Cisco Container Platform 1.5.0 User Guide

First Published: 2018-09-06

Last Modified: 2018-09-24

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

CHAPTER 1

Cisco Container Platform

Cisco Container Platform is a turnkey, production grade, extensible platform to deploy and manage multiple Kubernetes clusters. It runs on 100% upstream Kubernetes. Cisco Container Platform offers seamless container networking, enterprise-grade persistent storage, built-in production-grade security, integrated logging, monitoring and load balancing.

Cisco Container Platform provides authentication and authorization, security, high availability, networking, load balancing, and operational capabilities to effectively operate and manage Kubernetes clusters. Cisco Container Platform also provides a validated configuration of Kubernetes and can integrate with underlying infrastructure components such as Cisco HyperFlex and Cisco ACI. The infrastructure provider for Cisco Container Platform is Hyperflex.

Using the Cisco Container Platform web interface, you can create Kubernetes clusters on which you can deploy containerized applications. The clusters are created on the infrastructure provider platform.

The two user personas in Cisco Container Platform are as follows:

- The **Administrator** persona, which is associated with the **Administrator** role.
- The **User** persona, which is associated with the **User** role.

This chapter contains the following topics:

Administrator Workflow

The following table lists the workflow for Cisco Container Platform administrators.

Task	Related Section
Access the Cisco Container Platform web interface with *Administrator* credentials.	Accessing Cisco Container Platform Web Interface, on page 3
Set up the Cisco Container Platform infrastructure configuration.	Setting Up Cisco Container Platform, on page 3

Task	Related Section
Configure Cisco Smart Software Licensing for your Cisco Container Platform instance.	Configuring Cisco Smart Software Licensing, on page 4
Manage the Cisco Container Platform infrastructure configurations using which clusters are created.	Managing Cisco Container Platform Infrastructure Configuration, on page 9
Create Kubernetes clusters.	Creating Kubernetes Clusters, on page 15
Add users, assign appropriate roles, and associate the new users to the Kubernetes clusters that you have created.	Managing Users and RBAC, on page 17
Monitor Kubernetes clusters.	Monitoring Health of Cluster Deployments, on page 19 Viewing EFK Logs Using Kibana (Tenant Cluster), on page 20
Manage Kubernetes cluster using the Kubernetes Dashboard.	Managing Kubernetes Clusters, on page 23
Manage the lifecycle of Kubernetes clusters by scaling or upgrading the clusters.	Scaling Kubernetes Clusters, on page 16 Upgrading Kubernetes Clusters, on page 16

User Workflow

The following table lists the workflow for developers assigned with the *User* role.

Task	Related Section
Access the Cisco Container Platform web interface with user credentials.	Accessing Cisco Container Platform Web Interface, on page 3
Monitor Kubernetes clusters that are assigned to the user.	Monitoring Health of Cluster Deployments, on page 19 Viewing EFK Logs Using Kibana (Tenant Cluster), on page 20
Manage the assigned Kubernetes clusters using the Kubernetes Dashboard or CLI.	Managing Kubernetes Clusters, on page 23
Deploy applications on the assigned Kubernetes clusters.	Deploying Applications on Kubernetes Clusters, on page 43

Accessing Cisco Container Platform Web Interface

Before you begin

Ensure that you have configured the prerequisites for integrating ACI with Cisco Container Platform.

For more information, refer to the following documents:

- *ACI Integration Requirements* section of the *Cisco Container Platform Installation Guide*
- Planning and Prerequisites section of the Cisco ACI and Kubernetes Integration page

Ensure that you have powered on the installer VM on vCenter. The URL of the installer appears on the vCenter **Web console**.

Step 1 Obtain the URL to access the Cisco Container Platform web interface from the vCenter **Web console**.

Step 2 Access the URL using your web browser.

```
https://<Cisco Container Platform IP Address>:32443
```

Note We recommend that you use the Chrome, Safari, or Firefox browser to access the URL.

Step 3 Log in to the web interface as an *admin* user using the passphrase given during the Cisco Container Platform installation.

Setting Up Cisco Container Platform

Note This topic is applicable only for an ACI environment. In a non-ACI environment, the IP address range of the default VIP pool must be expanded to include the additional VIPs for tenant clusters. For more information, see Managing Networks, on page 11.

When you log in to Cisco Container Platform for the first time, you need to configure the Cisco Container Platform initial setup using the **Cisco Container Platform Setup** wizard.

Step 1 On the **Welcome** page, click **START THE SETUP**.

Step 2 In the **ACI Credentials** screen, specify information such as IP address, username, and passphrase of the APIC instance, click **CONNECT**, and then click **NEXT**.

Step 3 In the **ACI Configuration** screen, perform these steps:

a) In the **NAMESERVERS** field, enter the IP address of all the DNS servers that the ACI fabric can access.
b) From the **VMM DOMAIN** drop-down list, choose the Virtual Machine Manager Domain (VMMD) that you want to use.
c) In the **INFRASTRUCTURE VLAN ID** field, enter the VLAN number for layer 2 networking.
d) From the **VRF** drop-down list, choose the Virtual Routing and Forwarding (VRF) IP address.
e) From the **L3OUT POLICY NAME** drop-down list, choose the ACI object for allowing external internet connectivity.

f) From the **L3OUT NETWORK NAME** drop-down list, choose the external network that is reachable through the L3OUT object.
g) From the **AAEP NAME** drop-down list, choose an Attachable Access Entity Profile (AAEP) name to associate the VMM domain with an AAEP.
h) In the **STARTING SUBNET FOR PODS** field, enter the starting IP address for the IP pool that is used to allocate IP addresses to the pods.
i) In the **STARTING SUBNET FOR SERVICE** field, enter the starting IP address for the IP pool that is used to allocate IP addresses to the service VLAN.
j) In the **CONTROL PLANE CONTRACT NAME** field, enter the name of the contract that is provided by the Control Plane endpoint group to allow traffic from the Control Plane cluster to the tenant cluster.
k) In the **NODE VLAN START ID** field, enter the starting IP address for the IP pool that is used to allocate IP addresses to the node VLAN.
l) In the **NODE VLAN END ID** field, enter the ending IP address for the IP pool that is used to allocate IP addresses to the node VLAN.
m) In the **OPFLEX MULTICAST RANGE** field, enter a range for the Opflex multicast.
n) Click **CONNECT**.

Step 4 In the **Summary** screen, verify the configuration, and then click **FINISH**.
For more information on adding, modifying, or deleting an ACI profile, see Managing ACI Profile, on page 10.

Configuring Cisco Smart Software Licensing

You need to configure Cisco Smart Software Licensing to easily procure, deploy, and manage licenses for your Cisco Container Platform instance. The number of licenses required depends on the number of VMs necessary for your deployment scenario.

A Cisco Container Platform instance is available for a 90-day evaluation period after which, you need to register with Cisco Smart Software Manager (Cisco SSM). Cisco SSM enables you to manage your Cisco Smart Software Licenses from one centralized website. With Cisco SSM, you can organize and view your licenses in groups called virtual accounts. You can also use Cisco SSM to transfer the licenses between virtual accounts as needed.

You can access Cisco SSM from the Cisco Software Central homepage at software.cisco.com, under the **Smart Licensing** area.

If you do not want to manage licenses using Cisco SSM, either for policy reasons or network availability reasons, you can choose to install Cisco SSM satellite at your premises. Cisco Container Platform registers and reports license consumption to the Cisco SSM satellite as it does to Cisco SSM.

Note Ensure that you use Cisco SSM Satellite version 5.0 or later. For more information on installing and configuring Cisco SSM satellite, refer to http://www.cisco.com/go/smartsatellite.

License Usage and Compliance

Once you register Cisco Container Platform with Cisco SSM, you will receive the **Cisco Container Platform License with Support** license.

Cisco SSM or Cisco SSM satellite totals the license requirements for all your Cisco Container Platform instances and compares the total license usage to the number of licenses purchased, on a daily basis. After the data synchronization, your Cisco Container Platform instance displays one of the following status indicators:

- **Authorized**, when the number of licenses purchased is sufficient
- **Out of Compliance**, when the number of licenses is insufficient
- **Authorization Expired**, when the product has not communicated with Cisco SSM or Cisco SSM satellite for a period of 90 days.

Workflow of Cisco Smart Software Licensing

The following table describes the workflow of Cisco Smart Software Licensing.

Task	Related Section
Generate a product instance registration token in your virtual account	Generating Registration Token, on page 5
Configure the transport settings using which Cisco Container Platform connects to Cisco SSM or Cisco SSM satellite	Configuring Transport Settings, on page 6
Register the Cisco Container Platform instance with Cisco SSM or Cisco SSM satellite	Registering Cisco Container Platform License, on page 7
Manage licenses	Renewing Authorization, on page 7 Reregistering Cisco Container Platform License, on page 8 Deregistering Registration, on page 8

Generating Registration Token

You need to generate a registration token from Cisco SSM or Cisco SSM satellite to register the Cisco Container Platform instance.

Before you begin

Ensure that you have set up a Smart Account and a Virtual account on Cisco SSM or Cisco SSM satellite.

Step 1 Log in to your Smart Account on Cisco SSM or Cisco SSM satellite.

Step 2 Navigate to the Virtual account using which you want to register the Cisco Container Platform instance.

Step 3 If you want to enable higher levels of encryption for the products registered using the registration token, check the **Allow export-controlled functionality on the products registered with this token** check box.

Note This option is available only if you are compliant with the Export-Controlled functionality.

Step 4 Click **New Token** to generate a registration token.

Step 5 Copy and save the token for using it when you register your Cisco Container Platform instance.

For more information on registering your Cisco Container Platform instance, see Registering Cisco Container Platform License, on page 7.

Configuring Transport Settings

By default, Cisco Container Platform directly communicates with the Cisco SSM. You can modify the mode of communication by configuring the transport settings.

Before you begin

Ensure that you have obtained the registration token for the Cisco Container Platform instance.

Step 1 Log in to the Cisco Container Platform web interface.

Step 2 From the left pane, click **Licensing**.

If you are running Cisco Container Platform in the Evaluation mode, a license notification is displayed on the **Smart Software Licensing** pane.

Step 3 If a license notification is displayed, click the **edit the Smart Call Home Transport Settings** link.

Alternatively, click the **Licensing Status** tab, and then click the **View/Edit** link that appears under **Transport Settings**.

Step 4 In the **Transport Settings** dialog box, perform one of these steps:

- To configure Cisco Container Platform to send the license usage information to Cisco SSM using the Internet:
  1. Click the **DIRECT** radio button.
  2. Configure a DNS on Cisco Container Platform to resolve *tools.cisco.com*.

  This is the default setting.
- To configure Cisco Container Platform to send the license usage information to Cisco SSM using the Cisco SSM satellite:
  1. Click the **TRANSPORT GATEWAY** radio button.
  2. Enter the URL of the Cisco SSM satellite.
- To configure Cisco Container Platform to send the license usage information to Cisco SSM using a proxy server. For example, an off-the-shelf proxy, such as Cisco Transport Gateway or Apache:
  1. Click the **HTTP/HTTPS PROXY** radio button.
  2. Enter the IP address and port number of the proxy server.

Step 5 Click **SAVE**.

Registering Cisco Container Platform License

You need to register your Cisco Container Platform instance with Cisco SSM or Cisco SSM satellite before the 90-day evaluation period expires.

Before you begin

Ensure that you have configured the transport settings.

Step 1 Log in to the Cisco Container Platform web interface.

Step 2 From the left pane, click **Licensing**.

Step 3 In the license notification, click **Register**.
The **Smart Software Licensing Product Registration** dialog box appears.

Step 4 In the **Product Instance Registration Token** field, copy and paste the registration token that you generated using the Cisco SSM or Cisco SSM satellite.

For more information on generating a registration token, see Generating Registration Token, on page 5.

Step 5 Click **REGISTER** to complete the registration process.
Cisco Container Platform sends a request to Cisco SSM or Cisco SSM satellite to check the registration status and Cisco SSM or Cisco SSM satellite reports back the status to Cisco Container Platform, on a daily basis.

If registering the token fails, you can reregister the Cisco Container Platform instance using a new token.

For more information on reregistering Cisco Container Platform, see Reregistering Cisco Container Platform License, on page 8.

Renewing Authorization

By default, the authorization is automatically renewed every 30 days. However, Cisco Container Platform allows a user to manually initiate the authorization renew in case the automatic renewal process fails. The authorization expires if Cisco Container Platform is not connected to Cisco SSM or Cisco SSM satellite for 90 days and the licenses consumed by Cisco Container Platform are reclaimed and put back to the license pool.

Before you begin

Ensure that the Cisco Container Platform instance is registered with Cisco SSM or Cisco SSM satellite.

Step 1 Log in to the Cisco Container Platform web interface.

Step 2 From the left pane, click **Licensing**.

Step 3 From the **Actions** drop-down list, choose **Renew Authorization Now**.

Step 4 Click **OK** in the **Renew Authorization** dialog box.
Cisco Container Platform synchronizes with Cisco SSM or Cisco SSM satellite to check the license authorization status and Cisco SSM or Cisco SSM satellite reports back the status to Cisco Container Platform, on a daily basis.

Reregistering Cisco Container Platform License

You can reregister Cisco Container Platform with Cisco SSM or Cisco SSM satellite by deregistering it and registering it again, or by using a register force option.

Before you begin

Ensure that you have obtained a new registration token from Cisco SSM or Cisco SSM satellite.

Step 1 Log in to the Cisco Container Platform web interface.

Step 2 From the left pane, click **Licensing**.

Step 3 From the **Actions** drop-down list, choose **Reregister**.

Step 4 In the **Product Instance Registration Token** field of the **Smart Software Licensing Product Reregistration** dialog box, enter the registration token that you generated using Cisco SSM or Cisco SSM satellite.

For more information on generating a registration token, see Generating Registration Token, on page 5.

Step 5 Click **REGISTER** to complete the registration process.
Cisco Container Platform sends a request to Cisco SSM or Cisco SSM satellite to check the registration status and Cisco SSM or Cisco SSM satellite reports back the status to Cisco Container Platform, on a daily basis.

Deregistering Registration

You can deregister the Cisco Container Platform instance from Cisco SSM or Cisco SSM satellite to release all the licenses from the current Virtual account and the licenses are available for use by other products in the virtual account. Deregistering disconnects Cisco Container Platform from Cisco SSM or Cisco SSM satellite.

Before you begin

Ensure that the Cisco Container Platform instance is registered with Cisco SSM or Cisco SSM satellite.

Step 1 Log in to the Cisco Container Platform web interface.

Step 2 From the left pane, click **Licensing**.

Step 3 From the **Actions** drop-down list, choose **Deregister**.

Step 4 Click **DEREGISTER** in the confirmation dialog box.
Cisco Container Platform sends a request to Cisco SSM or Cisco SSM satellite to check the deregistration status and Cisco SSM or Cisco SSM satellite reports back the status to Cisco Container Platform, on a daily basis.

CHAPTER 2

Managing Cisco Container Platform Infrastructure Configuration

This chapter contains the following topics:

Managing Provider Profile

Cisco Container Platform enables you to define the provider profile on which clusters can be created.

You can configure multiple provider profiles in an instance of Cisco Container Platform and use the same provider profile for multiple clusters.

Adding Provider Profile

Before you begin

Cisco Container Platform interacts with vSphere through the user that you configure when you add a provider profile. Hence, you need to ensure that this user has the necessary privileges.

For more information on the vSphere user privileges, see User Privileges on vSphere, on page 49.

Step 1 From the left pane, click **Infrastructure Providers**.

Step 2 Click **NEW PROVIDER** and specify information such as name and description of provider, IP address, port, username and passphrase of the provider profile.

Step 3 Click **SUBMIT**.

Modifying Provider Profile

Step 1 From the left pane, click **Infrastructure Providers**.

Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Edit** corresponding to the provider profile that you want to modify.

Step 3 Change the provider details as necessary and click **SUBMIT**.

Deleting Provider Profile

Step 1 From the left pane, click **Infrastructure Providers**.

Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Delete** corresponding to the provider profile that you want to delete.

Step 3 Click **DELETE** in the confirmation dialog box.

Managing ACI Profile

Cisco Container Platform enables you to define ACI profiles using which tenant clusters can be created.

You can define multiple ACI profiles and use the same profile for multiple clusters.

Adding ACI Profile

Step 1 From the left pane, click **ACI Profiles**.

Step 2 Click **Add New ACI Profile** and perform these steps:

a) Specify information such as profile name, IP address, username, and passphrase of the ACI instance.

Note If there is more than one host, use a comma-separated host list in the **APIC IP ADDRESSES** field.

b) In the **NAMESERVERS** field, enter the IP address of all the DNS servers that the ACI fabric can access.

c) From the **VMM DOMAIN** drop-down list, choose the Virtual Machine Manager Domain (VMMD) that you want to use.

d) In the **INFRASTRUCTURE VLAN ID** field, enter the VLAN number for layer 2 networking.

e) From the **VRF** drop-down list, choose the Virtual Routing and Forwarding (VRF) IP address.

f) From the **L3OUT POLICY NAME** drop-down list, choose the ACI object for allowing external internet connectivity.

g) From the **L3OUT NETWORK NAME** drop-down list, choose the external network that is reachable through the L3OUT object.

h) From the **AAEP NAME** drop-down list, choose an Attachable Access Entity Profile (AAEP) name to associate the VMM domain with an AAEP.

i) In the **STARTING SUBNET FOR PODS** field, enter the starting IP address for the IP pool that is used to allocate IP addresses to the pods.

j) In the **STARTING SUBNET FOR SERVICE** field, enter the starting IP address for the IP pool that is used to allocate IP addresses to the service VLAN.
k) In the **CONTROL PLANE CONTRACT NAME** field, enter the name of the contract that is provided by the Control Plane endpoint group to allow traffic from the Control Plane cluster to the tenant cluster.
l) In the **NODE VLAN START ID** field, enter the starting IP address for the IP pool that is used to allocate IP addresses to the node VLAN.
m) In the **NODE VLAN END ID** field, enter the ending IP address for the IP pool that is used to allocate IP addresses to the node VLAN.
n) In the **OPFLEX MULTICAST RANGE** field, enter a range for the Opflex multicast.

Step 3 Click **SUBMIT**.

Modifying ACI Profile

Step 1 From the left pane, click **ACI Configuration**.
Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Edit** for the ACI profile that you want to modify.
Step 3 Change the ACI profile details as necessary and click **SUBMIT**.

Deleting ACI Profile

Step 1 From the left pane, click **ACI Configuration**.
Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Delete** for the ACI profile that you want to delete.
Step 3 Click **DELETE** in the confirmation dialog box.

Managing Networks

Note This section is applicable only for a non-ACI environment.

Cisco Container Platform enables you to select an existing network, create a subnet in that network, and then create a Cisco Container Platform Virtual IP Address (VIP) pool within that subnet.

VIP pools are reserved ranges of IP addresses that are assigned as virtual IP addresses within the Cisco Container Platform clusters. A minimum of two IP addresses are required for each tenant cluster, namely, one for the master VIP of the Kubernetes tenant cluster and an additional VIP for the external IP address of the Ingress controller. The range of IP addresses in the VIP pools must be outside of the IP addresses that are assigned by DHCP.

Modifying Networks

Step 1 From the left pane, click **Networks**.
The **Networks** page displays the default network.

Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Edit** for the network that you want to modify.

Alternatively, click the **SUBNETS** tab or the **POOLS** tab, and then click **EDIT** from the right pane to view the **Edit** dialog box.

Step 3 Modify the network name as necessary and click **SUBMIT**.

Adding Subnets

If you want to allocate VIP from a different subnet CIDR you need to add the subnet.

Step 1 From the left pane, click **Networks**, and then click the network to which you want to add a subnet.

Step 2 From the right pane, click **NEW SUBNET**.

Step 3 Enter a name and CIDR for the subnet.

Step 4 Click **SUBMIT**.

Modifying Subnets

Step 1 From the left pane, click **Networks**, and then click the network that contains the subnet you want to modify.

Step 2 Click the **SUBNETS** tab.

Step 3 From the drop-down list displayed under the **ACTIONS** column, choose **Edit** for the subnet that you want to modify.

Step 4 Modify the subnet name and CIDR as necessary, and then click **SUBMIT**.

Adding VIP Pool

Step 1 From the left pane, click **Networks**, and then click the network to which you want to add a VIP pool.

Step 2 From the right pane, click **NEW POOL**.

Step 3 Specify a name, subnet and IP address range for the VIP pool.

Step 4 Click **SUBMIT**.

Modifying VIP Pool

Step 1 From the left pane, click **Networks**, and then click the network that contains the VIP pool you want to modify.

Step 2 Click the **POOLS** tab.

Step 3 From the drop-down list displayed under the **ACTIONS** column, choose **Edit** for the VIP pool that you want to modify.

Step 4 Change the pool name and the IP address as necessary, and then click **SUBMIT**.

CHAPTER 3

Administering Kubernetes Clusters

You can create, modify, or delete Kubernetes clusters using the Cisco Container Platform web interface.

This chapter contains the following topics:

Creating Kubernetes Clusters

Step 1 From the left pane, click **Clusters**, and then click **NEW CLUSTER**.

Step 2 In the **Basic Information** screen, specify the following information, and then click **NEXT**:

- The infrastructure provider where the cluster needs to be created.

  For more information, see Adding Provider Profile, on page 9.

- The name, version of Kubernetes, and description to be used for creating the cluster.
- If you are using ACI, specify the ACI profile, see Adding ACI Profile, on page 10.

Step 3 In the **Provider Settings** screen, specify the data center, cluster, resource pool, network, HyperFlex local network, datastore, and VM template that you have configured on vSphere, and then click **NEXT**.

Note
- Ensure that DRS and HA are enabled on the cluster that you choose in this step. For more information on enabling DRS and HA on clusters, refer to the *Cisco Container Platform Installation Guide*.
- Ensure that the datastore that you choose in this step is accessible to the hosts in the cluster.

Step 4 In the **Node Configuration** screen, specify the following information, and then click **NEXT**:

- The number of worker and master nodes, and their VCPU and memory configurations.

- The SSH public key that you want to use for creating the cluster.
- The VM username that you want to use as the login for the VM.
- The subnet that you want to use for this cluster.
- The number of load balancer IP addresses for this cluster.

  For more information, see Load Balancer Services, on page 33.
- The IP addresses in CIDR notation that you want to use as the pod subnet.
- Whether or not you want to enable Istio
- A root CA certificate to allow tenant clusters to securely connect to additional services

Step 5 In the **Harbor Registry** screen, specify if you want to enable Harbor. If no, click **NEXT**. If yes, you must specify the following information, and then click **NEXT**:

a) Ensure the switch to enable Harbor is activated
b) A password for Harbor server admin
c) The immutable registry size in gigabits

Step 6 In the **Summary** screen, verify the configuration, and then click **FINISH**.

The cluster deployment takes few minutes to complete. The newly created cluster is displayed on the **Clusters** page.

For more information on deploying applications on clusters, see Deploying Applications on Kubernetes Clusters, on page 43.

Upgrading Kubernetes Clusters

Before you begin

Ensure that you have imported the latest tenant cluster OVA to the vSphere environment.

For more information on importing the tenant cluster OVA, refer to the *Cisco Container Platform Installation Guide*.

Step 1 From the left pane, click **Clusters**.

Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Upgrade**.

Step 3 In the **Upgrade Cluster** dialog box, choose a Kubernetes version and a new template for the VM, and then click **Submit**. It may take a few minutes for the Kuberenetes cluster upgrade to complete.

Scaling Kubernetes Clusters

You can scale clusters by adding or removing nodes to them based on the demands of the workloads you want to run.

Step 1 From the left pane, click **Clusters**.

Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Edit**, modify the number of worker nodes or load balancer IP addresses, and then click **UPDATE**.

Alternatively, follow these steps to scale the cluster:

a) Click the name of the cluster that you want to scale.
b) Click the **Nodes** tab.
c) From the right pane, click **EDIT**, modify the number of worker nodes or load balancer IP addresses, and then click **UPDATE**.

Deleting Kubernetes Clusters

Before you begin

Ensure that the cluster you want to delete is not currently in use, as deleting a cluster removes the containers and data associated with it.

Step 1 From the left pane, click **Clusters**.

Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Delete** for the cluster that you want to delete.

Step 3 Click **DELETE** in the confirmation dialog box.

Managing Users and RBAC

Cisco Container Platform provides Role-based Access Control (RBAC) through built-in static roles, namely the *Administrator* and *User* roles. Role-based access allows you to use local accounts and LDAP for authentication and authorization.

Configuring Local Users

Cisco Container Platform allows you to manage local users. An administrator can add a user, and assign an appropriate role and cluster(s) to the user.

Step 1 From the left pane, click **User Management**, and then click the **Users** tab.

Step 2 Click **NEW USER**.

Step 3 Specify information such as first name, last name, username, passphrase, and role for the user.

Step 4 Click **SUBMIT**.
The new user is displayed on the **User Management** page.

Note You can edit or delete a user by using the options available under the **ACTIONS** column.

Changing Login Passphrase

Step 1 From the left pane, click **User Management**, and then click the **Users** tab.

Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Edit** corresponding to your name.

Note Administrators can change passphrase and role for other users as well.

Step 3 Change the passphrase and role assigned as necessary, and click **SUBMIT**.

Configuring AD Servers

LDAP authentication is performed using a service account that can access the LDAP database and query for user accounts. You will need to configure the AD server and service account in Cisco Container Platform.

Step 1 From the left pane, click **User Management**, click the **Active Directory** tab, and then click **EDIT**.

Step 2 In the **SERVER IP ADDRESS** field, type the IP address of the AD server.

Step 3 In the **PORT** field, type the port number for the AD server.

Step 4 For improved security, we recommend that you check **STARTTLS**.

Step 5 In the **BASE DN** field, specify the domain name of the AD server for all the accounts that you have.

Step 6 In the **ACCOUNT USERNAME** field, specify the service account name that is used for accessing the LDAP server.

Step 7 In the **PASSPHRASE** field, type the passphrase of the AD account.

Step 8 Click **SUBMIT**.

Configuring AD Groups

Cisco Container Platform allows you to manage users using AD groups. An administrator can add users to AD groups, and then assign appropriate roles and clusters to the groups.

Before you begin

Ensure that you have configured the AD server that you want to use.

For more information on configuring AD servers, see Configuring AD Servers, on page 18.

Step 1 From the left pane, click **User Management**, and then click the **Groups** tab.

Step 2 Click **ADD GROUP**.

Step 3 Specify information such as the name of the AD group and the role you want to assign to the group.

Note If the AD group is associated with the *Administrator* role, by default, access is provided to all clusters. But, if the AD group is associated with the *User* role, you need to assign a cluster.

Step 4 From the **CLUSTERS** drop-down list, choose the names of the cluster that you want to assign to the AD group.

Step 5 Click **SUBMIT**.

Monitoring Health of Cluster Deployments

It is recommended to continuously monitor the health of your cluster deployment to improve the probability of early detection of failures and avoid any significant impact from a cluster failure.

Cisco Container Platform is deployed with Prometheus and Grafana configured to start monitoring and logging services automatically when a Kubernetes cluster is created.

Prometheus is an open-source systems monitoring and alerting toolkit and Grafana is an open source metric analytics and visualization suite.

Prometheus collects the data from the cluster deployment, and Grafana provides a general purpose dashboard for displaying the collected data. Grafana offers a highly customizable and user-friendly dashboard for monitoring purposes.

Note A user with *Administrator* role can view all the cluster deployments, but a user with *User* role can view only those clusters for which the user has permission to view.

Step 1 Access the Kubernetes cluster master node using ssh.

```
ssh -l <username> <IP address of master node>
```

Note Once you create a Kubernetes cluster, it may take a few minutes for the necessary services to start. If ssh to a cluster fails, we recommend that you try again after a few minutes.

Step 2 Obtain the password for Grafana, which is stored as a Kubernetes secret.

```
kubectl -n ccp get secrets ccp-addons-grafana -o yaml | grep admin-password | awk '{print $2}' |
base64 --decode
```

Note When you run the command on a Control Plane cluster, you can skip **-n ccp** in the command as these services run in the default namespace.

Step 3 Access the Grafana UI using a web browser.

```
https://<VIP>/grafana
```

Where <*VIP*> is the Virtual IP address of the control or tenant cluster as the case may be. In case of a tenant cluster, <*VIP*> is the Virtual IP address that is used by the cluster Ingress as described in Services and Networking, on page 27.

Step 4 Log in to the Grafana UI of your Kubernetes cluster using your username, and the password that you obtained in Step 2.

Note It is important to either change or retain the original login credentials since the secret that was used to initialize the Grafana login may be lost or changed with future upgrades.

Step 5 Add Prometheus as the data source and configure the Grafana dashboard to monitor the health of your cluster deployments.

Monitoring Logs from Cluster Deployments

The Elasticsearch, Fluentd, and Kibana (EFK) stack enables you to collect and monitor log data from containerized applications for troubleshooting or compliance purposes. These components are automatically installed when you install Cisco Container Platform.

Fluentd is an open source data collector. It works at the backend to collect and forward the log data to Elasticsearch.

Kibana is an open source analytics and visualization platform designed to work with Elasticsearch. It allows you to create rich visualizations and dashboards with the aggregated data.

Note A user with the *Administrator* role can view all logs, but a user with *User* role can view logs for only those clusters for which the user has permission to view.

Viewing EFK Logs Using Kibana (Tenant Cluster)

Before you begin

Ensure that you have installed the `kubectl` utility.

Step 1 Download the Kubeconfig file of the cluster whose logs you want to view, see Downloading Kubeconfig File, on page 43.

Step 2 Copy the contents of the downloaded Kubeconfig file to:

- Your local host `~/.kube/config`
- A local file and export KUBECONFIG=<*Downloaded Kubeconfig file*>

Step 3 Create a port-forward using kubectl to access Kibana from outside a cluster.

a) Determine the pod.

```
kubectl -n ccp get pods
```

For example, `kibana-logging-7db596d7f6-g9pxv`

b) Open a port-forward.

```
kubectl port-forward -n ccp
```

For example, `kibana-logging-7db596d7f6-g9pxv 5601:5601`

Step 4 Access the Kibana UI and view the data from the target tenant cluster using a web browser.

http://localhost:5601/app/kibana

For more information on customizing the Kibana UI, refer to the latest Kibana documentation.

Viewing EFK Logs Using Kibana (Control Plane Cluster)

Before you begin

Ensure that you have installed the `kubectl` utility.

Step 1 Access the Kubernetes cluster master node using ssh.

```
ssh ccpuser@control plane master node
sudo cat /etc/kubernetes/admin.conf
```

Step 2 Copy the contents of the downloaded Kubeconfig file to:

- Your local host `~/.kube/config`
- A local file and export KUBECONFIG=<*Full path of the Kubeconfig local file*>

For more information on setting Kubeconfig, see Configure Access to Multiple Clusters.

Step 3 Create a port-forward using kubectl to access Kibana from outside a cluster.

a) Determine the pod.

```
kubectl get pods
```

For example, `kibana-logging-7db596d7f6-g9pxv`

b) Open a port-forward.

```
kubectl port-forward
```

For example, `kibana-logging-7db596d7f6-g9pxv 5601:5601`

Step 4 Access the Kibana UI and view the data from the target tenant cluster using a web browser.

http://localhost:5601/app/kibana

For more information on customizing the Kibana UI, refer to the latest Kibana documentation.

Step 5

What to do next

Forwarding Logs to External Elasticsearch Server

Use the following Curl commands to configure forwarding of logs to an external Elasticsearch server:

Step 1 Open a terminal that has a curl client installed.

Step 2 Configure Cisco Container Platform login credentials.

```
export MGMT_HOST=https://<Cisco Container Platform IP address>:<Port>
export CCP_USER=<Username>
export CCP_PASSPHRASE=<Passphrase>
```

Step 3 Login to Cisco Container Platform and save the session cookie for future requests into the cookies.txt local file.

```
curl -k -j -c cookies.txt -X POST -H "Content-Type:application/x-www-form-urlencoded" -d
"username=$CCP_USER&password=$CCP_PASSWORD" $MGMT_HOST/2/system/login/
```

Step 4 Get the list of cluster names.

```
curl -s -k -b cookies.txt -H "Content-Type: application/json"
$MGMT_HOST/2/clusters/ | jq -r '.[].name'
```

Step 5 Set the CLUSTER_NAME environment variable to the cluster that you are working on.

```
export CLUSTER_NAME="<A cluster name from Step 2>"
```

Step 6 Configure the cluster UUID.

```
export CLUSTER_UUID=$(curl -s -k -b cookies.txt -H "Content-Type: application/json"
$MGMT_HOST/2/clusters/$CLUSTER_NAME | jq -r '.uuid')
```

Step 7 Configure the Elasticsearch server IP address and port number.

```
export EFK_SERVER=<IP address of Elasticsearch server>
export EFK_PORT=<Port number of Elasticsearch server>
```

Step 8 Install the helm chart to configure the custom Elasticsearch server.

```
curl -s -k -b cookies.txt -X POST --header 'Content-Type: application/json' --header 'Accept:
application/json' -d '{"chart_url": "/opt/ccp/charts/ccp-agent.tgz", "name": "ccpagent", "options":

"ccp-efk.localLogForwarding.enabled=False,ccp-efk.localLogForwarding.elasticsearchHost='$EFK_SERVER',ccp-efk.localLogForwarding.elasticsearchPort='EFK_PORT'"}'
 $MGMT_HOST/2/clusters/$CLUSTER_UUID/helmcharts
```

CHAPTER 4

Managing Kubernetes Clusters

The Cisco Container Platform web interface allows you to manage Kubernetes clusters by using the **Kubernetes Dashboard**. Once you set up the **Kubernetes Dashboard**, you can deploy applications on the authorized Kubernetes clusters, and manage the application and the cluster itself.

This chapter contains the following topic:

Setting up Kubernetes Dashboard

Step 1 From the left pane, click **Clusters**.

Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Download Token** to get the Kubernetes configuration (Kubeconfig) file of the cluster that you want to access using the Kubernetes Dashboard.

Step 3 Use the `Kubeconfig` file from Step 2 to login to the Kubernetes Dashboard.

Configuring Node Pools

Node pools allow the creation of worker nodes with varying configurations. Nodes belonging to a single node pool have identical characteristics.

Cisco Container Platform supports node pools of multiple types, such as vSphere and nodepool. A single Cisco Container Platform deployment can have only one type of node pool.

In the Cisco Container Platform vSphere implementation, a node pool has the following properties:

- vcpus
- memory
- template
- labels
- taints

Labels and taints are optional parameters. All nodes that belong to a nodepool are tagged with labels and they are tainted. Taints are key-value pairs, which are associated with an *effect*.

The following table describes the available *effects*.

Effect	Description
NoSchedule	Ensures that the pods that do not contain this taint are not scheduled on the node.
PreferNoSchedule	Ensures that Kubernetes avoids scheduling pods that do not contain this taint on the node.
NoExecute	Ensures that a pod is removed from the node if it is already running on the node, and is not scheduled on the node if it is not yet running on the node.

Each cluster has a unique node pool. During cluster creation, each cluster is assigned a default node pool.

Cisco Container Platform supports the ability for different master and worker configurations. To be able to support this, two new optional parameters, namely, *master_node_pool* and *worker_node_pool* are created as part of the cluster creation API. Upon cluster creation, the master node is created in the master-pool and the worker nodes are created in the default-pool.

Note You can not delete a master-pool when there is only one master node.

Customizing Master and Worker Configuration

A cluster with different node configuration for master and worker can be created by specifying the attributes *master_node_pool* and *worker_node_pool*. In this case, you cannot specify a cluster level template, memory, and vcpus.

This section contains the following examples on customizing the master and worker node configuration:

- Specifying Attributes of master_node_pool and worker_node_pool, on page 24
- Cluster Create Request Example, on page 25
- Adding Custom Node Pool to an Existing Cluster, on page 25
- Deleting Node Pool, on page 26
- Editing Node Pool, on page 26

Specifying Attributes of master_node_pool and worker_node_pool

Note Labels and taints are optional parameters and are valid for only a worker node pool.

```
"master_node_pool" : {
"vcpus": 4,
"memory": 8196,
      "template": "ccp-tenant-image-1.10.1-a39424c.ova",
     },
     "worker_node_pool" : {
      "vcpus": 2,
      "memory": 16384,
      "template": "ccp-tenant-image-1.10.1-a39424c.ova",
     "labels": "foo1=bar1,foo2=bar2"
     "taints": "key1=val1:NoSchedule,key2=val2:NoSchedule"
     }
}
```

Cluster Create Request Example

```
curl -k -X POST \
-H "X-Auth-Token:<AUTH_TOKEN>" \
-H "Content-Type: application/json" -d \
'{
   "deployer_type":"kubeadm",
   "datastore":"hx1-data",
   "name":"<NAME_OF_CLUSTER>",
   "networks":[
      "VLAN 1154 - 10.10.96.0 - 22"
   ],
   "workers":2,
   "master_vip":"10.10.96.143",
   "ssh_user":"<SSH_USERNAME>",
   "resource_pool":"hx1/Resources",
   "description":"vip",
   "deployer":{
      "provider":{
         "vsphere_datacenter":"Hyperflex",
         "vsphere_datastore":"hx1-data",
         "vsphere_client_config_uuid":"108b31fa-4b8c-4d9f-87f2-441b1b05921d",
         "vsphere_working_dir":"/Hyperflex/vm"
      },
      "provider_type":"vsphere"
   },
   "ssh_key":"<SSH_PUBLIC_KEY>",
   "datacenter":"Hyperflex",
   "cluster":"hx1",
   "masters":1,
   "type":1,
   "kubernetes_version":"1.10.1",
   "provider_client_config_uuid":"108b31fa-4b8c-4d9f-87f2-441b1b05921d",
   "master_node_pool" : {
    "vcpus": 4,
    "memory": 8196,
    "template": "ccp-tenant-image-1.10.1-a39424c.ova",
   },
   "worker_node_pool" : {
    "vcpus": 2,
    "memory": 16384,
    "template": "ccp-tenant-image-1.10.1-a39424c.ova",
    "labels": "foo1=bar1,foo2=bar2",
    "taints": "key1=val1:NoSchedule,key2=val2:NoSchedule"
    }
}' \
https://ccp-api/2/clusters
```

Adding Custom Node Pool to an Existing Cluster

```
curl -k -X POST \
-H "X-Auth-Token:<AUTH_TOKEN>" \
-H "Content-Type: application/json" -d \
'{
"name": "small-pool-1",
"vcpus": 1,
"memory": 8196,
"template": "ccp-tenant-image-1.10.1-a39424c.ova",
"size": 2,
"labels": "foo1=bar1,foo2=bar2",
"taints": "key1=val1:NoSchedule,key2=val2:NoSchedule"
}' \
https://ccp-api/2/clusters/fcd4a605-39a8-4e6e-9a0a-a1c7b1e7d9a7/nodepools
```

Deleting Node Pool

```
curl -k -X DELETE \
-H "X-Auth-Token:<AUTH_TOKEN>" \
https://ccp-api/2/clusters/fcd4a605-39a8-4e6e-9a0a-a1c7b1e7d9a7/nodepools/4
```

Editing Node Pool

```
curl -k -X PATCH \
-H "X-Auth-Token:<AUTH_TOKEN>" \
-H "Content-Type: application/json" -d \
'{
"size": 20,
"labels": "newFoo=newBar",
"taints": "newKey1=newVal1:NoSchedule"
}' \
https://ccp-api/2/clusters/fcd4a605-39a8-4e6e-9a0a-a1c7b1e7d9a7/nodepools/4
```

Suppose current NodePool is as follows:

```
"name": "small-pool-1",
"vcpus": 1,
"memory": 8196,
"template": "ccp-tenant-image-1.10.1-a39424c.ova",
"size": 2,
"labels": "foo1=bar1,foo2=bar2",
"taints": "key1=val1:NoSchedule,key2=val2:NoSchedule"
```

There are three cases in the **editNodePool** request:

- **Scale out operation:** When **sizerequested** is greater than current NodePool size.

  Scale in operation: When **sizeRequested** is less than current nodePool size.

  In this case, the requested number of nodes are deleted. If labels and taints are present in the request, they are applied to remaining nodes. Previous labels and taints are removed.

- When **sizeRequested** is equal to current nodePool size.

  No change occurs if labels and taints are not provided. If labels and taints are provided, they are applied to the nodes after removing the previous labels and taints.

CHAPTER 5

Services and Networking

This chapter contains the following topics:

Load Balancing Kubernetes Services using NGINX

Cisco Container Platform uses NGINX to offer advanced layer 7 load balancing solutions. NGINX can handle a large number of requests and at the same time, it can be run on Kubernetes containers.

The NGINX load balancer is automatically provisioned as part of Kubernetes cluster creation. Each Kubernetes cluster is provisioned with a single L7 NGINX load balancer. You can access the load balancer using its virtual IP address, which can be found by running the command `kubectl get svc -n ccp`.

To use the NGINX load balancer, you must create an Ingress resource. Ingress is a Kubernetes object that allows you to define HTTP load balancing rules to allow inbound connections to reach the cluster services. You can configure Ingress to create external URLs for services, load balance traffic, terminate SSL, offer name-based virtual hosting, and so on.

L7 Ingress

Cisco Container Platform supports the following types of L7 Ingresses:

- **Simple fanout**

  It enables you to access the website using http.

  Example

  ```
  cafe.test.com ->   10.1.1.1   ->   /tea      tea-svc:80
                                                           /coffee   coffee-svc:80
  ```

  For this type of Ingress, you need to create a yaml file that defines the Ingress rules.

  Sample yaml file

  ```
  apiVersion: extensions/v1beta1
  kind: Ingress
  metadata:
  name: cafe-ingress
  spec:
  ```

```
rules:
-host: cafe.test.com
http:
    paths:
    -path:/
    backend:
    serviceName: tea-svc
    servicePort: 80
    -path:/
    backend:
    serviceName: tea-svc
    servicePort: 80
```

- **Simple fanout with SSL termination**

  It enables you to access the website using https.

  Example

  ```
  https://cafe.test.com   ->   10.1.1.1   ->   /tea       tea-svc:80
                                               /coffee    coffee-svc:80
  ```

  For this type of Ingress, you need to create the following yaml files:

  - A yaml file that defines the Secret

    Sample yaml file

    ```
    apiVersion: v1
    kind: Secret
    metadata:
      name: cafe-secret
    type: Opaque
    data:
      tls.crt: base64 encoded cert
      tls.key: base64 encoded key
    ```

  - A yaml file that defines the Ingress rules

    Sample yaml file

    ```
    apiVersion: extensions/v1beta1
    kind: Ingress
    metadata:
      name: cafe-ingress
    spec:
      tls:
      -hosts:
      -cafe.test.com
       secretName: cafe-secret
      rules:
      -host: cafe.example.com
      http:
       paths:
       -path:/
       backend:
         serviceName: tea-svc
         sevicePort: 80
       -path:/
       backend:
         serviceName: coffee-svc
         servicePort: 80
    ```

- **Name based virtual hosting**

It enables you to access the website using multiple host names.

Example

```
 tea.test.com   --|          |-> tea.test.com      s1:80
                            |10.1.1.1  |
         coffee.test.com --|          |-> coffee.test.com  s2:80
```

For this type of Ingress, you need to create a yaml file that defines the Ingress rules.

Sample yaml file

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata
name: cafe-ingress
spec:
 rules:
 -host: tea.test.com
 http:
    paths:
    -path:/
    backend:
    serviceName: tea-svc
    servicePort: 80
-host: coffee.test.com
http:
paths:
-path:/
backend:
serviceName: coffee-svc
servicePort: 80
```

Note You can download the yaml files that are shown in this topic from the following link:

https://github.com/nginxinc/kubernetes-ingress/tree/master/examples/complete-example

For more information on a sample scenario of implementing Ingress, see Deploying Cafe Application with Ingress, on page 45.

L4 Ingress

NGINX supports L4 TCP and UDP Ingress load balancing. It uses the NGINX helm chart that contains the TCP or UDP service mappings, instead of the Ingress resources as in the case of L7 support.

Configuring L4 Load Balancing

Note NGINX supports either TCP or UDP L4 load balancing, but not both simultaneously.

Step 1 Access the Kubernetes cluster master node using ssh.

```
ssh -l <username> <IP address of master node>
```

Note Once you create a Kubernetes cluster, it may take a few minutes for the necessary services to start. If ssh to a cluster fails, we recommend that you try again after a few minutes.

Step 2 Get the current helm configuration values.

```
helm get values --all ccp-addons > /tmp/ccp_addons.yaml
```

Step 3 Navigate to the following location:

```
cd /tmp
```

Step 4 Edit the `ccp_addons.yaml` file.

You can search for *tcp* or *udp* in the `ccp_addons.yaml` file, and then add your L4 services.

The following example shows adding the tcp-test-svc TCP service that uses port 3333.

```
tcp:
    "9000": default/tcp-test-svc:3333
```

The following example shows adding the udp-test-svc UDP service that uses port 5005.

```
udp:
    "9001": default/udp-test-svc:5005
```

Step 5 Update the NGINX helm chart with the L4 service mappings.

```
helm upgrade --install ccp-addons /opt/ccp/charts/ccp-agent.tgz -f ccp_addons.yaml
```

Note You need to restart the NGINX Ingress controller pods for the new configuration to take effect.

Step 6 Verify that ingress has successfully mapped the port.

```
kubectl get svc -n ccp | grep ingress-controller
```

Ingress CA

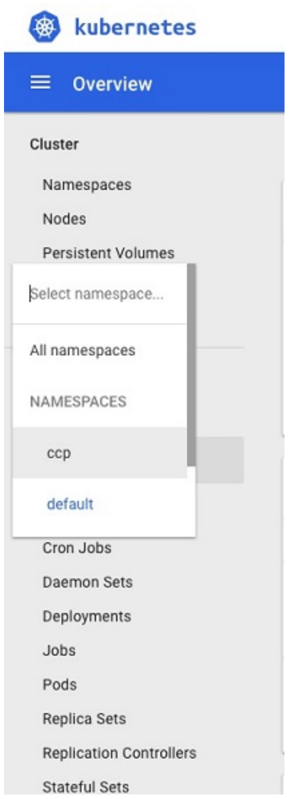
Cisco Container Platform by default creates an L7 Ingress service in order to support Monitoring Health of Cluster Deployments,Monitoring Logs from Cluster Deployments, and Setting up Kubernetes Dashboard. All of these services are exposed with TLS enabled, and the certificate authority (CA) that is used to sign the Ingress controller server certificate is self-signed and per cluster based.

In order to reach the services without triggering SSL warning, you can either add the CA as part of your application that needs to interact with services behind Cisco Container Platform ingress (preferred), or add the CA to your system trusted CA list. The following section describes how to obtain the CA certificate.

Step 1 Log in to the Kubernetes dashboard from browser as described in Setting up Kubernetes Dashboard section, download the kubeconfig file, and then use it to login to the Kubernetes dashboard.

Step 2 From the right pane, click the dropdown box under **Namespace**, click the **ccp** namespace.

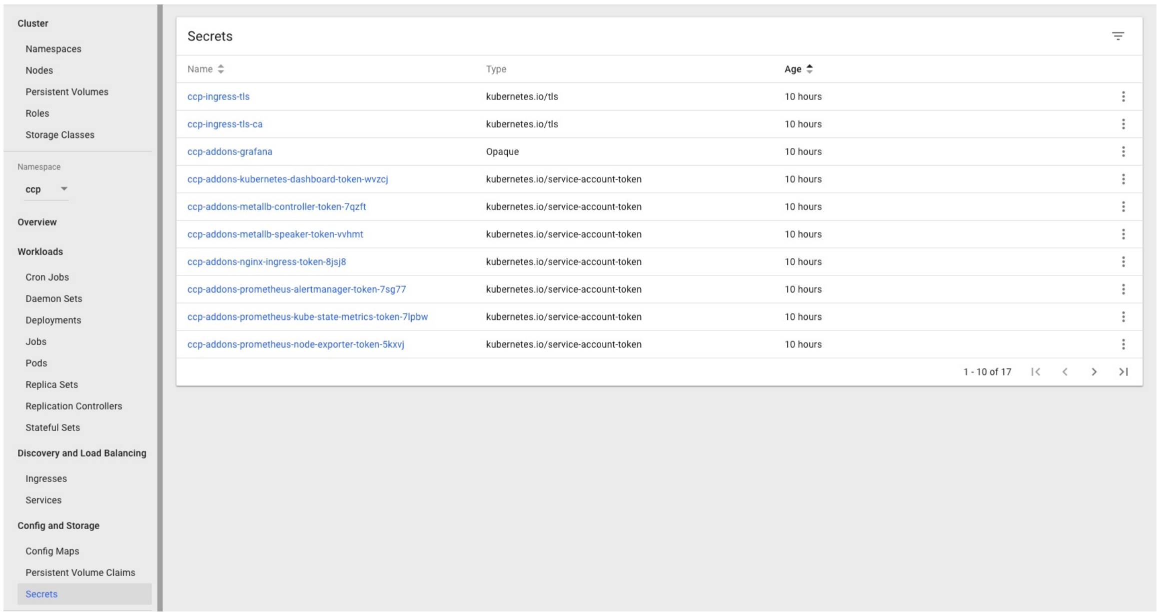
Figure 1: Kubernetes Dashboard



Step 3 Click the **Secrets** tab.

The **Secrets** pane appears.

Figure 2: Secrets Pane



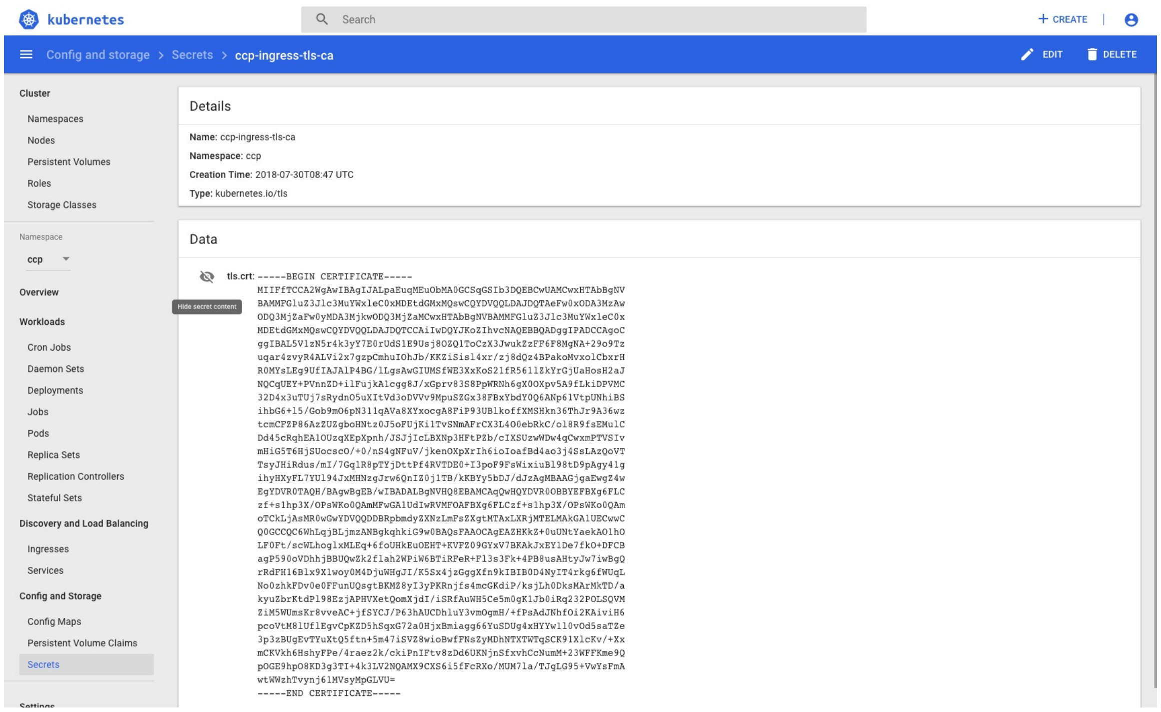
Secrets

Name	Type	Age
ccp-ingress-tls	kubernetes.io/tls	10 hours
ccp-ingress-tls-ca	kubernetes.io/tls	10 hours
ccp-addons-grafana	Opaque	10 hours
ccp-addons-kubernetes-dashboard-token-wvzcj	kubernetes.io/service-account-token	10 hours
ccp-addons-metallb-controller-token-7qzft	kubernetes.io/service-account-token	10 hours
ccp-addons-metallb-speaker-token-vvhmt	kubernetes.io/service-account-token	10 hours
ccp-addons-nginx-ingress-token-8jsj8	kubernetes.io/service-account-token	10 hours
ccp-addons-prometheus-alertmanager-token-7sg77	kubernetes.io/service-account-token	10 hours
ccp-addons-prometheus-kube-state-metrics-token-7lpbw	kubernetes.io/service-account-token	10 hours
ccp-addons-prometheus-node-exporter-token-5kxvj	kubernetes.io/service-account-token	10 hours

1 - 10 of 17

Step 4 Open the `ccp-ingress-tls-ca` secret and find the data for `tls.crt`.

Step 5 Click the **Eye** icon to view the details of a `tls.crt`.



Details

Name: ccp-ingress-tls-ca
Namespace: ccp
Creation Time: 2018-07-30T08:47 UTC
Type: kubernetes.io/tls

Data

```
tls.crt: -----BEGIN CERTIFICATE-----
MIIFfTCCA2WgAwIBAgIJALpaEuqMEuObMA0GCSqGSIb3DQEBCwUAMCwxHTAbBgNV
BAMMFGluZ3Jlc3MuYWxleC0xMDEtdGMxMQswCQYDVQQLDAJDQTAeFw0xODA3MzAw
ODQ3MjZaFw0yMDA3MjkwODQ3MjZaMCwxHTAbBgNVBAMMFGluZ3Jlc3MuYWxleC0x
MDEtdGMxMQswCQYDVQQLDAJDQTCCAiIwDQYJKoZIhvcNAQEBBQADggIPADCCAgoC
ggIBAL5V1zN5r4k3yY7E0rUdS1E9Usj8OZQ1ToCzX3JwukZzFF6F8MgNA+29o9Tz
uqar4zvyR4ALVi2x7gzpCmhuIOhJb/KKZiSisl4xr/zj8dQz4BPakoMvxolCbxrH
R0MYsLEg9UfIAJAlP4BG/lLgsAwGIUMSfWE3XxKoS21fR561lZkYrGjUaHosH2aJ
NQCqUEY+PVnnZD+ilFujkA1cgg8J/xGprv83S8PpWRNh6gX0OXpv5A9fLkiDPVMC
32D4x3uTUj7sRydnO5uXItVd3oDVVv9MpuSZGx38FBxYbdY0Q6ANp61VtpUNhiBS
ihbG6+l5/Gob9mO6pN311qAVa8XYxocgA8FiP93UBlkoffXMSHkn36ThJr9A36wz
tcmCFZP86AzZUZgboHNtz0J5oFUjKi1TvSNmAFrCX3L4O0ebRkC/ol8R9fsEMulC
Dd45cRqhEA1OUzqXEpXpnh/JSJjIcLBXNp3HFtPZb/cIXSUzwWDw4qCwxmPTVSIv
mHiG5T6HjSUocscO/+0/nS4gNFuV/jkenOXpXrIh6ioIoafBd4ao3j4SsLAzQoVT
TsyJHiRdus/mI/7Gq1R8pTYjDttPf4RVTDE0+I3poF9FsWixiuBl98tD9pAgy41g
ihyHXyFL7YUl94JxMHNzgJrw6QnIZ0j1TB/kKBYy5bDJ/dJzAgMBAAGjgaEwgZ4w
EgYDVR0TAQH/BAgwBgEB/wIBADALBgNVHQ8EBAMCAqQwHQYDVR0OBBYEFBXg6FLC
zf+s1hp3X/OPsWKo0QAmMFwGA1UdIwRVMFOAFBXg6FLCzf+s1hp3X/OPsWKo0QAm
oTCkLjAsMR0wGwYDVQQDDBRpbmdyZXNzLmFsZXgtMTAxLXRjMTELMAkGA1UECwwC
Q0GCCQC6WhLqjBLjmzANBgkqhkiG9w0BAQsFAAOCAgEAZHKkZ+0uUNtYaekAO1hO
LF0Ft/scWLhoglxMLEq+6foUHkEuOEHT+KVFZ09GYxV7BKAkJxEY1De7fkO+DFCB
agP590oVDhhjBBUQwZk2flah2WPiW6BTiRFeR+Fl3s3Fk+4PB8usAHtyJw7iwBgQ
rRdFH16Blx9X1woy0M4DjuWHgJI/K5Sx4jzGggXfn9kIBIB0D4NyIT4rkg6fWUqL
No0zhkFDv0e0FFunUQsgtBKMZ8yI3yPKRnjfs4mcGKdiP/ksjLh0DksMArMkTD/a
kyuZbrKtdPl98EzjAPHVXetQomXjdI/iSRfAuWH5Ce5m0gK1Jb0iRq232POLSQVM
ZiM5WUmsKr8vveAC+jfSYCJ/P63hAUCDhluY3vmOgmH/+fPsAdJNhfOi2KAiviH6
pcoVtM8lUflEgvCpKZD5hSqxG72a0HjxBmiagg66YuSDUg4xHYYw1l0vOd5saTZe
3p3zBUgEvTYuXtQ5ftn+5m47iSVZ8wioBwfFNsZyMDhNTXTWTqSCK91XlcKv/+Xx
mCKVkh6HshyFPe/4raez2k/ckiPnIFtv8zDd6UKNjnSfxvhCcNumM+23WFFKme9Q
pOGE9hpO8KD3g3TI+4k3LV2NQAMX9CXS6i5fFcRXo/MUM7la/TJgLG95+VwYsFmA
wtWWzhTvynj61MVsyMpGLVU=
-----END CERTIFICATE-----
```

Figure 3: Secrets Pane Showing Details of tls.crt

You can save the CA data into a file, and use it when a client is trying to connect to the Ingress service.

The following example uses **curl** to get to the dashboard using the saved CA certificate.

```
$ curl --cacert ./ca.crt -I https://10.10.99.185/dashboard
HTTP/1.1 200 OK
Server: nginx/1.13.12
Date: Mon, 30 Jul 2018 19:08:11 GMT
Content-Type: text/html; charset=utf-8
Connection: keep-alive
Vary: Accept-Encoding
Accept-Ranges: bytes
Cache-Control: no-store
Strict-Transport-Security: max-age=15724800; includeSubDomains
```

Network Policies

Cisco Container Platform supports Kubernetes NetworkPolicies. The NetworkPolicies are independent of the underlying container network plugin.

Load Balancer Services

Cisco Container Platform supports load balancer services on tenant clusters.

While creating a tenant cluster, you need to choose the number of load balancer IP addresses that you want to allocate for a tenant cluster from a VIP pool that you want to use.

Note The cluster creation operation fails if the number of requested load balancer IP addresses is more than the available IP addresses in the pool.

For more information, see Creating Kubernetes Clusters, on page 15.

Once load balancer IP addresses are allocated for a tenant cluster, externally reachable load balancer IP addresses are automatically provisioned for the load balancer services.

The following code provides an example of creating a service of type **LoadBalancer**.

```
apiVersion: v1
kind: Service
metadata:
     name: frontend
     labels:
             app: guestbook
             tier: frontend
     type: LoadBalancer
```

You can update the number of available load balancer IP addresses from the **Edit Cluster** screen. You need to be aware of the number of used addresses in order to update the number of allocated load balancer IP addresses.

For example:

Suppose the current tenant is allocated with five load balancer IP addresses. If there are three load balanced services running, you cannot reduce the number of load balancer IP addresses to three or less as there are services using those IP addresses already.

Note When you delete a tenant cluster, the allocated load balancer IP addresses are recycled to the VIP pool.

CHAPTER 6

Istio Service Mesh

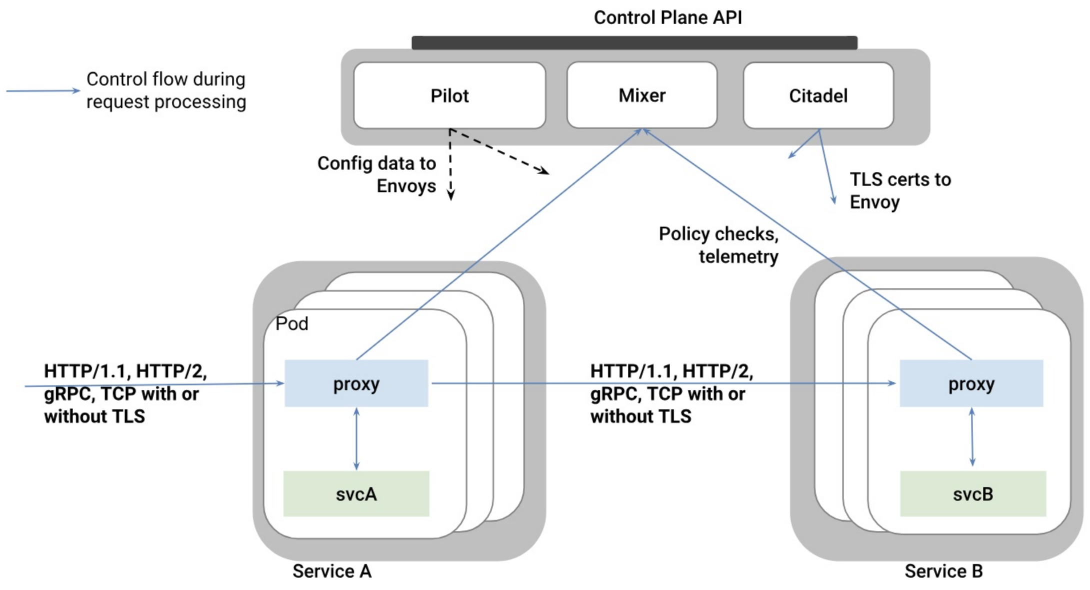
This chapter contains the following topics:

Introduction to Istio Service meshes

Cisco Container Platform includes support for Istio service meshes. An Istio service mesh is logically split into a Data Plane and a Control Plane. The Data Plane includes a set of intelligent proxies (Envoy) and the Control Plane provides a reliable Istio framework. The term Istio is sometimes also used as a synonym to refer to the entire service mesh stack that includes the Control Plane and the Data Plane components.

The service mesh technology allows you to construct North-South and East-West L4 and L7 application traffic meshes. It provides containerized applications a language-independent framework that removes several common tasks related to L4 and L7 application networking from the actual application code. The common tasks include L4 and L7 service routing and load balancing, support for polyglot environments in a language-independent manner and advanced telemetry. The service mesh technology enhances operational capabilities such as monitoring, security, load balancing and troubleshooting for the applications. You can deploy a service mesh in a multi-cloud topology allowing these functions to operate with applications that run across multiple independent cloud deployments.

The following figure shows the high-level architecture of an Istio service mesh.



Istio Architecture

In Cisco Container Platform, the components of Istio and Envoy are supported in the upstream Istio community. The Control and Data Plane components of the solution, such as Pilot, Mixer, Citadel and the Data Plane Envoy proxy for both North-South and East-West load balancing, are supported on Cisco Container Platform.

For more information on these technologies, refer to the upstream community documentation pages for Istio and Envoy.

Note Currently, the Istio service mesh feature is marked as a Tech Preview feature and uses the Istio community version v1.0. You need to contact your service representative for support on the version of Cisco Container Platform you have deployed.

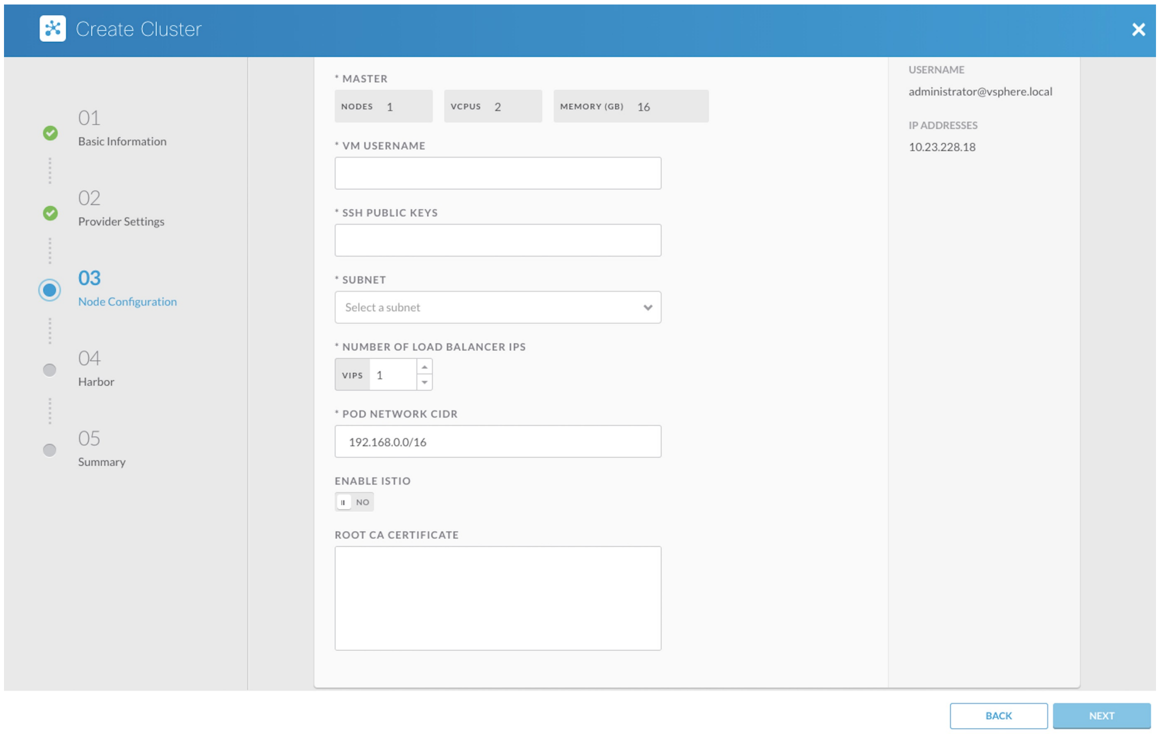
Configuring Istio Service meshes

An Istio service mesh is a configurable feature on Cisco Container Platform. You can configure a separate instance of the service mesh stack on each tenant cluster. Support for Istio must be configured at the time of creating a tenant Kubernetes cluster. You can perform this configuration using APIs or the Cisco Container Platform web interface.

Each instance of the Istio service mesh uses an IP address from the Virtual IP address pool that is associated with the tenant cluster. Consequently, you need to ensure that there is sufficient number of IP addresses free and available in the VIP pool before enabling Istio. Typically, at least three IP addresses are required, one each for the Kubernetes API, Kubernetes Ingress, and Istio Ingress gateway. This number may change in future when additional features require more virtual IP addresses.

For more information on the required number of virtual IP addresses for a given software version of Cisco Container Platform, refer to the Virtual IP address section.

The following figure shows the **Node Configuration** screen, using which you can enable the Istio service mesh on a tenant cluster of the Cisco Container Platform.



* MASTER

NODES 1 VCPUS 2 MEMORY (GB) 16

* VM USERNAME

* SSH PUBLIC KEYS

* SUBNET

Select a subnet

* NUMBER OF LOAD BALANCER IPS

VIPS 1

* POD NETWORK CIDR

192.168.0.0/16

ENABLE ISTIO

NO

ROOT CA CERTIFICATE

USERNAME

administrator@vsphere.local

IP ADDRESSES

10.23.228.18

In the current version of Cisco Container Platform, you can use a boolean flag to enable an Istio service mesh in a tenant Kubernetes cluster of Cisco Container Platform. If you enable the flag, a predetermined configuration of an Istio-based service mesh with Envoy as the Data Plane is configured in the tenant Kubernetes cluster. An internal instance of a service load balancer is automatically configured and a virtual IP address is automatically allocated for the Ingress gateway function of Istio.

Monitoring Service meshes

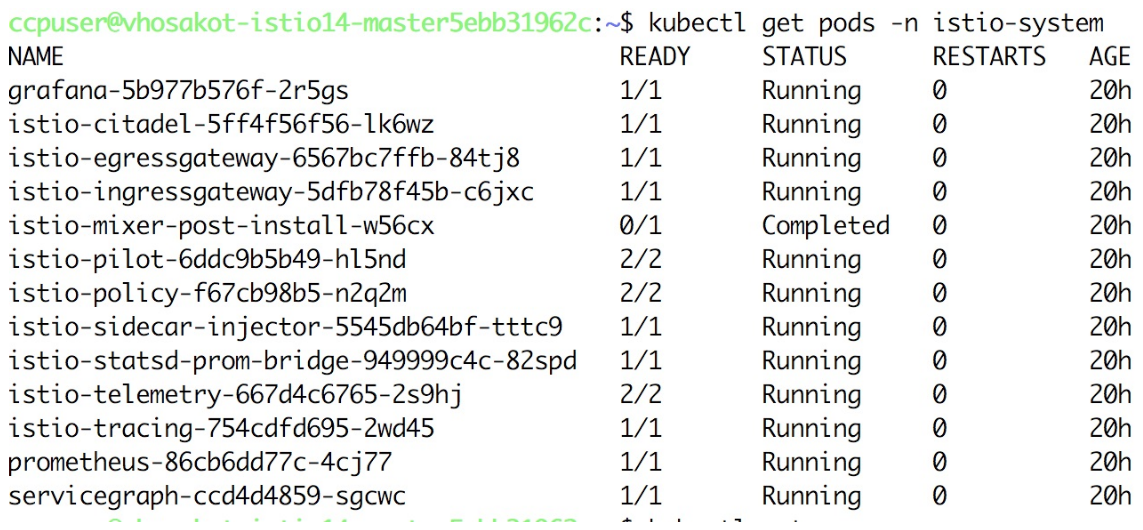
On Cisco Container Platform, the Istio Control Plane is deployed in a special **istio-system** namespace of a tenant Kubernetes cluster. This is similar to how other add-on services such as Prometheus based monitoring or NGINX based Kubernetes ingress are provided. In a production deployment, a tenant Kubernetes cluster administrator grants read-write access to your development namespaces but not to the namespaces of system add-on services such as Istio, thereby protecting the Control Plane of such services from getting over-written accidentally or maliciously by your application containers.

The following is a checklist of monitoring and troubleshooting steps when using Istio on Cisco Container Platform:

1. If Istio fails to be enabled on your tenant Kubernetes cluster, in addition to the usual troubleshooting steps for Cisco Container Platform, also ensure that there is a sufficient number of virtual IP addresses available in the pool configured for this Kubernetes tenant cluster. In the current version of Cisco Container Platform,

at least three IP addresses need to be free and available for a tenant Kubernetes cluster that has Istio enabled.

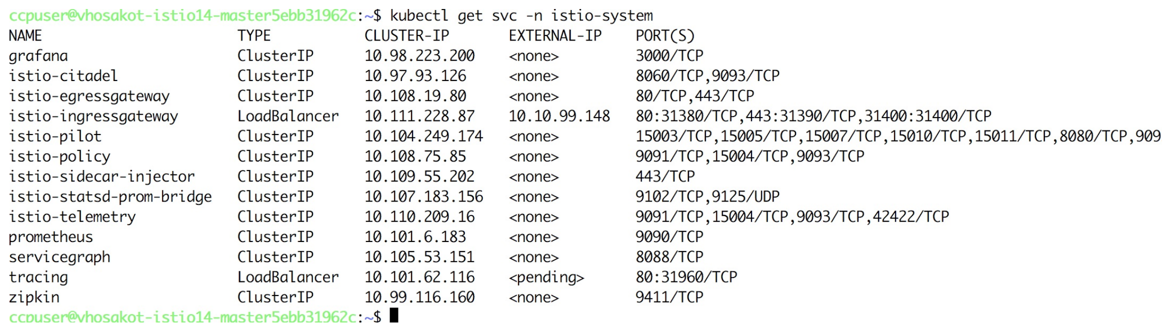
2. Confirm that all pods are running in the istio-system namespace of the tenant Kubernetes cluster. The following figure shows a sample CLI output indicating that all Istio control pods are running correctly in a tenant Kubernetes cluster. If one or more pods continuously fails to run, use **kubectl describe pod <name_of_pod>** to troubleshoot the issue.



```
ccpuser@vhosakot-istio14-master5ebb31962c:~$ kubectl get pods -n istio-system
NAME                                     READY     STATUS      RESTARTS   AGE
grafana-5b977b576f-2r5gs                 1/1       Running     0          20h
istio-citadel-5ff4f56f56-lk6wz           1/1       Running     0          20h
istio-egressgateway-6567bc7ffb-84tj8     1/1       Running     0          20h
istio-ingressgateway-5dfb78f45b-c6jxc    1/1       Running     0          20h
istio-mixer-post-install-w56cx           0/1       Completed   0          20h
istio-pilot-6ddc9b5b49-hl5nd             2/2       Running     0          20h
istio-policy-f67cb98b5-n2q2m             2/2       Running     0          20h
istio-sidecar-injector-5545db64bf-tttc9  1/1       Running     0          20h
istio-statsd-prom-bridge-949999c4c-82spd 1/1       Running     0          20h
istio-telemetry-667d4c6765-2s9hj         2/2       Running     0          20h
istio-tracing-754cdfd695-2wd45           1/1       Running     0          20h
prometheus-86cb6dd77c-4cj77              1/1       Running     0          20h
servicegraph-ccd4d4859-sgcwc             1/1       Running     0          20h
```

3. Confirm that all Istio services are running in the **istio-system** namespace of the tenant Kubernetes cluster.

   The following figure shows a CLI output with the Istio services up and running.



```
ccpuser@vhosakot-istio14-master5ebb31962c:~$ kubectl get svc -n istio-system
NAME                       TYPE           CLUSTER-IP       EXTERNAL-IP     PORT(S)
grafana                    ClusterIP      10.98.223.200    <none>          3000/TCP
istio-citadel              ClusterIP      10.97.93.126     <none>          8060/TCP,9093/TCP
istio-egressgateway        ClusterIP      10.108.19.80     <none>          80/TCP,443/TCP
istio-ingressgateway       LoadBalancer   10.111.228.87    10.10.99.148    80:31380/TCP,443:31390/TCP,31400:31400/TCP
istio-pilot                ClusterIP      10.104.249.174   <none>          15003/TCP,15005/TCP,15007/TCP,15010/TCP,15011/TCP,8080/TCP,909
istio-policy               ClusterIP      10.108.75.85     <none>          9091/TCP,15004/TCP,9093/TCP
istio-sidecar-injector     ClusterIP      10.109.55.202    <none>          443/TCP
istio-statsd-prom-bridge   ClusterIP      10.107.183.156   <none>          9102/TCP,9125/UDP
istio-telemetry            ClusterIP      10.110.209.16    <none>          9091/TCP,15004/TCP,9093/TCP,42422/TCP
prometheus                 ClusterIP      10.101.6.183     <none>          9090/TCP
servicegraph               ClusterIP      10.105.53.151    <none>          8088/TCP
tracing                    LoadBalancer   10.101.62.116    <pending>       80:31960/TCP
zipkin                     ClusterIP      10.99.116.160    <none>          9411/TCP
ccpuser@vhosakot-istio14-master5ebb31962c:~$
```

4. Confirm that the Ingress gateway service has an external IP address allocated and that this IP address is one of the previously available IP addresses in the virtual IP address pool associated with this tenant Kubernetes cluster. An example of this CLI output is shown in the preceding figure.

5. Deploy the bookinfo example application provided in the Istio upstream community web site.

6. The **istioctl** CLI utility is not deployed in the current version of the Cisco Container Platform. Most of the Istio functionality is now available through the **kubectl** CLI, but if you want to use **istioctl**, run these steps to deploy **istioctl** on a tenant Kubernetes cluster of the Cisco Container Platform:

```
export ISTIO_VERSION=1.0
    curl -L https://git.io/getLatestIstio | sh -
    chmod +x istio-${ISTIO_VERSION}/bin/istioctl
    sudo mv istio-${ISTIO_VERSION}/bin/istioctl /usr/local/bin/
    istioctl version
```

For more information and operational guidelines, refer to the Istio upstream documentation.

CHAPTER 7

Harbor Registry

Using a Harbor registry, you can host container images in a local, private Docker registry. Harbor is an extension of the basic Docker registry that implements access controls, identity management, and a graphical interface. Using imagePullSecrets, Kubernetes resources can connect to a Harbor Registry to retrieve container images on other systems.

This chapter contains the following topics:

Using Harbor Registry in Tenant Clusters

Follow these steps to create a new tenant cluster with access to the Harbor registry:

Step 1 Obtain the Ingress Root CA Certificate from the Kubernetes UI in one of the following ways:

- Use the steps in Ingress CA , on page 30.
- Run the following command on the tenant cluster where Harbor registry is installed.

```
$ kubectl get secrets -n ccp ccp-ingress-tls-ca -o jsonpath='{.data.tls.crt}' | base64 --decode
```

You can view the Harbor endpoint at `https://<LOAD_BALANCER_IP>:443` of the cluster where it is installed.

Step 2 Create a new tenant cluster.

For more information, see Creating Kubernetes Clusters, on page 15.

Step 3 In the **Node Configuration** screen, copy and paste the Root CA certificate obtained in Step 1.

Adding CA certificates to the Root CA is the only supported method of enabling secure registries in Cisco Container Platform tenant clusters.

Note Do not enable Harbor in the **Harbor Registry** screen.

Step 4 After tenant cluster creation, SSH to one of the VMs in the cluster and login to the Harbor registry with the password you provided during the installation of Harbor.

```
$ docker login -u admin -p *****
https://<LOAD_BALANCER_IP>:443
```

CHAPTER 8

Deploying Applications on Kubernetes Clusters

Once you have created Kubernetes cluster using the Cisco Container Platform web interface, you can deploy containerized applications on top of it.

This chapter contains the following topics:

Workflow of Deploying Applications

Task	Related Section
Create Kubernetes clusters using the Cisco Container Platform web interface.	Creating Kubernetes Clusters, on page 15
Download the kubeconfig file that contains the cluster information and the certificates required to access clusters.	Downloading Kubeconfig File, on page 43
Use the kubectl utility to deploy the application and test the scenario.	Sample Scenarios, on page 44

Downloading Kubeconfig File

You must download the cluster environment to access the Kubernetes clusters using command line tools such as `kubectl` or using APIs.

Step 1 From the left pane, click **Clusters**.

Step 2 Click the **Download** icon corresponding to the cluster environment that you want to download.

The `kubeconfig` file that contains the cluster information and the certificates required to access clusters is downloaded to your local system.

Sample Scenarios

This topic contains a few sample scenarios of deploying applications.

Deploying a Pod with Persistent Volume

This scenario describes deploying and configuring a pod with persistent volume.

Step 1 Go to the following URL:

https://github.com/kubernetes/examples/tree/master/staging/volumes/vsphere

Step 2 Download the following yaml files:

- vsphere-volume-sc-fast.yaml
- vsphere-volume-pvcsc.yaml
- vsphere-volume-pvcscpod.yaml

Step 3 Open the **kubectl** utility.

Step 4 Configure the Kubernetes cluster.

```
export KUBECONFIG=<Path to kubeconfig file>
```

Step 5 Create the storage class.

```
$ kubectl create -f vsphere-volume-sc-fast.yaml
```

Step 6 Verify if the storage cluster is created.

```
$ kubectl describe storageclass fast
```

```
Name: fast
IsDefaultClass: No
Annotations: <none>
Provisioner: kubernetes.io/vsphere-volume
Parameters: diskformat=zeroedthick,fstype=ext3
No events.```
```

Step 7 Create the persistent volume claim to request for storage.

```
$ kubectl create -f vsphere-volume-pvcsc.yaml
```

Step 8 Verify if the persistent volume claim (pvc) is created.

```
$ kubectl describe pvc pvcsc001
```

```
Name: pvcsc001
Namespace: default
StorageClass: fast
Status: Bound
Volume: pvc-83295256-f8e0-11e6-8263-005056b2349c
Labels: <none>
Capacity: 2Gi
Access Modes: RWO
Events:```
FirstSeen LastSeen Count From             SubObjectPath Type   Reason      Message
1m        1m       1     persistentvolume               Normal Provisioning Successfully provisioned
```

```
                                        -controller                          Succeeded    volume pvc-83295256-f8e0
                                                                                             -11e6-8263-005056b2349c
                                                                                               using
kubernetes.io/vsphere-volume
```

Persistent Volume is automatically created and is bounded to this pvc.

Step 9 Verify if the persistent volume claim is created:

```
$ kubectl describe pv pvc-83295256-f8e0-11e6-8263-005056b2349c
```

```
Name: pvc-83295256-f8e0-11e6-8263-005056b2349c
Labels: <none>
StorageClass: fast
Status: Bound
Claim: default/pvcsc001
Reclaim Policy: Delete
Access Modes: RWO
Capacity: 2Gi
Message:
Source:
Type: vSphereVolume (a Persistent Disk resource in vSphere)
VolumePath: [datastore1] kubevols/kubernetes-dynamic-pvc-83295256-f8e0-11e6-8263-005056b2349c.vmdk
FSType: ext3
No events.
```

Note VMDK is created inside the *kubevols* folder in the datastore, which is specified in the `vSphere cloudprovider config` file that is created during the setup of Kubernetes cluster on vSphere.

Step 10 Create a pod that uses persistent volume claim with storage class.

```
$ kubectl create -f vsphere-volume-pvcscpod.yaml
```

Step 11 Verify if the pod is up and running.

```
$ kubectl get pod pvpod
```

```
NAME     READY    STATUS     RESTARTS    AGE
pvpod    1/1      Running    0           48m
```

Step 12 While the pod is starting, access vCenter and view the dynamically provisioned VMDKs of the pod.

Deploying Cafe Application with Ingress

This scenario describes deploying and configuring the *Cafe application* with Ingress rules to manage incoming HTTP requests. It uses a **Simple fanout with SSL termination Ingress**.

For more information on Ingress, see Load Balancing Kubernetes Services using NGINX, on page 27.

Step 1 Go to the following URL:

https://github.com/nginxinc/kubernetes-ingress/tree/master/examples/complete-example

Step 2 Download the following yaml files:

- `tea-rc.yaml`
- `tea-svc.yaml`

- `coffee-rc.yaml`
- `coffee-svc.yaml`
- `cafe-secret.yaml`
- `cafe-ingress.yaml`

Step 3 Open the **kubectl** utility.

Step 4 Obtain the IP address of the L7 NGINX load balancer that Cisco Container Platform automatically installs:

```
$ kubectl get pods --all-namespaces -l app=ingress-nginx -o wide
```

```
NAMESPACE     NAME             READY  STATUS  RESTARTS AGE  IP            NODE
ingressnginx  nginx-            1/1   Running  0       3d   10.10.45.235  test-clusterwc5729f9ce2
              ingresscontroller
              -66974b775-jnmpl
```

Step 5 Deploy the Cafe application.

a) Create the coffee and the tea services and replication controllers:

```
kubectl create -f tea-rc.yaml<br>
kubectl create -f tea-svc.yaml<br>
kubectl create -f coffee-rc.yaml<br>
kubectl create -f coffee-svc.yaml
```

Step 6 Configure load balancing.

a) Create a Secret with an SSL certificate and a key:

```
kubectl create -f cafe-secret.yaml
```

b) Create an Ingress Resource:

```
kubectl create -f cafe-ingress.yaml
```

Step 7 Verify that the Cafe application is deployed.

```
$ kubectl get pods -o wide
```

```
NAMESPACE       READY STATUS     RESTARTS  AGE   IP                 NODE
coffee-rc-jb9sx 1/1   Running    0         3d    192.168.151.134   test-cluster-wb3d42afeff
coffee-rc-tjwgj 1/1   Running    0         3d    192.168.44.133    test-cluster-wc5729f9ce2
tea-rc-6qmvm    1/1   Running    0         3d    192.168.44.132    test-cluster-wc5729f9ce2
tea-rc-ms46j    1/1   Running    0         3d    192.168.151.132   test-cluster-wb3d42afeff
tea-rc-tnftv    1/1   Running    0         3d    192.168.151.133   test-cluster-wb3d42afeff
```

Step 8 Verify if the coffee and tea services are deployed.

```
$ kubectl get svc
```

```
NAME         TYPE        CLUSTER-IP    EXTERNAL-IP     PORT(S)     AGE
coffee-svc   ClusterIP   10.105.139.1  80/TCP          3d
kubernetes   ClusterIP   10.96.0.1     443/TCP         4d
tea-svc      ClusterIP   10.109.34.129 80/TCP          3d
```

Step 9 Verify if the Ingress is deployed.

```
$ kubectl describe ing
```

```
Name: cafe-ingress
Namespace: default
Address:
Default backend: default-http-backend:80 (<none>)
```

```
TLS: cafe-secret terminates cafe.example.com
Rules:

Host                Path       Backends
cafe.example.com
                    /tea       tea-svc:80 (<none>)
                    /coffee    coffee-svc:80 (<none>)

Annotations:
Events: <none>
```

Step 10 Test the application.

a) Access the load balancer IP address `10.10.45.235`, which is obtained in Step2.

b) Test if the Ingress controller is load balancing as expected.

```
$ curl --resolve cafe.example.com:443:10.10.45.235 https://cafe.example.com/coffee --insecure
<!DOCTYPE html>
...
<p><span>Server&nbsp;address:</span> <span>192.168.151.134:80</span></p>
...
$ curl --resolve cafe.example.com:443:10.10.45.235 https://cafe.example.com/coffee --insecure
<!DOCTYPE html>
...
<p><span>Server&nbsp;address:</span> <span>192.168.44.133:80</span></p>
...
```

APPENDIX A

User Privileges on vSphere

This appendix contains the following topic:

- User Privileges on vSphere, on page 49

User Privileges on vSphere

The following table provides the minimal set of privileges that are required by the vSphere user to execute the relevant operations in vCenter.

Roles	**Privileges**	**Entities**	**Propagate to Children**
manage-k8s-node-vms	Resource.AssignVMToPool System.Anonymous System.Read System.View VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.RemoveDisk VirtualMachine.Inventory.Create VirtualMachine.Inventory.Delete	Cluster, Hosts, VM Folder	Yes
manage-k8s-volumes	Datastore.AllocateSpace Datastore.FileManagement System.Anonymous System.Read System.View	Datastore	No

Roles	Privileges	Entities	Propagate to Children
k8s-system-read-and-spbmprofile-view	StorageProfile.View System.Anonymous System.Read System.View	vCenter	No
ReadOnly	System.Anonymous System.Read System.View	Datacenter, Datastore Cluster, Datastore Storage Folder	Yes
ccp-register-extension	Extension.Register Extension.Unregister Extension.Update	vCenter	No

For more information on adding a provider profile, see Adding Provider Profile, on page 9.