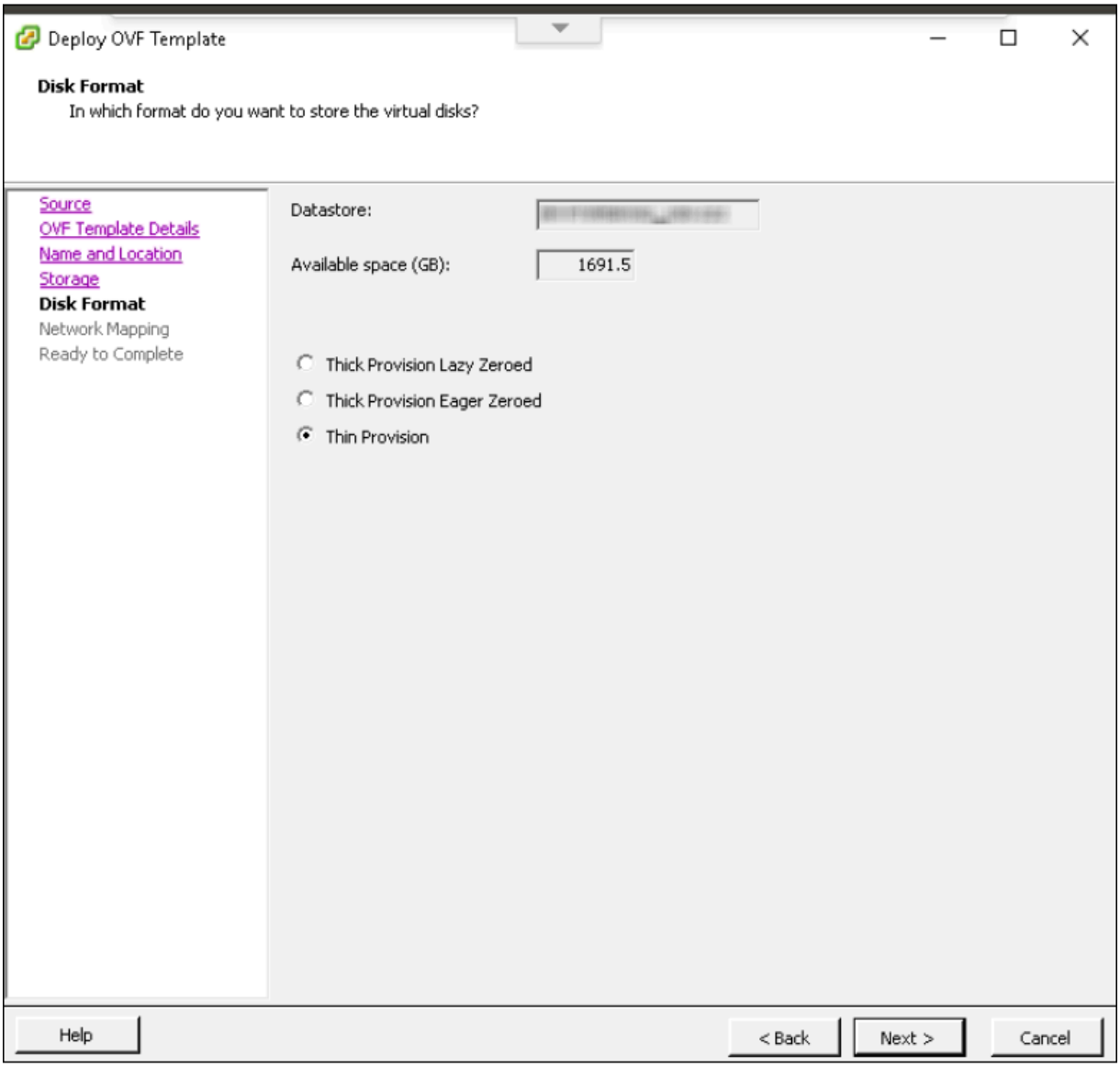
5. (Optional) In the **Name and Location** screen, change the server name to display.
6. Click **Next**.

   The **Storage** screen is displayed.



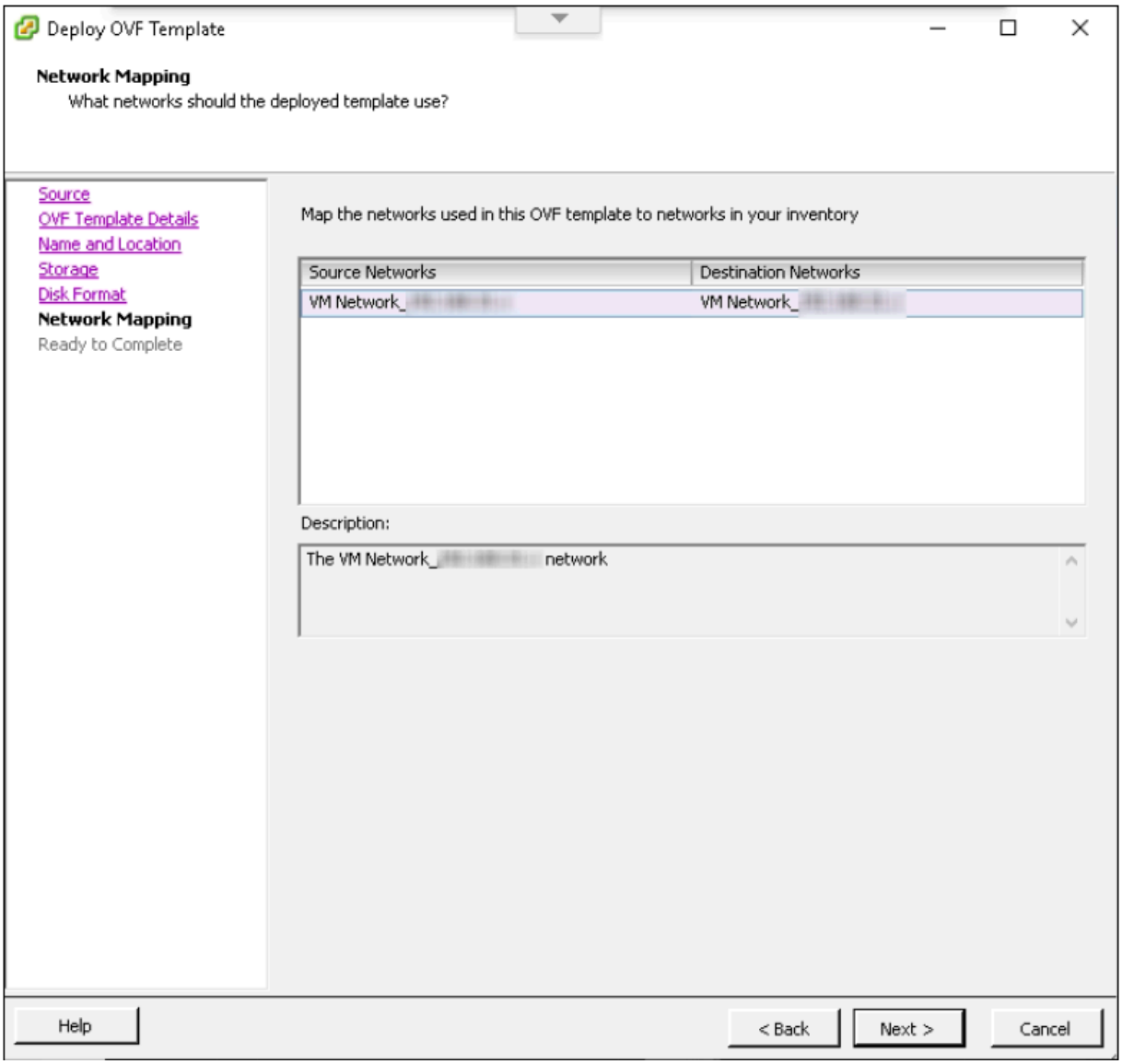
7. In the **Storage** screen, select a destination storage for the VM files.
8. Click **Next**.

   The **Disk Format** screen is displayed.



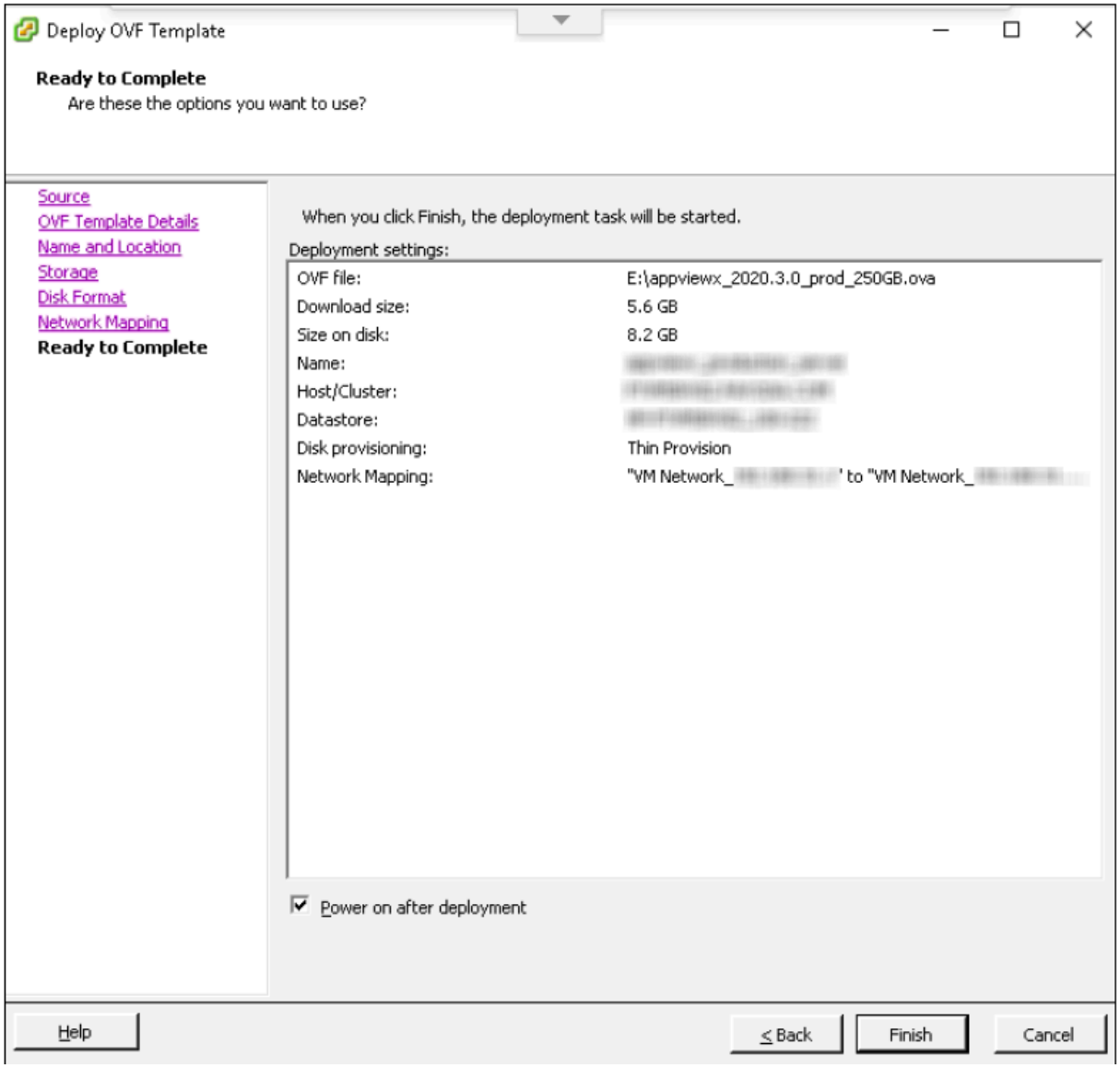
9. In the **Disk Format** screen, select a disk type.
10. Click **Next**.

    The **Network Mapping** screen is displayed.



11. In the **Network Mapping** screen, choose a network adapter.
12. Click **Next**.

    The **Ready to Complete** screen is displayed.



13. In the **Ready to Complete** screen, verify all the details.
14. Click **Finish**.
15. After the deployment, access the AppViewX VM console.

> **Note:** Contact the account rep for the root credentials as well as the credentials to access AppViewX.

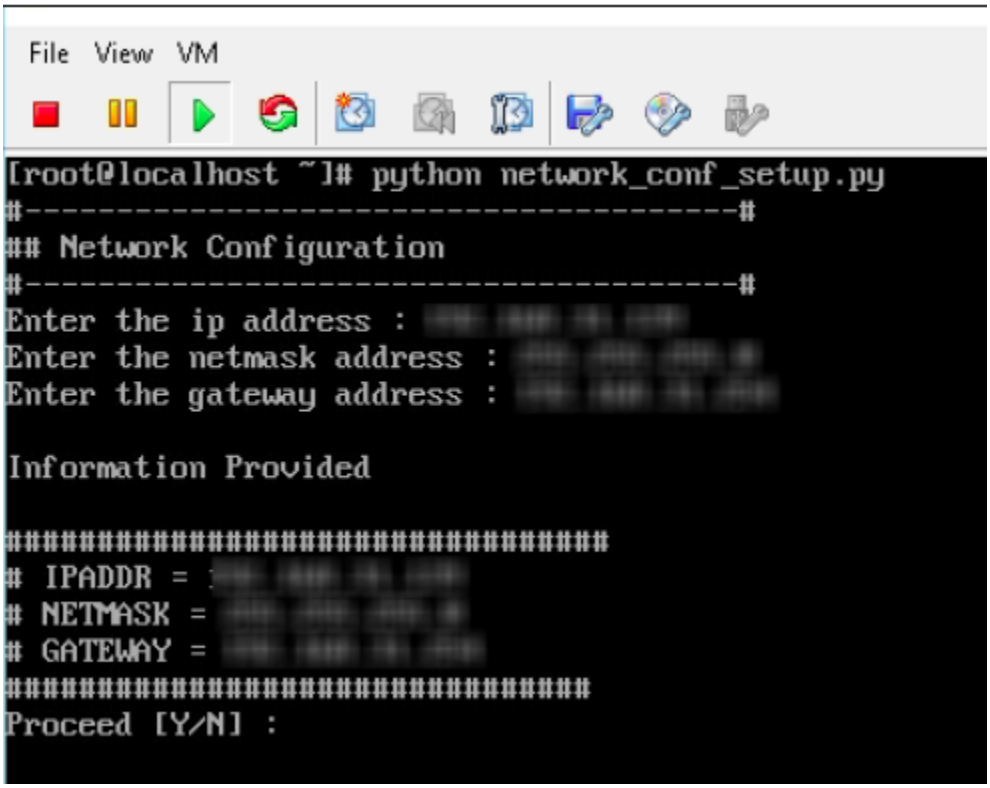
16. To navigate to the root folder as a root user, execute the command:`$ cd /root`

```
[root@pesrv03-regression02-98-13 ~]# cd /root/
[root@pesrv03-regression02-98-13 ~]# pwd
/root
```

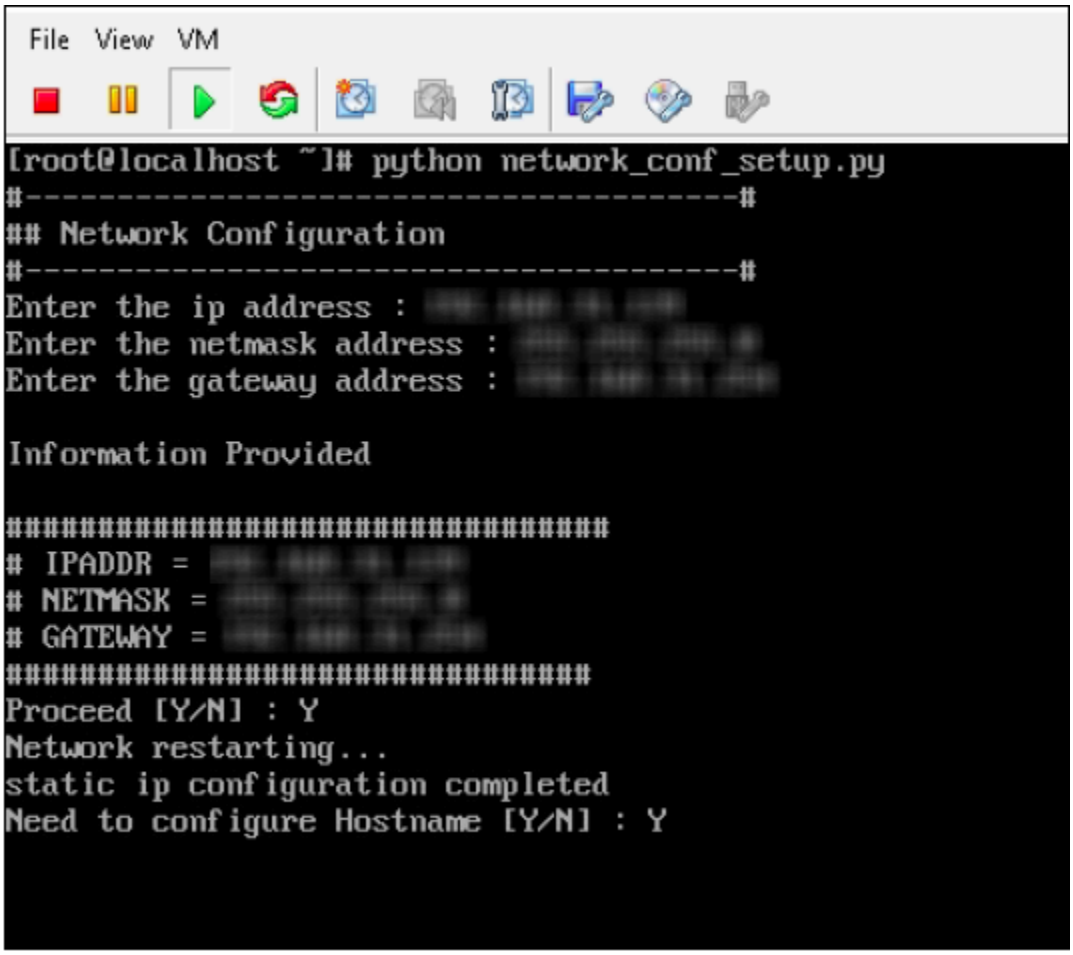
17. Execute the following command: `$ python network_conf_setup.py`

    The command starts the console and prompts you to enter the network configuration for the node.
18. Enter the following details at the prompt:
    - IP address
    - Netmask address
    - Gateway address

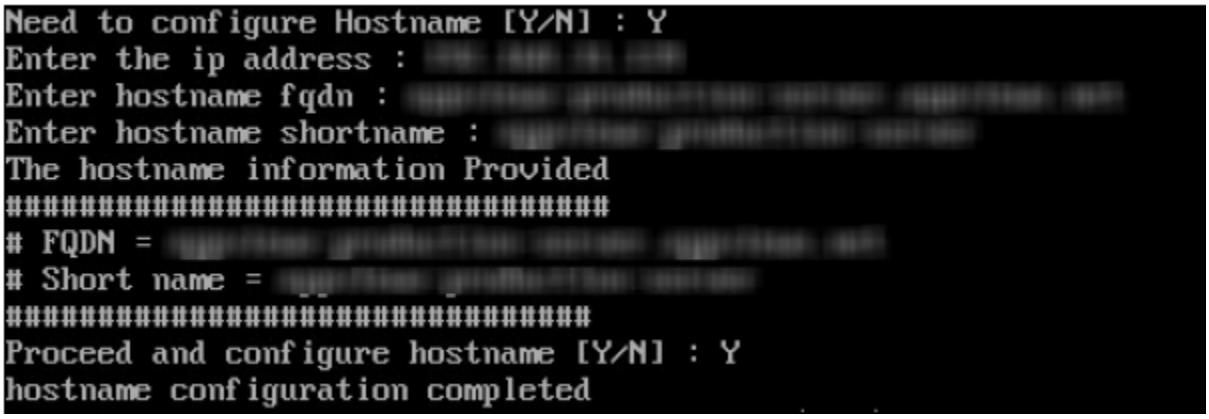
    A prompt to proceed is displayed.



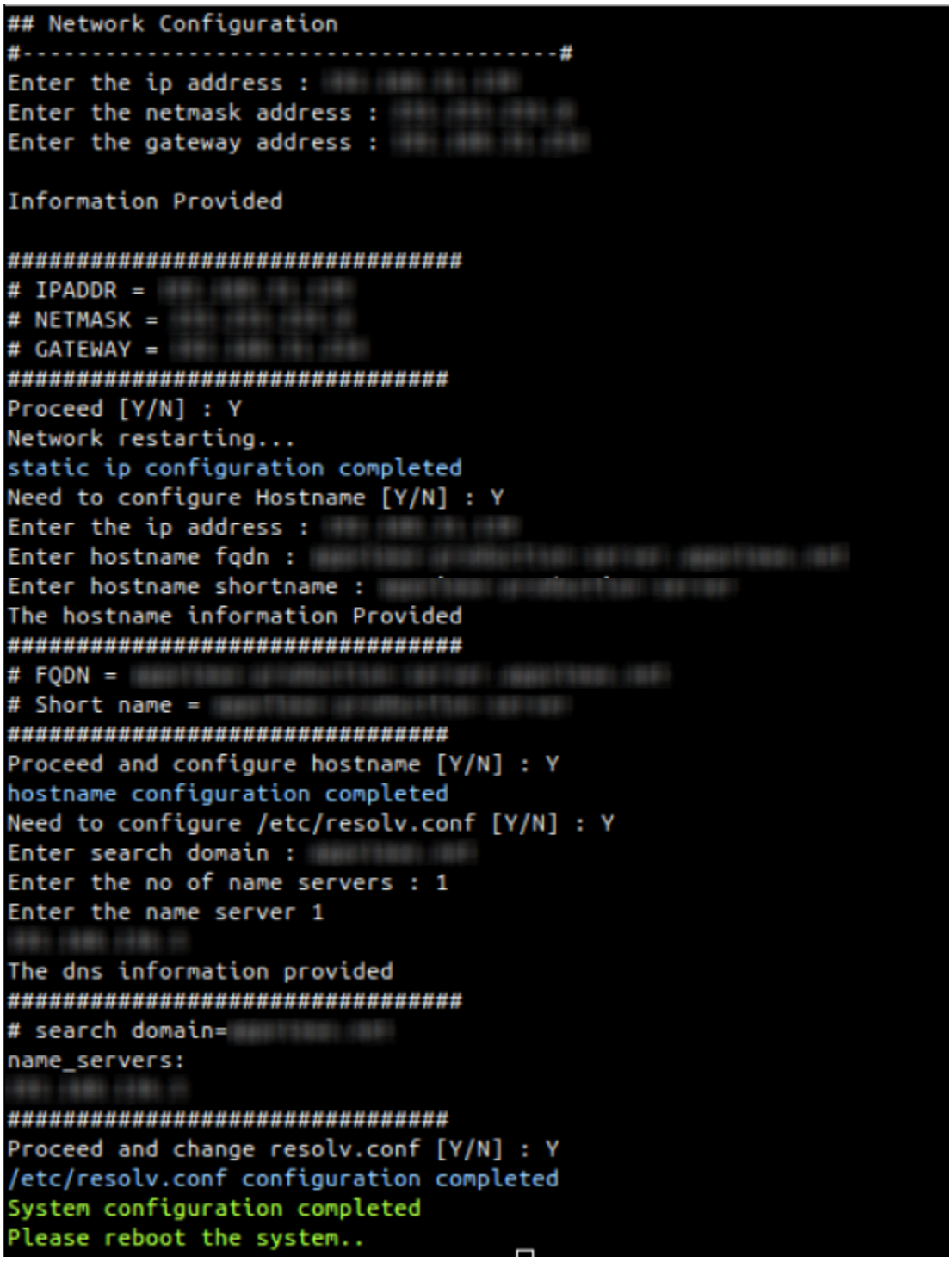
19. Type `Y` to proceed.
20. At the **Need to Configure Hostname [Y/N]** prompt, type `Y` to proceed.



21. Enter the **IP address**, desired **hostname**, and a **short name** for the hostname.



22. At the Proceed and Configure the Hostname [Y/N] prompt, type Y to proceed with the node configuration.



23. Repeat the steps from 2 to 21 to configure the other nodes.
24. After you complete the OVA and network configuration steps across all nodes, SSH into any one of the nodes as an AppViewX user to start the manual installation process.

Chapter 4: Installing AppViewX

Performing a Single Node or Standalone Installation

Prior to performing the installation, ensure the prerequisites success result is received after running the prerequisite tools. For running the prerequisites tool, see section Running the Prerequisite Tool.

1. Copy all the downloaded packages to the server.

> **Note:** The AppViewX installation must start from the node that is selected for the primary MongoDB host. For example, the first node specified under the `MONGODB_HOST` property in the `appviewx.conf` file.

2. SSH to the server in which packages are copied.
3. Open the terminal.
4. To extract the contents of the `appviewx_kubernetes_2021.1.0.tar.gz` file, execute the following command: `tar -xvf appviewx_kubernetes_2021.1.0.tar.gz`
5. To move the `appviewx_kubernetes_addons_2021.1.0.tar.gz` file to the appviewx_kubernetes folder, execute the following command: `mv appviewx_kubernetes_addons_2021.1.0.tar.gz appviewx_kubernetes/`

> **Note:** Refer to the Configuring POD and Service IP CIDR section before proceeding with the install to change the IP addresses/range used for pods and services.

6. To navigate to the <InstallerLocation>/appviewx_kubernetes/scripts directory, execute the following command:cd <InstallerLocation>/appviewx_kubernetes/scripts

> **Note:** If you have received the appviewx.conf file already from AppViewX support, you can skip steps 6 through 9. Copy the provided appviewx.conf file into InstallerLocation/appviewx_kubernetes/scripts/ and continue to Step 10.

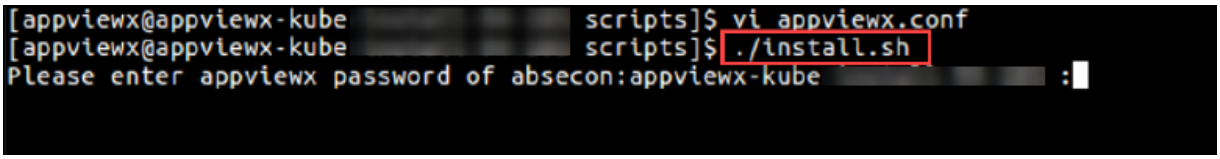
7. To copy the appviewx.conf.template file to the appviewx.conf file, execute the following command:cp appviewx.conf.template appviewx.conf

> **Note:** The entire installation process is driven by the values mentioned in the appviewx.conf file.

8. To open the appviewx.conf file, execute the following command: vi appviewx.conf
9. Enter the configuration values.

> **Note:** For more information, refer to the Configuring the appviewx.conf File to Install Appviewx section. Refer to the deployment diagram provided from help@appviewx.com or use the reference architecture provided by AppViewX

10. Save the changes to the file and exit the editor.
11. In the <InstallerLocation>/appviewx_kubernetes/scripts/ directory, execute the following command /install.sh
12. Enter the user credentials for the respective nodes.



> **Note:** The installerLocation is the path where the installer file is extracted. After you enter the credentials, the installation process starts and takes about 15 to 20 minutes to complete.

After the AppViewX installation is complete, a success message is displayed on the command prompt with the Web and Gateway URLs.

> **Note:**

- Users can also find the AppViewX Web and Gateway URLs in the appviewx.conf file in the installation location.
- Users can
  - verify the installation by following the instructions provided in the section Verifying the Installation
  - upload the license by referring to the instructions provided in the section
  - For troubleshooting issues, please refer to the Troubleshooting section.

Performing a Multi-node or High Availability Installation

This section explains the procedure to install AppViewX in a multi-node environment. The installation procedure is identical to the single node installation with the only difference being the cluster configuration and the POD and Service IP CIDR configuration.

Prior to performing the installation, ensure the prerequisites success result is received after running the prerequisite tools. For running the prerequisites tool, see section Running the Prerequisite Tool.

Recommendations:

- MongoDB is CPU and disk intensive. Therefore, it is recommended to run MongoDB on a worker node.
- The hostnames or IP addresses present in the configuration should be a subset of `SSH_HOSTS`.
- The items in the `SSH` list and the `SSH_HOSTS` list should be in the same order. In other words, if the index of the IP address is 3 in the `SSH` list, it should also be 3 in the `SSH_HOSTS` list.
- It is recommended to assign a data center to a plugin once it is enabled.
- For production environments, a single node deployment is **NOT** recommended because:
  - Single node does not support log monitoring using Kibana and Grafana.
  - Unablavaility of HA.
  - Syslog and statistics not available.

Configuring the appviewx.conf File to Install Appviewx

The installation of the application is driven by the **appviewx.conf** file available within the release package. For more information refer to the configuration file available in the following location:

<InstallerLocation>/appviewx_kubernetes/scripts/

The following parameters must be configured to install the application:

Parameter	Description

MULTINODE	Specifies the boolean value to describe if the installation is in a single node/multi-node environment.
	
	Example:
	
	`MULTINODE=TRUE (For multi-node)`
	
	`MULTINODE=FALSE (For single node)`

SAAS_ENABLED	Specifies the flag to enable SAAS model deployment **Note:** This parameter is only for SaaS installations. Example: `SAAS_ENABLED=false`
SAAS_DOMAIN	Specifies the domain name for SaaS installations **Note:** This parameter is only for SaaS installations. Example: `SAAS_DOMAIN=appvx.com`
VAULT_ENABLED	Specifies the flag to enable or disable the vault - for SAAS model deployment **Note:** This parameter is only for SaaS installations. Example: `VAULT_ENABLED=true`

Parameter	Description
PROVISIONING_ENABLED	Specifies the flag to be enabled for provisoning only the cluster **Note:** This parameter is only for SaaS installations. Example: `PROVISIONING_ENABLED=false`
TENANT_DEPLOYMENT_TYPE	Specifies the flag to set tenant_deployment_type. Expected values (any one of) - customer-prod, customer-non-prod, customer-additional, free-trial, poc-free, free-partner, free-training, internal-dev, internal-qa, internal-se, internal-training Example: `TENANT_DEPLOYMENT_TYPE=customer-prod`
SSH	Specifies the comma (,) separated values of node IPs in which the application is set to be deployed. Example: `SSH=192.168.XXX.XXX, 192.168.XXX.XXX, 192.168.XXX.XXX`
SSH_HOST	Specifies the comma (,) separated values of node hostnames in which the application is set to be deployed. **Note:** Execute the command hostname in the node and add that output to this field. The hostname of a node must be the output of the command "hostname". Ensure to give the IPs provided in the SSH and host name provided in the SSH_HOST must be in the same order. Example:

Parameter	Description
	SSH_HOST=master:appviewx- kube-95.214.appviewx.net,master:appviewx- kube-95.215.appviewx.net,master:appviewx- kube-95.216.appviewx.net,dc1:appviewx- kube-95.217.appviewx.net,appviewx- kube-95.218.appviewx.net,appviewx-kube-95.219.appviewx.net **Note:** For the master nodes, the recommendation is to have the hostname as master:hostname. Ensure that the SSH_HOST and SSH are in the same order.
CLOUD_CONNECTOR_DC	Comma separated values of DC names which will communicate via cloud connector (avx_vendors, avx_vendor_cert_network_discovery) Example: DC1, DC2 CLOUD_CONNECTOR_DC=absecon
INGRESS_HOST	To access AppViewX's Web UI, the INGRESS_HOST parameter must be configured. It can be configured with comma (,) separated values of Kubernetes worker node IP addresses where AppViewX needs to be accessed. **Note:** For single node AppViewX deployments, ensure that it is the IP address of the instance. To ensure high availability of the multiple DC deployments, it is recommended to add a minimum of one host per DC. Example: INGRESS_HOST=192.168.XXX.XXX,192.168.XXX.XXX,192.168.XXX.XXX

Parameter	Description
	Note: It is recommended to add the Kubernetes worker node IP addresses in this field. **Warning:** If the `INGRESS_HOST` parameter does not contain a host IP address, the AppVIewX UI will not be accessible.
INGRESS_LB_URL INGRESS_LB_PORT	In case the load balancer is used for ingress gateway service, provide the URL of the load balancer service and its port.
HSM_HOST	Comma separated values of node hostnames in which HSM pods will be scheduled **Note:** Execute the command "hostname" in the node and add that output to this field • For single node AppViewX deployments add the IP address of the instance where AppViewX is installed. • To ensure high availability in multiple DC deployments, It is recommended to add a minimum of one host per DC. Example: `HSM_HOST=$(hostname)`
INSTALLATION_PATH	Specifies the path in which AppViewX is installed. Example: `INSTALLATION_PATH=/home/appviewx/appviewx/`
ENABLE_IPV6	Specifies whether IP v6 is enabled.

Parameter	Description
	Example: `ENABLE_IPV6=False`
MONITORING	Specifies whether monitoring is enabled or not. When you set the value to `TRUE`, set the value of the `PROMETHEUS_HOST` and `GRAFANA_HOST` to one of the worker node for multinodes. Example: `MONITORING=TRUE`
PROMETHEUS_HOST	Specifies the hostname or IP address of the Prometheus node.
GRAFANA_HOST	Specifies the hostname or IP address of the Grafana node.
LOKI_HOST	Specifies the hostname or IP address of the Loki node.
ENABLED_PLUGINS	Specifies the list of plugins that needs to be enabled in the AppViewX installation. Example: `ENABLED_PLUGINS=appviewx_dependencies,avx_pkiaas_cert_ocsp_generator,avx_pkiaas_cert_ocsp_server,avx_commons,avx_crontab, avx_config_server,avx_platform_core,avx_platform_queue,avx_platform_gateway,avx_platform_web,avx_subsystems,avx_vendors,avx_subsystems_sync, avx_platform_report_generator,avx_visual_page_builder,avx_platform_logforwarding,avx_vendor_cert_network_discovery,avx_platform_hsm`
PLUGINS	Specifies the plugins to be installed in the datacenters. Example: `avx_config_server=absecon:appviewx-kube-150-146.appviewx.net, absecon:appviewx-kube-150-147.appviewx.net`

Parameter	Description
	avx_platform_core=absecon:appviewx-kube-150-146.appviewx.net, absecon:appviewx-kube-150-147.appviewx.net avx_platform_queue=absecon:appviewx-kube-150-146.appviewx.net, absecon:appviewx-kube-150-147.appviewx.net avx_subsystems=absecon:appviewx-kube-150-146.appviewx.net, absecon:appviewx-kube-150-147.appviewx.net avx_subsystems_sync=absecon:appviewx-kube-150-146.appviewx.net avx_vendors=absecon:appviewx-kube-150-146.appviewx.net, absecon:appviewx-kube-150-147.appviewx.net, absecon:appviewx-kube-150-148.appviewx.net avx_platform_gateway=absecon:appviewx-kube-150-146.appviewx.net, absecon:appviewx-kube-150-147.appviewx.net avx_platform_web=absecon:appviewx-kube-150-146.appviewx.net, absecon:appviewx-kube-150-147.appviewx.net
SSH_OTHER_USER SSH_OTHER_GROUP SSH_PORT	Specifies the Linux user account with which AppViewX is installed. **Note:** AppViewX can be installed only as a Sudo user. Refer to the document Commands executed during AppViewX installation to get the details of commands that the Sudo user needs access to. Example: SSH_OTHER_USER=appviewx SSH_OTHER_GROUP=appviewx SSH_PORT=22
MONGODB_MIN_REPLICA	This parameter is used for enabling 2DC,3 Nodes Setup. A maximum 2 nodes needs to be added in MONGODB_HOST. It is mandatory to update ARBITER_HOST.

Parameter	Description
MONGODB_HOST	Specifies the comma (,) separated values of node hostnames in which the MongoDB is set to be deployed. This parameter is applicable only in a multi-node installation. **Note:** Add the output of hostname command in each node in this field. Do not add the output of hostname –f. A minimum of three nodes must be added. Example: MONGODB_HOST=appviewx-kube-95.217.appviewx.net, appviewx-kube-95.218.appviewx.net, appviewx-kube-95.219.appviewx.net **Note:** A minimum of three nodes for MongoDB across three data centers are required to achieve HA at the data center level. It is recommended to run MongoDB only in the worker nodes.
ARBITER_HOST	This parameter is applicable only when AppViewX is deployed with two data centers. Arbiters are MongoDB instances that are part of a replica set but do not hold data. Arbiters participate in elections to break ties. Recommended to enable Arbiters only in AppViewX deployment with two data centers (DC) for high availability. In two DC environments, select the DC that has one Kubernetes master node, configure one of the Kubernetes worker nodes as an Arbiter node. **Note:** If AppViewX deployment is not in two DC environments, this parameter can be blank. Example:

Parameter	Description
	ARBITER_HOST=192.168.XXX.XXX **Note:** Do not add multiple IP addresses. Only one IP address is allowed.
VAULT_HOST	This parameter is valid only in multi-node installations. This parameter is comma (,) separated values of node hostnames in which the vault is set to be installed. **Note:** Add the output of `hostname` command in each node to this field. A minimum of three nodes must be added. Example: VAULT_HOST=appviewx-kube-95.217.appviewx.net, appviewx-kube-95.218.appviewx.net, appviewx-kube-95.219.appviewx.net **Note:** A minimum of three nodes for a vault across three data centers is required to achieve HA at the data center level. It is recommended to run the vault only in the worker hosts.
MASTER_HOST	Specifies the hostname of the node which you want to run as a Kubernetes Master. The total number of masters can be 1, 3, 5, 7, and so on. For example, for a three-master installation, enter one node in the master host and the other two nodes in the secondary master_host. **Note:** Add the output of <hostname> command in this parameter. Example:

Parameter	Description
	MASTER_HOST=appviewx-kube-install-94-179
SECONDARY_MASTER_HOST	Specifies the list of nodes that are designated to run as secondary masters. The total number of masters can be 1, 3, 5, 7, and so on. For example, for a three-master installation, enter one node in the master host and the other two nodes in the secondary master_host. This parameter is applicable only in multi-node installations. **Note:** For deployments with a single master, comment out the SECONDARY_MASTER_HOST section. Example: SECONDARY_MASTER_HOST=appviewx-kube-install-94-180, appviewx-kube-install-94-181
WORKER_HOST	Specifies the hostname of the list of Kubernetes worker nodes. **Note:** This parameter can be empty in a three-node setup. Example: WORKER_HOST=appviewx-kube-install-94-180, appviewx-kube-install-94-181 **Note:** Do not add the value given in the master_host in the worker_host. The worker and master nodes cannot be the same. This is again applicable only in multi-node installations.
TENANT_DB_S3BUCKET TENANT_MIGRATION_S3BUCKET	The following parameters specify the S3 bucket to be mounted for Tenant DB, Tenant migration, CC Binary, Mogo-backup in a SaaS multitenancy provisioning cluster.

Parameter	Description
CC_BINARY_S3BUCKET MONGO_S3BUCKET	**Note:** These parameters are only used for SaaS Installations.
EST_SERVER_ACCESS_CERT	Specifies the location for the digital enrollment certificate.
EST_SERVER_ACCESS_KEY	Specifies the location of the access key for the digital enrollment certificate.
EST_TRUSTED_CA_CERTS	Specifies the location of trusted certificate authorities for the EST server.
ELK	Specifies whether the `ELK` stash is enabled or not. You must specify a value for the `ELASTICSEARCH_HOST` parameter when you set `ELK` to `TRUE` for multinodes. Example: `ELK=FALSE`
ELASTICSEARCH_HOST	Specifies the hostname or IP address of the elastic search host node.
PUBSUB_ENABLED=false PUBSUB_PROJECT_ID= PUBSUB_TOPIC= PUBSUB_JSON_PATH=	Specifies the flags to enable the Google Pub/Sub for SaaS model deployment. **Note:** These labels below will be considered when ELK is set to true
SPLUNK_HEC_ENABLED=false SPLUNK_HEC_HTTPS_ENABLED=false SPLUNK_HEC_URL= SPLUNK_HEC_CERT= SPLUNK_HEC_TOKEN=	Specifies the flags to enable the Splunk for SaaS model deployment. **Note:** These labels below will be considered when ELK is set to true

Parameter	Description
XSS_PROTECTION	This parameter is used to enable the XSS Sanitisation in the API Gateway, and avoids any XSS related exploits. Example: `XSS_PROTECTION=true`
API_ADDRESS	Specifies the hostname of the API server.
INSIGHT	Specifies whether the Insight module is enabled or not. Example: `INSIGHT=TRUE`
SYSLOG	Specifies whether the Syslog module is enabled or not. Example: `SYSLOG=TRUE`
INSIGHT_ELASTICSEARCH_HOST	Specifies the hostname or IP address of the insight elastic search host node.
USER_GENERATED_PEM and PRIVATE_KEY_FILE_PATH	Set the value of the `USER_GENERATED_PEM` variable to `TRUE` if you want to perform a password-less installation. **Note:** • Update the value of the `PRIVATE_KEY_FILE_PATH` and set the value of the `USER_GENERATED_PEM` variable to `TRUE`. Otherwise, leave it empty. • Ensure that the value of the `PRIVATE_KEY_FILE_PATH` variable points to the private key file and not the directory. For example: `/tmp/user_generated_private.pem`.

Parameter	Description
REDIS_HOST	• The REDIS_HOST parameter is configured and applicable only in a multi-node setup. • Use only comma separated values of node hostnames in which the REDIS is to be deployed. **Note:** Add the output of the hostname command in each nodes to this field. Do not add output of hostname -f. • It is recommended to add only worker node(s) as the REDIS hosts but not the master hosts. • In case of two REDIS instances to be deployed on one node, add that node's hostname twice (e.g.: hostname1, hostname1, hostname2). • Add only two REDIS instances for a two-DC setup. Example: `REDIS_HOST=$(hostname)`
SENTINEL_DC	• The parameter, SENTINEL_DC is only needed for a two-DC setups. • It is preferred to be in the secondary DC (i.e. DC with less kubernetes Master) • The REDIS Sentinel will spin up only on the DC mentioned in this parameter. Example: `SENTINEL_DC=absecon`
SYSLOG_LOGSTASH_HOST	Specifies the hostname of the node where the syslog logstash needs to be deployed. Enter only one hostname, as shown below. `SYSLOG_LOGSTASH_HOST=$(hostname)`

Parameter	Description
ENABLE_LOWER_TLS	Set ENABLE_LOWER_TLS=True to enable TLSv1.0, TLSv1.1 in the application to manage devices with lower TLS versions.

OPTIMISE_ROUTING_FOR_LATENCY
 	
PREFERRED_DEFAULT_DC
	This parameter is used mainly if the application is installed across multiple DCs and the latency between the DCs is high. The local routing between the pods can be enabled by setting OPTIMISE_ROUTING_FOR_LATENCY=True and specifying the preferred DC name in PREFERRED_DEFAULT_DC to increase the application performance.
	
	Example:
	
	`OPTIMISE_ROUTING_FOR_LATENCY=FALSE`
	
	`PREFERRED_DEFAULT_DC=absecon`

MTU_VALUE	This option is used to change the MTU value for the calico during the appviewx installation. **Note:** This value should be changed before the application installation. Example: `MTU_VALUE=1350`
IPV4POOL_IPIP IPV4POOL_VXLAN	This option is used to enable the IPinIP/VXLAN tunneling for calico. **Note:**

Parameter	Description
	• 'Always' should be for any one of the protocols (IPIP or VXLAN), it should not be added for both. • This value should be changed before the application installation. Example: IPV4POOL_IPIP=Always IPV4POOL_VXLAN=Never
SERVICE_SUBNET POD_SUBNET	This option is used to configure the pod and service default IP subnet ranges. **Note:** • The IP range should not conflict with any of the internal IP ranges. • This value should be changed before the application installation. Example: SERVICE_SUBNET=10.96.0.0/12 POD_SUBNET=10.244.0.0/16
CALICO_PORT	This option is used to configure the default calico port. **Note:** This value should be changed before the application installation. Example: CALICO_PORT=179

Parameter	Description
SFTP_TRANSFER REMOTE_BACKUP_SERVER REMOTE_BACKUP_SERVER_SSH_PORT REMOTE_BACKUP_SERVER_USER MONGO_BACKUP_PATH VAULT_BACKUP_PATH	This option is used to configure the external SFTP Transfer for Mongo and Vault backup. It enables Passwordless communication between the remote backup server and the appviewx nodes. ***Pre-installation:*** Set SFTP_TRANSFER to true and configure the below listed variables ***Post-installation:*** Set SFTP_TRANSFER to true and configure the below listed variables and execute ./ sftp_transfer.sh script from the <appviewx-installer-location>/appviewx_kubernetes/scripts directory. The parameter description with examples is as follows: • *SFTP_TRANSFER=FALSE* – Enables SFTP transfer • *REMOTE_BACKUP_SERVER=* – Updates the SFTP server IP to store the vault and mongo backups • *REMOTE_BACKUP_SERVER_SSH_PORT=22* – Updates the External SFTP server's SSH port in case of a custom SSH port • *REMOTE_BACKUP_SERVER_USER=appviewx* – Contains the username of the remote backup server • *MONGO_BACKUP_PATH=/home/appviewx/* – Updates the External SFTP location to store the mongodb backup • *VAULT_BACKUP_PATH=/home/appviewx/* – Updates the External SFTP location to store the vault backup
ENABLE_CUSTOM_CA_CERTS=FALSE CERTIFICATE_PATHS=/home/appviewx/ appviewx/ca-bundle.crt	This option is used to enable custom certs for outbound site communication. Enter the absolute path of the certificate to add to java truststore in the comma delimited format. **Note:**

Parameter	Description
	• It is recommended to use CA-signed certificates for better security. • If you still want to go ahead and add any internal CA's or Self-signed ones, do so at your own risk.
DB_MIGRATION_JOB_TIMEOUT	This parameter is used to configure the timeout (in minutes) for the DB Migration job. Example: `DB_MIGRATION_JOB_TIMEOUT=65`

ISTIO_SECRET_TT	The istio secret TTL value is used to extend the workload certificates. The secret TTL value should always be in minutes. Execute the utility command to the configuration:
	`./appviewx.sh --update-secret-ttl`
	Example:
	`ISTIO_SECRET_TTL=8640m`

Configuring POD and Service IP CIDR

This section explains how to configure the number of POD/Service IP CIDRs that can run on a node. The Pods that run on a node are allocated an IP address from the node's Pod CIDR range.

Note: It is recommended to use the default settings for the POD and Service IP CIDR.

To configure POD/Service IP CIDRs:

1. Navigate to the <InstallerLocation>/appviewx_kubernetes/configs/kube/ directory.
2. To open the file, execute the following command: `vi kubeadm-config.yaml.tpl`

```
-bash-4.2$ cd  /home/appviewx/appviewx_kubernetes/configs/kube/
-bash-4.2$ vi kubeadm-config.yaml.tpl
-bash-4.2$
```

3. Under the networking section, check for serviceSubnet and change it as per requirements.`CIDR networking: serviceSubnet: <value> <change this default value to the desired CIDR>`
4. Under the networking section, check for podSubnet and change it as per requirements.`podSubnet: <value> <change this default value to the desired CIDR>`

```
apiVersion: kubeadm.k8s.io/v1beta2
kind: ClusterConfiguration
kubernetesVersion: v1.18.1
controlPlaneEndpoint: "${api_address}:6443"
networking:
  serviceSubnet: "[redacted]"
  podSubnet: "[redacted]"
  dnsDomain: "cluster.local"
apiServer:
  certSANs:
  - "${api_address}"
  extraArgs:
    service-account-signing-key-file: /etc/kubernetes/pki/sa.key
    service-account-key-file: /etc/kubernetes/pki/sa.pub
    service-account-issuer: api
    service-account-api-audiences: api,vault,factors
    authorization-mode: "Node,RBAC"
~
```

5. Save the changes and close the editor.
6. Once the above steps are complete, proceed with the AppViewX installation as mentioned in the Installing AppViewX section.

> **Note:**
> - After the successful installation, you can access the .appviewx_configuration file by following the procedure given in the Accessing the Management Console section.
> - Users can upload the license by referring to the instructions provided in the section Uploading the License Key.

Installation Support for 3 Nodes and 2 Datacenters

1. In the `appviewx.conf` file, set the value for the `Multinode` parameter as "`TRUE`".
2. Update the `SSH` and `SSH_HOST` parameters with the 1 master and min 2 workers as shown below.

```
# Comma separated values of node IPs in which the application is to be deployed
# For single node add this node ip
SSH=[redacted]

# Comma separated values of node hostnames in which the application is to be deployed
# Note: Execute the command hostname in the node and add that output to this field
# For single node add this node hostname
# Dont add datacenter as avx
SSH_HOST=master:[redacted]
```

3. Set the value of the `MONGODB_MIN_REPLICA` parameter as `TRUE`.

```
MONGODB_MIN_REPLICA=TRUE
```

4. Ensure that you add a minimum of 2 hosts to the `MONGODB_HOST` parameter. It is mandatory to add any one of the IP addresses of the mongodb host to the `ARBITER_HOST` parameter.

```
MONGODB_HOST=worker1.lab.net,worker2.lab.net
```

```
ARBITER_HOST=192.168.xx.2
```

5. Retain the `VAULT_HOST` parameter as is, because the system will automatically assign a vault host from each of the datacenters.
6. Update the `MASTER_HOST` and the `WORKER_HOST` parameters appropriately with the hostnames.
7. To navigate to the `<installer location./appviewx_kubernetes/scripts` `cd <installer location>/appviewx_kubernetes/scripts`
8. To run the installation script, execute the following command: `./install.sh`

Enabling the Load Balancer for the Kube API Server

Given below is an example configuration done on F5 devices and is needed only when we need to balance the load between multiple kube api servers in the case of multi DC support.

Prerequisite:

Create the TCP load balancer for Kube master apiserver.

> **Note:** This section is applicable only when the load balancer for the kube apiserver is not installed during the installation.

Sample Configuration:

Load balancer Configuration for Kube Master:

```
ltm virtual vs-appviewxmasterapi {
destination <IP Address>:sun-sr-https
ip-protocol tcp
mask XXX.XXX.XXX.XXX
pool pool-avxmasterapi
profiles {
```

```
fastL4 { }
}
serverssl-use-sni disabled
source 0.0.0.0/0
source-address-translation {
type automap
}
translate-address enabled
translate-port enabled
}
```

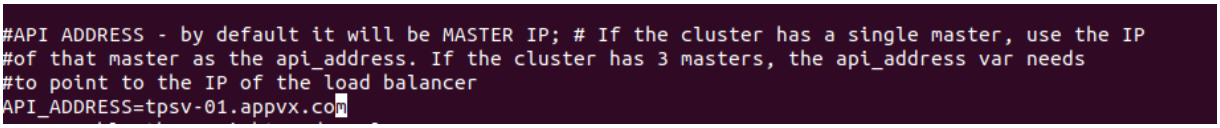
Pool Member Configuration for Kube Master

```
ltm pool pool-avxmasterapi {
 members {
 <Master Node IP Address>:sun-sr-https {
 address XXX.XXX.XXX.XXX
 session monitor-enabled
 state up
 }
 <Master Node IP Address>:sun-sr-https {
 address XXX.XXX.XXX.XXX
 session monitor-enabled
 state up
 }
 <Master Node IP Address>:sun-sr-https {
 address XXX.XXX.XXX.XXX
 session monitor-enabled
 state up
 }
 }
 monitor gateway_icmp
 }
```

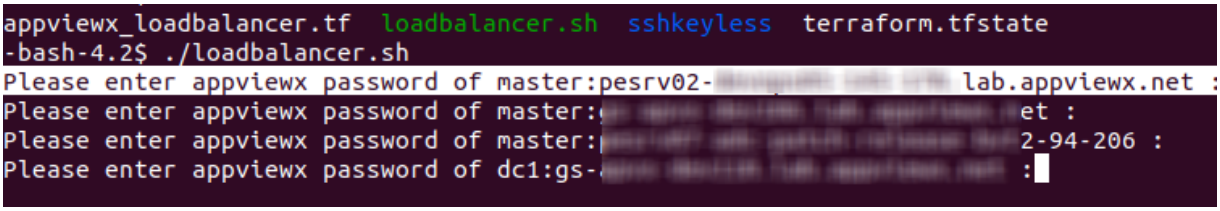
To enable the load balancer for Kube Master:

1. To verify whether the load balancer is functioning normally, execute the following command: `curl -k https://loadbalancer-ip:6443/version`

2. Apply the latest script patch from the release portal.
3. Navigate to the <installerLocation>/appviewx_kubernetes/scripts/ directory.
4. Open the `appviewx.conf` file.
5. Search for the `API_ADDRESS` parameter.
6. Modify the value of the `API_ADDRESS` parameter to reflect the IP Address or the FQDN of the load balancer.



7. Navigate to the <installerLocation>/appviewx_kubernetes/scripts/loadbalancer/ directory.
8. To run the load balancer script, execute the following command: `./loadbalancer.sh`



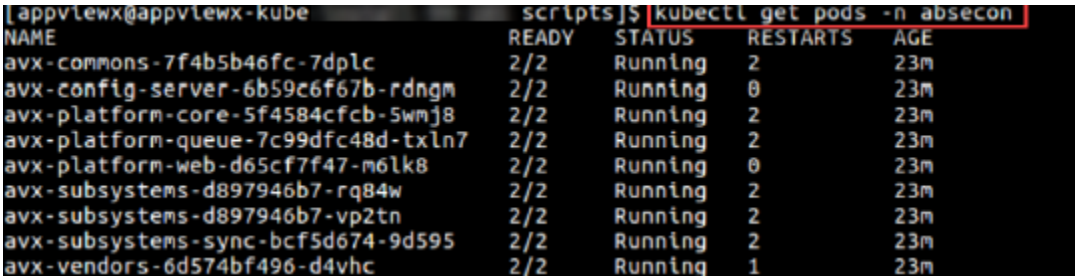
9. Enter the password of the nodes when prompted.
10. To verify the changes, execute the following command: `kubectl cluster-info`

    The output should contain the updated load balancer URL (IP Address or FQDN) of the kube API server.

Verifying the Installation

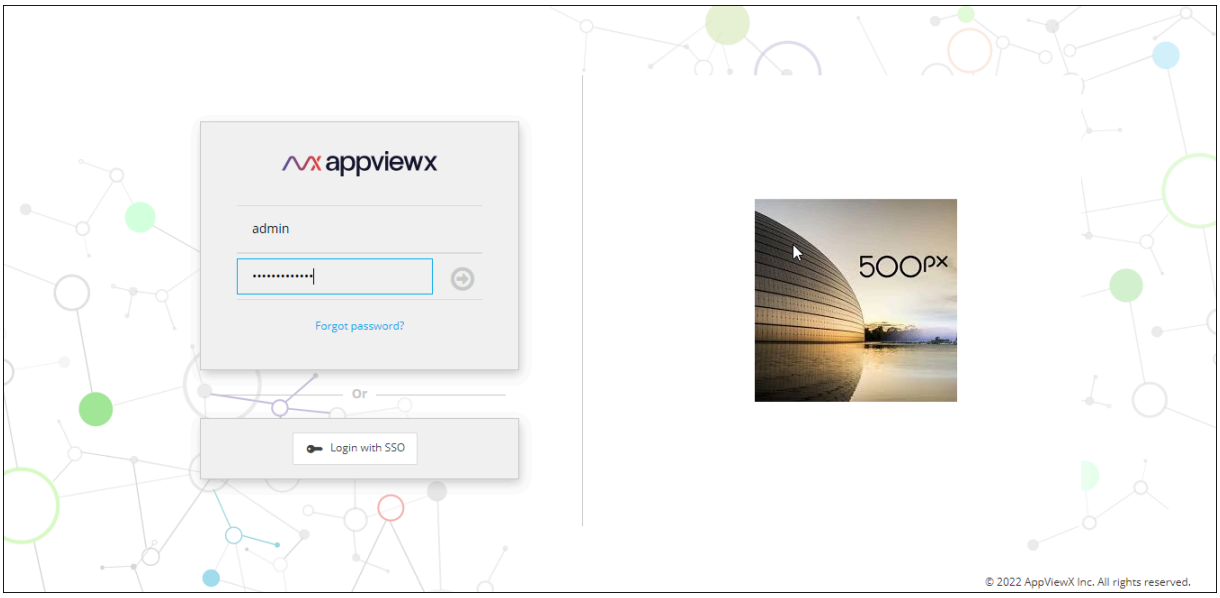
This section provides information on verifying whether the installation of AppViewX is successful. There are a few commands that will help you verify the installation. The commands are listed below.

1. To check the status of the pods, execute the following command: `kubectl get pods -A`If any of the pods show a different status, the application might not function as expected.
2. To restart the pod, execute the following command:`kubectl delete pods -n <dcname> <podname>`



3. Access the GUI using the AppViewX Web URL with valid credentials. (AppViewX provides the default credentials).

> **Note:** Refer to the `appviewx_configuration` file, available for the URL. The file is available in the <InstallerLocation>/appviewx__kubernetes/scripts/ directory.



Note: Multi-node installations come with a Redis cluster out-of-the-box. For single-node installations, there is a single Redis instance available that is enabled for PubSub only.

Note: For troubleshooting issues, refer to the Troubleshooting section.

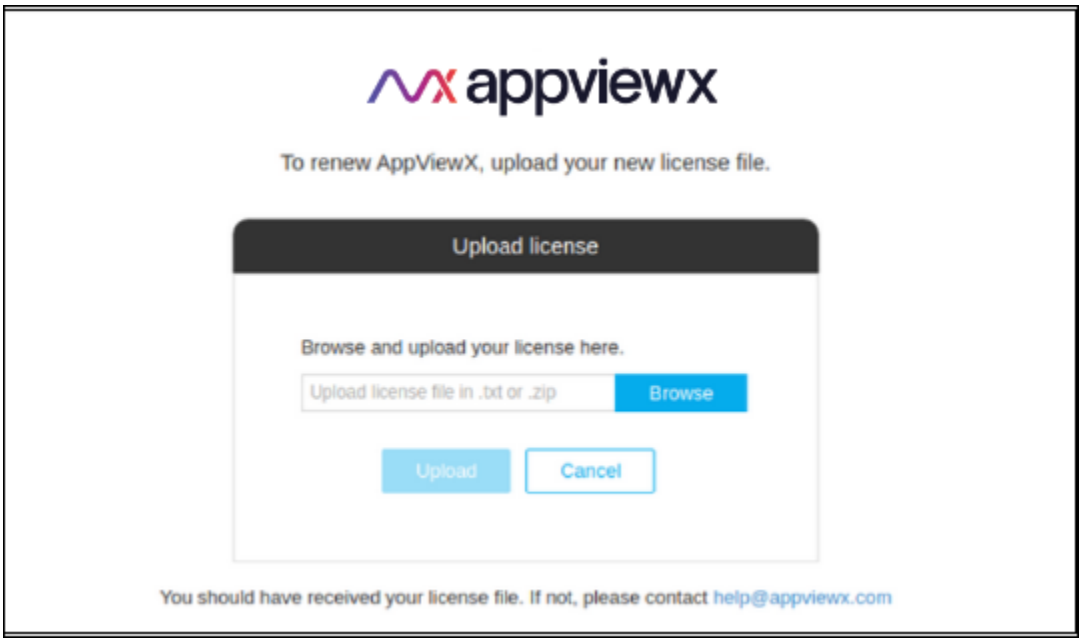
Uploading the License Key

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

To access the application, the user needs to upload a license. If you do not have a license key, send an email to help@appviewx.com with the hostname of the node in which the application is installed.

To upload the license key:

1. Log in to the application using the Web URL displayed in the success message of the installation. You are prompted to enter the username and password of the AppViewX admin account.
2. Click **Browse** and upload the license file.



A confirmation message is displayed after uploading a valid license.

Troubleshooting:

- If the license upload fails, ensure that the uploaded file is in the proper <.txt> (or) <.zip> format.
- If the license upload fails while activating the license, ensure that the output of the hostname command is provided during the generation of the license.
- If the license upload fails, trigger the following URL from your browser and try again after a few minutes.

`https://AppViewX GATEWAY URL/refresh`

Adding Third-party Libraries

AppViewX requires specified libraries to manage and control the devices. These libraries are specified by the manufacturers of the devices. AppViewX will be able to communicate with the devices only when these libraries are installed.

Please follow the steps in this section to add external proprietary jars in AppViewX.

- If the customer wants to use any third party integrations with earlier versions of AppViewX, ensure that the **.jar** files for these integrations are downloaded and extracted in the Installer/external_lib directory before the migration/installation process.
- If the customer wants to use any third party integrations with earlier versions of AppViewX after migration or installation, ensure that the corresponding **.jar** files are downloaded and extracted to the /home/appviewx/appviewx/external_libs directory.

iControl F5 Integration

iControl is an open API that enables applications to work in sync with the network based on the software integration. iControl uses SOAP/XML to ensure an open communication between dissimilar systems. It helps F5 customers, independent software vendors (ISVs), and solution providers leverage efficiency in automation and management of network objects and devices.

Users who want to use third party integrations to control their devices can integrate the required .jar file. The process begins with the user downloading the .jar file from the respective vendor. After downloading, the contents of the .jar file must be extracted into the external_libs directory. Finally, the plugin must be restarted for the changes to take effect.

1. To integrate the iControl library into the required project, copy the library and paste it into the <user_home_dir>/installer/external_libs/ directory (create a directory if it does not exist).
2. Visit devcentral f5 download page URL: https://devcentral.f5.com/s/articles/iControl-Library-For-Java-With-Source.
3. Download the latest iControl integration library file from the list of libraries.
4. Extract the downloaded zip file to: `iControlAssembly_13_1_0-Java\`.
5. Copy the `iControl.jar` file from the extracted package to the external_libs directory.
6. If AppViewX is already installed or upgraded from an earlier version of AppViewX, move the `iControl-13.1.0.jar` file to cp -r /lib/iControl-13.1.0.jar <user_home_dir>/appviewx_dependencies/external_libs/ directory.
7. If AppViewX is not installed, move the `iControl-13.1.0.jar` file to cp -r /lib/iControl-13.1.0.jar /home/appviewx/Installer/external lib directory.

8. In case of a multi node environment, copy the `iControl-13.1.0.jar` file to all the servers where the `avx_vendors` plugin is running.

> **Note:** To restart the avx_vendors plugin followed by the gateway plugin, refer to the Restarting a plugin section.

Thales

Users who want to use third party integrations to control their devices can integrate the required .jar file. The process begins with the user downloading the .jar file from the respective vendor. After downloading, the contents of the .jar file must be extracted into the `external_libs` directory. Finally, the plugin must be restarted for the changes to take effect.

> **Note:** Install the Thales client only on the node where AppViewX is installed.

1. To navigate to the directory where Thales client is installed, execute the following command: `cd /opt/nfast/java/classes`
2. Copy the jutils, kmjava, and nfjava jars from the directory and paste it to the external libs directory in AppViewX.
   - If AppViewX is already installed /migrated, execute the following command: `cp <jar_name>.jar <user_home_dir>/external_libs/`
   - If AppViewX is not installed/migrated, to copy the jar in the installer directory, execute the following command: `cp <jar_name.jar /home/appviewx/Installer/external lib`
3. Restart the `avx_vendors` plugin followed by the gateway plugin.

> **Note:** For more information on how to restart the plugin, refer to the Restarting a plugin section.

Safenet/Gemalto

Users who want to use third party integrations to control their devices can integrate the required .jar file. The process begins with the user downloading the .jar file from the respective vendor. After downloading, the contents of the .jar file must be extracted into the external_libs directory. Finally, the plugin must be restarted for the changes to take effect.

Note: Install the Safenet client only on the node where AppViewX is installed.

1. To navigate to the directory where Safenet is installed, execute the following command:`cd /usr/safenet/lunaclient/jcprov/lib`
2. Copy the `jcprov jar` from the directory and paste it to the `external_lib` directory in AppViewX.
   - If AppViewX is already installed /migrated:

     ```
     cp jcprov.jar <user_home_dir>/appviewx_dependencies/external_libs/
     ```

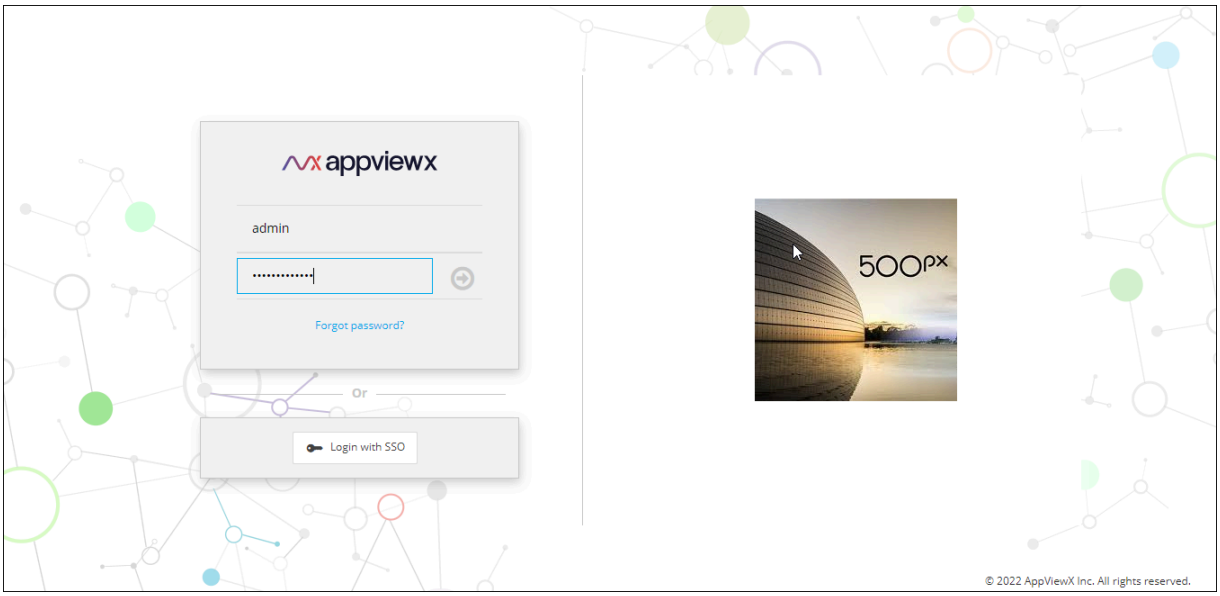
   - If AppViewX is not installed/migrated, copy the jar in the installer directory:

     ```
     cp jcprov.jar /home/appviewx/Installer/external_lib
     ```
3. Restart the `avx_vendors` plugin followed by the gateway plugin.

Note: For more information on how to restart the plugin, refer to the Restarting a plugin section.

Accessing the AppViewX Graphical User Interface

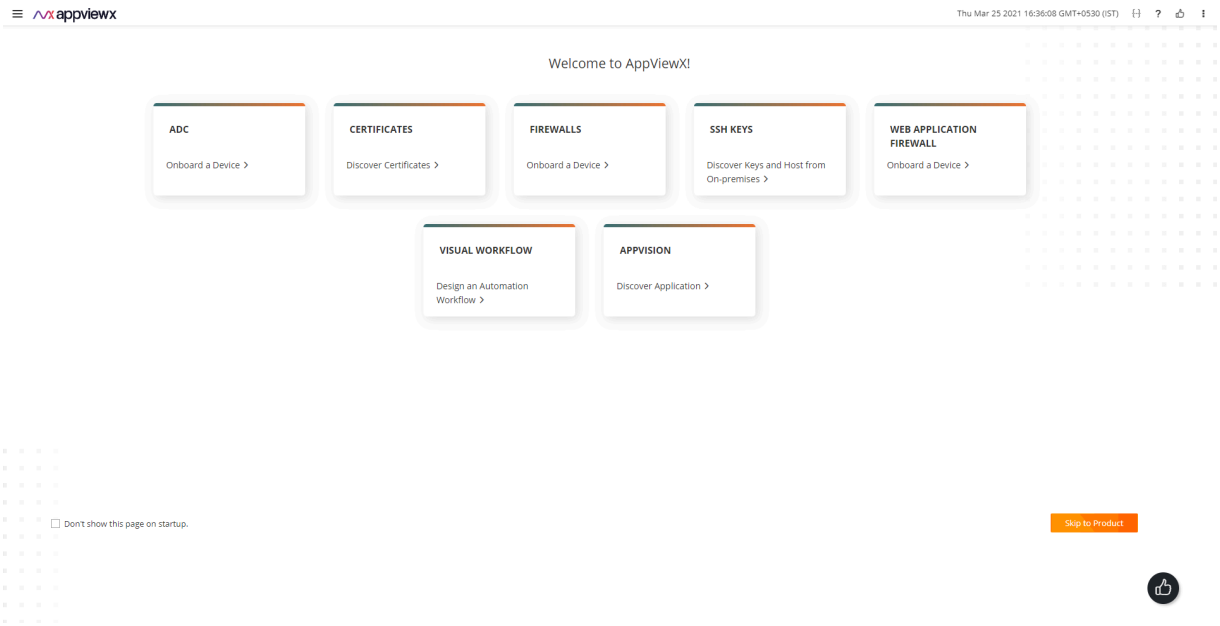
1. Access the graphical user interface (GUI) using the AppViewX Web URL with valid credentials.



Note: AppViewX provides default credentials to access the GUI.

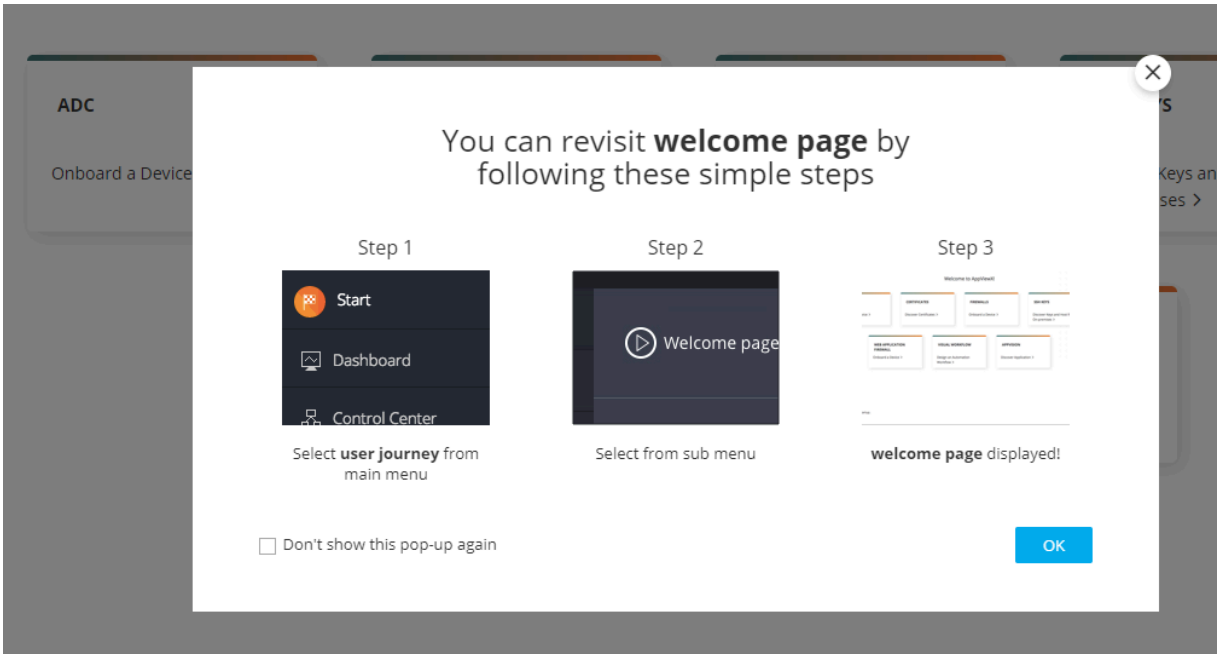
Note: Refer to the appviewx_configuration file, available for the URL. The file is available in the <InstallerLocation>/appviewx__kubernetes/scripts/ directory

Upon successful login, the **Welcome to AppViewX** page is displayed.



2. Click **Skip to Product**.

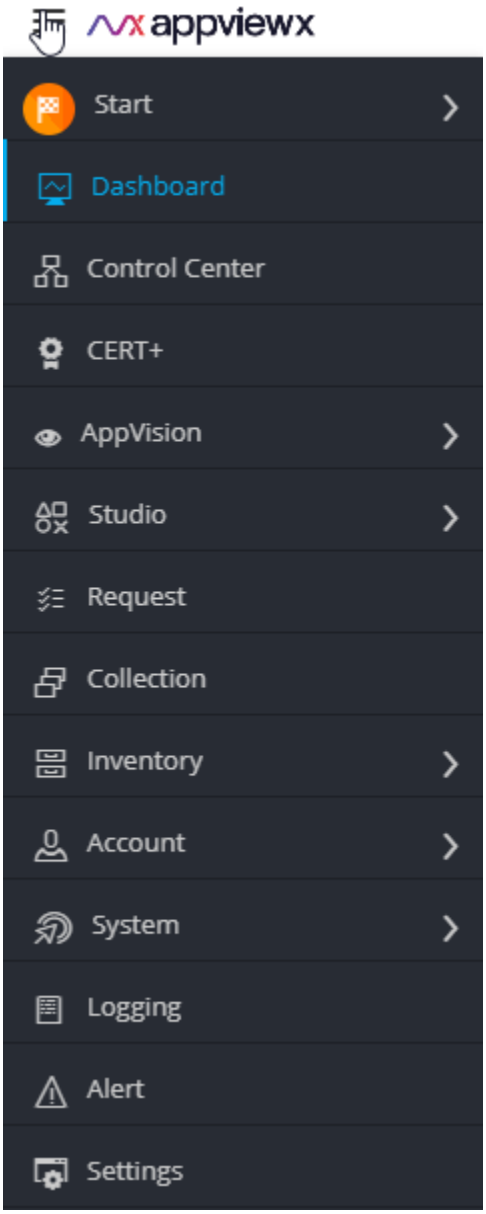
   The **Revisit Welcome** page is displayed.



3. Click **OK**.

   The system loads the dashboard page.

4. To access the different modules, click the icon to view the menu.



   Using this menu, you can navigate to the different modules and access the features of the application.

Installing a Fix Pack

This section provides instructions for applying patches on AppViewX v2022.1.0.

Downloading the Patch

Before installing the fix pack, ensure that you have downloaded the patch plugins and addons from the release portal.

Installing the Patch

The process of installing the fix pack is executed by a script.

Note: For more information and detailed steps, please refer to the AppViewX Patch Deployment Guide.

Upgrading to 2022.1.0 FP1

You can now upgrade to AppViewX 2022.1.0 FP1 from any of the legacy applications, mainly

- 2020.3.0 FP10
- 2021.1.0 FP3
- 2022.1.0

Refer to the Application Upgrade Guide from AppViewX documentation portal.

Chapter 5: Monitoring and Maintaining AppViewX

Installing ELK Components

Elasticsearch, Logstash, and Kibana (ELK) provide centralized logging to identify problems in servers or applications. It allows you to search all your logs and find issues that occur in multiple servers by

connecting their logs during a specific time frame. To enable log management of the present and historical CLI logs of AppViewX from a GUI, an ELK-based utility is utilized.

To install ELK:

1. Download the `appviewx_kubernetes_elk_2022.1.0.tar.gz` file.
2. To extract the contents of the file, execute the following command: `tar -xvf appviewx_kubernetes_elk_2022.1.0.tar.gz`
3. Move the contents of the extracted file to <InstallerLocation>/appviewx_kubernetes/yaml/appviewx_monitoring directory.
4. Edit the **appviewx.conf** file.
5. Update the following parameters with their corresponding values:

   a. `ELK=TRUE`

   b. /appviewx_kubernetes/yaml/appviewx_monitoring directory
6. Navigate to <InstallerLocation>/appviewx_kubernetes/Scripts/ directory.
7. To run the script, execute the following command: `./elk_install.sh`
   You can find the URLs to access the ELK stack in the command prompt after the installation is complete.

Installing Monitoring Components

1. Edit the `appviewx.conf` file
2. Set `Monitoring = True`.
3. Set the value of the `PROMETHEUS_HOST` and `GRAFANA_HOST` parameter to one of the worker nodes.
4. Navigate to `<InstallerLocation>/appviewx_kubernetes/Scripts/` directory.

5. To install the monitoring components, execute the following command: `./elk_install.sh`

> **Note:** For more information on installing the ELK stash, refer to the Installing ELK section.

Executing Commands for Maintenance

AppViewX is installed based on the Kubernetes engine. We can use a few of the basic Kubernetes commands to manage the components in AppViewX. The list below contains the basic commands to manage the Kubernetes cluster.

- View all the nodes in the cluster `kubectl get nodes`

```
[RPK-appviewx@ ]$ kubectl get nodes
NAME                    STATUS   ROLES    AGE   VERSION
pesrv07-devops-94-107   Ready    master   18d   v1.18.6
```

- View all the pods of AppViewX services `kubectl get pods -n avx -o wide`

```
[RPK-appviewx@ ]$ kubectl get pods -n avx -o wide
NAME                                    READY   STATUS             RESTARTS   AGE
IP              NODE                    NOMINATED NODE   READINESS GATES
avx-platform-gateway-586cdccd79-s7fl9   0/2     Pending            0          14d
<none>          <none>                  <none>           <none>
avx-platform-gateway-586cdccd79-xlxcd   1/2     Terminating        793        18d
<none>          pesrv07-devops-94-107   <none>           <none>
avx-platform-web-5c4595b87-cd2ml        2/2     Running            0          12d
                pesrv07-devops-94-107   <none>           <none>
mongo-configdb-0                        2/2     Running            0          12d
                pesrv07-devops-94-107   <none>           <none>
mongo-configdb-1                        2/2     Running            0          12d
                pesrv07-devops-94-107   <none>           <none>
mongo-configdb-2                        2/2     Running            0          12d
                pesrv07-devops-94-107   <none>           <none>
```

- View all the services `kubectl get services -n avx`

```
[RPK-appviewx@ ]$ kubectl get services -n avx
NAME                    TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)             AGE
avx-platform-gateway    ClusterIP                    <none>        5300/TCP            18d
avx-platform-web        ClusterIP                    <none>        5004/TCP,5555/TCP   18d
mongo-configdb-service  ClusterIP                    <none>        27017/TCP           18d
mongo-routerdb-service  ClusterIP                    <none>        27017/TCP           18d
mongo-shareddb-service  ClusterIP                    <none>        27017/TCP           18d
vault                   ClusterIP                    <none>        8200/TCP,8201/TCP   18d
vault-internal          ClusterIP                    <none>        8200/TCP,8201/TCP   18d
```

- Log in to a particular container of the pod `kubectl exec -it avx-platform-web-5c4595b87-cd2ml -n avx /bin/sh`

```
[RPK-appviewx@ ]$ kubectl exec -it avx-platform-web-5c4595b87-cd2ml -n avx -- /bin/sh
Defaulting container name to avx-platform-web.
Use 'kubectl describe pod/avx-platform-web-5c4595b87-cd2ml -n avx' to see all of the containers in this pod.
sh-4.2#
sh-4.2#
```

- List all the namespaces `kubectl get namespaces`

```
[RPK-appviewx@ ]$ kubectl get namespaces
NAME                   STATUS   AGE
absecon                Active   18d
avx                    Active   18d
avx-jobs               Active   18d
default                Active   18d
external-system        Active   18d
istio-operator         Active   18d
istio-system           Active   18d
kube-node-lease        Active   18d
kube-public            Active   18d
kube-system            Active   18d
kubernetes-dashboard   Active   18d
lens-metrics           Active   16d
```

- List all the configuration maps. This is used to view configuration related details. `kubectl get configmaps -n avx`

```
[RPK-appviewx@ ]$ kubectl get configmaps -n avx
NAME                          DATA   AGE
avx-common-config             6      18d
avx-platform-gateway-config   2      18d
avx-platform-web-config       1      18d
avx-vault-configmap           3      18d
istio-ca-root-cert            1      18d
vault-config                  1      18d
```

- List all the deployments `kubectl get deployments -n avx`

```
[RPK-appviewx@ ]$ kubectl get deployments -n avx
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
avx-platform-gateway   0/1     1            0           18d
avx-platform-web       1/1     1            1           18d
```

- Stop a deployment `kubectl scale --replicas=0 deployment/avx-vendor-haproxy -n avx`

```
[RPK-appviewx@ ]$ kubectl scale --replicas=0 deployment/avx-platform-gateway -n avx
deployment.apps/avx-platform-gateway scaled
```

- Start a deployment `kubectl scale --replicas=1 deployment/avx-vendor-haproxy -n avx`

```
[RPK-appviewx@ ]$ kubectl scale --replicas=1 deployment/avx-platform-gateway -n avx
deployment.apps/avx-platform-gateway scaled
```

- Edit a configuration `kubectl edit configmaps -n avx avx-common-config`

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  APS_MONGO_ENCRYPTED_PASSWORD: wnfzZa0MAvfOR/ULgeCMNA==
  DATA_CENTER: avx
  DEPENDENCY_PATH: /appviewx/dependencies
  MONGO_ENCRYPTED_PASSWORD: vault:v1:JY4O+YyoCLfcfUys7T84zWGAB/Vr9sNSk/8h9VYFNIA+jazThhggPeH49ZM5
  MONGO_KEY: t6ehrofmwa59g3hoakjh4d79s
  appviewx.properties: "#Below Vault are replaced in vault helm chart\nAPP_ROLE_ID=a5e54859-3304-13b3-3d74-6
\n#RELEASE_INFO\nRELEASE_DATE=2019-18-12_17-24-00\nBUILD_NUMBER=416\nRELEASE_DESCRIPTION=appviewX2020.1.0\n\
alhost:$APPVIEWX_SERVICE_PORT/services/\n\n#CERT_DELAY\nCERT_DISC_BATCH_AND_DELAY_IN_MILLISECONDS=220/2000\n
```

- Describe the pods `kubectl describe pods -n avx <plugin name>`

```
[RPK-appviewx@ ]$ kubectl describe pods -n avx avx-platform-web-5c4595b
87-cd2ml
Name:         avx-platform-web-5c4595b87-cd2ml
Namespace:    avx
Priority:     0
Node:
Start Time:   Thu, 25 Feb 2021 04:53:59 +0000
Labels:
Annotations:
```

- Log in to the database `kubectl exec -it mongo-routerdb-0 -n avx -- /bin/sh`

```
[RPK-appviewx@ ]$ kubectl exec -it mongo-routerdb-0 -n avx -- /bin/sh
Defaulting container name to mongo-routerdb-container.
Use 'kubectl describe pod/mongo-routerdb-0 -n avx' to see all of the containers in this pod.
#
#
```

Installing Trusted Certificate for GUI/API Access

To install a trusted certificate for GUI/API access:

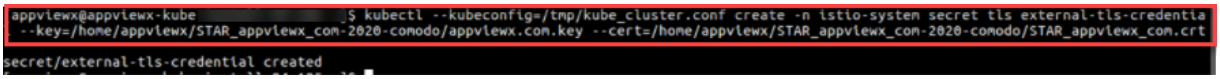
1. To create a secret external-tls-credential of type tls, execute the following command:`kubectl --kubeconfig=~/.kube/config create -n istio-system secret tls external-tls-credential --key=/etc/qualys/ssl/appviewx.com.key --cert=/etc/qualys/ssl/ssl-bundle.crt`

   For example:

   ```
   kubectl --kubeconfig=~/.kube/config create -n istio-system secret tls external-tls-credential --key=/etc/qualys/ssl/appviewx.com.key
   --cert=/etc/qualys/ssl/ssl-bundle.crt
   ```

   where:

   - *~/.kube/config* should be present in each node
   - *~/.kube* will be present in the home folder of the installing user

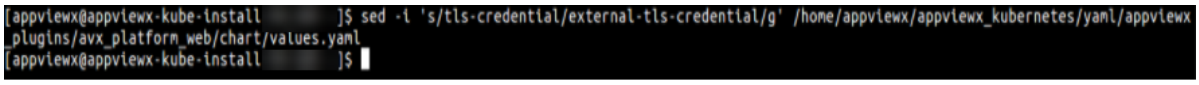


2. Replace secret name tls-credential with external-tls-credential in the `values.yaml` file.

   Note: The `values.yaml` file is available at installerLocation/appviewx_kubernetes/yaml/appviewx_plugins/avx_platform_web/chart/

   - To replace, execute the following command:

   ```
   sed -i 's/tls-credential/external-tls-credential/g' <installerLocation>/appviewx_kubernetes/yaml/appviewx_plugins/avx_platform_web/chart/values.yam
   ```



3. Update the Gateway to consume the latest changes:

   a. To navigate to the `<installerLocation>/appviewx_kubernetes/yaml/appviewx_plugins/avx_platform_web` directory, execute the following command:

   ```
   cd
     <installerLocation>/appviewx_kubernetes/yaml/appviewx_plugins/avx_platform_web
   ```

   b. To upgrade the avx-platform-web package to reflect changes, execute the following command:

   ```
   helm upgrade avx-platform-web ./chart
   ```

```
[appviewx@appviewx-kube-install                                    ]$ helm upgrade avx-platform-web ./chart
Release "avx-platform-web" has been upgraded. Happy Helming!
NAME: avx-platform-web
LAST DEPLOYED: Wed Sep  9 10:49:09 2020
NAMESPACE: default
STATUS: deployed
REVISION: 2
TEST SUITE: None
[appviewx@appviewx-kube-install                                    ]$
```

4. Verify the application URL to check SSL is enabled.
5. Verify the certificate by launching the Appviewx portal.

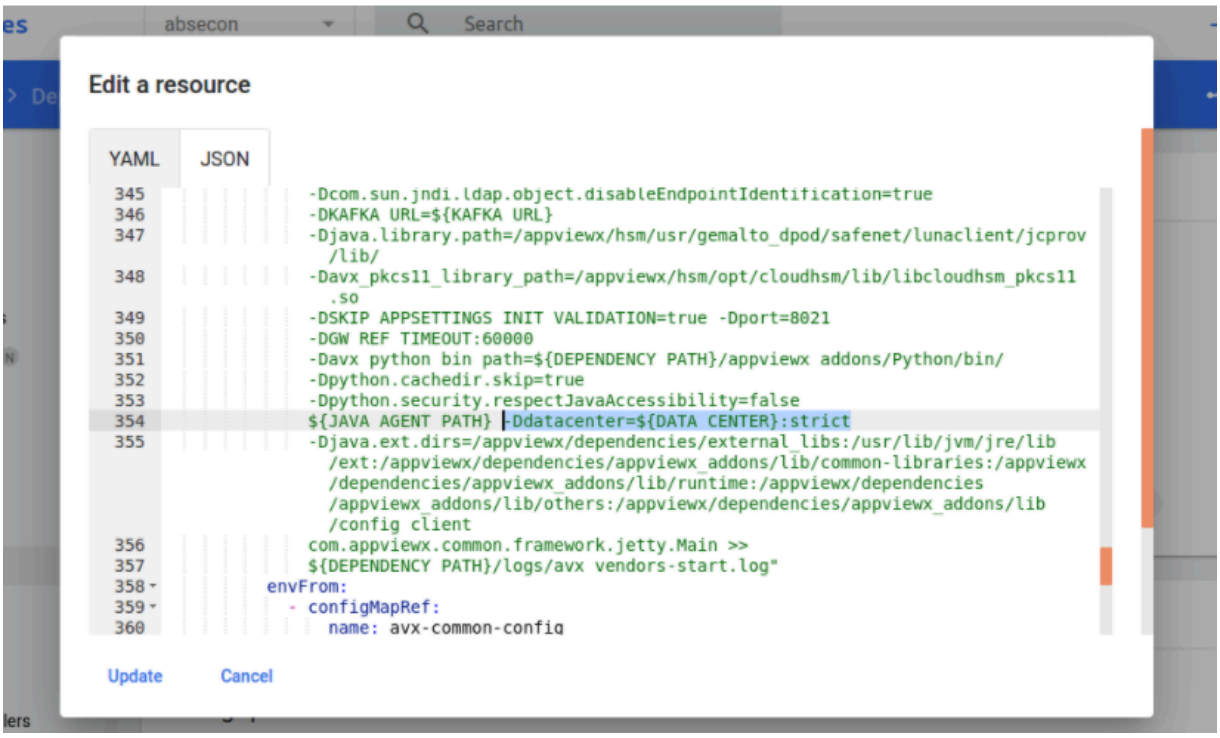
   The URL is https://<Service URL>:Port>/appviewx

Enabling Strict Data Center Routing

Strict data center routing is used to ensure that calls from AppViewX to a plugin/device through one data center are not routed to any other data center if there are no plugins available to serve traffic in the same data center.

To enable strict data center routing:

1. Log in to the Kubernetes dashboard of AppViewX.
2. On the left pane, under **Workloads**, click **Deployments**.
3. Search for the respective deployment to modify it.
4. Click **Edit**.
5. Add the argument **:strict** in **-Ddatacenter** jvm argument as shown below:



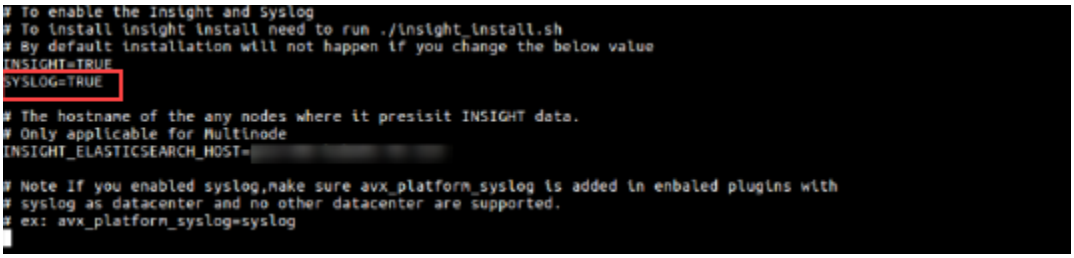
6. Click **Update**.

Enabling Device Syslog Processing

The Syslog module in AppViewX is used to receive syslogs from the device and update the necessary changes made in the device into the AppViewX database.

To enable Syslog parsing for the devices managed by AppViewX:

1. Navigate to the `/home/appviewx/appviewx_kubernetes/scripts` directory.
2. To open the `appviewx.conf` file, execute the following command: `vi appviewx.conf`
3. Search for the `SYSLOG` parameter.
4. Set the value of the `SYSLOG` parameter to `TRUE`.



5. Search for `Enabled Plugins`.
6. Add the following plugins:
   - `appviewx_dependencies`
   - `avx_platform_syslog`
   - `avx_platform_gateway`

   Note: Gateway must be added to register the new APIs from the plugins that are installed.

7. Update the data center as **syslog** for **avx_platform_syslog** plugin. (`avx_plaform_syslog=syslog`).

8. Save and exit the `appviewx.conf` file.
9. From the `/home/appviewx/appviewx_kubernetes/scripts` directory, execute the following command: `./insight_install.sh`
10. Execute the following command:

```
./plugins_install.sh
```

11. Execute the following command:

```
kubectl get services -n syslog
```

It displays the results as shown in the image below. Fetch the Syslog port from the service logstash-syslog-service. Here, the Syslog port is 30336.

This Syslog port changes for every installation/upgrade.

12. Connect to the MongoDB and open the **avx_app_metadata** collections. Edit this file by searching the parameter SYSLOG_RECEIVER_ENABLED and set it to TRUE. Save the file and move out of the DB.
13. To configure Syslog as `TRUE`, execute the following command:

```
kubectl edit configmaps -n "data center name" Set SYSLOG_RECEIVER_ENABLED=True,SYSLOG_HOST=192.168.XXX.XXX (Node IP where Syslog is installed)),SYSLOG_PORT=30047 (fetch the ports from point 8)
```

14. Save and exit the `<configmaps>` file.
15. To get the Pod name, execute the following command:

```
kubectl get pods -n "data center name"
```

16. To restart subsystems and vendors, execute the following command:

```
kubectl delete pods “Pod name” -n ‘‘data center name’’
```

For example: You can restart multiple pods and the config server by entering the name of the pod and config server in the command below with space.

```
kubectl delete pods avx-subsystems-7666cfb459-6q4rn avx-vendors-99c69cd69-jtr4w avx-config-server-85ff9dd46d-h5qnr-n ''data center name''
```

Enabling the Insight Module

The Insight module allows you to collect statistics from the devices that are managed by AppViewX. Also, it displays historical statistics on demand for users.

To install Insight for statistics collection:

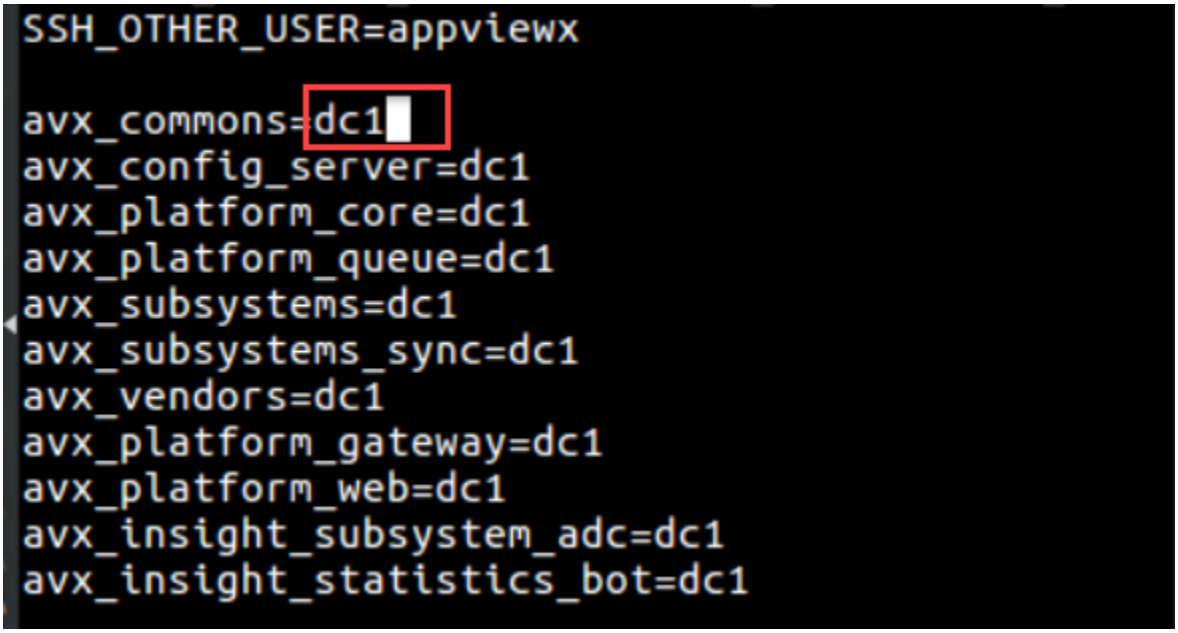
1. Open the terminal.
2. Navigate to the `/home/appviewx/appviewx_kubernetes/yaml` directory.
3. Download the `appviewx_kubernetes_insight_2022.1.0.tar.gz` file.
4. To extract the file, execute the following command:`tar -xvf appviewx_kubernetes_insight_2022.1.0.tar.gz`
5. Navigate to the `/home/appviewx/appviewx_kubernetes/scripts` directory.
6. To open the `appviewx.conf` file in the editor mode, execute the following command: `vi appviewx.conf` The value of the `INSIGHT` parameter is set to `TRUE` as shown in the image below.

```
# To enable the Insight and Syslog
# To install insight install need to run ./insight_install.sh
# By default installation will not happen if you change the below value
INSIGHT=TRUE
SYSLOG=TRUE

# The hostname of the any nodes where it presisit INSIGHT data.
# Only applicable for Multinode
INSIGHT_ELASTICSEARCH_HOST=

# Note If you enabled syslog,make sure avx_platform_syslog is added in enbaled plugins with
# syslog as datacenter and no other datacenter are supported.
# ex: avx_platform_syslog=syslog
```

7. Search for `Enabled Plugins` and add the following plugins:
   - `appviewx_dependencies`
   - `avx_insight_subsystem_adc`
   - `avx_insight_statistics_bot`
   - `avx_platform_gateway`
8. Update the data center for insight plugins as shown in the image below:



9. Save and exit the `appviewx.conf` file.
10. To install Insight, navigate to the `/home/appviewx/appviewx_kubernetes/scripts` directory.
11. Execute the following command:

```
./insight_install.sh
```

12. Execute the following command:

```
./plugins_install.sh
```

13. Execute the following command:

```
kubectl edit configmaps -n absecon
```

14. Search for the `ELASTIC_ENABLE` parameter and set the value to `TRUE`.
15. Ensure that the following parameters have the default values as given below:
    - `ELASTIC_ENABLE=TRUE,`
    - `ELASTIC_CLUSTER_NAME=elasticsearch`
    - `ELASTIC_HOST=elasticsearch-insight.statistics.svc.cluster.local`
    - `ELASTIC_PORT=9200`
    - `ELASTIC_HTTPS=FALSE`
    - `ELASTIC_TRANSPORT_PORT=9300`
16. Save and exit the `configmaps` file.
17. To restart subsystems and vendors, execute the following command:

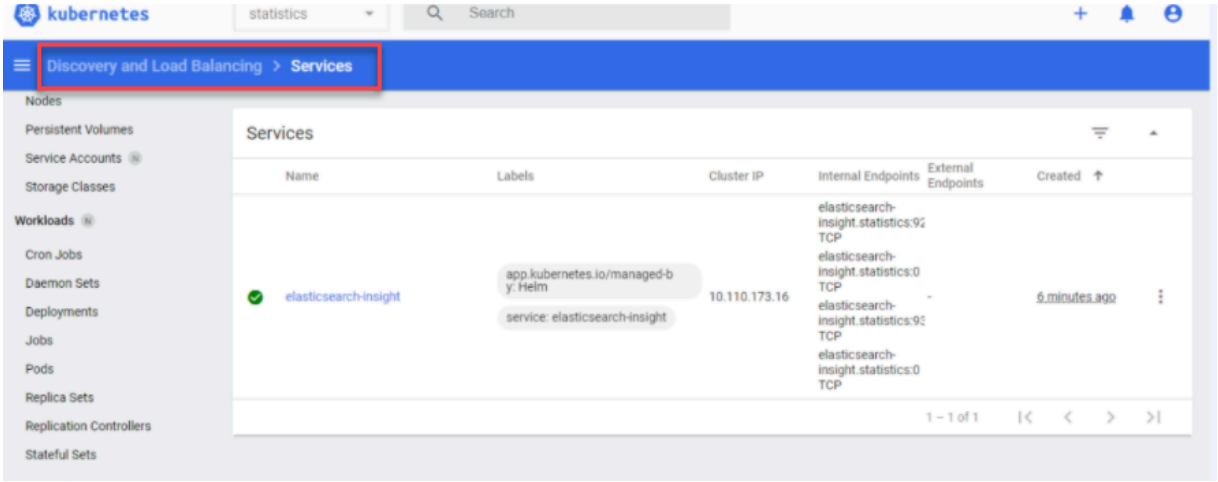
```
kubectl delete pods "Pod name" -n "datacenter name"
```

    For example:

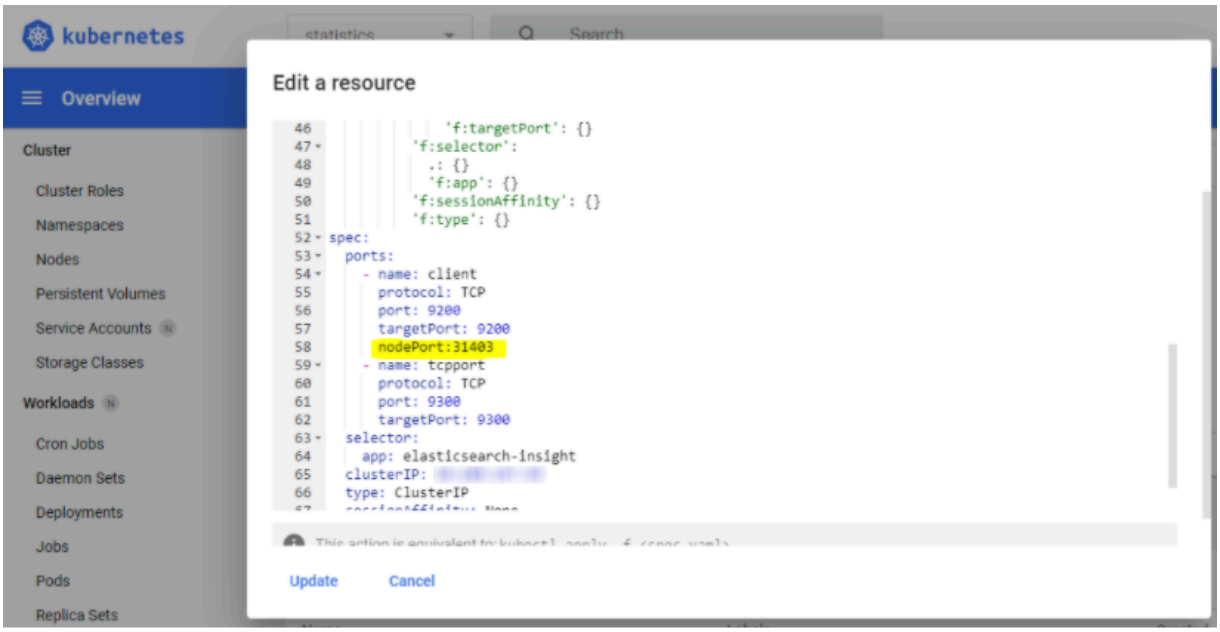
```
kubectl delete pods avx-insight-statsitics-bot-3499c69cd6-4sdfs,avx-insight-subsystem-adc-4399c69ed6-4sdfs,avx-subsystems-7666cfb459-6q4rn -n absecon
```

    To restart multiple Pods, enter the name of the pod in the above command with space.

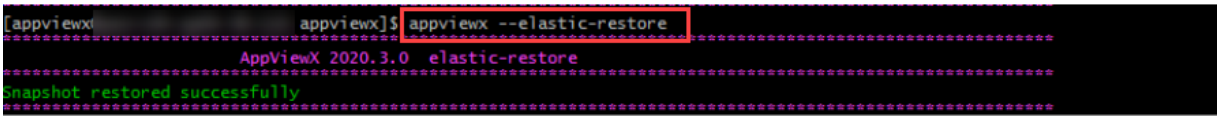
18. In the case of Insight migration, continue till point 11.
19. Log in to the Kubernetes dashboard, enter statistics as the namespace.
20. Select services and search for **elasticsearch-insight**.



21. On the Pod, click **Edit**.
22. Enter the port details as **nodePort: 31403** and save as shown in the image below:



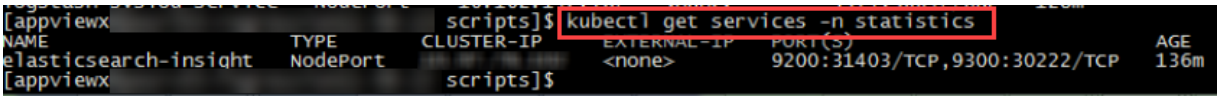
23. To restore the elastic data, go to the old installation path of AppViewX, and execute the following command:`appviewx --elastic-restore`



24. To connect to the elastic database, execute the following command:

```
kubectl get services -n statistics
```

It displays the results as shown in the image below:



Understanding Commands Executed during Installation

The section lists the commands executed by the AppViewX installer that requires Sudo access.

To restrict the commands that a Sudo user has access to, enable the following commands:

- `sudo kubeadm`
- `sudo kubectl`
- `sudo yum remove`
- `sudo yum install`

- `sudo systemctl daemon-reload`
- `sudo rpm -ivh --force *.rpm`
- `sudo modprobe br_netfilter`
- `sudo swapoff -a`
- `sudo iptables -F`
- `sudo date --set`
- `sudo -S setenforce 0`
- `sudo -S sysctl -w net.bridge.bridge-nf-call-iptables=1`

Apart from the above commands, Sudo user must be able to read/write/execute in the following directories:

- `/etc/`
- `/root/`
- `/var/lib`
- `/tmp`
- `/usr/local/bin`
- `/home/SSH_OTHER_USER` (Other user is user-defined in /scripts/appviewx.conf)

Enabling Sudo Access

To enable Sudo access and grant access to all commands:

1. Log in as an Administrator.
2. Log in to the node with root credentials.

Creating a New Sudo User

To create a new Sudo user:

1. Open the terminal.
2. Execute the following command:

```
adduser UserName
```

Note: Replace the UserName with the new user's name.

3. To create a password for the new user, execute the following command:

```
passwd UserName
```

The system prompts you to set and confirm a password for your new user account. If successful, the system responds with “all authentication tokens updated successfully.”

Note: A strong secure password has more characters and a few special characters (such as numbers, symbols, or capitals). Ensure that you are choosing an appropriately strong password for your system.

Adding Users to the Sudo Group

For CentOS

By default, CentOS 7 has a user group called the Wheel group. Members of the wheel group are automatically granted Sudo privileges. Adding a user to this group grants Sudo privileges to the user.

To add a user to the wheel group, refer to Adding a User to the Wheel Group.

For Ubuntu

In Ubuntu OS, a user in “sudo” group is granted sudo permissions.

To add a user to the sudo group:

Execute the following command:

```
usermod –aG scudo UserName
```

Note: Replace the UserName with the new user's name to grant Sudo privileges.

Verifying if the Wheel Group is Enabled

To verify whether CentOS 7 installation has the wheel group enabled or disabled:

1. To open the configuration file, execute the following command:

```
vi sudo
```

2. Search for the following entry in the configuration file:

```
## Allows people in group wheel to run all commands %wheel ALL=(ALL) AL
```

   If the second line begins with the # sign, it indicates that the line is marked as a comment and the feature is disabled.

3. Delete the # sign at the beginning of the second line as given below.

```
%wheel ALL=(ALL) ALL
```

4. Save the file and exit the editor.

> **Note:** If there is no # sign at the beginning of the line, do not make any changes. The wheel group is already enabled.

Adding a User to the Wheel Group

Adding a user to the wheel group is applicable for CentOS.

To add a user to the wheel group:

Execute the following command:

```
usermod –aG wheel UserName
```

> **Note:** Replace the UserName with the new user's name to grant Sudo privileges.

Switching to the Sudo User

To switch to the new (or newly-elevated) user account with the su (substitute user):

1. Execute the following command:

```
su - UserName
```

2. Enter the password if prompted.
3. To list the contents of the /root directory, execute the following command:

```
sudo ls -la /root
```

4. Enter the password if prompted.
   The terminal displays the list of directories. Since listing the contents of the /root directory requires Sudo privileges, this is an easy way to prove that the new user can use the Sudo command.

Understanding the Best Practices on Reboot Sequence

This section provides information on the best practices to be followed for rebooting the operating system after security patching.

Note: Before you perform these steps, ensure that all prerequisites are complied with as mentioned in the Configuring YUM section.

The steps are to be executed in the order given below.

1. Log in into the AppViewX worker node from where the installation has been initiated.
2. Navigate to `<installer directory path>/appviewx_kubernetes/scripts`
3. Take a backup of the scripts directory from `/appviewx_kubernetes/scripts`
4. Download the latest `scripts.tar.gz` from the release portal.
5. Copy the existing `appviewx.conf` file from the older scripts folder to the newly downloaded scripts folder from the release portal.
6. Execute the commands from the installer location/scripts folder.

   Note: The Stop all and Start all commands are applicable only for a multi node setup.

7. To drain all the pods, execute the following command:

   ```
   ./appviewx.sh --stop -all
   ```

   The command will drain the pods in the nodes in the following order; Worker, Secondary master(if any), Master.
8. Shut down the nodes in the order mentioned in step 7.
9. Start the nodes in the reverse order; Primary master, Secondary masters, and Workers from the primary mongo as per `appviewx.conf` entries.
10. To start all the pods in the nodes, execute the following command:

    ```
    ./appviewx.sh --start -all
    ```

    This command will start the pods in the nodes in the following order; Master, Secondary master(if any), Worker.

Working with Alerts

Alerts are used to notify users when a predefined target or a condition is met. For example, if the memory usage for a cluster exceeds 90%, you can set an email notification to be sent to the users. This type of notification helps in mitigating the dangers of application downtime that might occur when parameters or go unnoticed.

The following alerts are available:

- Application Alerts
- System Alerts
- Enabling an email Alert
- Troubleshooting Alerts

Enabling an email Alert

AppViewX enables the administrator to send out an email to designated email addresses if the `appviewx.conf` file is modified.

To enable an email alert when the `appviewx.conf` file is modified:

1. Open the terminal.
2. Navigate to the `<avx_installed_path>/conf` directory.
3. To open the `appviewx.conf` file, execute the following command:

   ```
   vi appviewx.conf
   ```

4. Update the following SMTP fields in the `appviewx.conf` file.SMTP_SERVER = <email server>:<port> SMTP_SENDER_USER = <sender email address> SMTP_RECEIVER_USER = <sender email address>
5. To get an email alert if the file is tampered, execute the following command: `./appviewx --conf_change_alert cron`
6. To set the command in crontab, complete the following steps:

   ```
   crontab -e

   <cron freq> cd /home/appviewx/appviewx/scripts && ./appviewx --conf_change_alert

   cron 2>>/home/appviewx/appviewx/logs/cron_logs 1>/dev/null
   ```

Troubleshooting Alerts

Note: For troubleshooting issues, please refer to the Troubleshooting section.

Working with Backup and Restore

The application level backups are no longer supported in AppViewX. You can back up the mongodb and vault and restore the same in the event of any failure. To facilitate this process, there are scripts available for mongodb and vault backup and restore. You can download them from the release portal.

- Downloading the Scripts
- Performing a Backup for MongoDB and Vault
- Restoring a MongoDB Backup
- Restoring the Vault Backup
- Troubleshooting Backup and Restore Operations

Downloading the Scripts

The scripts are used to trigger the backup and restore operations. The backup files will be created under the directory mentioned in the scripts.

Download the following scripts from the release portal:

- `mongo_backup.sh`
- `vault-backup.sh`
- `vault_restore.sh`
- `mongo_restore.sh`

Copy all the files to the `<appviewx_installer_location_path>/appviewx_kubernetes/scripts` directory.

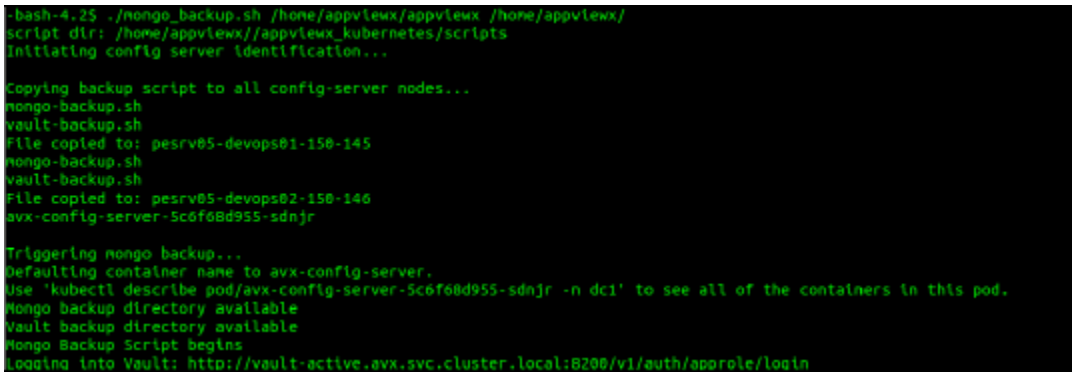
Performing a Backup for MongoDB and Vault

1. Open the terminal.
2. Execute the following command:

```
sh mongo_backup.sh <appviewx_installed_Path> <appviewx_installer_location_path>
```

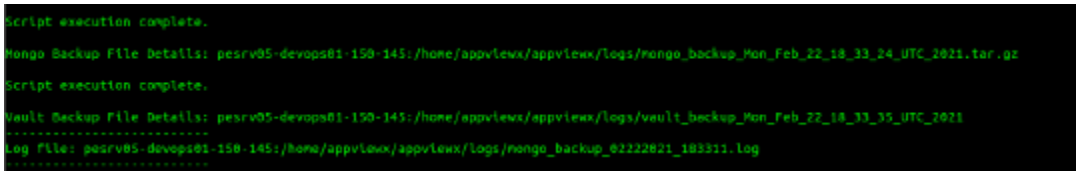
where,

- `appviewx_installed_Path` - specifies the path where AppViewX is installed
- `appviewx_installer_location_path` - specifies the location of the AppViewX installer file.



The command creates a backup of both MongoDB and Vault. The backup files will be available as listed below:

- Mongo Backup file path: `<Installation path>/logs/<mongo file with timestamp>`
- Vault Backup file path: `<Installation path>/logs/<vault file with timestamp>`



The backup is maintained for a period of five days. The backup data will be purged after this period.

Restoring a MongoDB Backup

1. Open the terminal.
2. Execute the following command with parameters like Installation Path, appviewx installer location Path, target pod name (avx-config-server), and mongo backup file name.

```
sh mongo_restore.sh <appviewx_installed_Path> <appviewx_installer_location_path> <avx-config-server> <mongo-backup-file-path>
```

Restoring the Vault Backup

1. Open the terminal.
2. Execute the following command:

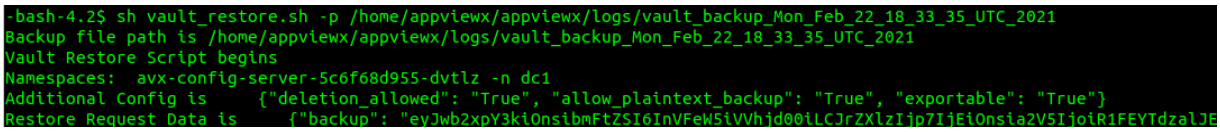
```
sh vault_restore.sh -p <vault_backup_file_path>
```

where,

- `vault_backup_file_path` - specifies the location of the vault backup file to be restored.

Example:

```
sh vault_restore.sh -p /home/appviewx/appviewx/logs/vault_backup_Mon_Feb_22_18_33_35_UTC_2021
```



Troubleshooting Backup and Restore Operations

Note: For troubleshooting issues, please refer to the Troubleshooting section.

Working with Logs

In any application, log files are used to record all events.It provides information about the customer usage patterns, the names of modules that are used frequently. In addition, they also help users analyse the issues depending on the events.

In any application, there are mainly two types of logs that are collected. One of them is application logs that are required to monitor the performance. Another type of log that is maintained is infrastructure logs. These logs are used to monitor the status of the hardware infrastructure like memory usage, disk usage, and CPU usage.

In AppViewX, the log files are collected and maintained for plugins. To manage logs, AppViewX uses Kibana.

- Managing Logs using Kibana
- Managing Logs using AppViewX Nodes

Managing Logs using Kibana

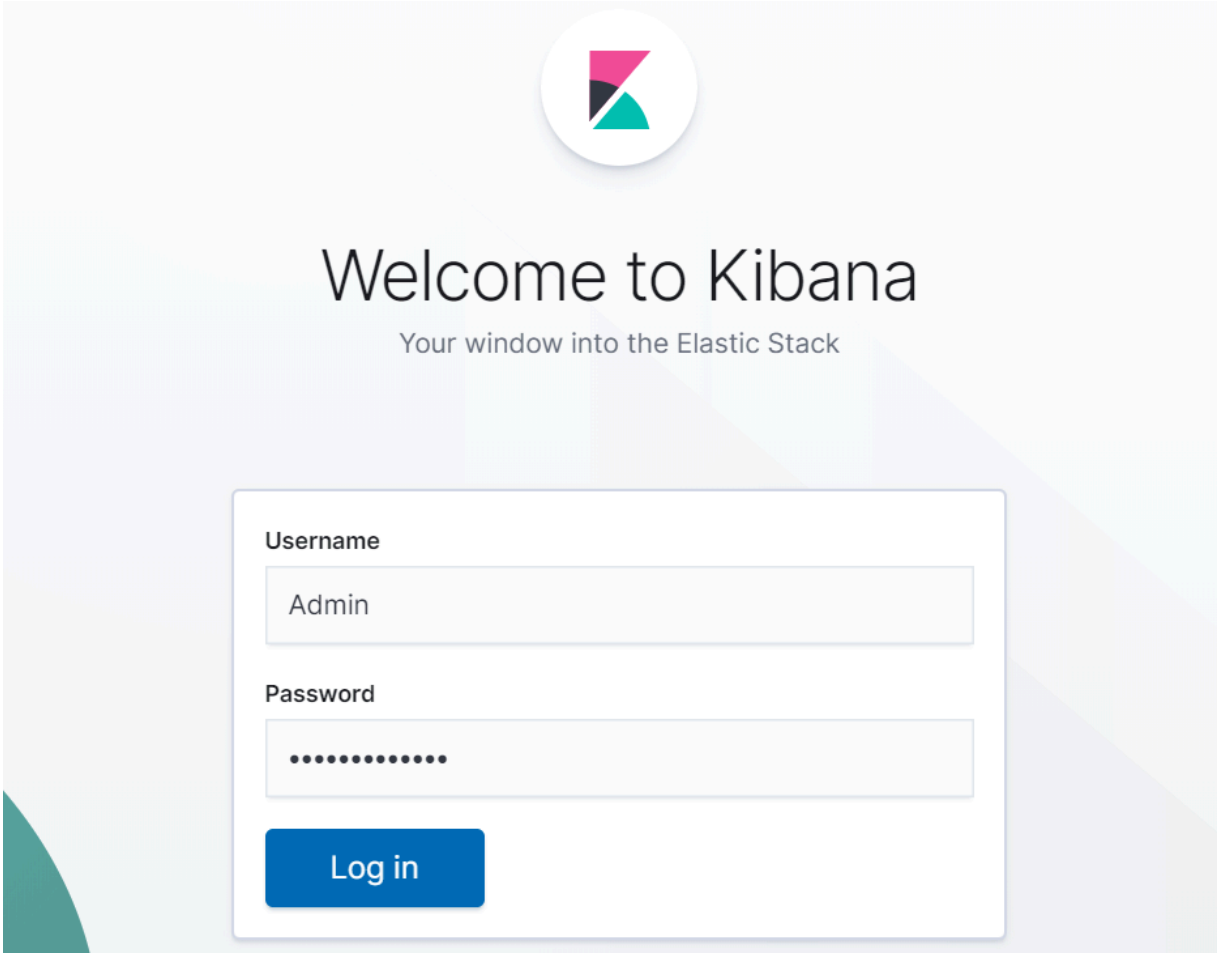
Kibana is an open user interface that enables you to graphically represent the log files and monitor system performance.

Before using Kibana, ensure that you have the following:

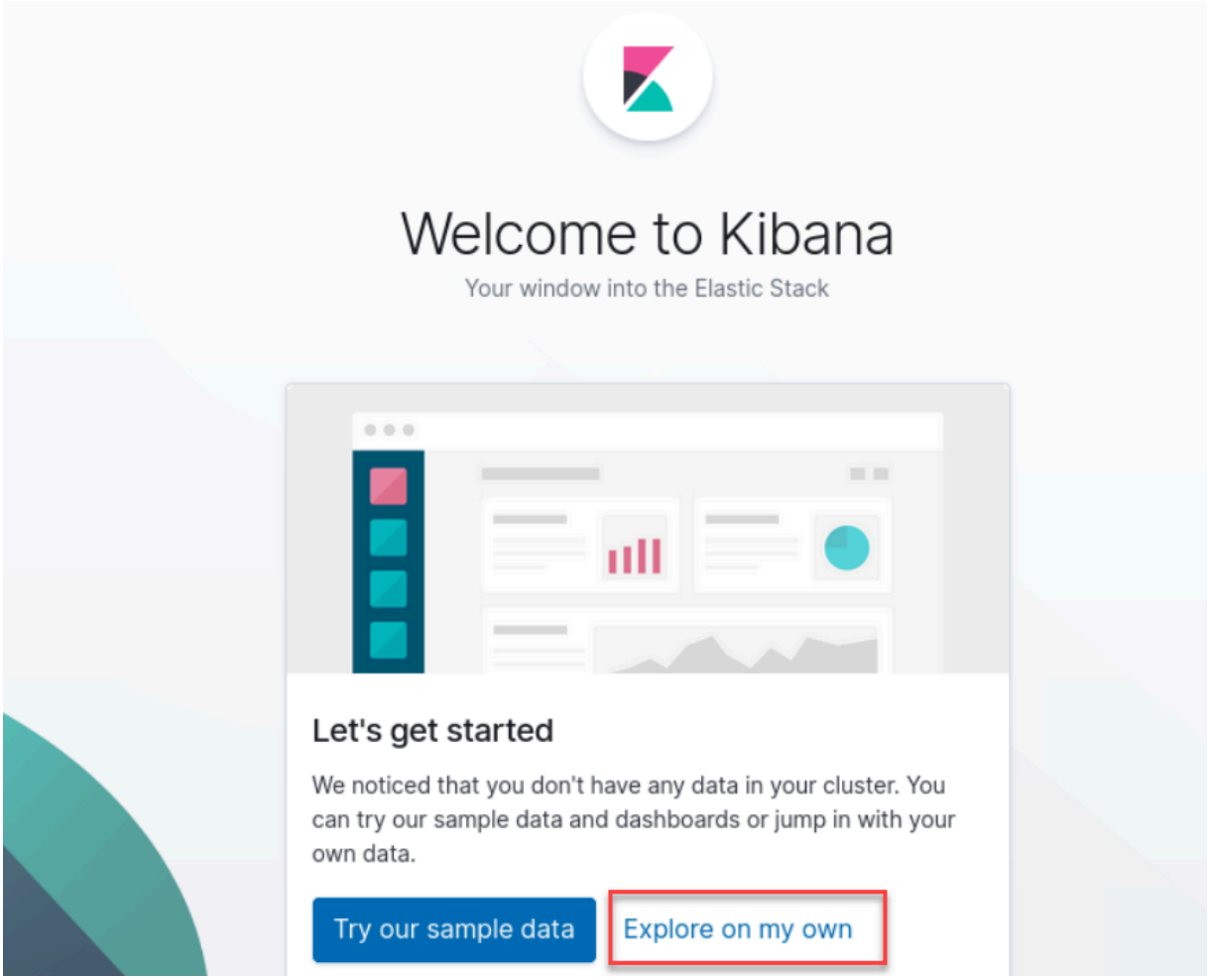
- `Kibana Web URL` - mentioned in the `<INSTALLATION_PATH>/.appviewx_configuration` file
- Kibana Username - mentioned in the `<INSTALLATION_PATH>/.appviewx_configuration` file
- `Kibana Password` - mentioned in the `<INSTALLATION_PATH>/.appviewx_configuration` file

Accessing Kibana

1. Open the Kibana Web URL.
2. Enter the credentials.
3. Click **Log in**.

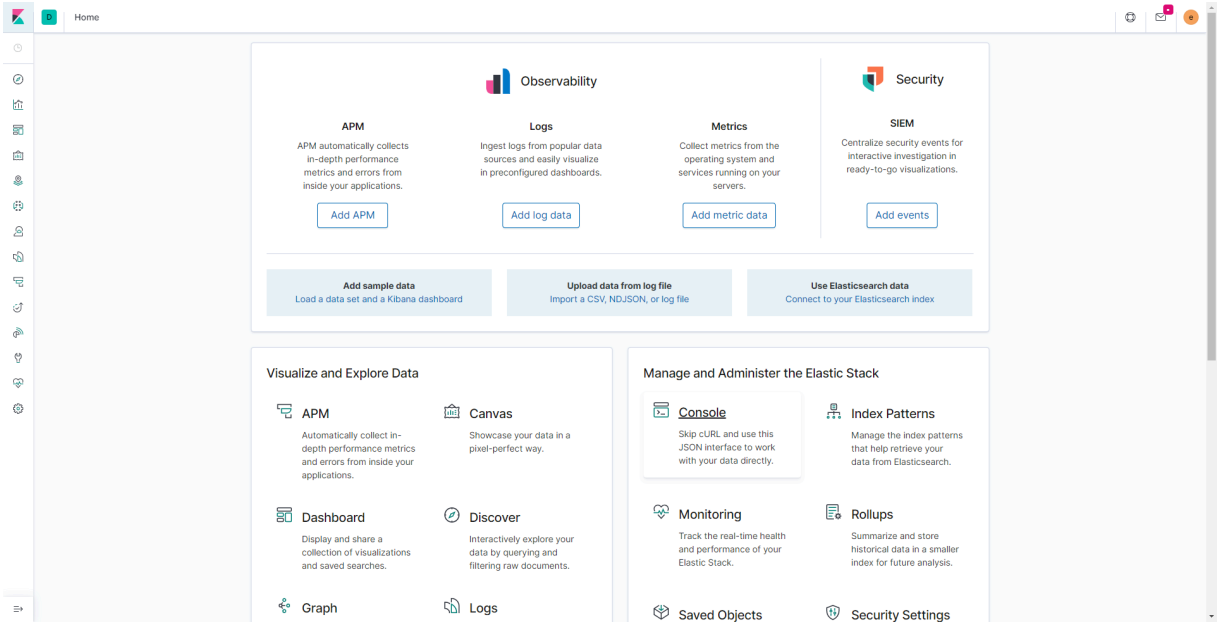


The **Let's get started** page is displayed.



4. Click **Explore on my own**.

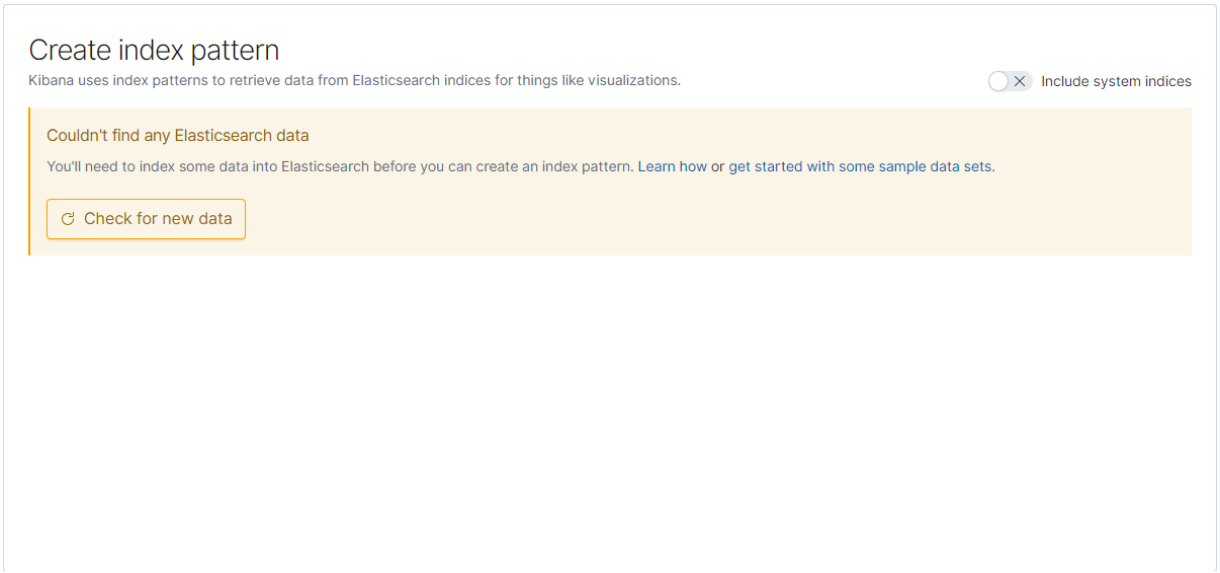
The **Home page** is displayed.



Creating an Index Pattern

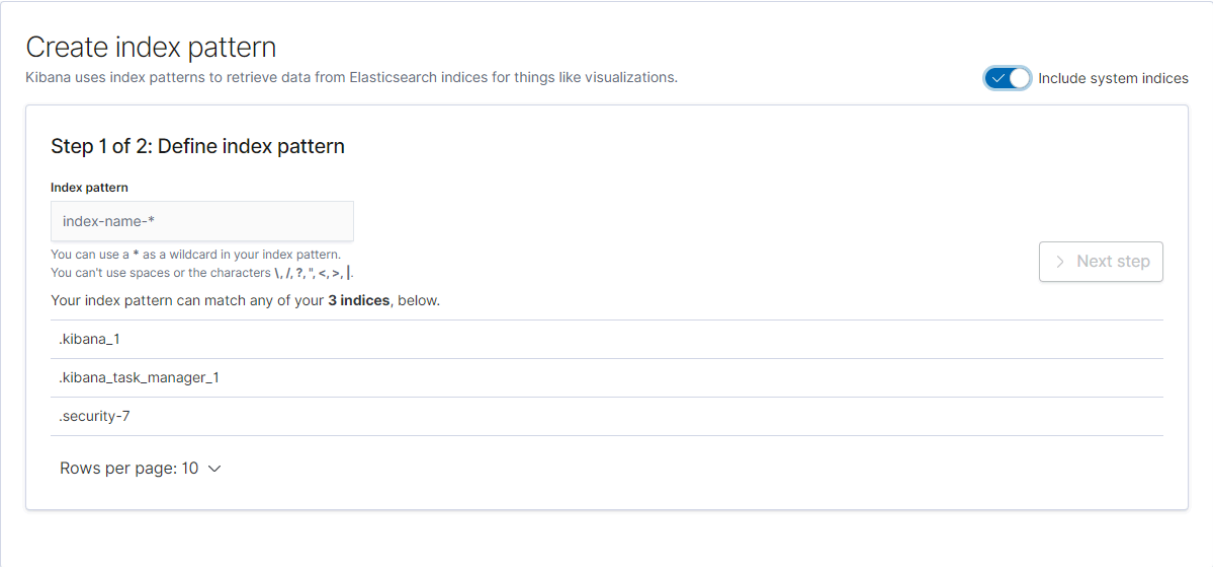
1. Login to Kibana.
2. Under **Visualize and Explore Data**, click **Visualize**.

   The **Create index pattern** page is displayed.



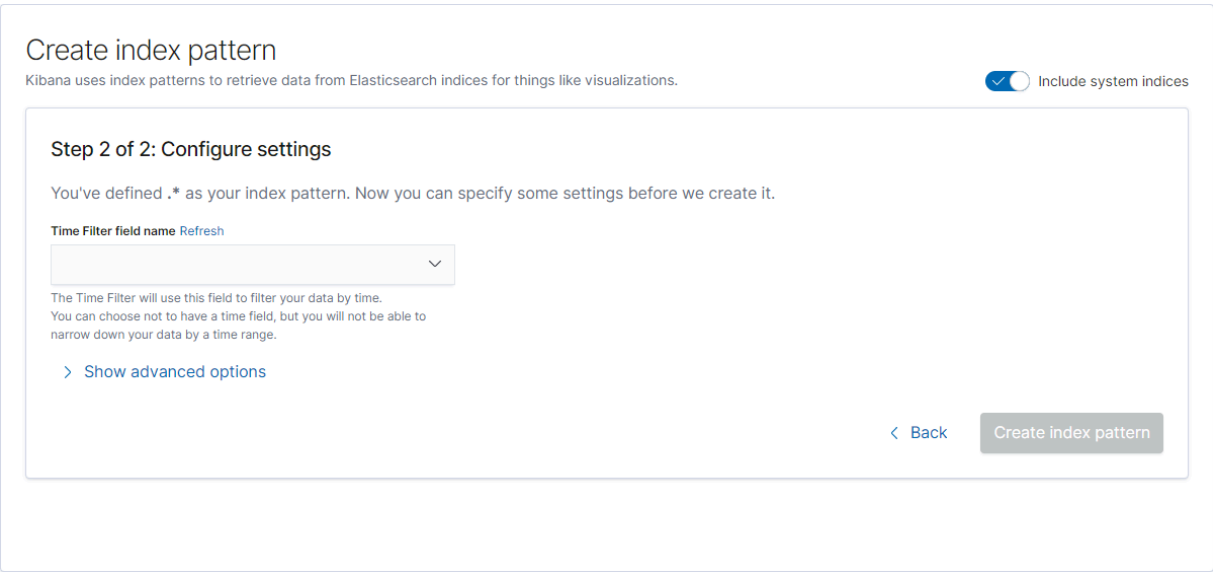
3. Enable the **Include system indices** option.

   The **Define index pattern** page is displayed.



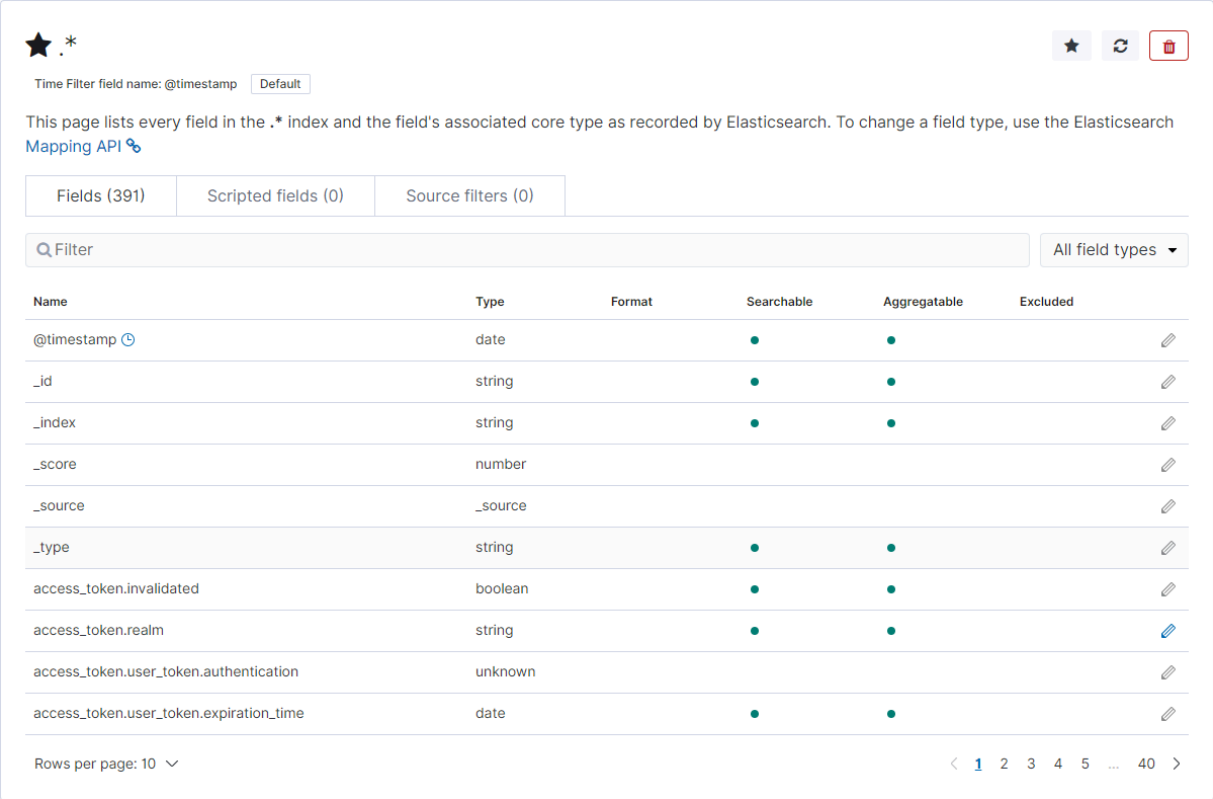
4. Under Index pattern, enter .*.
5. Click **Next step**.

   The **Configure settings** page is displayed.



6. From the **Time Filter field name** list, select **@timestamp**.
7. Click **Create Index Pattern**.

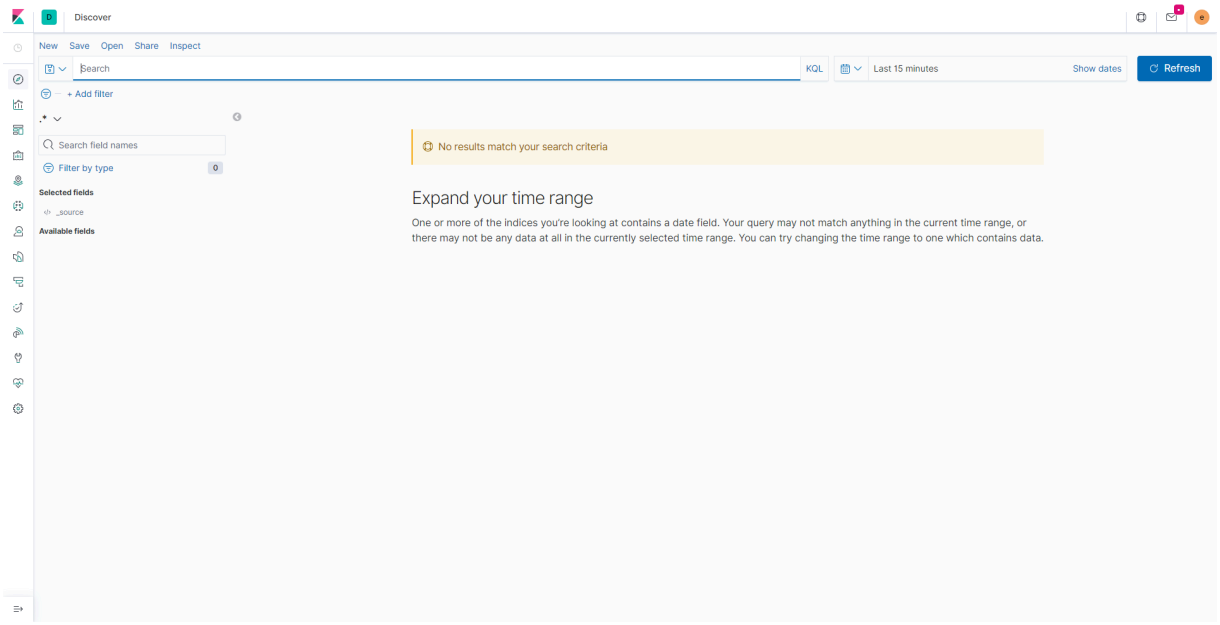
The system creates an index pattern.



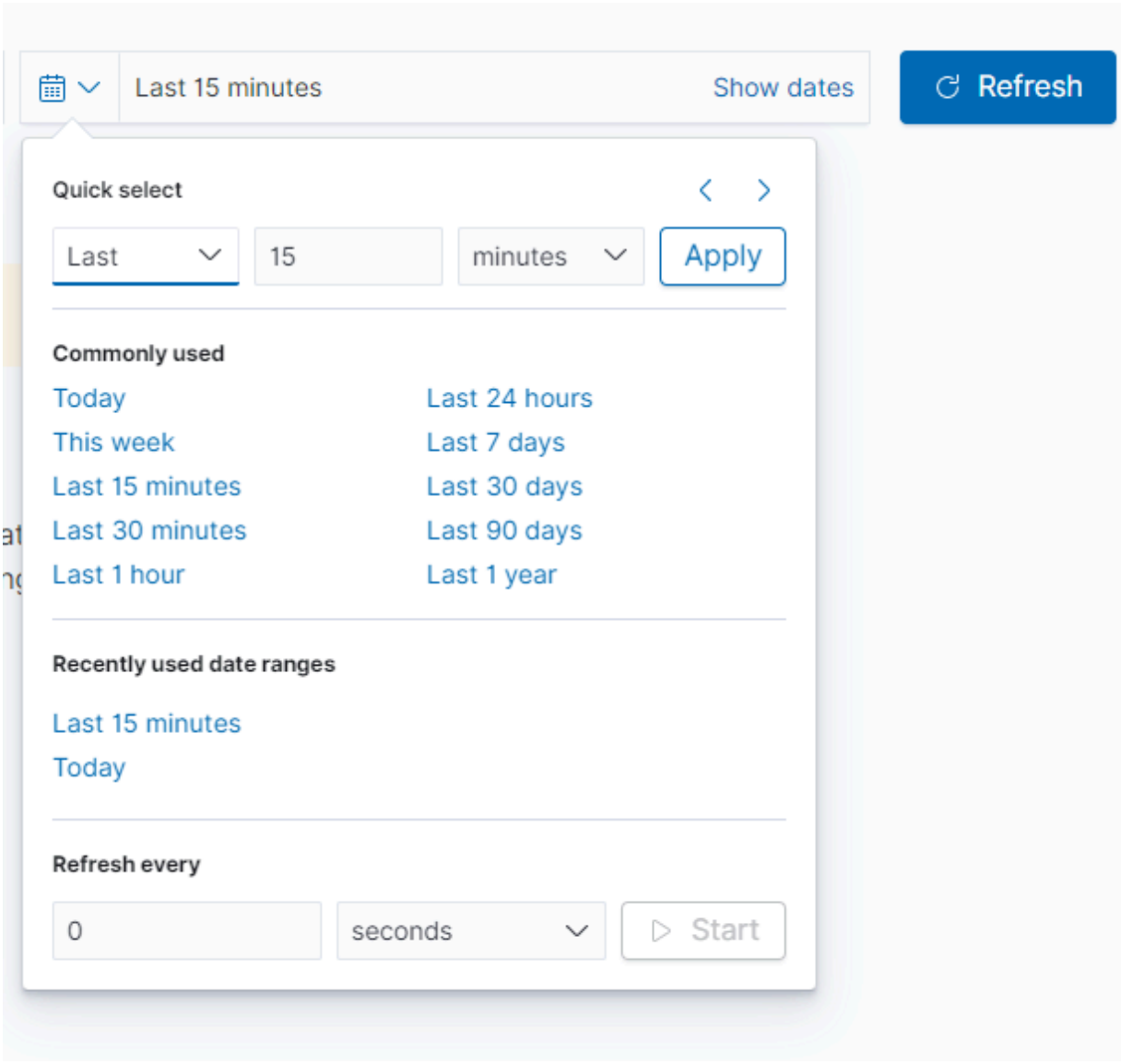
Viewing Logs

1. Login to Kibana.
2. Under **Visualize and Explore Data**, click **Discover**.

   The **Discover** page is displayed.



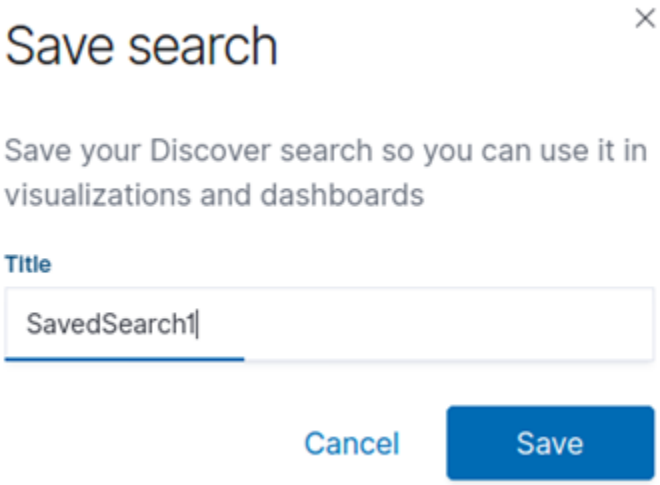
3. In the time frame section, select the time frame within which the logs need to be captured.



4. To view the updated logs, click **Refresh**.
5. To save the search:

a. Click **Save**.

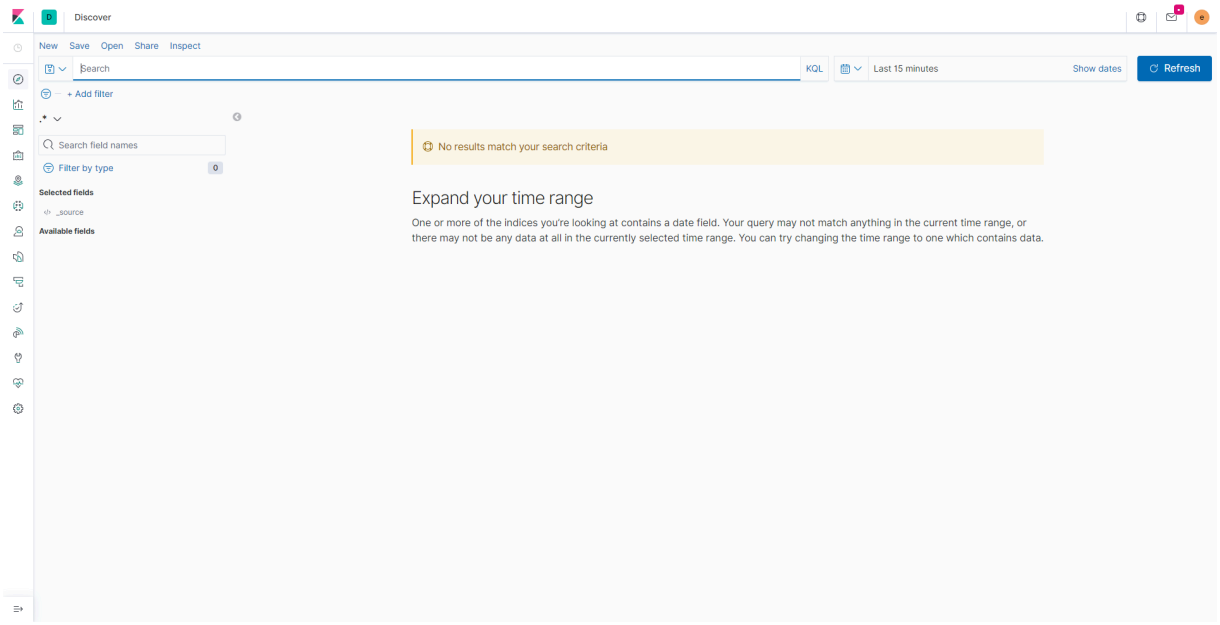
b. Enter a valid name to save the search.



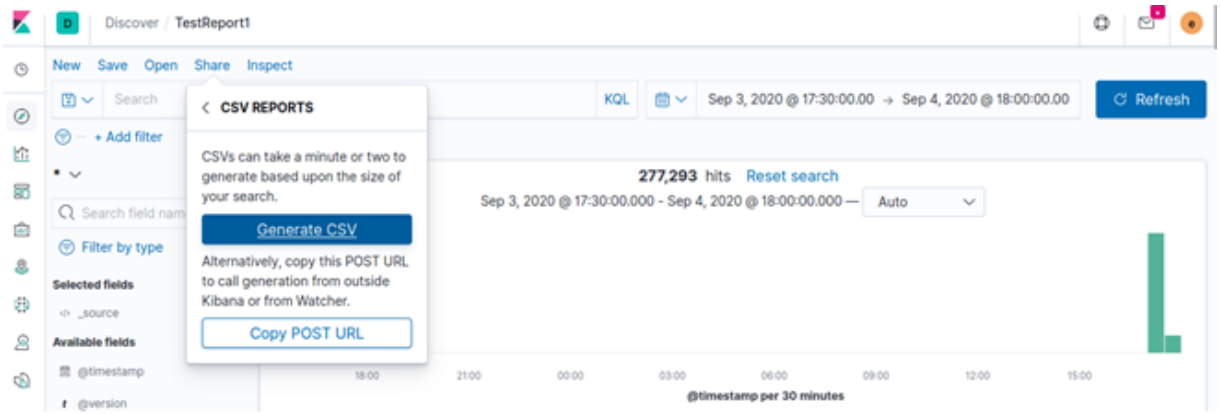
Generating a Report

Kibana enables you to generate a report in CSV format. In order to generate the report, you must copy the <.ndjson> ext files from the <InstallerLocation>/appviewx_kubernetes/yaml/appviewx_monitoring/kibana/deploy location and import into the import section (for example, <gateway.ndjson> and <platform.ndjson>).

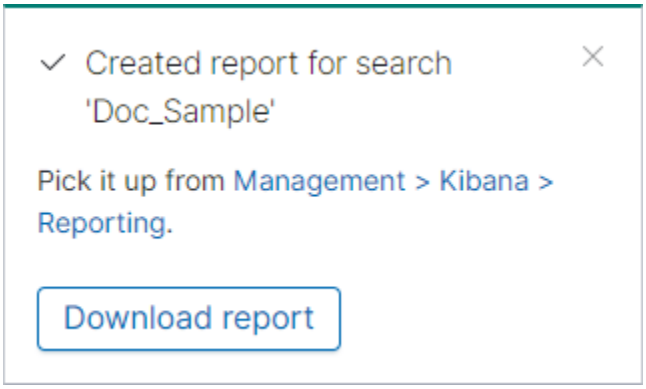
1. Login to Kibana.
2. Under **Visualize and Explore Data**, click **Discover**.
   The **Discover** page is displayed.



3. Select **Share > CSV reports > Generate CSV**.



The system generates the report and prompts to download the same.



4. To download the report, click **Download report**.
   The system downloads the report to the default download location.

Managing Logs using AppViewX Nodes

You can also view and manage the log files even if you do not have Kibana installed. In this case, you can use the AppViewX nodes to view and manage the log files. You can also view logs using the command line interface before you install the ELK.

Note: Logs are maintained as per the retention policy. Any log exceeding 30 MB will be rotated and archived as part of the data retention policy.

To view the logs:

1. Log in to the respective node.
2. Navigate to the appviewx/dependencies/logs directory.

   You can view the CLI logs for pods in the same node.

To view the logs from the AppViewX nodes:

1. Using the command line interface, log in to the AppViewX node.
2. To fetch the node name in which the pod is running, execute the following command:

   ```
   kubectl get pods- n <dc> -o wide
   ```

3. Log in to the respective node using SSH.
4. Navigate to `<INSTALLATION_PATH>/logs` for all log files.

   For example, If you want to view the logs for the subsystem plugin in the datacenter DC1, execute the following command to get the node name of the pod:

```
kubectl get pods -n DC1 -o wide
```

```
[appviewx@gs-apvx-dev86 ~]$ kubectl get pods -n absecon -o wide
NAME                                   READY   STATUS    RESTARTS   AGE     IP           NODE         NOMINATED NODE
 READINESS GATES
avx-commons-696f66b88f-68pnq           2/2     Running   14         4d19h   [redacted]   [redacted]   <none>
 <none>
avx-config-server-765bc549c8-h92wt     2/2     Running   13         4d19h   [redacted]   [redacted]   <none>
 <none>
avx-platform-core-97d99cddd-c9q6c      2/2     Running   14         4d19h   [redacted]   [redacted]   <none>
 <none>
avx-platform-gateway-7c957fdd4f-2br5d  2/2     Running   15         4d19h   [redacted]   [redacted]   <none>
 <none>
avx-platform-queue-9dbcc9ccb-txmn5     2/2     Running   14         4d19h   [redacted]   [redacted]   <none>
 <none>
avx-platform-web-6b4df49fb6-2phqs      2/2     Running   0          4d19h   [redacted]   [redacted]   <none>
 <none>
avx-subsystems-75db48b9b4-5gfgk        2/2     Running   13         4d19h   [redacted]   [redacted]   <none>
 <none>
avx-subsystems-75db48b9b4-8xnl5        2/2     Running   18         10d     [redacted]   [redacted]   <none>
 <none>
avx-subsystems-75db48b9b4-9hwlv        2/2     Running   13         4d19h   [redacted]   [redacted]   <none>
 <none>
avx-subsystems-75db48b9b4-mm22c        2/2     Running   18         10d     [redacted]   [redacted]   <none>
 <none>
avx-subsystems-sync-7f59dc8b9-mlsc6    2/2     Running   18         10d     [redacted]   [redacted]   <none>
 <none>
avx-vendors-586f9db568-8vmcv           2/2     Running   0          4d16h   [redacted]   [redacted]   <none>
 <none>
[appviewx@gs-apvx-dev86 ~]$
```

Note: For troubleshooting issues, please refer to the Troubleshooting section.

Working with Plugins

- Adding a New Plugin
- Removing a Plugin
- Restarting a Plugin
- Scaling a Plugin
- Changing the Memory for a Plugin

Adding a New Plugin

During the AppViewX installation, the user may not enable all the plugins that are required. Therefore, the user can enable those plugins after the AppViewX installation.

To enable a plugin after installation:

1. Navigate to the `/home/appviewx/appviewx_kubernetes/scripts` directory.
2. Open the `appviewx.conf` file.
3. Modify the `ENABLED_PLUGINS` as new plugins that need to be installed.

> **Warning:** It is not recommended to delete the appviewx_dependencies in the ENABLED_PLUGINS value. For example, ENABLED_PLUGINS= avx_dependencies,avx_vendors.

```
ENABLED_PLUGINS=appviewx_dependencies,avx_platform_amc,avx_platform_gateway

SSH_OTHER_USER=appviewx

avx_platform_amc=dc1,dc2
avx_config_server=dc1,dc2
```

4. Enter the data center value in which the plugin needs to be installed.
   For example, `avx_vendors=dc1`.

```
-bash-4.2$ kubectl get pods -A | grep amc
dc1                  avx-platform-amc-68b9fbc7f-fj7wr                  2/2     Running     1          2d2h
dc2                  avx-platform-amc-68b9fbc7f-kv8k8                  2/2     Running     2          2d2h
-bash-4.2$
```

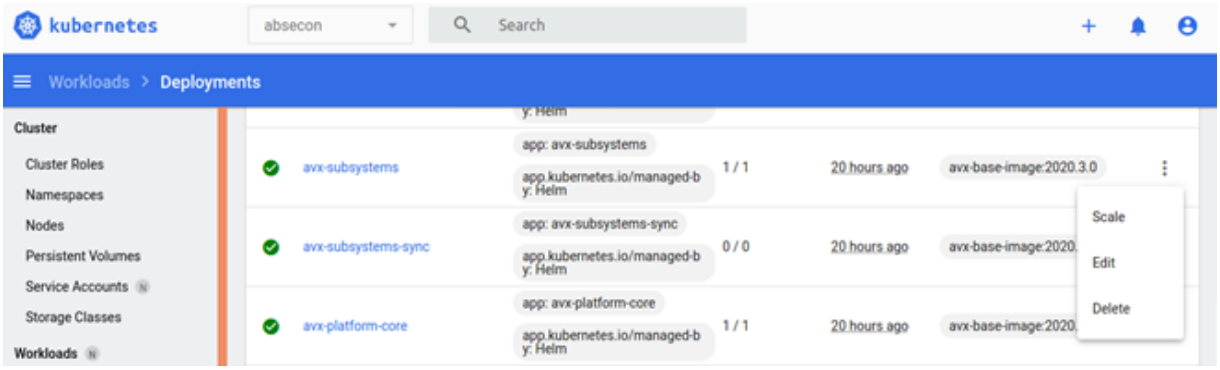
5. Save and exit the `appviewx.conf` file.
6. Navigate to the `scripts` directory.
7. In the scripts directory, execute the following command:

```
script plugins_install.sh
```

Removing a Plugin

To remove a plugin for maintenance purposes:

1. Log in into the kubernetes management console.
2. From the top list, select the required namespace or datacenter.
3. From the left pane, click **Deployments**.
4. Search for the specific deployment/plugin that needs to be stopped.
5. Against the name of the pod, click the three dots and select **Scale**.



The **Scale a Resource** page is displayed.

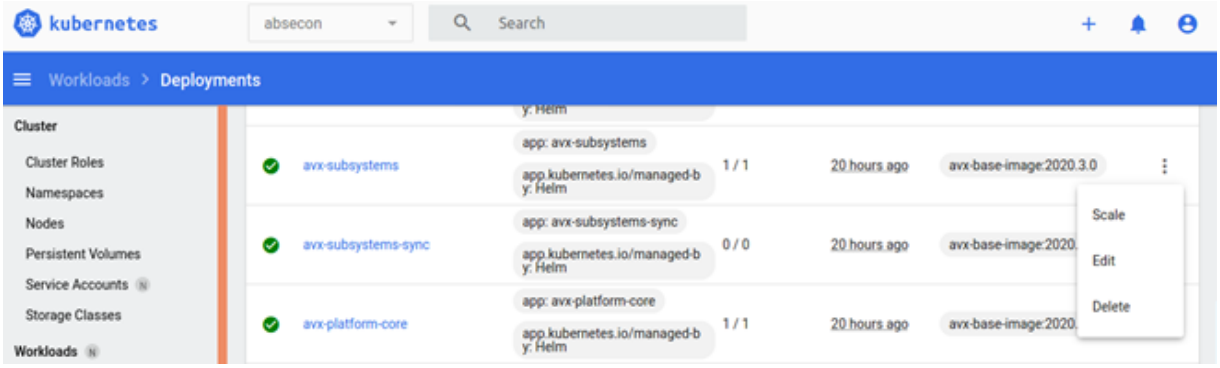
6. Set the value for **Desired replicas** to 0.
   This will delete all the pods and does not spin any new pod for that plugin.

Restarting a Plugin

1. Log in into the kubernetes management console.
2. From the top list, select the required namespace or datacenter.
3. From the left pane, click **Deployments**.

4. Search for the specific deployment/plugin that needs to be restarted.
5. Against the name of the pod, click the three dots and select **Delete**.

   This will stop the current pod and create a new pod.

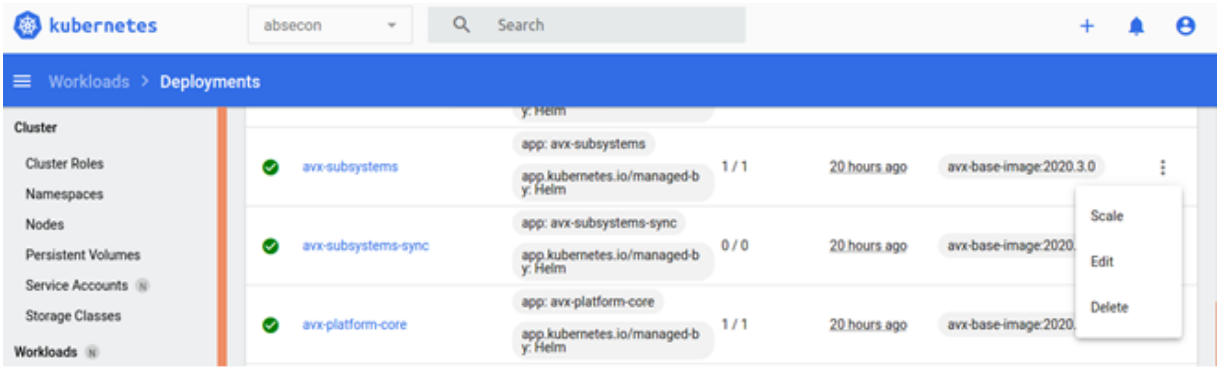


Scaling a Plugin

Scale refers to an increase or decrease in the number of plugins manually. You have an option to scale it from the Kubernetes management console.

To increase/decrease the number of plugins of a specific type:

1. Log in into the kubernetes management console.
2. From the top list, select the required namespace or datacenter.
3. From the left pane, click **Deployments**.
4. Search for the specific deployment/plugin that needs to be scaled.
5. Against the name of the pod, click the three dots and select **Scale**.



The **Scale a Resource** page is displayed.

6. Update the value of the **Desired replicas** parameter to increase or decrease the number of pods for a plugin.
7. Click **Scale**.

Changing the Memory for a Plugin

Every plugin inside the node runs on a dedicated memory. It can be adjusted to the maximum and minimum memory that a pod can use.

To increase or decrease the plugins memory:

1. Log in to the Kubernetes dashboard of AppViewX.
2. From the left pane, under **Workloads**, click **Deployments**.
3. Search for the respective deployment to modify it.
4. Click **Edit**.

5. Modify the xmx and xms values to the required values as shown below.

Working with the Management Console

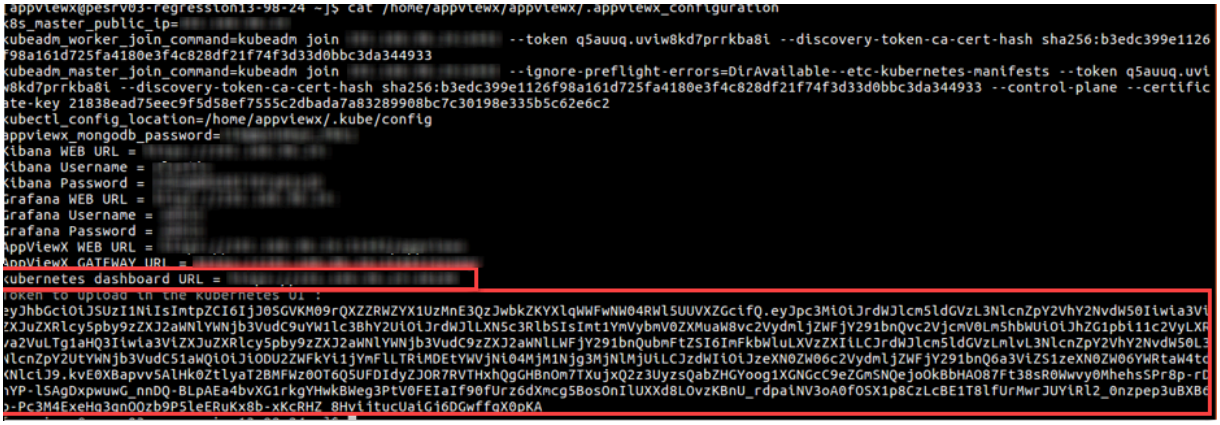
The management console allows you to monitor, maintain, and manage the application as well as the performance. The console provides a graphical interface to view and monitor the application instance.

- Accessing the Management Console
- Viewing the POD Status
- Accessing the POD Console
- Accessing the Database Command Line
- Exporting a Database Collection

Accessing the Management Console

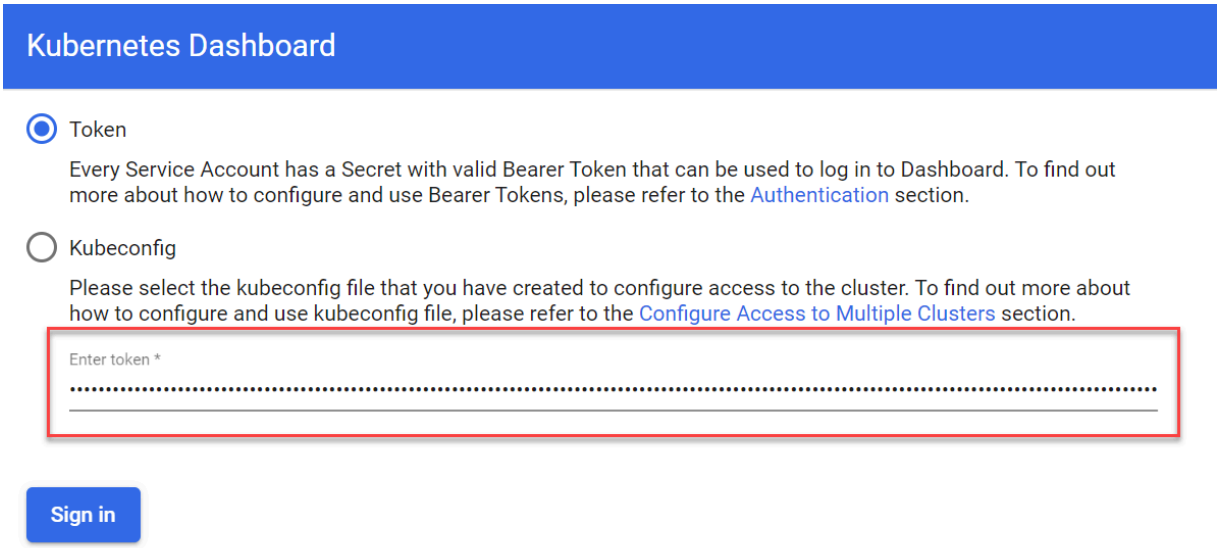
1. Navigate to the installation directory.
2. Open the .appviewx_configuration file.

   For example: cat/home/appviewx/appviewx/.appviewx_configuration
3. Search for Kubernetes dashboard URL and copy the URL to a web browser.

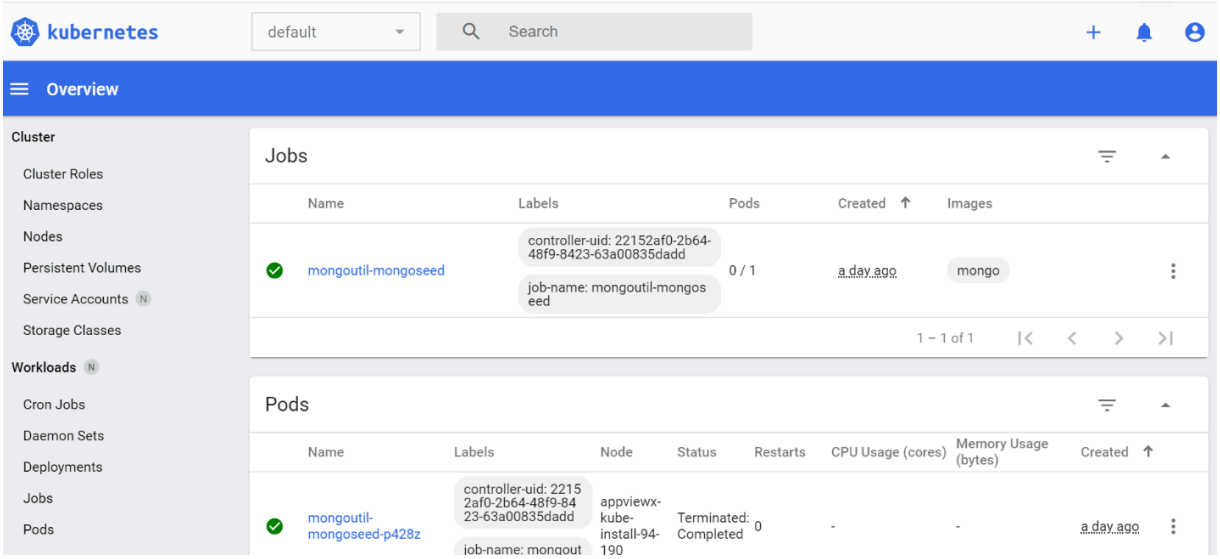


4. On the **Kubernetes Dashboard** login screen, enter the token value
5. Click **Sign In**.

> **Note:** You can find the token value from the .appviewx_configuration file.

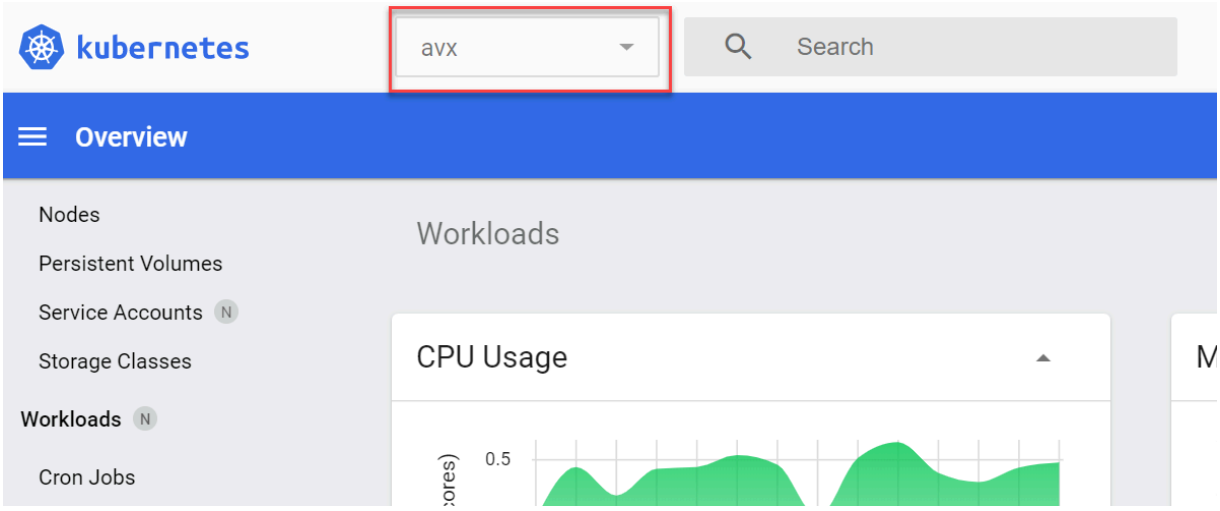


After you log in, you can access the Kubernetes management console and manage AppViewX components.



Viewing the POD Status

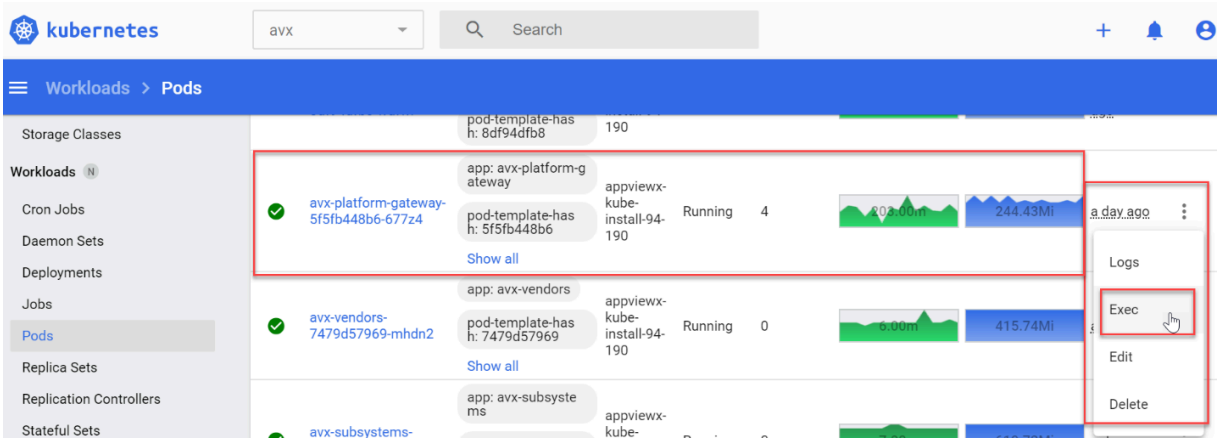
1. Open the Kubernetes management console.
2. Select a namespace from the top list.
3. Select **Pods** on the left menu.



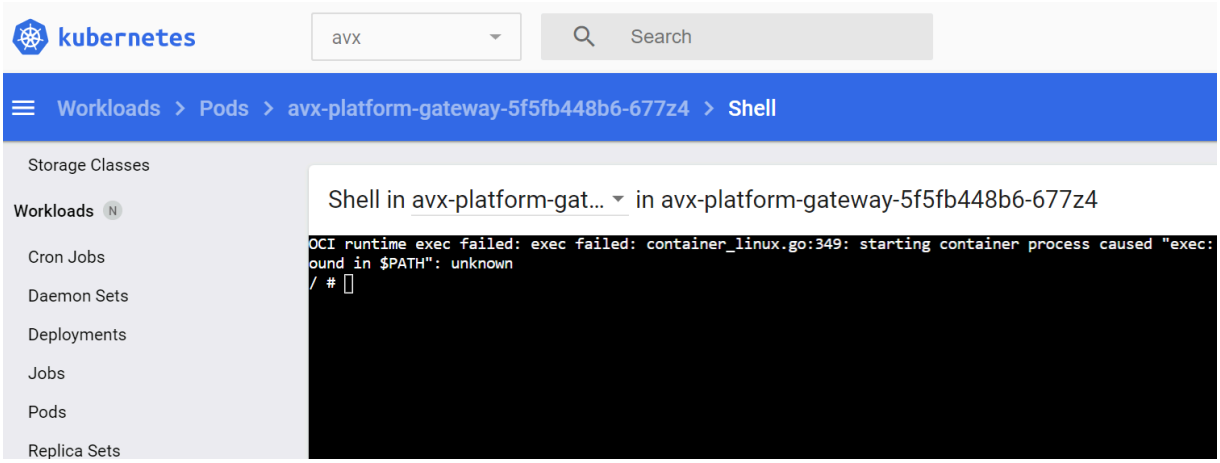
Accessing the POD Console

1. Open the Kubernetes management console.
2. Select the required namespace.
3. Under **Workloads**, click **Pods**.

   The **Pods** page is displayed.
4. Click on the three dots next to the pod and select **Exec**.

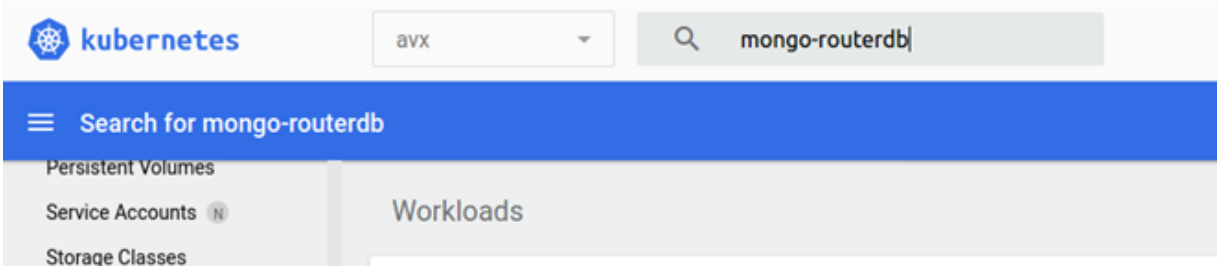


The Pod command line shell is displayed.

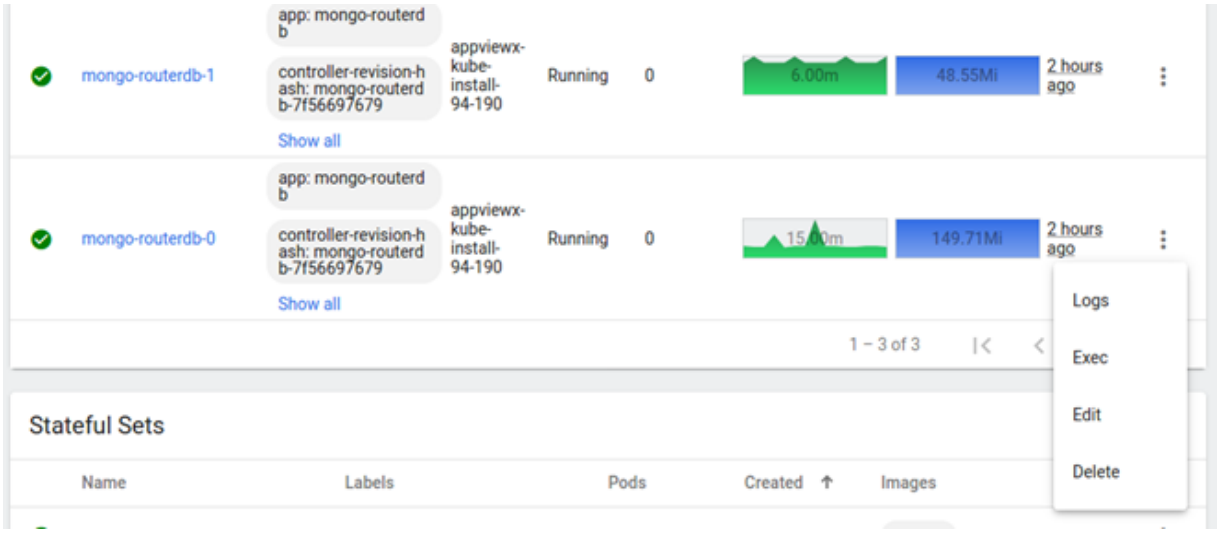


Accessing the Database Command Line

1. Open the Kubernetes management console.
2. Select **avx** in the namespace.
3. Search for **mongo-routerdb**.



4. Click on the three dots next to mongo-routerdb-0 pod and select **Exec**.



5. To launch the mongo db prompt, execute the following command: `<mongo>`
6. Execute the following command:

```
<use admin>
```

7. Execute the following command:

```
db.auth("admin","<mongodbpasssword>")
```

Note: The password can be taken from the value of the `appviewx_mongodb_password` variable from the `<INSTALLATION_PATH>/.appviewx_configuration` file.

Exporting a Database Collection

Collections serve as generic repositories that hold any data in key-value pair format. It acts as interfaces to enter and modify data into the AppViewX Mongo database. Data from collections is consumed as a part of the provisioning request process or by any other scripts that are triggered by AppViewX. The structure of the collections is based on the Mongo database.

To export a mongo database collection:

1. Login to Kubernetes dashboard UI with the token.
2. From the top section, select the **avx** namespace.
3. Click pods and search for **mongo-routerdb-0**.
4. Click on the three dots icon and select **Exec**.
5. To navigate to the logs directory, execute the following command:

   ```
   cd /appviewx/dependencies/logs
   ```

6. Check if export_collection directory is available. Otherwise, to create the directory, execute the following command:`mkdir export_collection`
7. To navigate to the export_collection directory, execute the following command: `cd /appviewx/dependencies/logs/export_collection`
8. To export the database collection, execute the following command:`mongoexport --username admin --password <password> --db=appviewx --collection=<collectionName> --out=<fileName>.json --authenticationDatabase admin` Change the fields highlighted in bold with the desired values according to your setup. This command will

export the collection and the file will be available at the following location: `/appviewx/dependencies/logs/export_collection`

Note: The exported file is also available at the following location on the host where the mongodb pod is running: `INSTALLATION_PATH/appviewx/logs/export_collection`

Offline Patching for CentOS

This section provides the steps, in chronological order, to perform patching in the customer environment offline. The steps are categorized into tasks for the customer and tasks for the AppViewX team.

Steps for Customers

Follow the steps below to patch the customer nodes where internet access is restricted.

1. Run the following command to generate a log file with the list of installed RPMs in customer nodes separately:

   ```
   rpm -qa >> installed_rpms_<node_name>.log
   ```

   This command will create a file *installed_rpms_<node_name>.log* that contains a list of all the installed rpms.

2. SCP the log files of each node to your laptop and upload them as input in the following Jenkins job: http://ci.appviewx.in/job/Offline_patching_of_the_centos_packages/
3. Do not edit the default HOST_IP and HOST_PASSWORD.
4. After successful execution of the Jenkins job, download the rpms_dir.tar.gz and SCP to respective nodes.
5. Unzip rpms_dir.tar.gz and cd to rpms_dir. Run the following command to install the latest RPMs:

   ```
   tar -xvf rpms_dir.tar.gz

   cd rpms_dir

   rpm -Uvh * –-force
   ```

Steps for the AppViewX Team

1. SCP the log files of each node to your laptop and upload it as input in the following Jenkins job: http://%3Cjenkins_server%3E/job/Offline_patching_of_the_centos_packages

Note: Replace jenkins_server with the actual appviewx jenkins server IP or hostname.

2. Do not edit the default HOST_IP and HOST_PASSWORD.
3. After the successful execution of Jenkins job, download the rpms_dir.tar.gz.
4. Install the generated rpms with the lab setup.
5. Perform BVT and Dev Sanity test case execution to ensure the system is healthy.
6. After verification, provide the RPMs to the customer in a tarball rpms_dir.tar.gz.

Chapter 6: External Certificate for Kubernetes

Certificate Specifications

Certificates must be generated individually for each of the common names listed in the table below. All master nodes (IP address and hostname) listed in the table must be added in the SAN of the certificates for a multi-node environment.

Common Name	Type	O (in Subject)	SAN (refer notes below)	Parent CA	Cert and Location
kube-etcd	server		<master_hostnames>, <master_Host_IPs>, <kube_api_address>, localhost, 127.0.0.1,<service_ip>	etcd-ca	etcd/server.crt,etcd/server.key
kube-etcd-peer	server		<master_hostnames>, <master_Host_IPs>, <kube_api_address>, localhost, 127.0.0.1, <service_ip>	etcd-ca	etcd/peer.crt,etcd/peer.key
kube-etcd-healthcheck-client	client			etcd-ca	etcd/healthcheck-client.crt,etcd/healthcheck-client.key
kube-apiserver-etcd-client	client	system:masters		etcd-ca	pki/apiserver-etcd-client.key, pki/apiserver-etcd-client.crt

Common Name	Type	O (in Subject)	SAN (refer notes below)	Parent CA	Cert and Location
kube-apiserver	server		<master_hostnames>, <master_Host_IPs>, <kube_api_address>, localhost, 127.0.0.1,<service_ip>, kubernetes, kubernetes.default, kubernetes.default.svc, kubernetes.default.svc.cluster, kubernetes.default.svc.cluster.local	kubernetes-ca	pki/apiserver.key, pki/apiserver.crt
kube-apiserver-kubelet-client	client	system:masters		kubernetes-ca	pki/apiserver-kubelet-client.key, pki/apiserver-kubelet-client.crt
front-proxy-client	client			kubernetes-front-proxy-ca	client.key,pki/front-proxy-client.crt
kubernetes-admin	client	system:masters		kubernetes-ca	admin.crt, admin.key
system:kube-controller-manager	client			kubernetes-ca	controller-manager.crt, controller-manager.key
system:kube-scheduler	client			kubernetes-ca	scheduler.crt, scheduler.ke
system:node:<hostname>	client	system:nodes		kubernetes-ca	kubelet.crt, kubelet.key

SAN values are as follows:

<master_hostnames> Hostname of the master.

<master_Host_IPs> IPs of the master.

<service_ip> The service IP can be obtained by executing the `kubectl get svc` from any node.

```
[appviewx@pe-iu-node18 scripts]$ kubectl get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   23h
[appviewx@pe-iu-node18 scripts]$
```

<kube_api_address> Load balancer for kube apiserver (if configured).

system:node:<hostname>The *kubelet.crt* should be generated for all the servers (master and worker) separately. For example, if the setup consists of three nodes, the certificates must be generated for all three nodes. The value of the **<hostname>** should be entered from the output of the `hostname` command.

Note:

- Enter only the `hostname` command output in SAN; do not enter the `hostname -f` command output.
- A general rule for the SAN of the certificate is to add the IP address in the *IP Address* field and the hostnames in the *DNS* field.
- An example of a wrong entry is shown below:

Entering All Certificates in the appviewx.conf File

Navigate to the **<appviewx_installed_location>/appviewx_kubernetes/scripts** and open *appviewx.conf* file.

1. To enable the external CA for kubeadm, enter the command `KUBE_EXTERNAL_CERT=TRUE`

```
#Manage kubernetes with the external certficates
#Replace /home/appviewx/external_p12_multinode with the absolute path of the certificate file
#Certificate should be in .p12 format
#Follow the the guide for the certificate speficifications
#Execute ./appviewx.sh --password-encrypt command to encrypt the CERT_PASSWORD password

KUBE_EXTERNAL_CERT=TRUE
```

2. Enter the encrypted certificate password in the `CERT_PASSWORD` key. To encrypt the password,

a. navigate to **<appviewx_installer_location>/appviewx_kubernetes/scripts**

b. execute the command `./appviewx.sh --password-encrypt`

```
#Manage kubernetes with the external certficates
#Replace /home/appviewx/external_p12_multinode with the absolute path of the certificate file
#Certificate should be in .p12 format
#Follow the the guide for the certificate speficifications
#Execute ./appviewx.sh --password-encrypt command to encrypt the CERT_PASSWORD password

KUBE_EXTERNAL_CERT=TRUE
CERT_PASSWORD=vault:v1:XJufPkIER2fdnAYt6bZEEZHhq66r7VPGhNw7AhZ/UQPGaBNNs5WJNg==
```

3. Enter the absolute path of the certificate which is generated for the common name **kube-etcd** in `KUBE_ETCD_PATH`
4. Enter the absolute path of the certificate which is generated for the common name **kube-etcd-peer** in `KUBE_ETCD_PEER_PATH`
5. Enter the absolute path of the certificate which is generated for the common name **kube-etcd-healthcheck-client** in `KUBE_ETCD_HEALTHCHECK_CLIENT_PATH`
6. Enter the absolute path of the certificate which is generated for the common name **kube-apiserver-etcd-client** in `KUBE_APISERVER_ETCD_CLIENT_PATH`
7. Enter the absolute path of the certificate which is generated for the common name **kube-apiserver** in `KUBE_APISERVER_PATH`
8. Enter the absolute path of the certificate which is generated for the common name **kube-apiserver-kubelet-client** in `KUBE_APISERVER_KUBELET_CLIENT_PATH`
9. Enter the absolute path of the certificate which is generated for the common name **front-proxy-client** in `FRONT_PROXY_CLIENT_PATH`
10. Enter the absolute path of the certificate which is generated for the common name **kubernetes-admin** in `KUBERNETES_ADMIN_PATH`
11. Enter the absolute path of the certificate which is generated for the common name **system:kube-controller-manager** in `KUBE_CONTROLLER_MANAGER_PATH`
12. Enter the absolute path of the certificate which is generated for the common name **system:kube-scheduler** in `KUBE_SCHEDULER_PATH`

```
KUBE_ETCD_PATH=/home/appviewx/external_p12_multinode/kube-etcd_17_BA_FA_51_75_3A_CE_0D_E5_86_9B_20_A5_5A_4D_14_00_35_89_DD.p12
KUBE_ETCD_PEER_PATH=/home/appviewx/external_p12_multinode/kube-etcd-peer_51_A3_CE_5F_51_35_9A_72_3C_15_1B_54_BE_83_5C_25_ED_94_CB_C4.p12
KUBE_ETCD_HEALTHCHECK_CLIENT_PATH=/home/appviewx/external_p12_multinode/kube-etcd-healthcheck-client_31_54_F6_E1_3E_68_AB_B1_65_EC_02_99_E2_FB_A9_A7_5D_0C_D5_D3.p12
KUBE_APISERVER_ETCD_CLIENT_PATH=/home/appviewx/external_p12_multinode/kube-apiserver-etcd-client_27_FC_1E_94_84_0A_A8_90_D8_5D_99_5F_98_BB_B9_10_BF_E8_B5_4A.p12
```

```
KUBE_APISERVER_PATH=/home/appviewx/external_p12_multinode/kube-apiserver_19_33_6A_BE_B7_5E_F0_90_E6_2A_A8_F8_5D_C3_A0_2C_2A_78_BD_D1.p12
KUBE_APISERVER_KUBELET_CLIENT_PATH=/home/appviewx/external_p12_multinode/kube-apiserver-kubelet-client_7D_5F_B2_78_2C_51_03_D1_39_17_BF_FD_26_6E_A2_1A_60_93_1C_BF.p12
FRONT_PROXY_CLIENT_PATH=/home/appviewx/external_p12_multinode/front-proxy-client_61_97_2B_D9_E8_13_2B_24_3F_7E_85_B3_1A_F9_3A_AF_10_4C_5F_45.p12
KUBERNETES_ADMIN_PATH=/home/appviewx/external_p12_multinode/kubernetes-admin_2D_A0_1B_5E_A0_CF_27_2E_6B_9C_34_02_D9_E0_CA_60_95_BD_92_E0.p12
KUBE_CONTROLLER_MANAGER_PATH=/home/appviewx/external_p12_multinode/system_kube-controller-manager_31_32_15_2E_5F_4A_9C_B9_0E_2A_11_9B_CE_15_AA_59_5D_B7_FC_D1.p12
KUBE_SCHEDULER_PATH=/home/appviewx/external_p12_multinode/system_kube-scheduler_6A_FF_10_E1_F1_C9_9F_3C_0F_9D_82_88_18_38_EB_01_FB_3D_02_70.p12
```

13. Enter the **Kubelet certificates** in a colon ':' separated format, such as *<hostname>:<kubelet_certificate.p12>*. There should not be any spaces and also no colon (:) in the certificate file name.

Note:

a. If the kubelet certificate is generated for the host **pe-iu-node20.lab.appviewx.net**, the entry should be in the format **KUBELET_CERT_PATH=<hostname>:<absolute certificate file path>**. The entry for the host would be KUBELET_CERT_PATH=pe-iu-node20.lab.appviewx.net:/home/appviewx/external_p12_multinode/system_node_pe-iu-node20.lab.appviewx.net.p12

b. Enter all certificates that match the hosts in a comma-separated format, as given in the example below:

```
KUBELET_CERT_PATH=pe-iu-node20.lab.appviewx.net:/home/appviewx/external_p12_multinode/system_node_pe-iu-node20.lab.appviewx.net.p12,pe-iu-node16.lab.appviewx.net:/home/appviewx/external_p12_multinode/system_node_pe-iu-node16.lab.appviewx.net.p12,pe-iu-node17.lab.appviewx.net:/home/appviewx/external_p12_multinode/system_node_pe-iu-node17.lab.appviewx.net.p12,pe-iu-node18.lab.appviewx.net:/home/appviewx/external_p12_multinode/system_node_pe-iu-node18.lab.appviewx.net.p12,pe-iu-node19.lab.appviewx.net:/home/appviewx/external_p12_multinode/system_node_pe-iu-node19.lab.appviewx.net.p12,pe-iu-node20.lab.appviewx.net:/home/appviewx/external_p12_multinode/system_node_pe-iu-node20.lab.appviewx.net.p12,pe-iu-node21.lab.appviewx.net:/home/appviewx/external_p12_multinode/system_node_pe-iu-node21.lab.appviewx.net.p12
```

Warning: Entering wrong certificates in the paths mentioned above will compromise the functioning of the application.

14. After adding all the certificate entries in the *appviewx.conf*

a. Navigate to the **<appviewx_installer_location>/appviewx_kubernetes/scripts**

b. Execute the command `./appviewx.sh --enable-kube-external-ca`

```
[appviewx@pe-iu-node18 scripts]$ ./appviewx.sh --enable-kube-external-ca
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
MAC verified OK
```

15. The command prompt for the passwords of all the nodes once the validations are completed. Enter the passwords, and hit the keyboard **Enter** key to proceed further.

```
node/pe-iu-node20.lab.appviewx.net drained
NAME                            STATUS                     ROLES                  AGE   VERSION
pe-iu-node16.lab.appviewx.net   Ready,SchedulingDisabled   <none>                 25h   v1.20.7
pe-iu-node17.lab.appviewx.net   Ready,SchedulingDisabled   <none>                 25h   v1.20.7
pe-iu-node18.lab.appviewx.net   Ready,SchedulingDisabled   <none>                 25h   v1.20.7
pe-iu-node19.lab.appviewx.net   Ready,SchedulingDisabled   control-plane,master   25h   v1.20.7
pe-iu-node20.lab.appviewx.net   Ready,SchedulingDisabled   control-plane,master   25h   v1.20.7
pe-iu-node21.lab.appviewx.net   Ready,SchedulingDisabled   control-plane,master   25h   v1.20.7
/home/appviewx/FP6/appviewx_kubernetes/scripts/script_util
Please enter appviewx password of master:pe-iu-node20.lab.appviewx.net :
Please enter appviewx password of master:pe-iu-node21.lab.appviewx.net :
Please enter appviewx password of master:pe-iu-node19.lab.appviewx.net :
Please enter appviewx password of absecon:pe-iu-node16.lab.appviewx.net :
Please enter appviewx password of antartica:pe-iu-node17.lab.appviewx.net :
Please enter appviewx password of antartica:pe-iu-node18.lab.appviewx.net :
null_resource.ssh_connectivity[5]: Creating...
null_resource.ssh_connectivity[3]: Creating...
null_resource.ssh_connectivity[2]: Creating...
null_resource.ssh_connectivity[4]: Creating...
null_resource.ssh_connectivity[1]: Creating...
null_resource.ssh_connectivity[0]: Creating...
null_resource.ssh_connectivity[3]: Provisioning with 'remote-exec'...
null_resource.ssh_connectivity[2]: Provisioning with 'remote-exec'...
```

The following message is displayed on the successful completion of the execution:

```
Starting all the components..
node/pe-iu-node20.lab.appviewx.net uncordoned
node/pe-iu-node19.lab.appviewx.net uncordoned
node/pe-iu-node21.lab.appviewx.net uncordoned
node/pe-iu-node16.lab.appviewx.net uncordoned
node/pe-iu-node17.lab.appviewx.net uncordoned
node/pe-iu-node18.lab.appviewx.net uncordoned
NAME                            STATUS   ROLES                  AGE   VERSION
pe-iu-node16.lab.appviewx.net   Ready    <none>                 25h   v1.20.7
pe-iu-node17.lab.appviewx.net   Ready    <none>                 25h   v1.20.7
pe-iu-node18.lab.appviewx.net   Ready    <none>                 25h   v1.20.7
pe-iu-node19.lab.appviewx.net   Ready    control-plane,master   25h   v1.20.7
pe-iu-node20.lab.appviewx.net   Ready    control-plane,master   25h   v1.20.7
pe-iu-node21.lab.appviewx.net   Ready    control-plane,master   25h   v1.20.7
/home/appviewx/FP6/appviewx_kubernetes/scripts/script_util

Components will take few mins to start..Please wait..
Old certificates and conf file backups can be found under /etc/kubernetes/external_ca_bkp_05-03-2022_22_51_42 and /var/lib/kubelet/pki_05-03-2022_22_51_42
Logs can be found under /home/appviewx/FP6/appviewx_kubernetes/scripts/script_util/../../logs/kubeadm-external-ca_05-03-2022_22_51_42.log
[appviewx@pe-iu-node18 scripts]$
[appviewx@pe-iu-node18 scripts]$
```

Rollback Steps For Failure in Certificate Updates

This section describes the commands that can be executed to restore the certificates and config files to their previous state, in the event of a certificate update failure.

```
Error: error executing "/tmp/terraform_1942716618.sh": Process exited with status 1



Error: error executing "/tmp/terraform_1680268861.sh": Process exited with status 1


Certificate update failed!
Rolling back to previous state
Stopping all the components..
node/pe-iu-node16.lab.appviewx.net already cordoned
node/pe-iu-node16.lab.appviewx.net drained
node/pe-iu-node17.lab.appviewx.net already cordoned
node/pe-iu-node17.lab.appviewx.net drained
node/pe-iu-node18.lab.appviewx.net already cordoned
node/pe-iu-node18.lab.appviewx.net drained
node/pe-iu-node19.lab.appviewx.net already cordoned
node/pe-iu-node19.lab.appviewx.net drained
node/pe-iu-node21.lab.appviewx.net already cordoned
node/pe-iu-node21.lab.appviewx.net drained
```

Note: The pods can either be in the **Init:CrashLoopBackOff** state or the **Pending** state.

1. **Init:CrashLoopBackOff :** If the pod is in this state, delete the pods by executing the command `Kubectl delete pod <podname> -n <namespace> –force`
2. **Pending:** If the pod is in this state, execute the commands in the order mentioned below:
   a. `kubectl scale –replicas=0 deploy/<component name> -n <namespace>`
   b. `kubectl get pods --all-namespaces | awk '{if ($4=="Terminating") print "kubectl delete pod " $2 " -n " $1 " --force --grace-period=0 ";}' | sh > /dev/null 2>&1`
   c. `kubectl scale –replicas=3 deploy/<component name> -n <namespace>`

   Replicas can be changed based on the initial setup.

Chapter 7: Uninstalling AppViewX

Troubleshooting Uninstall Issues

Note: For troubleshooting issues, please refer to the Troubleshooting section.

Uninstalling AppViewX

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

To uninstall an application package safely:

1. Open the terminal window.
2. To navigate to the appviewx_kubernetes directory, execute the following command:
   cd /home/appviewx/appviewx_kuberbetes/scripts/uninstall
3. To start the uninstallation process, execute the following command:/uninstall.sh
4. Enter the node's credentials when prompted.

```
[appviewx@pesrv03-regression02-98-13 uninstall]$ cd
[appviewx@pesrv03-regression02-98-13 ~]$ cd /home/appviewx/                    /scripts/uninstall/
[appviewx@pesrv03-regression02-98-13 uninstall]$ ./uninstall.sh
Please enter appviewx password of master:pesrv03-regression02-98-13 :
Please enter appviewx password of dc1:pesrv03-regression03-98-14 :
Please enter appviewx password of dc2:pesrv03-regression04-98-15 :
```

5. Reboot all the nodes after completion of the AppViewX uninstallation.

• Troubleshooting Uninstall Issues

• Uninstalling AppViewX

Chapter 8: Troubleshooting

AppViewX Installation Failed

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

Frequently Faced Errors

- Pre requisites not met- port not opened, insufficient disk/cpu, time not in sync, packages not found, hostname incorrect in configuration etc. please check for all the above items.
- Error while installing the docker

  In some custom OS which the customer brings in, the linux packages that we bundle along with the installer might not be compatible with the OS. In such cases, we might need to install the relevant package to proceed further. The same can be seen from the log messages stating error while installing a package.
- Error while installing the docker

  In some cases, we have seen there are intermittent errors from the OS while installing the docker. When you face an error in this stage, please try doing an uninstall of the application and reboot all the nodes and proceed with the installation.
- Error while initializing the kube master/worker

  In certain cases, when uninstallation does not clean up the data properly, we can see errors while initializing kube master and worker. Please perform an uninstall, reboot all the nodes and go ahead with the install once when we face this error. Also there are cases where the installation fails because of the port connectivity issues. Please check if 6443, 10250, 2379 and 2380 ports are opened properly if a failure occurs in this stage.

- Error while initializing the mongodb chart

  This specific error occurs after a timeout of 5 minutes to initialize the mongodb charts. When we face this error, it happens because the pods are not able to communicate between themselves. Use the following commands to verify that:

Kubectl describe statefulset -n avx mongo-shareddb - If we face any connectivity errors, this will give the specific error stating connection timed out.

- Node is enabled with IPv6 but the application is not.

  Check the output of ifconfig | grep -i inet6 when this shows an IPv6 address, we need to enable ipv6 in the appviewx.conf file, else the communication does not happen properly.

- IP in IP tunnelling is not enabled

  When the IP in IP traffic is not enabled, (IPv4 protocol is not allowed) we will be facing the same issue. The prerequisite check script does not capture this. Therefore, we must have this confirmed.

- Error while installing the AppViewX plugins

  When there is an error while installing AppViewX plugins, most probably there is an error in the configuration file. Please double check the configuration file and proceed with the execution of plugins_install.sh to install the plugins alone.