CN-Series Deployment Guide

PAN-OS 10.0

paloaltonetworks.com/documentation

techDOCS

Contact Information

Corporate Headquarters:
Palo Alto Networks
3000 Tannery Way
Santa Clara, CA 95054
www.paloaltonetworks.com/company/contact-support

About the Documentation

- For the most recent version of this guide or for access to related documentation, visit the Technical Documentation portal www.paloaltonetworks.com/documentation.
- To search for a specific topic, go to our search page www.paloaltonetworks.com/documentation/document-search.html.
- Have feedback or questions for us? Leave a comment on any page in the portal, or write to us at documentation@paloaltonetworks.com.

Copyright

Last Revised

October 16, 2020

Table of Contents

CN-Series Firewall for Kubernetes

The Palo Alto Networks Container Native Firewalls (CN-Series) are natively integrated into Kubernetes (k8s) to provide complete L7 visibility, application level segmentation, DNS Security, and protection from advanced threats for traffic going across trusted zones in public cloud or data center environments. It enables you to isolate and protect workloads, application stacks, and services, even as individual containers scale up, down, or across hosts and consistently apply security policies that are based on Kubernetes labels.

App deployment in a Kubernetes environment is dynamic and the following teams are often involved in the container lifecycle:

- **Platform (PAAS) Admin**—Manages the Kubernetes clusters and other infrastructure components in public cloud and data centers.
- **App Teams**—Deploy their individual containerized and other applications in Kubernetes namespaces/projects provided by PAAS admin.
- **Security Admin**—Provisions security for the entire deployment including Kubernetes clusters and individual containerized applications.

In this dynamic scenario and interplay with multiple teams, security management and monitoring pose a challenge. The CN-Series firewall enables your security administrator to provision security for the containerized applications across a wide range of environments including Cloud Provider Managed k8s such as GKE, EKS, AKS, and Customer Managed k8s such as Openshift, and Native k8s on the public cloud or on premises data centers. The CN-Series firewall uses Kubernetes constructs and metadata driven policy so that the teams can automate the deployment and efficiently enforce security policy to consistently protect from known and unknown threats.

Start here:

- CN-Series Core Building Blocks
- CN-Series Deployment—Supported Environments
- Components Required to Secure Kubernetes Clusters with CN-Series Firewall
- IP-Address-to-Tag Mapping of Kubernetes Attributes
- Editable Parameters in CN-Series Deployment YAML Files

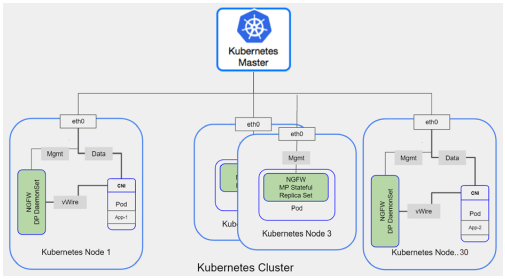
CN-Series Core Building Blocks

The CN-Series firewall is the containerized next-generation firewall that provides visibility and security for your containerized application workloads on Kubernetes clusters. The CN-Series firewall uses native Kubernetes (K8s) constructs and Palo Alto Networks components to make this possible.

The core building blocks to Deploy the CN-Series Firewalls are:

- **Distributed PAN-OS architecture with CN-MGMT and CN-NGFW pods**—The management plane (CN-MGMT) and data plane (CN-NGFW) of the containerized firewall are separate to enable better runtime protection for applications and to support a smaller footprint. The CN-MGMT and CN-NGFW are deployed using container images and YAML manifest files with ConfigMap objects. This architecture enables you to place the CN-NGFW DaemonSet pod on each node that you want to protect workloads in a cluster, and a pair of CN-MGMT pods can connect to and manage up to 30 CN-NGFW pods within a cluster. For more information on limits, see CN-Series Supported Scale Factors.
  - CN-MGMT runs as a StatefulSet to ensure that it has persistent volume and is exposed as a K8s service that can be discovered using DNS in the Kubernetes environment. The CN-MGMT provides fault tolerance and a single CN-MGMT pod can manage the existing CN-NGFW pods in the event of a restart or a failure of a CN-MGMT pod.
  - CN-NGFW runs as a DaemonSet. Each instance of the CN-NGFW pod can secure 30 application pods deployed within the cluster.

    If for example, you have a cluster with 120 nodes, you need 4 CN-MGMT pods (in pairs because it is a StatefulSet) and each service/pair can secure up to 30 nodes that have CN-NGFW pods.



- **PAN-CNI plugin for network insertion**—The PAN-CNI plugin is responsible for the allocation of network interfaces on every pod, which enables network connectivity to the CN-NGFW pod. The YAML files that enable you to deploy the CN-Series include the PAN-CNI DaemonSet, which insert the PAN-CNI plugin into the CNI plugin chain on each node within the cluster. The plugin reads the annotation on each application pod as it comes up to determine whether to enable security and redirect traffic to the CN-NGFW pod for inspection as it ingresses and egresses the pod.

  The pan-cni secures traffic on the default "eth0" interface of the pod only and does not support multi-homed pods.
- **Panorama for centralized management**—Panorama functions as the hub for managing the configuration and licensing of the containerized firewalls. It also hosts the Kubernetes plugin, which enables monitoring of the Kubernetes clusters, and centralized Security policy management. You can use a physical or virtual Panorama appliance, and deploy it on-premises or in a public cloud environment. Panorama must have network connectivity to the firewall management plane pods (CN-MGMT) to ensure that it can license the (CN-NGFW) firewalls and push configuration and policies using Panorama templates and device groups. Palo Alto Networks recommends deploying Panorama in an HA configuration.

You need standard Kubernetes tools such as kubectl or Helm to deploy and manage your Kubernetes clusters, apps, and firewall services. Panorama is not designed to be an orchestrator for Kubernetes cluster deployment and management. Templates for cluster management are provided by Managed Kubernetes providers. You can also use the community-supported templates for deploying CN-Series with Helm and Terraform.

- **Kubernetes Plugin on Panorama**—The Kubernetes plugin manages the licenses for the CN-Series firewall. Licensing is based on the number of nodes within a cluster. Each CN-NGFW pod uses a license token, and the tokens are managed locally on Panorama after you activate the auth code and retrieve the specified number of tokens from the Palo Alto Networks license server. As each CN-NGFW comes up on the Kubernetes nodes, Panorama distributes the license tokens locally.

  The Kubernetes plugin on Panorama also enables you to monitor your clusters and leverage Kubernetes labels that you use to organize Kubernetes objects such as pods, services, deployments and the associated identifying attributes, so that you can create context-aware Security policy rules. The Kubernetes plugin communicates with the API server and retrieves metadata to enable visibility into the applications running within the cluster. The Kubernetes plugin collects namespaces, services, and labels from your Kubernetes clusters to create tags for the IP-address-to-tag mapping for the associated objects within the cluster which can then be used in Security policies. For details see IP-Address-to-Tag Mapping of Kubernetes Attributes.

  It also collects information on the ports specified in your application YAML and creates Service Objects.

  While these tags and service objects are automatically shared with the CN-NGFW pods in each cluster, you can also enable sharing of the tags and service objects with hardware-based or VM-Series firewalls. The tags become available as match criteria in Dynamic Address Groups, which you can then use to secure traffic between pods or namespaces, to an internet-exposed service, or outbound connections.

  Palo Alto Networks recommends deploying Panorama in an HA configuration so that the Panorma peer continues to receive IP address updates in the event of a failure. If you deploy a single instance of Panorama, in the event of a failure the traffic from any existing applications pods are not impacted, and the current policies are enforced on the CN-NGFW pods. When a new pod comes up, all the rules with the source "ANY" will match to this new pod, and traffic from this new pod will be allowed or blocked depending on your policy rules. For example, if there is an Anti-Spyware policy rule to block outbound access from *any* source to the outside world, then this rule will apply to the new pod, and the profile can secure traffic. If there is a default *Deny* rule, then traffic from this new pod will be denied.

  You can use the Kubernetes plugin to distribute IP-address-to-tag mapping for pods, nodes, namespaces, and services deployed within the Kubernetes cluster to physical or VM-Series firewalls, even if you have not deployed CN-Series firewall in that cluster.

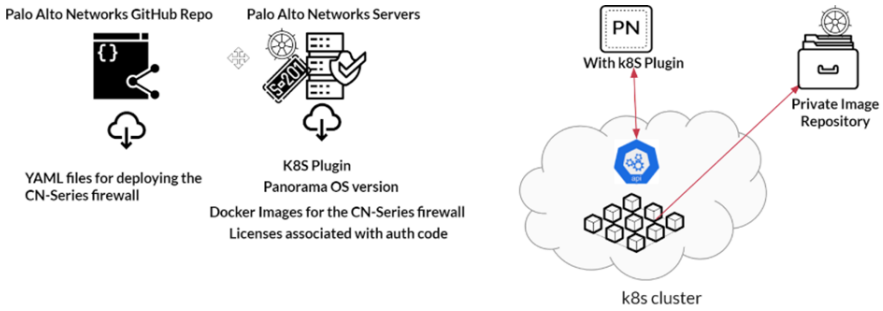
CN-Series Deployment—Supported Environments

You can deploy the CN-Series firewall in the following environments:

Product	Version
Container runtime	Docker, CRI-O
Kubernetes version	1.14, 1.15, 1.16, 1.17, 1.18
Cloud provider managed Kubernetes	• AWS EKS (1.14, 1.15, 1.16, 1.17) • Azure AKS (1.14, 1.15, 1.16, 1.17, 1.18) • GCP GKE (1.14, 1.15, 1.16, 1.17)
Customer managed Kubernetes	On the public cloud or on-premise data center. Make sure that the Kubernetes version, CNI Types, and Host VM OS versions are as listed in this table.
Kubernetes Host VM OS	• Ubuntu 16.04 • Ubuntu 18.04 • RHEL/Centos 7.3 and later • CoreOS 21XX, 22XX • Container-Optimized OS The Linux Kernel Netfilter: Iptables
CNI Plugins	CNI Spec 0.3 and later: • AWS-VPC • Azure • Calico • Flannel • Weave For Openshift • OpenshiftSDN • Multus (supported on PAN-OS 10.0.1 and later) • Bridge (supported on PAN-OS 10.0.1 and later) • SR-IOV (supported on PAN-OS 10.0.2 and later) • Macvlan (supported on PAN-OS 10.0.2 and later)
OpenShift	Version 4.2, 4.4, 4.5

Also review the CN-Series Prerequisites, before you Deploy the CN-Series Firewalls.

Components Required to Secure Kubernetes Clusters with CN-Series Firewall



The following is a list of what you need to Deploy the CN-Series Firewallsand secure the applications deployed within Kubernetes clusters.

- **Panorama**—A hardware-based or virtual appliance that can connect to the Kubernetes clusters where the applications and CN-Series firewalls are deployed. Panorama is required for license management and configuration management of the CN-Series firewalls. For more information, see CN-Series Core Building Blocks.
- **Kubernetes Plugin on Panorama**—Because of the rate of change with containerized applications, this plugin is required for visibility into container activity within a cluster and for managing the license token allocation for the firewall deployed on each node within a cluster.

  The Kubernetes plugin connects to Kubernetes clusters using service account credentials. From there it retrieves resource attributes and labels and creates tags and service objects. The tags can be used to create Dynamic address groups and reference them in Security policy for IP traffic enforcement. You can also use the service objects in Security policy to allow or deny traffic based on ports as well as IP addresses. The tags and service objects give you visibility and granular control for traffic enforcement within your Kubernetes cluster.
- **Docker Images**—To support the distributed architecture, the CN-Series firewall has four docker images that are available on the Palo Alto Networks Customer Support Portal (CSP). These images are published as three compressed tar archives (tar.gz format), and you must get these images unzip and do a Docker push to your image registry.

  Note: Make sure that the images and YAML files versions are compatible.The compressed files are:

  - **PanOS_cn-10.0.0-<bn>.tgz**—This archive includes the firewall management plane (CN-MGMT) and firewall dataplane (CN-NGFW) images.

    The unzipped image names are, for example: `panos_cn_ngfw:10.0.0-b7` and `panos_cn_mgmt:10.0.0-b7`

    - **Pan_cn_mgmt_init-1.0.0-<xn>.tgz**—This archive includes the init container (CN-INIT) that contains the utilities required to deploy the management plane on the firewall. The init container enables secure IPSec communication between the CN-MGMT and CN-NFGW Pods. The unzipped image name is for example: `pan_cn_mgmt_init:1.0.0-b1-c1`.
    - **Pan_cni-1.0.0-<xn>.tgz**—This archive includes the CNI plugin that enables connectivity between the CN-MGMT and CN-NFGW and reconfigures the network interfaces on the application pods to redirect traffic to the CN-NGFW pod on each node. The unzipped image name is for example: `pan_cni:1.0.0-b1-c3`.
- **YAML Files**—The YAML files that include the required fields and object specifications for deploying the resources in your Kubernetes clusters, and are published on GitHub.

All the YAML files you need, for a supported environment such as native Kubernetes or GKE, are combined and zipped in one folder for your convenience.

- CN-MGMT has three YAML files—`pan-cn-mgmt.yaml`, `pan-cn-mgmt-configmap.yaml`, and `pan-cn-mgmt-secret.yaml`.
- CN-NGFW has two YAML files—`pan-cn-ngfw.yaml`, and `pan-cn-ngfw-configmap.yaml`.
- CNI plugin has two YAML files—`pan-cni.yaml`, and `pan-cni-configmap.yaml`.

  - *There is also a `pan-cni-serviceaccount.yaml` that is referenced in the service account creation section below.*
  - *For OpenShift deployments there is an additional `pan-cni-net-attach-def.yaml`.*
- **Service Account Creation**—Three YAML files, `pan-mgmt-serviceaccount.yaml`, `pan-cni-serviceaccount.yaml`, and `plugin-serviceaccount.yaml`.

  `pan-mgmt-serviceaccount.yaml` and `pan-cni-serviceaccount.yaml` are for the CN-MGMT and CN-NGFW pods to authenticate to the cluster.

  The `plugin-serviceaccount.yaml` is for the Kubernetes plugin on Panorama to authenticate to the cluster.
- **Persistent volume YAML for Native Kubernetes deployments**—`pan-cn-pv-manual.yaml` and `pan-cn-pv-local.yaml`.

  The `pan-cn-pv-manual.yaml` is only provided for PoC with single node clusters. Palo Alto Networks strongly recommends the use of dynamically provisioned persistent volumes for storing the configuration and logs for the CN-MGMT pods that are referenced in the `pan-cn-mgmt.yaml`. Make sure to set up a persistent volume within the cluster for both the CN-MGMT pods.

- **License auth code**—The auth code enables you to license each instance of the CN-NGFW pod deployed on each node within a cluster.

  The license auth code is tied to the CN-Series license bundle you purchased. You must provide the license bundle information in the `pan-cn-mgmt-configmap.yaml` and the corresponding auth code on the Panorama (Kubernetes plugin) in order to ensure that the CN-NGFW pods can register with the CN-MGMT pods, which communicates with Panorama.

  If the license bundle and auth code do not match, licensing will fail and you will need to redeploy the CN-MGMT pods with the correct license bundle and auth code combination. For available licenses, see Register the CN-Series Firewall Auth Code.

  Note the following details when your license information is updated on a renewal, mid-term upgrade, or expiration:

  - *Renewals- When you renew your existing bundle licenses, if you upgrade or downgrade the license bundles, you must delete the existing deployment, and redeploy the firewalls.*
  - *Mid-term upgrade- On a mid-term upgrade you can increase the quantity of tokens associated with your auth code or upgrade the subscription bundle (for example Basic to Bundle 1 or Bundle 1 to Bundle 2). When you upgrade the subscription bundle, you must delete the existing deployment, and redeploy the firewalls for using the additional subscriptions that you have purchased.*
  - *License Expiration-If your license expires and you do not renew it within the 30-day grace period, the licenses are deactivated. The CN-NGFW pods use the failover mode —failopen or failclosed—as defined in your PAN-CN-MGMT-CONFIG yaml. You must renew the license, register it on the CSP, and redeploy the firewalls.*

IP-Address-to-Tag Mapping of Kubernetes Attributes

The Kubernetes plugin on Panorama creates tags for predefined tags in your Kubernetes clusters, user defined labels for Pods and Services, and service objects.

The plugin creates tags for the following Kubernetes objects:

- Pod Classes: ReplicaSets, DaemonSets, StatefulSets
- Service Types: ClusterIP, NodePort, LoadBalancer
- Service Objects: port, targetPort, and nodePort

By default, the Kubernetes plugin on Panorama retrieves the following predefined tags from every Kubernetes cluster that you are monitoring on Panorama, and creates tags in the format listed below. You can then use these tags as match criteria in Dynamic Address Groups and enforce Security policy for the underlying IP addresses associated with each tag.

You can use the Kubernetes plugin to distribute IP-address-to-tag mapping for pods, nodes, namespaces, and services deployed within the Kubernetes cluster to physical or VM-Series firewalls, even if you have not deployed CN-Series firewall in that cluster.

Predefined Tags	Tag Format on Panorama	IP Address Collected
DaemonSet	k8s.cl_<cluster-name>.ns_<namespace>.ds_<pod-name>	Pod IP addresses
ReplicaSet	k8s.cl_<cluster-name>.ns_<namespace>.rs_<pod-name>	Pod IP addresses
StatefulSet	k8s.cl_<cluster-name>.ns_<namespace>.ss_<pod-name>	Pod IP addresses
Service	k8s.cl_<cluster-name>.ns_<namespace>.svc_<svc-name>	Cluster IP addresses Pod IP addresses
External Service	k8s.cl_<cluster-name>.ns_<namespace>.exsvc_<svc-name>	External Service IP addresses LoadBalancer IP addresses
Nodes	k8s.cl_<cluster-name>.nodes	Private IP addresses of all nodes
External Nodes	k8s.cl_<cluster-name>.ex_nodes	Public IP addresses of all nodes
Namespace	k8s.cl_<cluster-name>.ns_<namespace>	All Cluster IP addresses in the namespace All Pod IP addresses in the namespace

If you use labels to organize the Pods and services within the Kuberbetes cluster, the Kubernetes plugin on Panorama can query these labels and create tags for you. The following user-defined labels are supported:

User-Defined Tags	Tag Format on Panorama	IP Address Collected
Label	k8s.cl_<cluster-name>.ns_<namespace>.<label-key>.<label-value>	All Cluster IP addresses in that namespace that match the specified label. All Pod IP addresses in that namespace that match the specified label.
Label Selector	k8s.cl_<cluster-name>.<selector-name>	All Cluster IP addresses that match the specified selector. All Pod IP addresses that match the specified selector.

The label selector matches for the specified label against Pods and Services within the Kubernetes cluster and maps the IP addresses associated with the label to a single tag. The Kubernetes plugin supports set-based and equality-based selectors for label key and label value.

The following equality-based selectors are supported:

- `key = value; key ==`
- `value; key != value, for example, app = redis`

You can also specify multiple selectors in an expression as a comma separated list. For example:

```
app == web, tier != backend
```

The following set-based selectors are supported:

- key in (value1, value2)
- key notin (value1, value2), for example, tier notin (frontend, backend)
- key
- !key

For the monitored Service Objects, the plugin generates ports for the port, targetPort, and nodePort service objects using the following naming scheme:

```
<namespace>-<svc_name>-<type>-<port_value>-<hash>
```

The hash ensures that even if you have overlapping namespaces and service names across k8s clusters, the service objects are unique.

Editable Parameters in CN-Series Deployment YAML Files

The YAML files include several editable parameters, the following tables list the ones you must modify to Deploy the CN-Series Firewalls successfully.

- PAN-CN-MGMT-CONFIGMAP
- PAN-CN-MGMT-SECRET
- PAN-CN-MGMT
- PAN-CN-NGFW-CONFIGMAP
- PAN-CN-NGFW
- PAN-CNI-CONFIGMAP
- PAN-CNI

PAN-CN-MGMT-CONFIGMAP

PAN-CN-MGMT-CONFIGMAP	
Panorama IP Address `PAN_PANORAMA_IP:`	Include the Panorama IP address to which the CN-MGMT Pod will connect. If you have configured your Panorama management servers in a high availability (HA) configuration, provide the IP address of the primary-active Panorama.
Device Group Name `PAN_DEVICE_GROUP:`	Specify the device group name to which you want to assign the CN-NGFW Pods. From Panorama, you will push identical policies to all CN-NGFW Pods that are managed by a pair of CN-MGMT Pods (or that belong to a PAN-SERVICE-NAME).
Template Stack Name `PAN_TEMPLATE_STACK:`	Allows you to configure the settings that enable firewalls (CN-NGFW Pods) to operate on the network.
Log Collector group name `PAN_PANORAMA_CGNAME:`	Enables log storage for the logs generated on the CN-NGFW firewalls. Without a Collector Group, the firewall logs are not saved.
License bundle `PAN_BUNDLE_TYPE:`	The license bundle you purchased— CN-X-BASIC, CN-X-BND1, or CN-X-BDN2 You must enter the authorization code associated with this license bundle on the Kubernetes plugin on Panorama. The tokens associated with the authorization code are used to license the CN-NGFW Pods that are managed by the CN-MGMT Pods. If there is a mismatch between the license bundle, for example CN-X-BND2, and the auth code you enter on Panorama, licensing will fail and you will need to

PAN-CN-MGMT-CONFIGMAP	
	redeploy the CN-MGMT pods with the correct license bundle/auth code combination.
(Optional) `#CLUSTER_NAME:`	Specify the cluster name. The hostname of the CN-MGMT pod combines the StatefulSet name defined in the `PAN-CN-MGMT.yaml` and this optional CLUSTER_NAME. This hostname enables you to identify pods that are associated with different clusters, if you manage multiple clusters on the same Panorama appliance. As a best practice, use the same name here and on the Kubernetes plugin on Panorama.
(Optional) Panorama HA peer IP address `#PAN_PANORAMA_IP2:`	IP address of the Panorama peer (passive-secondary) that is configured in a high availability setup. Verify that the PAN_PANORAMA_IP is that of the primary-active Panorama.

The following default values are defined in the `pan-cn-mgmt-configmap.yaml` file.

```
metadata:
  name: pan-mgmt-config
  namespace: kube-system
data:
  PAN_SERVICE_NAME: pan-mgmt-svc
  PAN_MGMT_SECRET: pan-mgmt-secret
```

These default values allow you to use these files for a quick proof-of-concept. If you want to modify these e.g., to deploy more than one fault-tolerant pair of PAN-MGMT Pods that manage up to 30 PAN-NGFW Pods, you must modify `pan-mgmt-svc` to use another service name. When you modify these values, you must update the corresponding references in the other YAML files to match the values you define in this file.

PAN-CN-MGMT-SECRET

PAN-CN-MGMT-SECRET	
VM auth key `PAN_PANORAMA_AUTH_KEY:`	Allows Panorama to authenticate the firewalls so that it can add each firewall as a managed device. The VM auth key is required for the lifetime of the deployment. Without a valid key in the connection request, the CN-Series firewall will be unable to register with Panorama. See 6.
Device certificate for the CN-Series `CN-SERIES-AUTO-REGISTRATION-PIN-ID`	The firewall requires the device certificate to get any site license entitlements and securely access the Palo Alto cloud-delivered services. Generate

PAN-CN-MGMT-SECRET	

`CN-SERIES-AUTO-REGISTRATION-PIN-VALUE`	the PIN ID and the PIN value on the Palo Alto Networks CSP, and use the PIN before it expires. For example:
	`CN-SERIES-AUTO-REGISTRATION-PIN-ID:`
	`"01cc5-0431-4d72-bb84-something”`
	`CN-SERIES-AUTO-REGISTRATION-PIN-VALUE:`
	`"12.....13e"`
	The following additional field for CN-SERIES-AUTO-REGISTRATION-API-CSP is commented out and is not required: `"certificate.paloaltonetworks.com"`
	See Generate the Auto-Registration PIN for the CN-Series.

PAN-CN-MGMT

PAN-CN-MGMT	

Image path for the Init container image for the CN-MGMT firewall
`initContainers:`
`  - name: pan-mgmt-init`
`  image: <your-private-registry-image-path>`
	The init container generates certificates which are used for securing communication between instances of CN-MGMT Pods and between CN-MGMT pods and CN-NGFW pods.
	Edit the image path to point to the location to which you have uploaded the docker image for the CN-MGMT container.

Image Path for the CN-MGMT image containers:
`- name: pan-mgmt`
`  image: <your-private-registry-image-path>`
	Edit the image path to point to the location to which you have uploaded the docker image for the CN-MGMT container.

Hostname of the CN-MGMT firewall
`kind: StatefulSet`
`metadata:`
`  name: pan-mgmt-sts`
	The hostname of the CN-MGMT firewall is derived by combining the StatefulSet name and the optional cluster name that you may have defined in the `pan-cn-mgmt-configmap.yaml`.
	The default hostname of the CN-MGMT pods is pan-mgmt-sts-0 and pan-mgmt-sts-1, because the StatefulSet name is pan-mgmt-sts and the cluster name is not defined.

PAN-CN-MGMT	
	If the hostname is more than 30 characters, the name will be truncated at 30 characters.
(Only for an on-premises or self-managed Native Kubernetes deployment) `storageClassName: local`	For self-managed deployment, the default config has “storageClassName: local”. If your cluster has dynamically provisioned Persistent Volumes (PV), you must modify the “storageClassName: local” to match that storageClass or remove these lines if DefaultStorageClass is being used. If your cluster doesn’t have dynamically provisioned PV, cluster admin can create static PVs with provided `pan_cn_pv_local.yaml` which has 2 sets of few PVs, one each for each PAN-CN-MGMT statefulSet pods. You can modify `pan_cn_pv_local.yaml` to match the volumes in your setup and deploy it before deploying the `PAN-CN-MGMT.yaml`.
Jumbo frame mode `PAN_JUMBO_FRAME_ENABLED: "True"`	The attribute that enables jumbo frame mode on the CN-Series firewalls. Use this option when you have a primary CNI that does not use jumbo frames and secondary CNI is enabled for jumbo frames. You must make this change before the CN-MGMT StatefulSet is deployed. Requires PAN-OS 10.0.1

PAN-CN-NGFW-CONFIGMAP

You do not need to modify any PAN-values unless you need to change the following:

- `PAN_SERVICE_NAME: pan-mgmt-svc`

  The service name should match what you defined on the PAN-CN-MGMT-CONFIGMAP.
- `FAILOVER_MODE: failopen`

  You can change this to failclose. It comes into effect only when CN-NGFW fails to get a license.
  - In fail-open mode the firewall will receive the packet and send it out without inspecting it. Transitioning to fail-open mode causes an internal restart and a brief disruption to traffic.
  - In fail-close mode, the firewall will drop all the packets it receives. The fail-close mode also brings down the CN-NFGW and releases the slot allocated to let other licensed CN-NFGW use that slot.

PAN-CN-NGFW

PAN-CN-NGFW	

Image path for the CN-NGFW container image
image
`containers:`
`- name: pan-ngfw-container`
`image:`
`<your-private-registry-image-path>`
	Edit the image path to point to the location to which you have uploaded the docker image for the CN-NGFW container.

Note:
• The following annotation identifies the PAN-NGFW daemonset:
`paloaltonetworks.com/app: pan-ngfw-ds`
Do not modify this value.
• The following annotation identifies the firewall name (“pan-fw”):
`paloaltonetworks.com/firewall: pan-fw`
In `pan-cni-configmap.yaml`, this firewall name must match exactly in the `cni_network_config: “firewall”`
And this annotation should match exactly in the application yaml that you use to deploy each application pod.
	The CN-NGFW Pod on each node secures the application pods and namespaces that have the annotation:
	`paloaltonetworks.com/firewall: pan-fw`
	Keep this annotation as is.

PAN-CNI-CONFIGMAP

PAN-CNI-CONFIGMAP	
List of firewall names that the application pod might belong to: `"firewall": [ "pan-fw" ]`	While no modifications are required, if you change the annotation `paloaltonetworks.com/firewall: pan-fw` in the `pan-cn-ngfw.yaml`, you must replace the value in `"firewall": [ "pan-fw" ]` to match.
`"exclude_namespaces": []`	While no modifications are required, if you want to exclude specific namespaces, add it to **`“exclude_namespaces"`**, so that the application pod annotation in that namespace is ignored and traffic is not redirected to the CN-NGFW pod for inspection.

PAN-CNI-CONFIGMAP	
`"security_namespaces": [ "kube-system" ]`	Add the namespaces in which you have deployed the CN-NGFW daemonset in security_namespaces. The default namespace is kube-system.
`“interfaces”`	Add the interfaces in the application pods from which you want to redirect traffic to the CN-NGFW pod for inspection. By default, only eth0 traffic is inspected, and you can add additional interfaces as a comma-separated list of strings e.g. [“eth0”, “net1”].

PAN-CNI

PAN-CNI	

Image path for the PAN-CNI container image that has the CNI binaries and the CNI network config file on each node.
`containers:`
`  name: install-pan-cni`
`  image: <your-private-registry-image-path>`
	Edit the image path to point to the location to which you have uploaded the docker image for the PAN-CNI container.

Secure Kubernetes Workloads with CN-Series

CN-Series offers threat protection for inbound, outbound, and east-west traffic between container trust zones and other workload types, without slowing the speed of development.

Deploy the CN-Series for Layer 7 visibility into container traffic and enforce security policies with threat prevention profiles to protect allowed traffic across Kubernetes namespace boundaries, and share that context with the hardware and VM-Series firewalls to ensure a consistent policy enforcement model across your entire hybrid cloud environment.

CN-Series Prerequisites

System Requirements for the Kubernetes Cluster

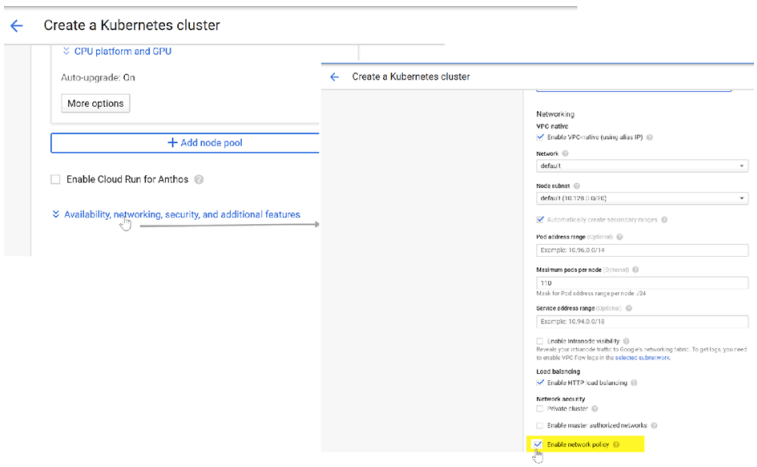
System requirements for the cluster in which you are deploying the CN-Series firewall.

While the CPU, memory and disk storage will depend on your needs, here are some guidelines:

Resource	CN-MGMT (StatefulSet Pod for Fault Tolerance)	CN-NGFW (DaemonSet Pod)
Memory (min)	2G	2G
Memory (max)	4G	2.5G
CPU (Min)	2	1
CPU (Max)	None	None
Disk	50 GB	N.A.

- Kubernetes cluster running Kubernetes version -1.14 or 1.15.

If your cluster is on GKE, make sure to enable the Kubernetes Network Policy API to allow the cluster administrator to specify which pods are allowed to communicate with each other. This API is required for the CN-NGFW and CN-MGMT Pods to communicate.



- Container Images—4 docker files. See Components Required to Secure Kubernetes Clusters with CN-Series Firewall.
- YAML files for your environment. See Components Required to Secure Kubernetes Clusters with CN-Series Firewall.
- Panorama OS version 10.0.0 (minimum version)

Panorama must be able to establish network connectivity with the Kubernetes cluster API server endpoint. In addition, you must add the ports that Panorama uses to fetch updates and communicate with the managed devices to an allow list, see Ports Used on Panorama.

- Kubernetes plugin on Panorama version 1.0.0 (minimum version)

For information on scaling, see CN-Series Supported Scale Factors and for the supported environments CN-Series Deployment—Supported Environments.

System Requirements for On-Premises Kubernetes Deployments

Review the following prerequisites for your on-premises deployments:

- Ensure that the container images are accessible to all nodes in the Kubernetes cluster.
- Set up a persistent volume within the cluster for both the CN-MGMT pods. Because the CN-MGMT pods are deployed as a StatefulSet, which actively manage the CN-NGFW pods, both instances must have access to the persistent volume.

Register the CN-Series Firewall Auth Code

CN-Series firewall licensing is managed by the Kubernetes plugin on Panorama. The CN-Series firewalls are licensed based on the number of Kubernetes nodes you want to secure with one token is allocated to each CN-NGFW pod that secures a Kubernetes node. Node-based licensing provides you the flexibility of running as many pods as you need on a node that is being secured with the CN-Series firewall. Since the pods can move to the different nodes within the cluster the actual throughput the firewall will need per node can vary over time. Given the dynamic nature of Kubernetes, with node-based licensing you do not need to predict your throughput needs upfront. Your security administrator can provide an initial estimate of the total number of nodes they want to protect, and as your security needs increase, you can then purchase additional tokens as required and add it to Panorama. The license bundle can be CN-X-BASIC, CN-X-BND1, or CN-X-BND2.

- The basic bundle includes the firewall capacity license and support entitlement.
- Bundle 1 includes Threat Prevention and support entitlement.
- Bundle 2 includes Threat Prevention, URL Filtering, DNS Service, Enterprise DLP and WildFire subscription and support entitlement.

For more details on the licenses, see Subscriptions.

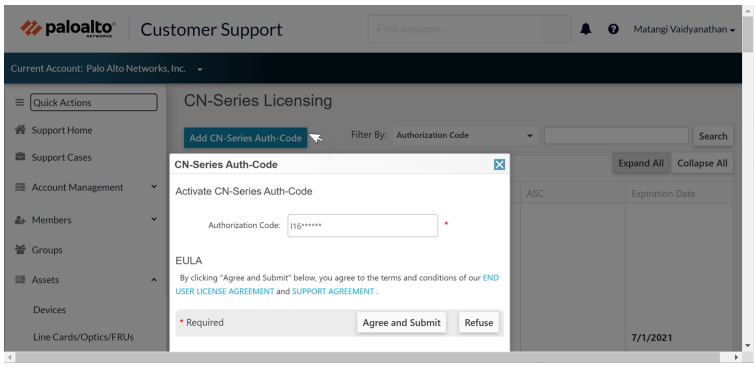
Before you begin with Install the Kubernetes Plugin and Set up Panorama for CN-Series , you must register the auth code for the CN-Series firewall.

STEP 1 | Log in to the Palo Alto Networks Customer Support website with your account credentials.

If you need a new account, see Create a Support Account.

STEP 2 | Select **Assets** > **CN-Series Licensing** > **Add CN-Series Auth-Code**.

STEP 3 | Enter the auth code you received by email in **Add CN-Series Auth-Code**, and **Agree and Submit** to save your input.



The page refreshes to display the list of auth codes registered to your support account. You can track the total number of CN-Series tokens you purchased and the number of tokens that are still available for use against each auth code. When all the available tokens are used, the auth code does not display on the CN-Series Auth-Codes page. To view all the assets that are deployed, select **Assets** > **Devices**.

Allocate CN-Series Tokens to Panorama

The CN-Series firewalls connect to Panorama and retrieve the appropriate licenses based on the CN-Series license bundle you purchased and registered on the CSP. If your Panorama is not connected directly to the internet, you need to allocate tokens to the Panorama on which you plan to install the Kubernetes plugin

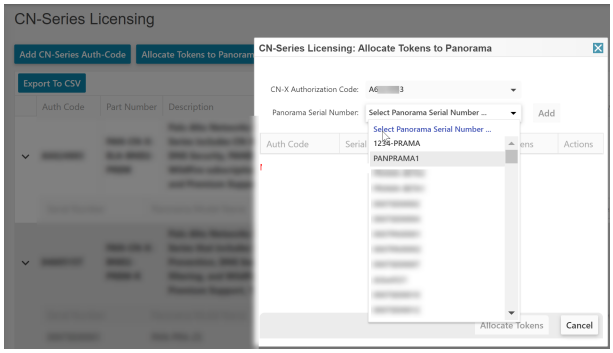
and manage the CN-Series deployment, download this license file from the CSP, and manually upload it to Panorama. This process allows you to make sure that Panorama has the tokens to successfully the license the CN-Series firewall instance on each node within the Kubernetes cluster.

You must carefully plan the number of tokens you want to allocate to Panorama. If you need to change the number of tokens, you must redeploy the CN-Series firewalls.

STEP 1 | Log in to the Palo Alto Networks Customer Support website with your account credentials.

STEP 2 | Select **Assets** > **CN-Series Licensing** > **Allocate Tokens to Panorama**.

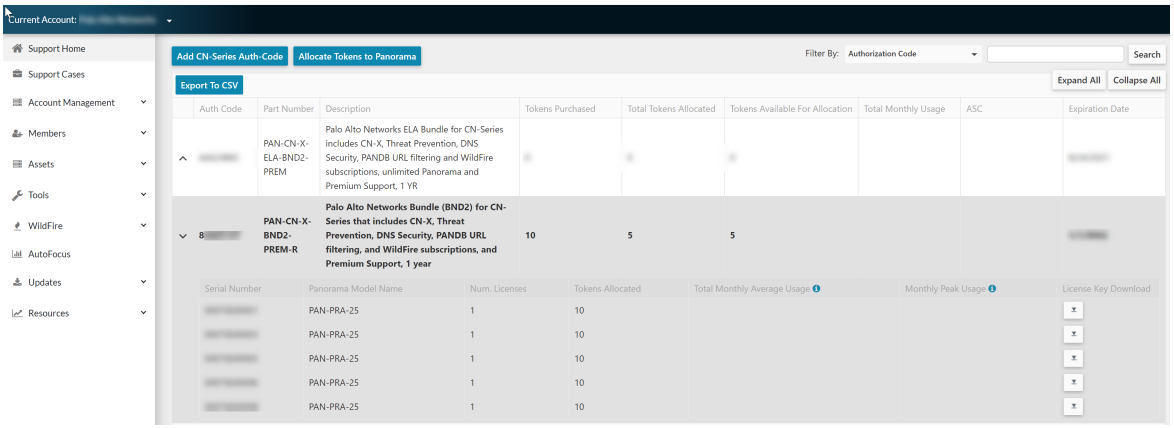
STEP 3 | Select the CN-Series auth code that you have activated, and the Panorama Serial number and **Allocate Tokens**.



The page refreshes to display the CN-Series tokens you allocated and the license file is ready for download.

STEP 4 | Download the license key.

Save the key. You must upload this key to Panorama when you Install the Kubernetes Plugin and Set up Panorama for CN-Series .



Generate the Auto-Registration PIN for the CN-Series

For automated deployments, you must generate the auto-registration PIN password on the Customer Support Portal and these values are unique to your Palo Alto Networks support account. The firewalls require this PIN ID and value to get the device certificate, which authorize access to any site license entitlements and secure access the Palo Alto cloud-delivered services.

STEP 1 | Log in to the Palo Alto Networks Customer Support website with your account credentials.

If you need a new account, see How to Create a New Customer Support Portal User Account.

STEP 2 | Select **Assets** > **Device Certificates** > **Generate Registration PIN**.

STEP 3 | Save the PIN ID and value.

Save the PIN ID and value. This PIN ID and value are inputs in the `pan-cn-mgmt-secret.yaml` file used to Deploy the CN-Series Firewalls. Make sure to launch the firewall before the PIN expires.

Create Service Accounts for Cluster Authentication

The CN-Series firewall requires three Service accounts with the minimum permissions that authorize it to communicate with your Kubernetes cluster resources. The service account (pan-plugin-user) created with the `plugin-serviceaccount.yaml` enables the Kubernetes plugin on Panorama to authenticate with the Kubernetes cluster for retrieving metadata on the pods.The other two yaml files, `pan-mgmt-serviceaccount.yaml` and `pan-cni-serviceaccount.yaml`, create the pan-mgmt-sa and the pan-cni-sa service accounts to enable the authentication between the fault tolerant CN-Mgmt pods, and between the CN-MGMT pod and the CN-NGFW pods.

By default, the YAML files create the service account and the secret in the kube-system namespace; the Kubernetes plugin will only look for the secret in the kube-system namespace.

To create the service accounts, your Kubernetes cluster should be ready.

STEP 1 | Run the service account YAML for the **`plugin-serviceaccount.yaml`**.

This service account enables the permissions that Panorama requires to authenticate to the GKE cluster for retrieving Kubernetes labels and resource information. This service account is named pan-plugin-user by default.

1. **`kubectl apply -f plugin-serviceaccount.yaml`**
2. **`kubectl -n kube-system get secrets | grep pan-plugin-user-token`**

   To view the secrets associated with this service account.
3. **`kubectl -n kube-system get secrets <secrets-from-above-command> -o json >> cred.json`**

   Create the credential file, named cred.json in this example, that includes the secrets and save this file. You need to upload this file to Panorama to set up the Kubernetes plugin for monitoring the clusters in Install the Kubernetes Plugin and Set up Panorama for CN-Series .

STEP 2 | Run the **`pan-mgmt-serviceaccount.yaml`** and **`pan-cni-serviceaccount.yaml`**.

The `pan-mgmt-serviceaccount.yaml` creates a service account named pan-sa, and is required to enable the CN-MGMT and CN-NGFW Pods to communicate with each other, the PAN-CNI, and the Kubernetes API server. If you modify this service account name, you must also update the YAML files that you use to deploy the CN-MGMT and CN-NFGW Pods.The `pan-cni-serviceaccount.yaml` creates a service account named pan-cni-sa.

`kubectl apply -f pan-mgmt-serviceaccount.yamlkubectl apply -f pan-cni-serviceaccount.yaml`

`kubectl apply -f pan-cni-serviceaccount.yaml`

STEP 3 | Verify the service accounts.

`kubectl get serviceaccounts -n kube-system`

Install the Kubernetes Plugin and Set up Panorama for CN-Series

You can deploy the Panorama appliance on-premises or in the cloud, as long as the Panorama appliance can connect with the Kubernetes clusters where you want to deploy the CN-Series firewalls. This workflow takes you through the process of installing the Kubernetes plugin, activating the auth code and setting up the Kubernetes plugin to monitor your clusters.

STEP 1 | Deploy a Panorama with software version 10.0 and install the minimum content version.

1. **Check Now** (**Panorama** > **Dynamic Updates**) for the minimum content release version on PAN-OS 10.0.

   See PAN-OS Release Notes.
2. **Check Now** (**Panorama** > **Software**) for the software version.

   Locate and download the model-specific file for the release version to which you are upgrading. For example, to upgrade an M-Series appliance to Panorama 10.0.0, download the Panorama_m-10.0.0 image; to upgrade a Panorama virtual appliance to Panorama 10.0.0, download the Panorama_pc-10.0.0 image.

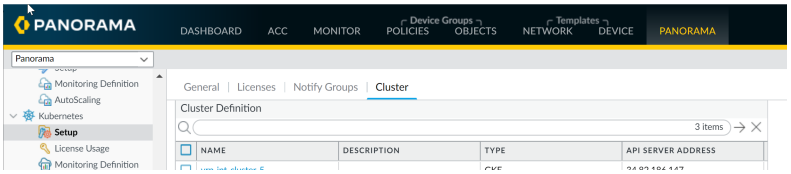
   After a successful download, the **Action** column changes from Download to Install for the downloaded image.

STEP 2 | Verify that your Panorama is in Panorama mode, if you want Panorama to collect the firewall logs.

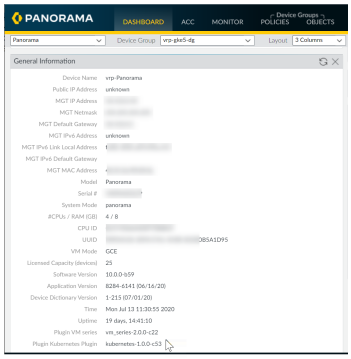
STEP 3 | Install the Kubernetes plugin on Panorama.

1. Log in to the Panorama Web Interface, select **Panorama** > **Plugins** and click **Check Now** to get the list of available plugins.
2. Select **Download** and **Install** the Kubernetes plugin

   After you successfully install, Panorama refreshes and the Kubernetes plugin displays on the **Panorama** tab.

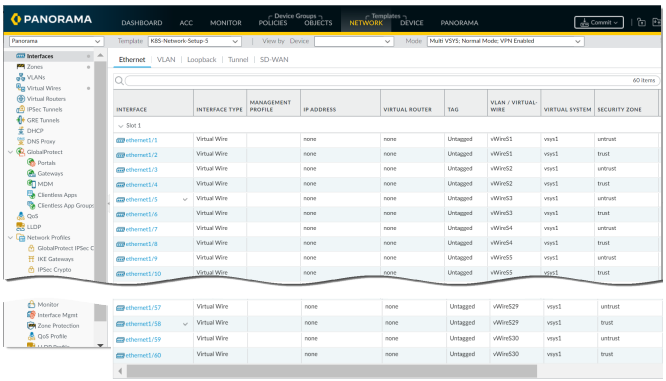


You can also verify the General Information widget on the Panorama **Dashboard**.



STEP 4 | Commit your changes on Panorama.

Click **Commit to Panorama**. The commit creates the interfaces and virtual wires and the associated template named **K8S-Network-Setup**. It can take up to one minute for the interfaces to display on Panorama. The template has 30 virtual wires; a pair of interfaces that are part of a virtual wire to secure an application. Therefore, the CN-NGFW can secure a maximum of 30 application pods on a node.



STEP 5 | Get the CN-Series license tokens on Panorama.

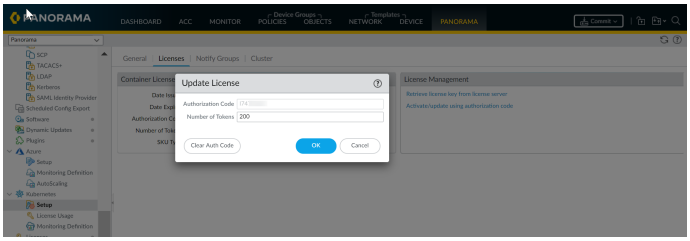
1. Choose your workflow.

   If your Panorama has internet access you can activate the authcode on the Kubernetes plugin. If your Panorama does not have internet access, you must Allocate CN-Series Tokens to Panorama and then upload the license key.

   - Activate the authcode on the Kubernetes plugin.

     1. Select **Panorama** > **Plugins** > **Kubernetes** > **Setup** > **Licenses**.
     2. Select **Activate/update using authorization code**, and enter the auth code and the number of licenses you need for each of the nodes you want to protect with CN-Series firewalls.

        You must activate the auth code to enable the CN-MGMT to connect with Panorama. Every node in the cluster uses a token.

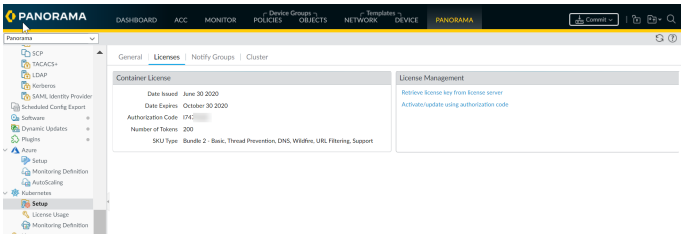
        If you deploy the CN-Series firewall without activating the license, you have a 4-hour grace period after which the firewalls stop processing traffic. After the grace period, the CN-NGFW instances will either failopen (default) or failclosed based on the (FAILOVER_MODE) defined in the `pan-cn-ngfw-configmap.yaml`.In fail-open mode the firewall will receive the packets and send it out without applying any security policies. Transitioning to fail-open will require a restart and cause a brief disruption of traffic during that (expected around 10-30 seconds). In fail-closed mode, the firewall will drop all the packets it receives. A fail-close will bring down the CN-NGFW Pod and release the tokens to the available token pool for licensing new CN-NGFW Pods.
   - Upload the license key to Panorama.

     1. Log in to the CLI on Panorama.

        Use SSH to log into the CLI.
     2. Enter **`request plugins kubernetes manually-upload-license file`**.

3. Paste the contents of the file.

2. Verify that the number of available license tokens is updated.



STEP 6 | Generate VM Auth Key.

Log in to the Panorama CLI, and use the following operational command:

```
request bootstrap vm-auth-key generate lifetime <1-8760>
```

For example to generate a key that is valid for 24 hrs, enter the following:

```
request bootstrap vm-auth-key generate lifetime 24
```

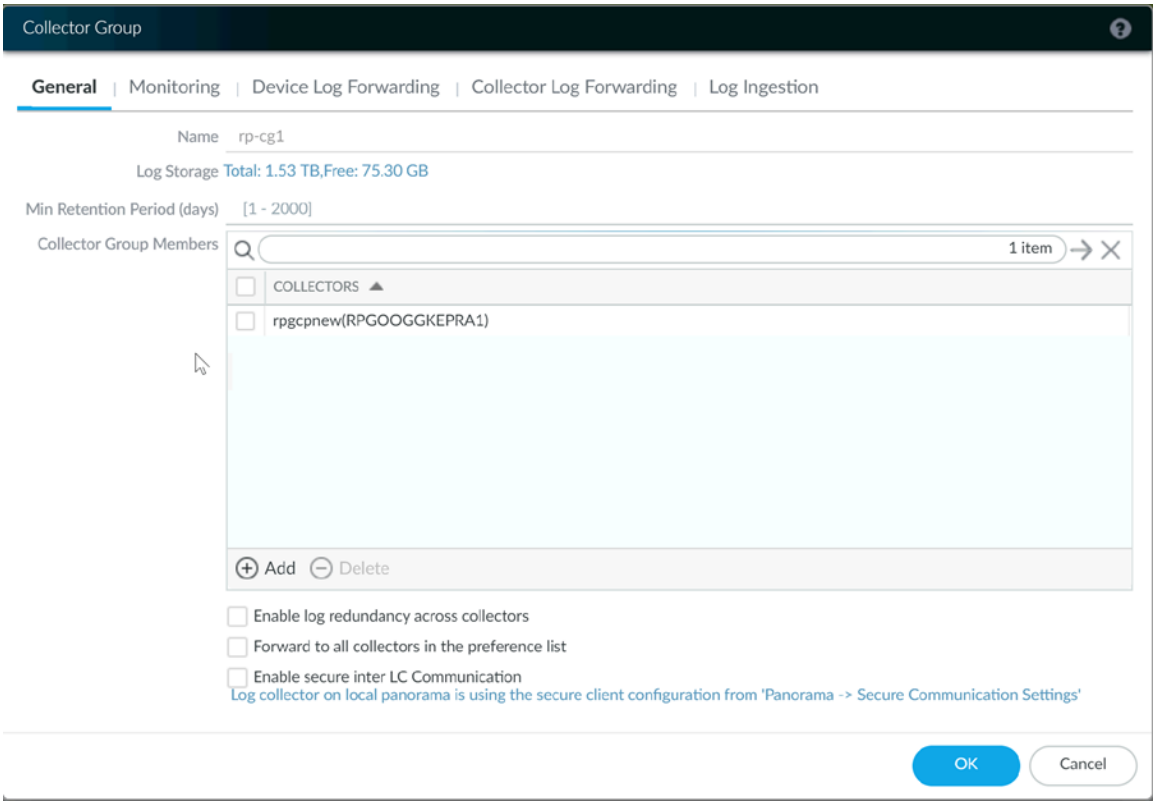
```
VM auth key 755036225328715 generated.
Expires at: 2020/01/29 12:03:52
```

STEP 7 | Create a parent Device Group and Template Stack.

You must create a template stack and a device group, and you will later reference this template stack and device group when you edit the YAML file to deploy the CN-MGMT Pods. The Kubernetes plugin on Panorama creates a template called K8S-Network-Setup, and this template will be part of the template stack you define here.

1. Create a template stack and add the K8S-Network-Setup template the template stack.
    1. Select **Panorama** > **Templates** and **Add Stack**.
    2. Enter a unique **Name** to identify the stack.
    3. Add and select the K8S-Network-Setup template.
    4. Click **OK**.
2. Create a device group.
    1. Select **Panorama** > **Device Groups** and click **Add**.
    2. Enter a unique **Name** and a **Description** to identify the device group.
    3. Select the **Parent Device Group** (default is **Shared**) that will be just above the device group you are creating in the device group hierarchy.
    4. Click **OK**.
3. (If you are using a Panorama virtual appliance) Create a Log Collector and add it to a Log Collector Group.
    1. Select **Panorama** > **Collector Groups** and **Add** a Collector Group.
    2. Enter a **Name** for the Collector Group.
    3. Enter the **Minimum Retention Period** in days (1 to 2,000) for which the Collector Group will retain firewall logs.

       By default, the field is blank, which means the Collector Group retains logs indefinitely.
    4. **Add** Log Collectors (1 to 16) to the Collector Group Members list.



5. Select **Commit** > **Commit and Push** and then **Commit and Push** your changes to Panorama and the Collector Group you configured.

STEP 8 | Set up the Kubernetes plugin for monitoring the clusters.

Add the Kubernetes cluster information so that Panorama can access the API endpoint for the cluster and authenticate using the service account credentials in order to query the API server. You can add up to 32 service account credentials on Panorama; Panorama supports only one service account credential for a Kubernetes cluster.

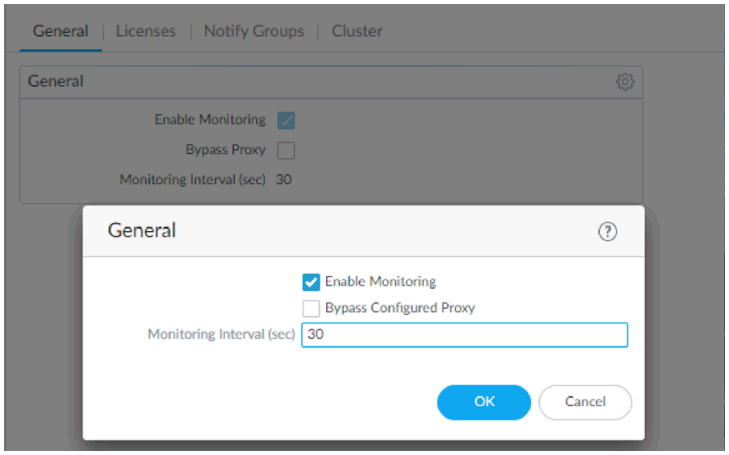
To ensure that the plugin and the Kubernetes clusters are in sync, the plugin polls the Kubernetes API server at a configured interval and listens for notifications from the Kubernetes Watch API at a predefined interval (not user configurable).

After you add the cluster information, Panorama always retrieves the such as service, node, replica set and creates tags for them to enable you to gain visibility and to control traffic to and from these clusters. Optionally, you can specify whether you want Panorama to retrieve information on the Kubernetes labels and create tags for these also. See IP-Address-to-Tag Mapping of Kubernetes Attributes for a list of supported attributes.

1. Check the monitoring interval.

   The default interval at which Panorama polls the Kubernetes API server endpoint is 30 seconds.

   1. Select **Panorama** > **Plugins** > **Kubernetes** > **Setup** > **General**.
   2. Verify that **Enable Monitoring** is selected.
   3. Click the gear icon to edit the **Monitoring Interval** and change to a range of 30-300 seconds.



2. Select **Panorama** > **Plugins** > **Kubernetes** > **Setup** > **Cluster**, and **Add Cluster**.

   Make sure that you do not add the same Kubernetes cluster to more than one Panorama (single instance or HA pair) appliance because you may see inconsistencies in how the IP-address-to mappings are registered to the device groups.
3. Enter a **Name** and the **API Server Address**.

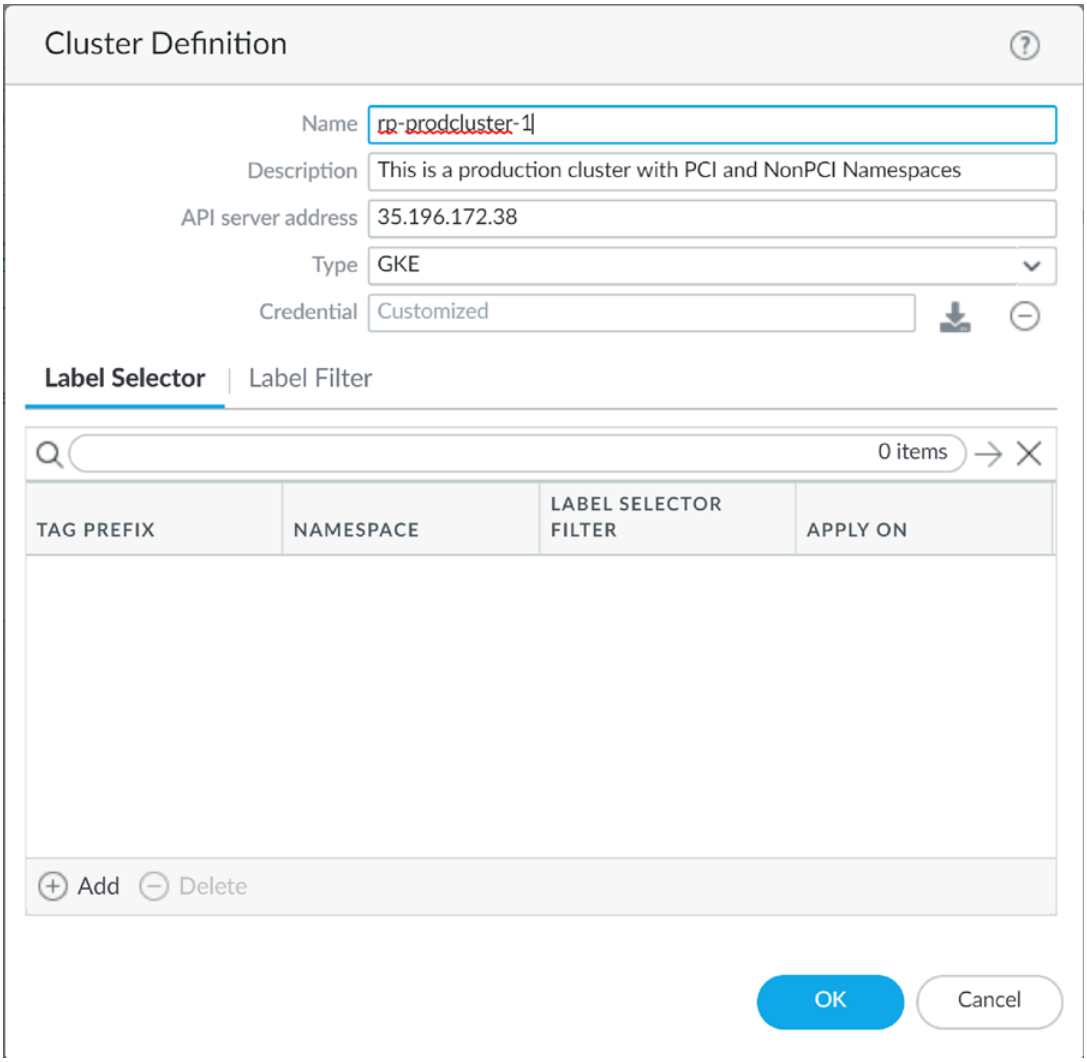
   This is the Endpoint IP address for the cluster that you must get from your Kubernetes deployment. Enter a name, up to 20 characters, to uniquely identify the name of the cluster. You cannot modify this name because Panorama uses the cluster name when it creates tags for the pods, nodes, services it discovers within the cluster.

   The format of the API server address can be a hostname or an IP address:port number, and you do not need to specify the port if you are using port 443, which is the default port.
4. Select the **Type** of environment on which your cluster is deployed.

   The available options are AKS, EKS, GKE, Native Kubernetes, OpenShift, and Other.
5. Upload the service account **Credential** that Panorama requires to communicate with the cluster. In the previous step of this workflow, this filename for this service account was `plugin-svc-acct.json`.

   If your service credential file is over 10KB, you must gzip the file and then do a base64 encoding of the compressed file before you upload or paste the contents of the file into the Panorama CLI or API.
6. Click **OK**.

   You can leave the Label Filter and Label Selector configuration for later. This is an optional task that enables you to retrieve any custom or user-defined labels for which you want Panorama to create tags.



STEP 9 | (Optional) Configure a proxy for each cluster.

Unlike the other plugins, the Kubernetes plugin does not use the proxy configured under **Panorma** > **Setup** > **Services**. Instead if you want to enable or bypass a proxy, you must enter the proxy for each cluster. When configured, the Kubernetes plugin uses this proxy server IP address to make all API calls to the API server for this cluster.

1. Log in to the CLI on Panorama.
2. Enter the following CLI commands to configure the proxy server for this Kubernetes cluster.

```
> configure> set plugins kubernetes setup cluster-credentials <cluster-name>  cluster-proxy enable-proxy <yes/no> proxy-port <port>  proxy-server <IP> proxy-user <username>  secure-proxy-password <password>
*** username and password are optional ***
```

STEP 10 | Next steps:

1. Get the Images and Files for the CN-Series Deployment
2. Deploy the CN-Series Firewalls
3. Configure Panorama to Secure a Kubernetes Deployment

Get the Images and Files for the CN-Series Deployment

Use the following instructions to get the YAML files from GitHub and to download the docker images from the Palo Alto Networks CSP and push it to your container registry before you continue to Deploy the CN-Series Firewalls.

Download the docker images and YAML files.

1. Get the compressed tar archives from the Palo Alto Networks Customer Support Portal (CSP).
   - PanOS_cn-10.0.0-b7.tgz - for CN-MGMT and CN-NGFW Pods.
   - Pan_cn_mgmt_init-1.0.0-b1-c1.tgz - for the init container that runs as a part of the CN-MGMT Pod.
   - Pan_cni-1.0.0-b1-c3.tgz - for the PAN-CNI Pod.
2. Get the YAML files from GitHub.
   - Get the files from the Native-k8s folder for use with native Kubernetes on-premises or cloud deployments.
   - Get the files from the respective Managed Kubernetes folder for AKS, EKS, or GKE.
3. Retrieve the docker images and push it to your container registry.

   For example, on a GKE deployment, you will upload the images to a Container Registry on GKE and get the image path for referencing in the YAML files.Use the following commands on a client system running the docker engine.

   1. Load the images.

      ```
      docker load -i PanOS_cn-10.0.0-b1.tgz
      ```

      ```
      docker load -i Pan_cn_mgmt-init-0.0.1-b1.tgz
      ```

      ```
      docker load -i Pan_cni-0.0.1-b1.tgz
      ```

      After these steps, "docker images" will display the image, for example, "paloaltonetworks/panos_cn_mgmt:10.0.0-b1".
   2. Tag these images to include your private registry detail.

      ```
      docker tag paloaltonetworks/panos_cn_mgmt:10.0.0-b1 <your_registry>/
      paloaltonetworks/panos_cn_mgmt:10.0.0-b1
      ```

      ```
      docker tag paloaltonetworks/panos_cn_ngfw:10.0.0-b1 <your_registry>/
      paloaltonetworks/panos_cn_ngfw:10.0.0-b1
      ```

      ```
      docker tag paloaltonetworks/pan_cn_mgmt_init:10.0.0-b1 <your_registry>/
      paloaltonetworks/pan_cn_mgmt_init:0.0.1-b1
      ```

      ```
      docker tag paloaltonetworks/pan_cni:10.0.0-b1 <your_registry>/
      paloaltonetworks/pan_cni:0.0.1-b1
      ```
   3. Push these images to your private registry.

      ```
      docker push <your_registry>/paloaltonetworks/panos_cn_mgmt:10.0.0-b1
      ```

      ```
      docker push <your_registry>/paloaltonetworks/panos_cn_ngfw:10.0.0-b1
      ```

      ```
      docker push <your_registry>/paloaltonetworks/pan_cn_mgmt_init:0.0.1-b1
      ```

      ```
      docker push <your_registry>/paloaltonetworks/pan_cni:0.0.1-b1
      ```

Deploy the CN-Series Firewalls

After you review the CN-Series Core Building Blocks, and the high-level overview of the workflow in Secure Kubernetes Workloads with CN-Series you can start with deploying the CN-Series firewalls to secure traffic between containers within the same cluster, and between containers and other workload types, such as virtual machines and bare-metal servers.

If you on the OpenShift environment, see Deploy the CN-Series on OpenShift.

STEP 1 | Set up your Kubernetes cluster.

1. Verify that the cluster has adequate resources.
   - Make sure that cluster has the CN-Series Prerequisites resources to support the firewall:
     - For the CN-NGFW (DaemonSet pods), you need a minimum of 1 CPU/ 2 GB on each node.
     - For each instance of CN-MGMT (StatefulSet pods), you need a minimum of 2 CPU, 2 GB memory and 50 GB disk storage (persistent volume) in the cluster. Provision and run 2 such instances for fault-tolerance.

       The CPU, memory and disk storage allocation will depend on your needs. See CN-Series Supported Scale Factors.
   - Collect the Endpoint IP address for setting up the API server on Panorama. Panorama uses this IP address to connect to your Kubernetes cluster.
   - Collect the template stack name, device group name, Panorama IP address, VM-auth-key, auto-registration PIN ID and value, and optionally the Log Collector Group Name.
   - Location of the container image repository to which you downloaded the images.

STEP 2 | Edit the YAML files to provide the details required to deploy the CN-Series firewalls.

You need to replace the image path in the YAML files to include the path to your private registry and provide the required parameters. See Editable Parameters in CN-Series Deployment YAML Files for details.

STEP 3 | Deploy the CNI DaemonSet.

The CNI container is deployed as a DaemonSet (one pod per node) and it creates two interfaces on the CN-NGFW pod for each application deployed on the node. When you use the kubectl commands to run the pan-cni YAML files, it becomes a part of the service chain on each node.

1. Verify that you have modified the pan-cni-configmap and pan-cni YAML files.
2. Verify that you have created the service account using the pan-cni-serviceaccount.yaml.

   See Create Service Accounts for Cluster Authentication.
3. Use Kubectl to run the pan-cni-configmap.yaml.

   ```
   kubectl apply -f pan-cni-configmap.yaml
   ```
4. Use Kubectl to run the pan-cni.yaml.

   ```
   kubectl apply -f pan-cni.yaml
   ```

STEP 4 | Deploy the CN-MGMT StatefulSet.

By default, the management plane is deployed as a StatefulSet that provides fault tolerance. Up to 30 firewall CN-NGFW pods can connect to a CN-MGMT StatefulSet.

1. (Required for statically provisioned PVs only) Deploy the Persistent Volumes (PVs) for the CN-MGMT StatefulSet.

1. Create the directories to match the local volume names defined in the pan-cn-pv-local.yaml.

   You need six (6) directories on at least 2 worker nodes. Log in to each worker node on which the CN-MGMT StatefulSet will be deployed to create the directories. For example, to create directories named /mnt/pan-local1 to /mnt/pan-local6, use the command:

   ```
   mkdir -p /mnt/pan-local1 /mnt/pan-local2 /mnt/pan-local3 /mnt/pan-
   local4 /mnt/pan-local5 /mnt/pan-local6
   ```

2. Modify pan-cn-pv-local.yaml.

   Match the hostname under `nodeaffinity`, and verify that you have modified the directories you created above in `spec.local.path` then deploy the file to create a new storageclass pan-local-storage and local PVs.

2. Verify that you have modified the pan-cn-mgmt-configmap and pan-cn-mgmt YAML files.

   Sample pan-cn-mgmt-configmap from EKS.

   ```
    Session Contents Restored
   apiVersion: v1
   kind: ConfigMap
   metadata:
     name: pan-mgmt-config
     namespace: kube-system
   data:
     PAN_SERVICE_NAME: pan-mgmt-svc
     PAN_MGMT_SECRET: pan-mgmt-secret
     # Panorama settings
     PAN_PANORAMA_IP: "x.y.z.a"
     PAN_DEVICE_GROUP: "dg-1"
     PAN_TEMPLATE_STACK: "temp-stack-1"
     PAN_CGNAME: "CG-EKS"
     # Intended License Bundle type - "CN-X-BASIC", "CN-X-BND1", "CN-X-
   BND2"
     # based on the authcode applied on the Panorama K8S plugin"
     PAN_BUNDLE_TYPE: "CN-X-BND2"
   #Non-mandatory parameters
     # Recommended to have same name as the cluster name provided in
    Panorama Kubernetes plugin - helps with easier identification of pods
    if managing multiple clusters with same Panorama
     #CLUSTER_NAME: "Cluster-name"
     #PAN_PANORAMA_IP2: "passive-secondary-ip"
     # Comment out to use CERTs otherwise bypass encrypted connection to
    etcd in pan-mgmt.
     # Not using CERTs for etcd due to EKS bug
     ETCD_CERT_BYPASS: ""          # No value needed
     # Comment out to use CERTs otherwise PSK for IPSec between pan-mgmt
    and pan-ngfw
     # IPSEC_CERT_BYPASS: ""         # No values needed
   ```

   Sample pan-cn-mgmt.yaml

   ```
   initContainers:        - name: pan-mgmt-init          image: <your-
   private-registry-image-path>
   ```

   ```
    containers:        - name: pan-mgmt          image: <your-
   private-registry-image-path>          terminationMessagePolicy:
    FallbackToLogsOnError
   ```

3. Use Kubectl to run the yaml files.

   .**kubectl apply -f pan-cn-mgmt-configmap.yaml**

   kubectl apply -f pan-cn-mgmt.yaml

   kubectl apply -f pan-cn-mgmt-secret.yaml

   You must run the pan-mgmt-serviceaccount.yaml, only if you had not previously completed the Create Service Accounts for Cluster Authentication.
4. Verify that the CN-MGMT pods are up.

   It takes about 5-6 minutes.

   Use **kubectl get pods -l app=pan-mgmt -n kube-system**

   ```
   NAME READY STATUS RESTARTS AGEpan-mgmt-sts-0 1/1 Running 0 27hpan-mgmt-
   sts-1 1/1 Running 0 27h
   ```

STEP 5 | Deploy the CN-NGFW pods.

By default the firewall dataplane CN-NGFW pod is deployed as a DaemonSet. An instance of the CN-NFGW pod can secure traffic for up to 30 application Pods on a node.

1. Verify that you have modified the YAML files as detailed in PAN-CN-NGFW-CONFIGMAP and PAN-CN-NGFW.

   containers: - name: pan-ngfw-container image: <your-private-registry-image-path>
2. Use Kubectl apply to run the pan-cn-ngfw-configmap.yaml.

   kubectl apply -f pan-cn-ngfw-configmap.yaml
3. Use Kubectl apply to run the pan-cn-ngfw.yaml.

   kubectl apply -f pan-cn-ngfw.yaml
4. Verify that all the CN-NGFW Pods are running. (one per node in your cluster)

   This is a sample output from a 4-node on-premises cluster.

   kubectl get pods -n kube-system -l app=pan-ngfw -o wide

   ```
   NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES
   pan-ngfw-ds-8g5xb 1/1 Running 0 27h 10.233.71.113 rk-k8-node-1 <none>
   <none>
   pan-ngfw-ds-qsrm6 1/1 Running 0 27h 10.233.115.189 rk-k8-vm-worker-1
   <none> <none>
   pan-ngfw-ds-vqk7z 1/1 Running 0 27h 10.233.118.208 rk-k8-vm-worker-3
   <none> <none>
   pan-ngfw-ds-zncqg 1/1 Running 0 27h 10.233.91.210 rk-k8-vm-worker-2 <none>
   <none>
   ```

STEP 6 | Verify that you can see CN-MGMT, CN-NGFW and the PAN-CNI on the Kubernetes cluster.

kubectl -n kube-system get pods

```
pan-cni-5fhbg 1/1 Running 0 27hpan-cni-9j4rs 1/1 Running 0 27hpan-cni-ddwb4
1/1 Running 0 27hpan-cni-fwfrk 1/1 Running 0 27hpan-cni-h57lm 1/1 Running 0
27hpan-cni-j62rk 1/1 Running 0 27hpan-cni-lmxdz 1/1 Running 0 27hpan-mgmt-
sts-0 1/1 Running 0 27hpan-mgmt-sts-1 1/1 Running 0 27hpan-ngfw-ds-8g5xb
```

```
1/1 Running 0 27hpan-ngfw-ds-qsrm6 1/1 Running 0 27hpan-ngfw-ds-vqk7z 1/1
Running 0 27hpan-ngfw-ds-zncqg 1/1 Running 0 27h
```

STEP 7 | Annotate the application yaml or namespace so that the traffic from their new pods is redirected to the firewall.

You need to add the following annotation to redirect traffic to the CN-NGFW for inspection:

```
annotations:            paloaltonetworks.com/firewall: pan-fw
```

For example, for all new pods in the “default” namespace:

```
kubectl annotate namespace default paloaltonetworks.com/firewall=pan-fw
```

On some platforms, the application pods can start when the pan-cni is not active in the CNI plugin chain. To avoid such scenarios, you must specify the volumes as shown here in the application pod YAML.

```
volumes:      - name: pan-cni-ready        hostPath:
    path: /var/log/pan-appinfo/pan-cni-ready          type:
 Directory
```

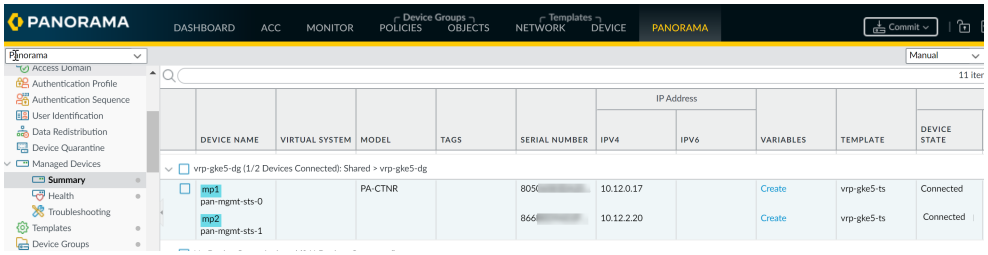
STEP 8 | Deploy your application in the cluster.

Configure Panorama to Secure a Kubernetes Deployment

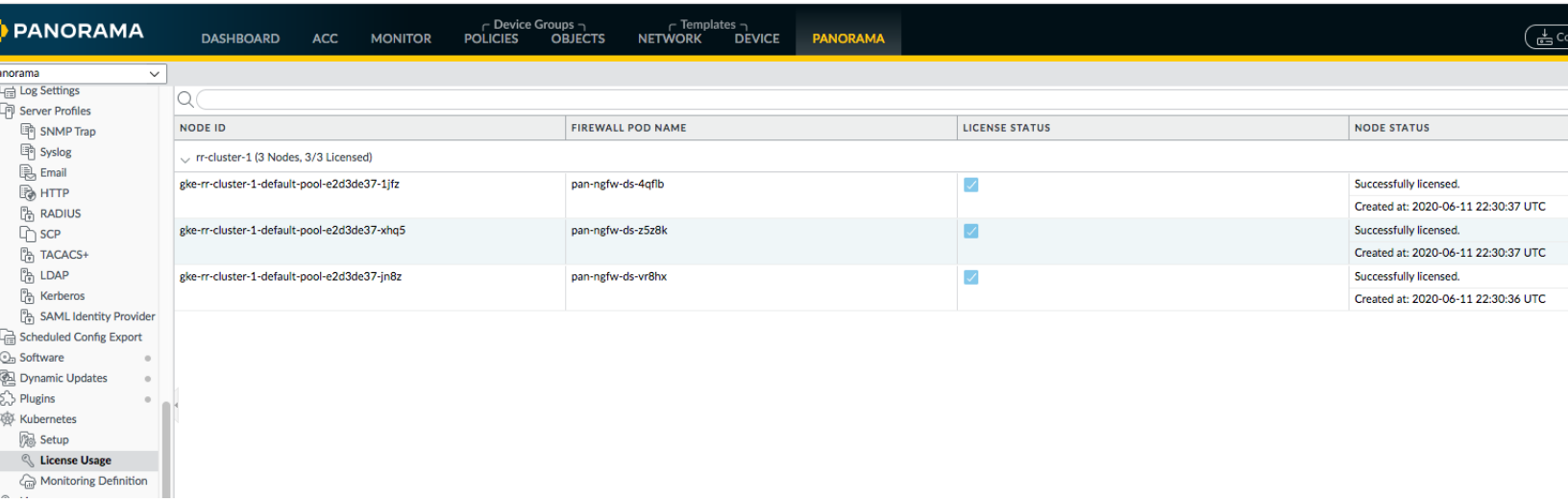
After you Install the Kubernetes Plugin and Set up Panorama for CN-Series and Deploy the CN-Series Firewalls, to monitor the Kubernetes cluster and configure the Security policies that enable traffic enforcement, you need to complete the following tasks.

STEP 1 | Verify that the CN-MGMT pods are registered on Panorama and the CN-NGFW pods are licensed.

1. Select **Panorama** > **Managed Devices** > **Summary**.



2. Select **Panorama** > **Plugins** > **Kubernetes** > **License Usage** to verify that each node within the cluster is allocated a license token.



STEP 2 | Create a Log Forwarding profile to forward logs to Panorama.

The profile defines the destinations for the different logs that will be generated on the firewall.

1. Select the device group you created for the k8s deployment from the **Device Group** drop-down.
2. Select **Objects** > **Log Forwarding** and click **Add**.
3. Enter a **Name** to identify the profile. If you want to automatically assign the profile to new security rules and zones, enter **`default`**. If you don't want a default profile, or you want to override an existing default profile, enter a **Name** that will help you identify the profile when assigning it to security rules.
4. **Add** the log types to forward.
5. Click **OK**.

STEP 3 | Configure the Kubernetes plugin to push the tags to the specified device groups.

You must add a monitoring definition that includes the name of the Kubernetes cluster from which Panorama retrieves predefined labels and optionally a notify group.

A notify group is a list of device groups that receive tag updates. For the Kubernetes plugin, the notify group should include firewalls external to the cluster (meaning that they do not belong to the same device group as the Kubernetes cluster from which you are collecting attributes).

Because you specify the device group name in the YAML files that are used to deploy the CN-Series firewalls, the Kubernetes plugin automatically learns of all device groups that are internal to the cluster and it automatically pushes all predefined tags to those device groups by default.

The Kubernetes plugin uses Kubernetes Secrets to dynamically learn of the device groups within each cluster. Each time you deploy a CN-MGMT StatefulSet, the Secret is published to the Kubernetes API server and Panorama learns of it in the next monitoring interval.

1. Set up the Kubernetes plugin for monitoring the clusters.
2. Add notify groups.Add a notify group and select the device groups that receive the tags related to your Kubernetes cluster.
    1. Select **Panorama** > **Plugins** > **Kubernetes** > **Setup** > **Notify Groups**, and **Add**.
    2. Enter a **Name** of up to 31 characters for the notify group.
    3. Select **Enable sharing internal tags with Device Groups** if you want to share internal tags in addition to the external tags (default) created for the cluster.
    4. Select the device groups to which you want to register the tags.

        For the Notify Group that you select, Panorama only pushes the external tags.

        An external tag is any tag that is reachable from outside the cluster such as tags generated for an external service IP address and port for a cluster IP address, external IP address for all nodes and node ports, and external load balancer IP address and port or node port.

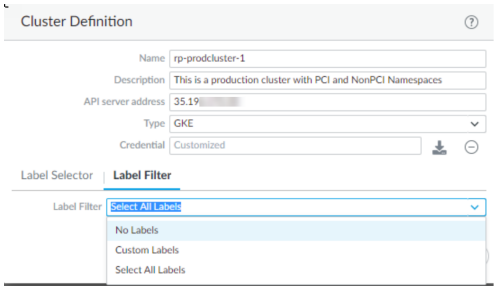
        The internal tags include details on the internal cluster IP addresses, pod IP address, nodes and node ports.

        By default, Panorama pushes all the tags it discovers (based on the Label Filters you select) to the Device Group associated with the cluster as defined in the YAML file that you used to deploy the CN-MGMT pods.
3. Add a Monitoring Definition for each cluster.
    1. Select **PanoramaPlugins** > **Kubernetes** > **Monitoring Definition** and **Add**.
    2. Enter a **Name** for the monitoring definition.
    3. Select the **Cluster** you want to monitor.
    4. (Optional) Select a **Notify Group** to which you want to send the IP-address-to-tag mapping information.

        By default the tags are shared with all CN-NFGW pods within the cluster.
    5. Click **OK** to save your changes.
4. **Commit** to Panorama.

STEP 4 | (Optional) Set up the Kubernetes plugin to retrieve user-defined labels from your application YAML files.

1. Select **PanoramaPlugins** > **Kubernetes** > **Setup** > **Cluster**, and select the cluster definition from the list.
2. Select the label filter from the following options:



1. **No Labels**—Does not create any tags for Kubernetes labels.
2. **Custom Labels**—Creates tags for only the labels you care about.

   To use custom labels, you must first annotate the YAML files inyour Kubernetes deployment, and then use any of the followingcombinations to generate custom tags for the corresponding IPaddresses:

   Specify the namespace, key, and value. Use * for all. The plugincreates tags when all three inputs are valid.

   Specify the namespace, and key, to create tags for allmatching keys within that namespace.

   Specify the namespace only to create a tag for every label withinthat namespace.
3. **Select All Labels**—Create tags for all Kubernetes labels including any custom labels.

3. Add a label selector expression.

   The label selector matches the specified label within the Kubernetes cluster and maps the IP addresses associated with the label to a single tag. For a list of supported prefixes see IP-Address-to-Tag Mapping of Kubernetes Attributes.

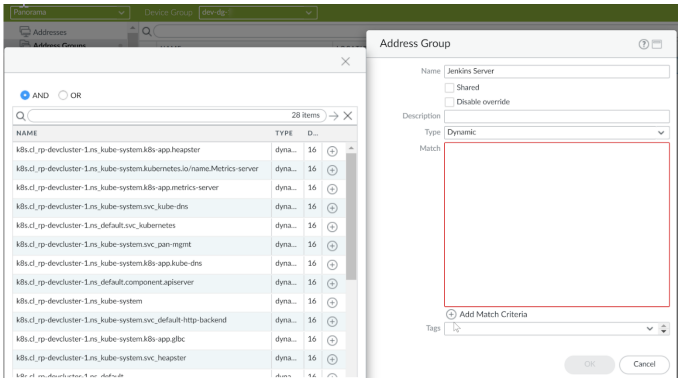
   For each label selector, Panorama generates a tag that becomes available as match criteria in dynamic address groups and enable you to enforce security policies:

   1. **Tag Prefix**—Phrase that each tag ends with to help you easily identify the tag. For example, the label selector k8s.cl_<clustername>.<selector-name>, matches on all ClusterIPs that match theselector, all Pod IPs that match the selector. These could be in all namespaces or specific one depending on what you configure.
   2. **Namespace**— * for all namespaces, or enter a value for the namespace.
   3. **Label Selector Filter**—The Kubernetes plugin supports set-based andequality-based selectors for label key and label value. The followingequality-based selectors are supported —key = value; key ==value; key != value, for example, app = redis.You can also specify multiple selectors in an expression as a commaseparated list, such as app == web, tier != backendThe following set-based selectors are supported—key in (value1,value2), key notin (value1, value2), key, !key, for example, tier notin (frontend, backend).
   4. **Apply On**—The type of resource to apply this on is Service, Pod, All.

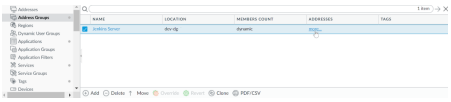
STEP 5 | Set up the Dynamic Address Groups.

1. Select your Device Group for managing the CN-NGFW pods.
2. Select **Objects** > **Address Groups**.
3. Click **Add** and enter a **Name** and a **Description** for the address group.
4. Select **Type** as **Dynamic**.

STEP 6 | Click **Add Match Criteria**, and select the **AND** or **OR** operator and select the attributes that you would like to filter for or match against.



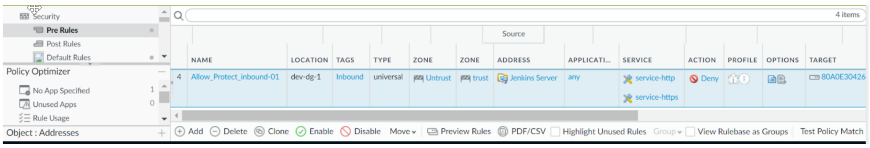
STEP 7 | Click **OK** and **Commit on Panorama**.



Use the **more...** link to view the IP addresses associated with the object, the Jenkins servers in your cluster in this example.

STEP 8 | Create Security Policy rules for traffic enforcement.

1. Select **Policies** > **Security**.
2. Click **Add** and enter a **Name** and a **Description** for the policy.
3. Add the **Source Zone** to specify the zone from which the traffic originates.
4. Add the **Destination Zone** at which the traffic is terminating.
5. For the **Destination Address**, select the Dynamic address group you just created.
6. Specify the action— **Deny**—for the traffic, and optionally attach the default security profiles to the rule.
7. Select **Actions** and select the **Log Forwarding** profile you created.
8. Click **Commit**.



You can also apply Security policy for east-west traffic within namespaces. If for example, you have two namespaces stage-ns and db-ns within a cluster named staging cluster where the frontend pods for a voting application are deployed in stage-NS, and the Redis backend pods are running in the DB-NS namespace. When you add this cluster to the Kubernetes plugin on Panorama for monitoring, it retrieves the label metadata to create tags. You can use these tags to enforce Security policy rules. To do this you need to

- Make sure that the Namespace or YAML files you use to deploy the frontend and backend applications are annotated with paloaltonetworks.com/firewall: pan-fw.
- Create the dynamic address group for the frontend and the backend pods.

  You must configure the dynamic address groups in the device group associated with the cluster, and select the tags for the frontend servers first. Then, repeat the process to create another dynamic address group for the backend servers.
- Add the Security policy rule to allow traffic for the Redis app from the frontend pods to backend pods.

The source is the dynamic address group of the frontend servers and the destination is the dynamic address group for the backend servers, and the action is allow .

Deploy the CN-Series on OpenShift

The pan-cni secures traffic on the default "eth0" interface of the application pod. If you have multi-homed pods, you can configure the CN-NGFW pod to secure additional interfaces that are configured with a bridge-based connection to communicate with other pods or the host. Depending on the annotation in the application YAML, you can configure the CN-Series firewall to inspect traffic from all the interfaces or a selected number of interfaces attached to each pod.

The pan-cni doesn't create any network and hence doesn't need IP addresses like other CNI plugins.

STEP 1 | Deploy your cluster.

Refer to the cloud platform vendor's documentation and verify that the OpenShift versions and CNI are supported for the CN-Series.

Review the following:

- Get the Images and Files for the CN-Series Deployment
- Editable Parameters in CN-Series Deployment YAML Files

STEP 2 | Use the workflow included in Secure Kubernetes Workloads with CN-Series.

You must create the service credentials, and deploy the firewall YAMLs.

Note: If your service credential file is over 10KB, you must gzip the file and then do a base64 encoding of the compressed file before you upload or paste the contents of the file into the Panorama CLI or API.

STEP 3 | Configure the PAN-CNI plugin to work with the Multus CNI plugin.

The Multus CNI on OpenShift functions as a "meta-plugin" that calls other CNI plugins. For each application you must:

1. Deploy the PAN-CNI NetworkAttachmentDefinition in every pod namespace

   `kubectl apply -f pan-cni-net-attach-def.yaml -n <target-namespace>`
2. Modify the Application YAML.

   After you deploy the pan-cni-net-attach-def.yaml, in the app pod yaml add the annotation:

   `paloaltonetworks.com/firewall: pan-fw`

   `k8s.v1.cni.cncf.io/networks: pan-cni`

   If you have other networks in the above annotation, add **`pan-cni`** after the networks that need to be inspected. The networks that follow **`pan-cni`** are not redirected and inspected.

   If your pod has multiple network interfaces, you must specify the interface names for which you want the CN-NGFW pod to inspect traffic, under "interfaces" in the pan-cni-configmap.yaml.

   For example:

```
template:
    metadata:
      annotations:
         paloaltonetworks.com/firewall: pan-fw
```

```
                k8s.v1.cni.cncf.io/networks: bridge-conf, macvlan-conf,
sriov-conf, pan-cni
```

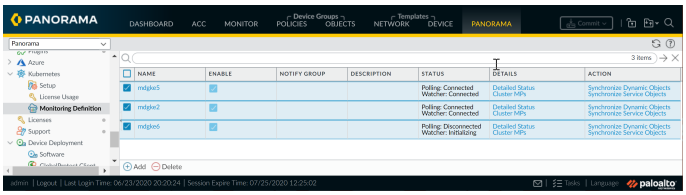
Uninstall the Kubernetes Plugin on Panorama

Use the following workflow to uninstall the Kubernetes plugin on Panorama so that you can successfully return all tokens to the Palo Alto Networks licensing servers, and then clear the authcode. This workflow enables you to ensure that the tokens are available for use on another Panorama.If you have deployed your Panorama management server in a high-availability configuration, you must complete the steps on the active-primary Panorama before you move to the passive-primary Panorama peer.

STEP 1 | Log in to your active-primary Panorama peer, if deployed in an HA configuration.

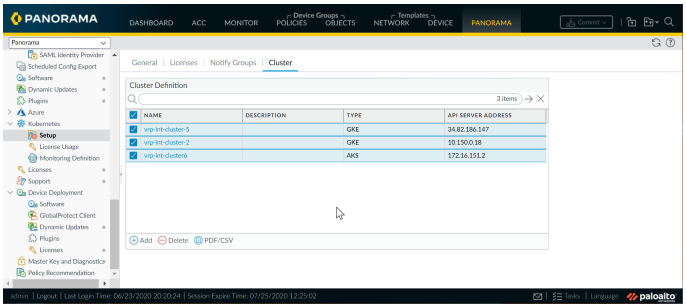
1. Remove all the cluster configuration from the plugin.
    1. Delete the monitoring definitions.

        Select **Plugins** > **Kubernetes** > **Monitoring Definition**, select the monitoring definitions and **Delete**.

    2. Delete the Kubernetes cluster definitions.

        Select **Plugins** > **Kubernetes** > **Set up** > **Cluster**, select the cluster definitions and **Delete**.

2. Commit your changes on Panorama.

    Commit > **Commit to Panorama**.
3. Verify that the used tokens count is zero.

    To confirm that all the tokens are returned back to the licensing server.
4. Perform a clear authcode and make sure the license column authcode is None.
5. Remove configuration and commit your changes.
    1. Select **Plugins** and find the Kubernetes plugin version you have installed, and **Remove Config**.
    2. **Commit** > **Commit to Panorama**.
6. Uninstall the Kubernetes plugin.
7. Suspend the active Panorama peer.

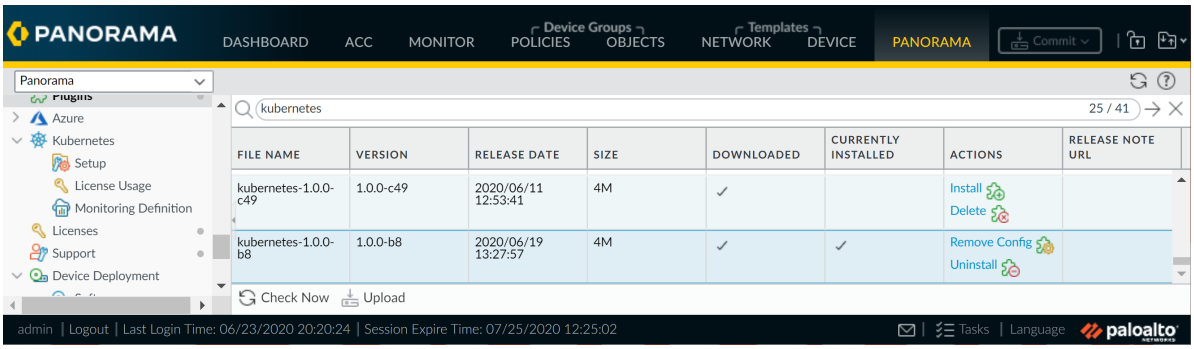
    Select **Panorama** > **High Availability**, and then click **Suspend local Panorama** link in the Operational Commands section.

STEP 2 | Log in to your other Panorama peer.

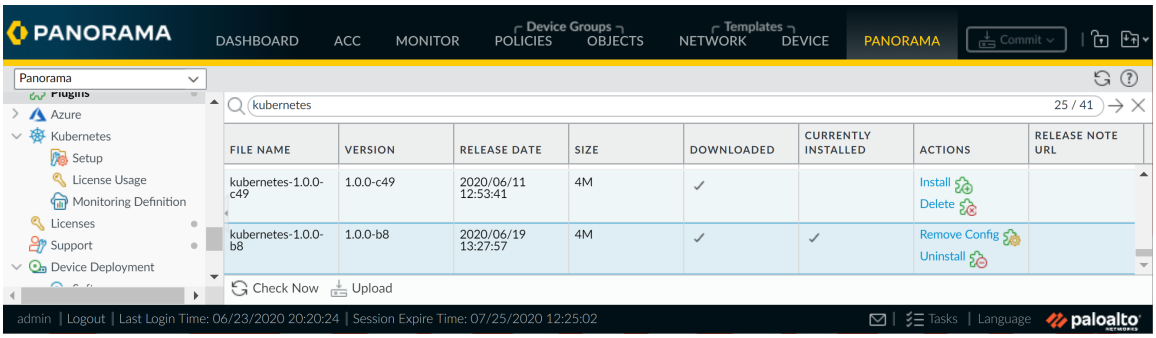
Now this peer is the active-secondary peer.

1.

2. Select **Plugins** and find the Kubernetes plugin version you have installed, and **Remove Config**.



3. Uninstall the plugin.
    1. Select **Plugins** and find the Kubernetes plugin version you have installed, and **Uninstall**.



    2. Verify that the uninstall was successful.

Clear the Auth Code for the CN-Series Firewalls on Panorama

Use the workaround listed below only if you removed the plugin configuration and committed your changes before clearing the authcode. This workaround enables you to release the toekns back to the licensing server so that you can use it on another Panorama appliance.

STEP 1 | 1. Add a new plugin user and commit your changes.

1. Select **Panorama** > **Administrators**.
2. **Add** a new user called **__kubernetes**.
3. **Commit** > **Commit to Panorama**.

STEP 2 | Clear the auth code on Panorama.

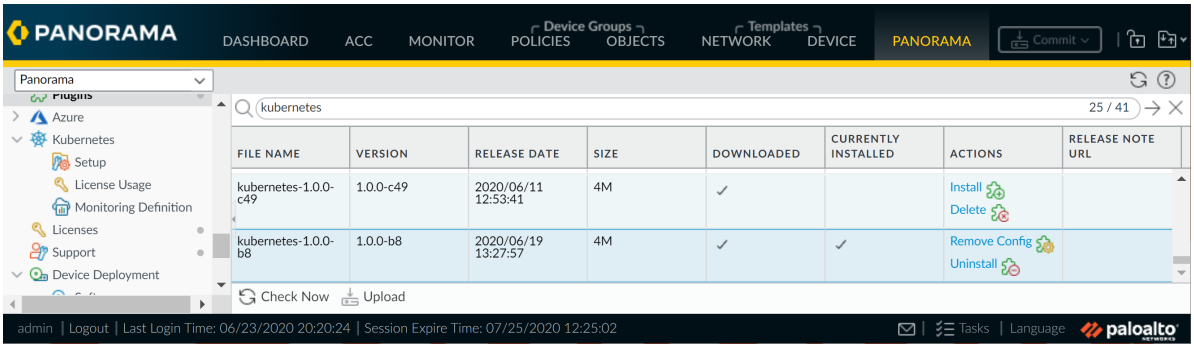
1. Select **Panorama** > **Plugins** > **Kubernetes** > **Setup** > **Licenses**.
2. Select **Activate/update using authorization code**, and **Clear Auth Code**.
3. Verify that the license column displays auth code **None**.

STEP 3 | Delete the plugin user **__kubernetes** you created in Step 1.

STEP 4 | Commit your changes.

STEP 5 | Uninstall the plugin.

1. Select **Plugins** and find the Kubernetes plugin version you have installed, and **Uninstall**.



2. Verify that the uninstall was successful.

Features Not Supported on the CN-Series

The following capabilities supported on PAN-OS are not available for the CN-Series:

- Authentication
- Interfaces other than virtual wires are not supported.
- IoT Security
- NAT
- Policy Based Forwarding
- QoS
- Tunnel Content Inspection
- User-ID
- WildFire Inline ML
- No Support for GlobalProtect and Software updates from the Device Deployment tab on Panorama.

  Only Plugin and Dynamic Updates for content release versions are supported.

CN-Series Supported Scale Factors

The scale numbers that the different components required to Secure Kubernetes Workloads with CN-Series are listed in the following sections:

- Scale Supported on the CN-Series Components
- Scale Supported on the Kubernetes Plugin on Panorama

Scale Supported on the CN-Series Components

Attribute	CN-Series Scale
Maximum CN-MGMT pairs per K8s cluster	4 CN-MGMT (or 8 CN-MGMT pod instances in a cluster)
Maximum CN-NGFW pods per CN-MGMT pair	30
Kubernetes pods secured by CN-NGFW (per K8s node)	30
Maximum Number of TCP/IP Sessions per CN-NGFW	20,000 sessions
Maximum Dynamic Address Groups IP addresses* per CN-MGMT pair	2500
Tags per IP address* per CN-MGMT pair	32

*See the Firewall comparison tool.

Scale Supported on the Kubernetes Plugin on Panorama

Attribute	Kubernetes Plugin Scale
Maximum Clusters on a K8s Panorama Plugin	16 (across all supported environments such as native K8s, AKS, EKS, GKE)
Maximum pods per cluster in Kubernetes plugin	900 (30*30)
Maximum Services per K8s cluster (Internal + External)	40
Maximum IP addresses (Pods + Services) across clusters per device group in the Kubernetes plugin	32*30 + 40 * 16 = 1560 (MP supports 2500)

Upgrade the CN-Series Firewall

The CN-MGMT pods (management plane) and the CN-NGFW pods (data plane) must always be on the same PAN-OS version. There are two ways to upgrade or downgrade your CN-Series firewall deployment. For either method, you must schedule the upgrade or downgrade during a planned maintenance window.

- **Upgrade the CN-Series Firewall—Redeploy**—Delete your existing CN-Series firewall deployment and replace the existing deployment completely. In this workflow, you must plan for a longer maintenance window because all the firewalls will be offline at the same time, and all the secured application traffic will be impacted until the firewalls pods are up again.
- **Upgrade the CN-Series Firewall—Rolling Update**—Use the new version to deploy a new CN-MGMT statefulset and service in the cluster, so you have two pairs of CN-MGMT pods running simultaneously while you perform a rolling upgrade of the CN-NGFW pods on each node. When the CN-NGFW pod is upgraded it registers to the new CN-MGMT pods and when the process completes for all secured nodes in your cluster, you can delete the old CN-MGMT statefulset and service in the cluster.

Both these methods create a new serial number for the CN-MGMT pods, and you must install the dynamic content updates for the subscriptions you have purchased. Review the Release Notes for the PAN-OS version to verify the minimum content version that is required and install it on the CN-MGMT pods.

Upgrade the CN-Series Firewall—Rolling Update

Use one of the following options to perform a rolling update, upgrade or downgrade to a supported PAN-OS version.

- Rolling Update where you upgrade the CN-MGMT StatefulSet and then upgrade the CN-NGFW pods.
- Rolling Update with Additional CN-MGMT StatefulSet

  When you use this option, review the Compare the Old and New PAN-CN-MGMT.yaml and make sure that you update the relevant section of the yaml file.

Rolling Update

This process enables you to first upgrade the CN-MGMT StatefulSet and then upgrade the CN-NGFW pods. The disruption to application traffic is minimal because the CN-NGFW pods are functioning during the CN-MGMT StatefulSet upgrade, and the rolling update for the CN-NGFW pods occurs one instance of the CN-NGFW pod at a time.

If you have a large Kubernetes cluster with a significant number of CN-NGFW pods and want a faster upgrade, you can schedule a maintenance window to delete the CN-NGFW yaml and upgrade all CN-NGFW pods at once.

During the CN-MGMT upgrade, logging is impacted. Additionally, both kubectl logs and System log messages are generated for temporary version mismatch and connection restarts between the CN-NGFW and the CN-MGMT pods.

STEP 1 | Upgrade the CN-MGMT StatefulSet.

1. Use one of the following options.
   - Option 1— Update the image name in the pan-cn-mgmt.yaml and apply the changes.

     ```
     containers:
      - name: pan-mgmt
      image:<your-private-registry-image-path-new-image>
     ```

     ```
     kubectl apply -f pan-cn-mgmt.yaml
     ```

   - Option 2—Use kubectl. When using kubectl, you are not updating the yaml files and therefore must keep track of the image used for the upgrade.

     ```
       kubectl -n kube-system set image sts/pan-mgmt-sts pan-mgmt=<your-
     private-registry-image-path-new-image>
     ```

2. Verify that the CN-MGMT StatefulSet is deployed.
   1. Use **`kubectl -n kube-system get sts/pan-mgmt-sts -o wide`**
   2. Check the status of the upgrade.

      `kubectl exec -it pan-mgmt-sts-0 -n kube-system -- su admin`

      `admin@pan-mgmt-sts-0> show jobs all`

      ```
      admin@pan-mgmt-sts-0.Basc-cluster-180> show jobs all
      ```

```
Enqueued              Dequeued           ID  PositionInQ
                Type                          Status Result Completed
-------------------------------------------------------------------------------
2020/08/25 14:11:11   14:11:11            2
             AutoCom                             FIN     OK 14:11:44
```

STEP 2 | Upgrade the CN-NGFW pods.

1. Use one of the following options.
   - Option1— Update the image name in the pan-cn-ngfw.yaml and apply the changes.

     ```
     containers:
      - name: pan-ngfw-container
      image:<your-private-registry-image-path-new-image>
     ```

     ```
     kubectl apply -f pan-cn-ngfw.yaml
     ```

   - Option 2—Use kubectl. When using kubectl, you are not updating the yaml files and therefore must keep track of the image used for the upgrade.

     ```
      kubectl -n kube-system set image ds/pan-ngfw-ds pan-ngfw-
     container=<your-private-registry-image-path-new-image>
     ```

2. Check the status of the upgrade.

   Use **`kubectl -n kube-system get ds/pan-ngfw-ds -o wide`**

STEP 3 | Required only if the images are updated for the PAN-OS version Update the init container and pan-cni images.

1. Modify the Init container image in the pan-cn-mgmt.yaml for the CN-MGMT firewall.

   ```
   initContainers:
   - name: pan-mgmt-init
   image:<your-private-registry-image-path>
   ```

2. Edit the image path for the PAN-CNI container image in the pan-cni.yaml.

   ```
   containers:
    name: install-pan-cni
   image:<your-private-registry-image-path>
   ```

Rolling Update with Additional CN-MGMT StatefulSet

STEP 1 | Before you begin.

1. Verify that the nodes in your cluster have the memory and CPU resources required for the additional CN-MGMT StatefulSet.
2. (Required for statically provisioned PVs only) Verify that you have PVs available for the additional CN-MGMT StatefulSet.

   The pan-cn-pv-local.yaml creates the directories required to deploy the CN-MGMT.

STEP 2 | Set up the new pan-cn-mgmt-configmap.yaml.

Edit the PAN_SERVICE_NAME: value to match what you added above in the new pan-cn-mgmt.yaml.

apiVersion: v1

kind: ConfigMap

metadata:

name: pan-ngfw-config

namespace: kube-system

data:

PAN_SERVICE_NAME: **pan-mgmt-svc2**

Retain the same values for the # Panorama settings and # Intended License Bundle type in the new file to reduce any updates on Panorama.

STEP 3 | Set up the new pan-cn-mgmt.yaml file.

There are multiple places you need to replace the service names, apps, and labels. See Compare the Old and New PAN-CN-MGMT.yaml.

STEP 4 | Required only if the images are updated for the PAN-OS version Update the Init container and pan-cni images

Image path for the Init container image in the pan-cn-mgmt.yaml for the CN-MGMT firewall

```
initContainers: - name: pan-mgmt-init image: <your-private-registry-image-path>
```

Image path for the PAN-CNI container image that has the CNI binaries and the CNI network config file on each node.

```
containers: name: install-pan-cni image: <your-private-registry-image-path>
```

STEP 5 | Apply the new CN-MGMT yaml files.

```
kubectl apply -f pan-cn-mgmt-configmap-new.yaml
```

```
kubectl apply -f pan-cn-mgmt-new.yaml
```

STEP 6 | Verify that the new CN-MGMT StatefulSet is deployed.

```
kubectl -n kube-system get sts -o wide
```

```
NAME READY AGE CONTAINERS IMAGES
```

```
pan-mgmt-sts 2/2 16h pan-mgmt 018147215560.dkr.ecr.ap-southeast-1.amazonaws.com/test/panos_ctnr/10.0.0/b/mp:63
```

```
pan-mgmt-sts-new 2/2 50m pan-mgmt-new 018147215560.dkr.ecr.ap-southeast-1.amazonaws.com/test/panos_ctnr/10.0.1/b/mp:64
```

STEP 7 | Edit the CN-NGFW pod yaml files with the new service name.

1. Update the pan-cn-ngfw-configmap.yaml.

   When you modify the PAN_SERVICE_NAME: value referenced in the pan-cn-ngfw-configmap.yaml to match the value you defined in the pan-cn-mgmt.yaml Service name, the pods that use the new image will connect to the new StatefulSet.

   apiVersion: v1

   kind: ConfigMap

   metadata:name: pan-ngfw-config

namespace: kube-system

data:

PAN_SERVICE_NAME: **pan-mgmt-svc2**

2. Deploy the pan-cn-ngfw-configmap.yaml

   kubectl apply -f pan-cn-ngfw-configmap.yaml

3. Edit the image path referenced in the pan-cn-ngfw.yaml.

   For example, you can use `kubectl set image ds/pan-cn-ngfw-ds -n kube-system pan-ngfw-container=018147215560.dkr.ecr.ap-southeast-1.amazonaws.com/test/panos_ctnr/10.0.2/b/dp:62`

4. Check the status of the rolling update.

   UP-TO-DATE column displays the number of replicas that have been updated successfully.

   kubectl get ds/pan-ngfw-ds -n kube-system -o wide

   ```
   NAME DESIRED CURRENT READY UP-TO-DATE AVAILABLE NODE SELECTOR AGE
   CONTAINERS IMAGES SELECTOR
   ```

   ```
   pan-ngfw 4 4 3 1 3 <none> 16h pan-ngfw-container 018147215560.dkr.ecr.ap-
   southeast-1.amazonaws.com/test/panos_ctnr/10.0.0/b/dp:22 app=pan-ngfw
   ```

STEP 8 | Verify that the CN-NGFW pods are deployed.

kubectl -n kube-system get pods -l app=pan-ngfw

```
NAME                READY STATUS RESTARTS AGE
pan-ngfw-ds-8b5gp 1/1 Running 0       40m
pan-ngfw-ds-h8xc6 1/1 Running 0        40m
pan-ngfw-ds-sn62b 1/1 Running 0       40m
pan-ngfw-ds-vxfqp 1/1 Running 0       40m
```

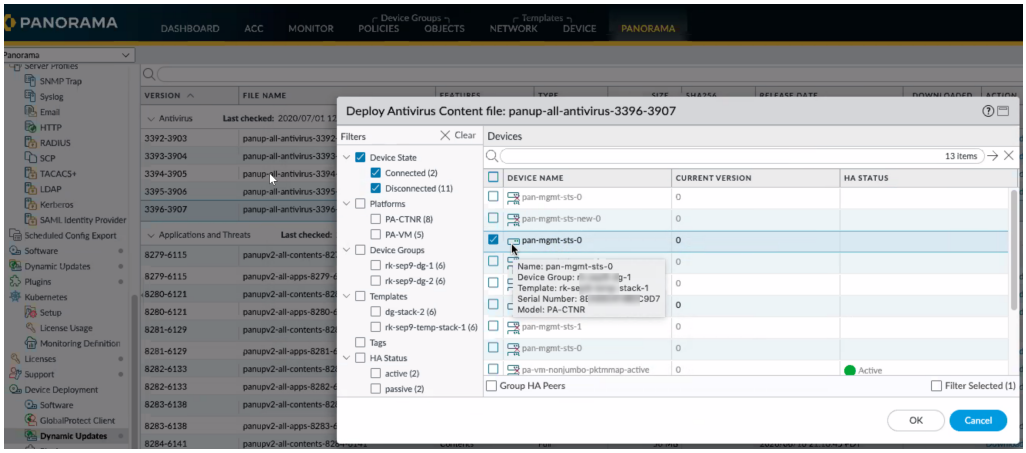
STEP 9 | Get the Serial Number for the CN-MGMT pods.

kubectl exec -it pan-mgmt-sts-0 -n kube-system -- su admin

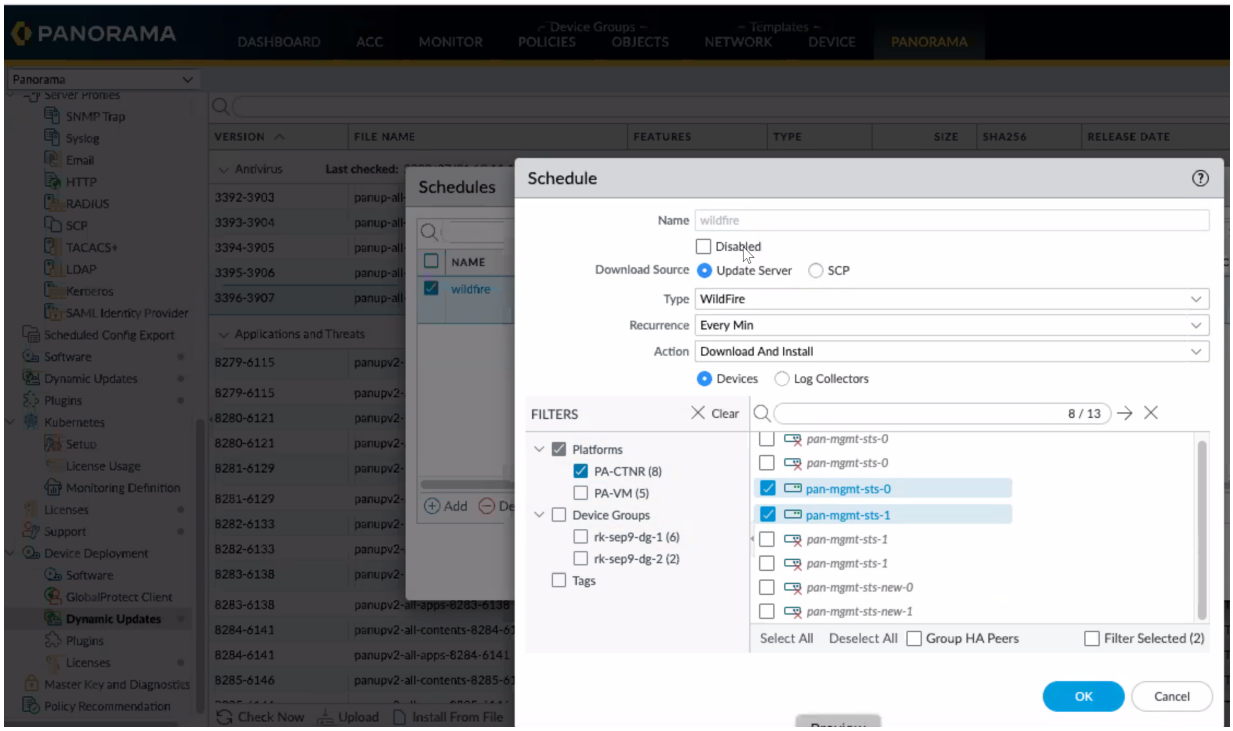
```
Warning: Your device is still configured with the default admin account
credentials. Please change your password prior to deployment.admin@pan-mgmt-
sts-0>
```

STEP 10 | Install the dynamic content updates for the subscriptions you have purchased.

You can either install it manually or set up a schedule. Verify the serial numbers of the CN-MGMT pods when selecting them for the dynamic updates.



or on a recurring schedule.



Compare the Old and New PAN-CN-MGMT.yaml

Review the different places where you need to update the service names, apps, and labels within the yaml file when you deploy a new CN-MGMT StatefulSet.

Old	New
apiVersion: v1 kind: Service **metadata:** **name: pan-mgmt-svc**	apiVersion: v1 kind: Service **metadata:** **name: pan-mgmt-svc2**
namespace: kube-system	namespace: kube-system

Old	New
labels: **app: pan-mgmt-svc**	**labels:** **app: pan-mgmt-svc2**
spec: ports: - protocol: UDP port: 4500 name: ipsec **selector:** **appname: pan-mgmt-sts**	spec: ports: - protocol: UDP port: 4500 name: ipsec **selector:** **appname: pan-mgmt-sts-new**
apiVersion: apps/v1 kind: StatefulSet **metadata:** **name: pan-mgmt-sts**	apiVersion: apps/v1 kind: StatefulSet **metadata:** **name: pan-mgmt-sts-new**
namespace: kube-system spec: selector: **matchLabels:** **appname: pan-mgmt-sts** **serviceName: pan-mgmt-svc** # Replicas are for fault-tolerance. Max 2 replicas supported. replicas: 2 updateStrategy: type: RollingUpdate podManagementPolicy: Parallel template: metadata: **labels:** app: pan-mgmt **appname: pan-mgmt-sts**	namespace: kube-system spec: selector: **matchLabels:** **appname: pan-mgmt-sts-new** **serviceName: pan-mgmt-svc2** # Replicas are for fault-tolerance. Max 2 replicas supported. replicas: 2 updateStrategy: type: RollingUpdate podManagementPolicy: Parallel template: metadata: **labels:** app: pan-mgmt **appname: pan-mgmt-sts-new**
labelSelector: matchExpressions:- key: "appname" operator: In	labelSelector: matchExpressions:- key: "appname" operator: In

Old	New
values: **pan-mgmt-sts**	**values:** **pan-mgmt-sts-new**
topologyKey: "kubernetes.io/hostname" initContainers: - name: pan-mgmt-init mountPath: /var/log/pan/ envFrom: **configMapRef:** **name: pan-mgmt-config**	topologyKey: "kubernetes.io/hostname" initContainers: - name: pan-mgmt-init mountPath: /var/log/pan/ envFrom: **configMapRef:** **name: pan-mgmt-new-config**
# sw-secret in pan-cn-ngfw.yaml and hard-coded in ipsec.conf value: pan-fw **containers:** name: pan-mgmt **image: 018147215560.dkr.ecr.ap-southeast-1.amazonaws.com/test/panos_ctnr/10.0.0/b/mp:63**	# sw-secret in pan-cn-ngfw.yaml and hard-coded in ipsec.conf value: pan-fw **containers:** name: pan-mgmt-new **image: 018147215560.dkr.ecr.ap-southeast-1.amazonaws.com/test/panos_ctnr/10.0.1/b/mp:64**
volumes name: dshm envFrom: **configMapRef:** **name: pan-mgmt-config**	volumes name: dshm envFrom: **configMapRef:** **name: pan-mgmt-new-config**

Upgrade the CN-Series Firewall—Redeploy

This option enables you to deploy the CN-Series firewalls afresh with an updated PAN-OS version (upgrade or downgrade to a supported PAN-OS version). This workflow is the simpler of the two options although it requires a little more downtime.

1. Delete the Existing CN-Series Firewall Deployment
2. Update the CN-Series Docker Images
3. Deploy the CN-Series Firewalls

Delete the Existing CN-Series Firewall Deployment

STEP 1 | Delete the existing CN-MGMT and CN-NGFW pods.

1. **`kubectl delete -f pan-cn-mgmt.yaml`**
2. **`kubectl delete -f pan-cn-ngfw.yaml`**

STEP 2 | Verify that the pods are deleted.

1. **`kubectl get pods -n kube-system -l app=pan-mgmt`**
2. **`kubectl get pods -n kube-system -l app=pan-ngfw`**

STEP 3 | Delete the existing persistent volume claims (PVCs) and persistent volumes (PVs)

1. Use **`kubectl -n kube-system get pvc -l appname=pan-mgmt-sts`** to find all the PVCs and PVs associated with the pan-cn-mgmt.yaml.

   `pan-mgmt-sts` is the default appname selector for the CN-MGMT pods. If you modified the yaml to specify a different name, you must replace the appname to match. The following is a sample output from EKS:

   ```
   NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE
   panconfig-pan-mgmt-sts-0 Bound pvc-<id> 8Gi RWO gp2 15h
   panconfig-pan-mgmt-sts-1 Bound pvc-<id> 8Gi RWO gp2 15h
   panlogs-pan-mgmt-sts-0 Bound pvc-<id> 20Gi RWO gp2 15h
   panlogs-pan-mgmt-sts-1 Bound pvc-<id> 20Gi RWO gp2 15h
   panplugincfg-pan-mgmt-sts-0 Bound pvc-<id> 1Gi RWO gp2 15
   panplugincfg-pan-mgmt-sts-1 Bound pvc-<id> 1Gi RWO gp2 15
   panplugins-pan-mgmt-sts-0 Bound pvc-<id> 1Gi RWO gp2 15h
   panplugins-pan-mgmt-sts-1 Bound pvc-<id> 1Gi RWO gp2 15h
   varcores-pan-mgmt-sts-0 Bound pvc-<id> 20Gi RWO gp2 15h
   varcores-pan-mgmt-sts-1 Bound pvc-<id> 20Gi RWO gp2 15h
   varlogpan-pan-mgmt-sts-0 Bound pvc-<id> 20Gi RWO gp2 15h
   varlogpan-pan-mgmt-sts-1 Bound pvc-<id> 20Gi RWO gp2 15h
   ```

   - *For statically provisioned PVs, to delete the PVs (typically used on-premises deployments) you must explicitly delete the pan-cn-pv-local.yaml file and the directories that contain data on each node which hosts the CN-MGMT pods.*

Use the command ***rm -rf /mnt/pan-local1/**** *for deleting the PVs for pan-local 1 through 6.*

- *For dynamically provisioned PVs, such as on the Managed Services/Cloud Platforms, when you delete the PVCs, the PVs are automatically deleted.*

Update the CN-Series Docker Images

STEP 1 | Upload the new images, for the version to which you want to upgrade, to the container registry.

See Get the Images and Files for the CN-Series Deployment.

STEP 2 | Update the image and image path on the CN-MGMT and CN-NGFW yaml files.

Image path for the CN-NGFW container image in the pan-cn-ngfw.yaml

```
containers: - name: pan-ngfw-container image: <your-private-registry-image-path>
```

Image path for the CN-MGMT container image in the pan-cn-mgmt.yaml

```
Image Path for the CN-MGMT image containers: - name: pan-mgmt image: <your-private-registry-image-path>
```

STEP 3 | Required only if the images are updated for the PAN-OS version Update the init container and pan-cni images.

Image path for the Init container image in the pan-cn-mgmt.yaml for the CN-MGMT firewall

```
initContainers: - name: pan-mgmt-init image: <your-private-registry-image-path>
```

Image path for the PAN-CNI container image in the pan-cni.yaml.

```
containers: name: install-pan-cni image: <your-private-registry-image-path>
```

Deploy the CN-Series Firewalls

For details on the YAML files and information on the set up, see Editable Parameters in CN-Series Deployment YAML Files and CN-Series Prerequisites.

The pan-cn-mgmt.yaml and pan-cn-ngfw.yaml are required to redeploy the CN-Series firewall, and you need to redeploy other yaml files only if you have changes. When deploying, begin with the pan-cni.yaml, pan-cn-mgmt.yaml and the last file you deploy is the pan-cn-ngfw.yaml.

STEP 1 | Deploy the yaml files.

1. Only required if you made changes, to these files:

```
kubectl apply -f pan-cni-configmap.yaml
kubectl apply -f pan-cn-mgmt-secret.yaml
kubectl apply -f pan-cn-mgmt-configmap.yaml
kubectl apply -f pan-cn-ngfw-configmap.yaml
kubectl apply -f plugin-serviceaccount.yaml
kubectl apply -f pan-mgmt-serviceaccount.yaml
kubectl apply -f pan-cni-serviceaccount.yaml
```

2. Only required if you have statically provisioned PVs:

   kubectl apply -f pan-cn-pv-local.yaml
3. Only required if you modified the pan-cni.yaml:

   `kubectl apply -f pan-cni.yaml`

   This command triggers a rolling update, and the pan-cni daemonset is updated on one node at a time.

   The cni takes 30-45 seconds to restart and become available on a node. During this restart, there is no impact to the applications and CN-NGFW pods that are running. Traffic from any new application pods that start on a node in this period are not be secured by the CN-NGFW pod.
4. **`kubectl apply -f pan-cn-mgmt.yaml`**
5. **`kubectl apply -f pan-cn-ngfw.yaml`**

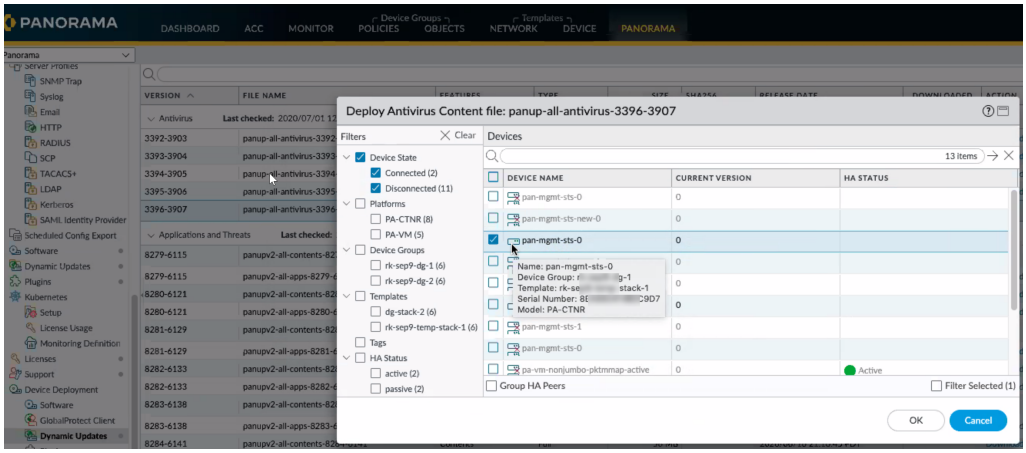
STEP 2 | Get the Serial Number for the CN-MGMT pods.

```
kubectl exec -it pan-mgmt-sts-0 -n kube-system -- su admin

Warning: Your device is still configured with the default admin account
credentials. Please change your password prior to deployment.admin@pan-mgmt-
sts-0>
```

STEP 3 | Install the dynamic content updates for the subscriptions you have purchased.

You can either install it manually or set up a schedule. Verify the serial numbers of the CN-MGMT pods when selecting them for the dynamic updates.



or on a recurring schedule.