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.

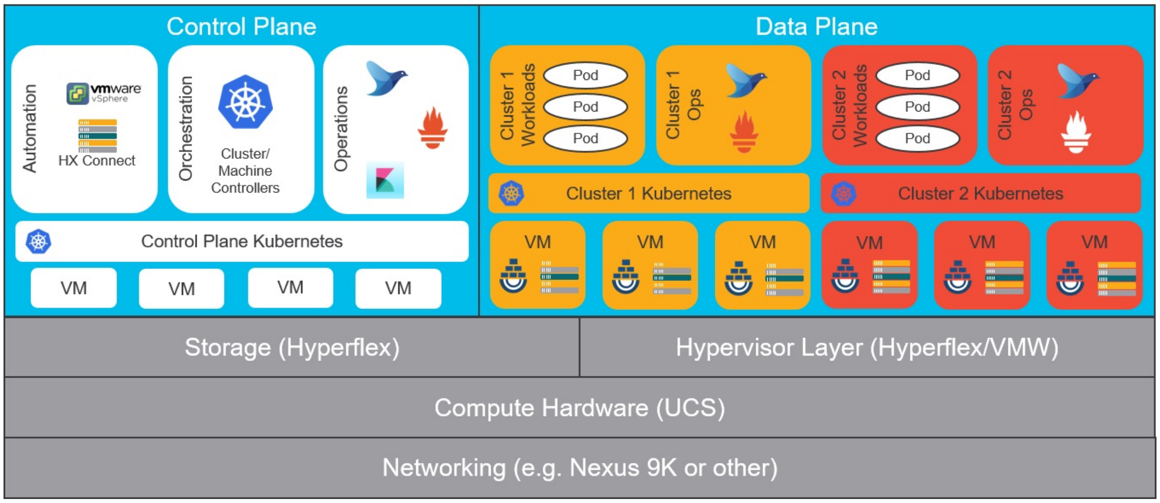
This chapter contains the following topics:

- Cisco Container Platform Architecture Overview, on page 1
- Sample Deployment Topology, on page 2
- Container Network Interface Plugins, on page 4
- System Requirements, on page 5
- ACI Integration Requirements, on page 10
- Getting Cisco Container Platform Software, on page 13

Cisco Container Platform Architecture Overview

The following figure shows the architecture of Cisco Container Platform deployment with HyperFlex and ACI integration.

Figure 1: Cisco Container Platform Architecture Overview



Note Cisco Container Platform can run on top of an ACI networking fabric as well as on a non-ACI networking fabric that performs standard L3 switching.

At the bottom of the stack, there is an ACI fabric that consists of Nexus switches, Application Policy Infrastructure Controllers (APICs) and Fabric Interconnects (FIs). The next layer up is the UCS servers running the HyperFlex software. HyperFlex provides virtualized compute resources through VMware, and distributed storage resources through the HyperFlex converged data platform.

The next layer up is the Cisco Container Platform Control Plane and Data Plane. In the preceeding figure, Cisco Container Platform Control Plane runs on the four VMs on the left.

Kubernetes tenant clusters are preconfigured to support Persistent Volumes using vSphere Cloud Provider and FlexVolumes using HyperFlex volume plugin. Both implementations use the underlying replicated, highly available HyperFlex data platform for storage.

Components of Cisco Container Platform

The following table describes the components of Cisco Container Platform.

Function	Component
Container Runtime	Docker CE
Operating System	Ubuntu
Orchestration	Kubernetes
IaaS	vSphere
Infrastructure	HyperFlex
Container Network Interface (CNI)	ACI, Contiv, Calico
SDN	ACI
Container Storage	HyperFlex Flex Driver
Load Balancing	NGINX, Envoy
Service Mesh	Istio, Envoy
Monitoring	Prometheus, Grafana
Logging	Elasticsearch, Fluentd, and Kibana (EFK) stack

Sample Deployment Topology

This section describes a sample deployment topology of the Cisco Container Platform and illustrates the network topology requirements at a conceptual level. Future sections of the document such as System

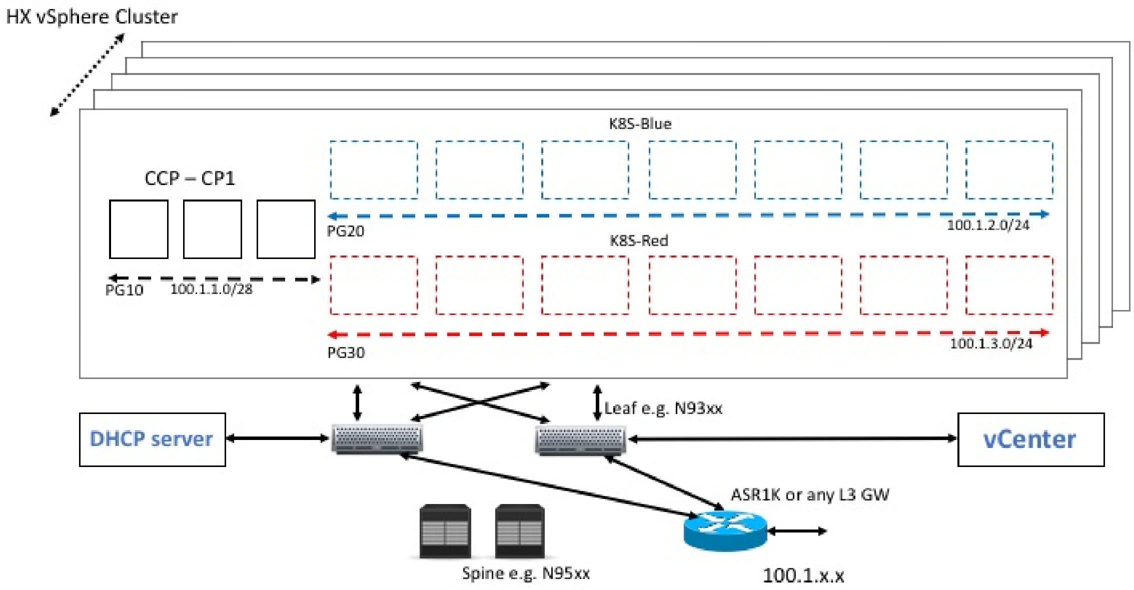
Requirements, on page 5 and Installing Cisco Container Platform on vSphere Web Client provide additional configuration details based on these concepts.

Note In this example, the deployment target is a VMware vSphere virtualization platform, and Cisco Container Platform is using a non-ACI CNI such as Calico or Contiv. Other deployment environments are conceptually similar but with some slight differences appropriate to those environments.

In this case, it is expected that the vSphere based cluster is set up, provisioned and fully functional for virtualization and Virtual Machine functionality before any installation of Cisco Container Platform. You can refer to the standard VMware documentation for details on vSphere installation.

The following figure illustrates an example vSphere cluster on which Cisco Container Platform is to be deployed.

Figure 2: Example vSphere Cluster



Once the vSphere cluster is ready to provision VMs, the admin then provisions one or more VMWare port groups (for example PG10, PG20 and PG30 in the figure) on which virtual machines will subsequently be provisioned as container cluster nodes. Basic L2 switching using VMWare vswitch functionality can be used to implement these port groups. IP subnets should be set aside for use on these port groups and the VLANs used to implement these port groups should be terminated on an external L3 gateway (such as the ASR1K shown in the figure). The control plane cluster and tenant plane Kubernetes clusters of Cisco Container Platform can then be provisioned on these port groups.

All provisioned Kubernetes clusters may choose to use a single shared port group or separate port groups may be provisioned (1 per Kubernetes cluster) depending on the isolation needs of the deployment. Layer 3 network isolation may be used between these different port groups as long as the following conditions are met:

- There must be L3 IP connectivity between the port group used for the Control Plane cluster and each of the Tenant cluster port groups

- The IP address of the vCenter server must be reachable from the Control plane cluster
- For current versions of Cisco Container Platform, a DHCP server must be provisioned for assigning IPs to the cluster nodes and it must be reachable from all cluster nodes

The simplest functional topology would be to use a single shared port group for all clusters with a single IP subnet to be used to assign IPs for all container cluster VMs. This IP subnet can be used to assign one IP per cluster VM and up to four virtual IPs per Kubernetes cluster, but would not be used to assign individual Kubernetes pod IPs. Hence, a reasonable capacity planning estimate for the size of this IP subnet is as follows:

(The expected total number of container cluster VMs across all clusters) + 3 x (The total number of expected Kubernetes clusters)

Container Network Interface Plugins

Cisco Container Platform supports multiple Kubernetes CNI plugins such as:

- ACI is the recommended plugin for use with an ACI fabric. It is optimized for use with an ACI fabric. ACI is fully supported by Cisco.
- Calico is recommended when an ACI fabric is not used. It can be used for quick evaluation of Cisco Container Platform. Calico is an integrated CNI plugin and is not fully supported under the Cisco commercial support agreement.
- Contiv (Tech Preview) is a user space switch that is optimized for high performance and scale.

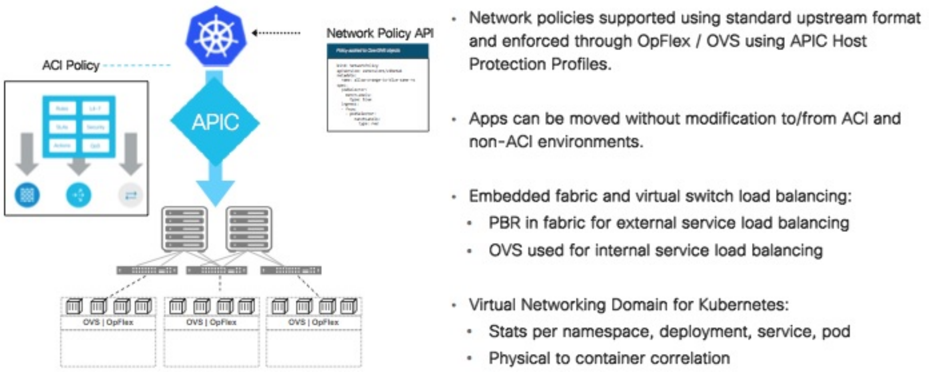
Operationally, all the CNI plugins offer the same experience to the customer. The container network connectivity is seamless and network policies are applied using Kubernetes NetworkPolicies. Under-the-hood, both ACI and Contiv offer advanced feature support. ACI allows you to map CNI NetworkPolicies to an ACI fabric and supports richer underlay policies such as common policies for containers/virtual machines/physical servers and inter-Kubernetes cluster policies. Additionally, ACI supports Kubernetes Type LoadBalancer using PBR policies in the ACI fabric.

ACI

ACI is tightly integrated with the ACI fabric. It supports underlay integration with the ACI fabric and hardware accelerated load balancing.

The following figure shows the architecture of ACI.

Figure 3: Architecture of ACI



- Network policies supported using standard upstream format and enforced through OpFlex / OVS using APIC Host Protection Profiles.
- Apps can be moved without modification to/from ACI and non-ACI environments.
- Embedded fabric and virtual switch load balancing:
  - PBR in fabric for external service load balancing
  - OVS used for internal service load balancing
- Virtual Networking Domain for Kubernetes:
  - Stats per namespace, deployment, service, pod
  - Physical to container correlation

System Requirements

This section describes the software, hardware, storage, and network requirements that are necessary to deploy Cisco Container Platform.

Supported Version Matrix

Cisco Container Platform uses various software and hardware components. The following table provides information on the validated versions of each component.

Component	Validated Version
Kubernetes	1.10 1.11
ACI	3.2(2o)
HyperFlex software	3.0(1b)+ 3.5
vSphere	vSphere 6.0 (u2)+ vSphere 6.5

Note Cisco Container Platform is supported on all hardware configurations that are supported by the required HyperFlex software versions. For more information on HyperFlex hardware configurations, refer to the UCS HyperFlex product documentation.

Requirements for VMware on UCS

The following are the requirements for a topology where VMware is deployed on a UCS platform, and HyperFlex is not used:

- Use the Enterprise Plus license to set up the VMware clusters with HA and DRS enabled. For more information on the supported versions of VMware, see Supported Version Matrix, on page 5.
- Use a processor that has an Ivy Bridge (UCS C220 M4) or a newer microarchitecture so that the CPU RDRAND instruction set is available.
- If you are enabling VMware EVC Mode, you must use an Ivy Bridge or a later micro-architecture so that the CPU RDRAND instruction set is available.
- Ensure that the following requirements are met on the network that you want to use for deploying the Cisco Container Platform VMware instances:
    - DHCP is enabled.
    - A static pool of IP addresses is available in the network for assigning IP addresses to the master nodes and the VIP pools.
    - The network is routable to and from the VMware vCenter server.
    - The client install machine is routable to the network during the Cisco Container Platform control plane install.
    - The network allows communication between Cisco Container Platform VM instances. You must not use a private LAN.
- By default, the Cisco Container Platform control plane VXLAN network uses 192.168.0.0/16 network. If you have routed IP addresses in that space, you must assign another RFC1918 range for your VXLAN network. It does not need to be a full /16 network, a /22 network is adequate for the Cisco Container Platform control plane.
- If the DHCP server does not provide the location of the NTP service, enter the NTP address in the Installer UI, under **Control Plane Settings** > **Advanced Settings**.
- Ensure that a shared datastore that is accessible to the VMware hosts such as NFS or iSCSI or FC, with at least 200 GB of storage is available.

Software Requirements

Ensure that the following software applications are installed in the deployment environment:

- VMware vCenter server 6.5
- VMware client integration plugin
- vSphere Flash client
- A supported version of HyperFlex software

  For more information on installing HyperFlex and accessing the HyperFlex Connect UI, refer to the latest HperFlex documentation.

Hardware Requirements

- In Cisco Container Platform 1.3.0 or later, the hypervisor hosts need to run CPUs with an Ivy Bridge (UCS C220 M4) or newer micro-architecture.

Storage Requirements

Once HyperFlex is installed, you need to configure a shared datastore that is accessible to the hosts in the cluster for the following purposes:

- Persistent volume storage
- Deploying the Cisco Container Platform tenant base VM

Configuring Shared Datastore

Step 1 Log in to the **HX Connect UI** using the VMware vCenter SSO administrator credentials.

Step 2 In the left pane, click **Manage** > **Datastores**.

Step 3 Perform these steps to create a datastore for provisioning the Kubernetes persistent volume storage and deploying the Cisco Container Platform tenant base VM:

a) In the right pane, click **Create Datastore**.

b) In the **Name** field, enter `ds1`, and then enter a size and block size for the datastore.

Note We recommend that you use `1TB` size and `8K` block size.

c) Click **Create Datastore**.

The newly created datastore is available on vCenter.

Configuring Link-local Network for HyperFlex iSCSI Communication

The FlexVolume plug-in requires a host-only link between each VM that runs Kubernetes and the Internet Small Computer System Interface (iSCSI) target on the ESX host.

For HyperFlex 3.5+

Step 1 Log in to the **HX Connect UI**.

Step 2 Choose **Settings** > **Integrations** > **Kubernetes**.

Step 3 Click **Enable All Node** and wait until the **KUBERNETES STORAGE PROVISIONING** option is enabled.

The HyperFlex infrastructure is configured and ready to use for Cisco Container Platform with Kubernetes persistent volume support.

For HyperFlex 3.0.x

Step 1 Open an SSH session to the HyperFlex 3.0 Platform Installer VM or one of the HyperFlex Controller VMs and log in as a root user.

Step 2 Perform these steps to get the vCenter details that you need to enter when you run the `add_vswitch.py` script.

a) Run the following command to get the vCenter datacenter name and vCenter cluster name.

```
stcli cluster info | grep -i vcenter
```

b) Run the following command to validate the reachability of vCenter IP address.

```
ping <vcenter URL>
```

Step 3 Navigate to the following location:

`/usr/share/springpath/storfs-misc/hx-scripts/`

Step 4 Run the `add_vswitch.py` script.

```
python add_vswitch.py --vcenter-ip <vCenter IP address>
```

When prompted, specify the vCenter credentials, datacenter name, and cluster name that you got from the output of Step 2.

The HyperFlex infrastructure is configured and ready to use for Cisco Container Platform with Kubernetes persistent volume support.

Resource Management Requirements

Enabling DRS and HA on Clusters

It is required that you enable DRS and HA on vCenter for the following reasons:

- DRS continuously monitors resource utilization across vSphere servers and intelligently balances VMs on the servers.
- HA provides easy to use, cost-effective high availability for applications running on virtual machines.

Step 1 In the vSphere Web Client, navigate to the host or cluster on which you want to deploy Cisco Container Platform.

Step 2 Click the **Configure** tab.

Step 3 Under **Services**, click **vSphere DRS**, and then click **Edit**.

Step 4 In the right pane of the **Edit Cluster Settings** window, check the **Turn ON vSphere DRS** check box, and then click **OK**.

Step 5 Under **Services**, click **vSphere Availability**, and then click **Edit**.

Step 6 In the right pane of the **Edit Cluster Settings** window, check the **Turn ON vSphere HA** check box, and then click **OK**.

Network Requirements

Provisioning a Port Group for Cisco Container Platform VM Deployment

Cisco Container Platform creates VMs that are attached to a Port Group on either a vSphere Standard Switch (VSS) or a Distributed Virtual Switch (DVS). The HyperFlex installer creates VSS switches in vSphere for the networks that are defined during installation. You need to create either VSS or DVS Switches for managing the VM traffic.

The following topics provide information on configuring a VSS or a DVS.

Configuring vSphere Standard Switch

Step 1 In the vSphere Web Client, navigate to the host or cluster on which you want to deploy Cisco Container Platform.

Step 2 Click the **Configure** tab.

Step 3 Expand **Networking**, and then select **Virtual switches**.

Step 4 Click **Add host networking**.

Step 5 Choose **Virtual Machine Port Group for a Standard Switch** as the connection type for which you want to use the new standard switch and click **Next**.

Step 6 Select **New standard switch** and click **Next**.

Step 7 Add physical network adapters to the new standard switch.

Step 8 Under **Assigned adapters**, click **Add adapters**.

Step 9 Select one or more physical network adapters from the list.

Step 10 From the **Failover order group** drop-down list, choose from the Active or Standby failover lists.

Step 11 For higher throughput and to provide redundancy, configure at least two physical network adapters in the Active list.

Step 12 Click **OK**.

Step 13 Enter connection settings for the adapter or the port group as follows:

a) Enter a network Label or the port group, or accept the generated label.
b) Set the VLAN ID to configure VLAN handling in the port group.

Step 14 On the **Ready to Complete** screen, click **OK**.

Configuring DHCP Server

Cisco Container Platform requires a DHCP server to be present. The Cisco Container Platform installer VM, Control Plane VMs and tenant cluster VMs get their primary interface IP addresses from the DHCP server. You must ensure that you have configured a DHCP server.

Reserving IP Addresses for Static Allocation

A static IP address is used during Cisco Container Platform installation for the **CCP Control Plane master node virtual IP** to support Cisco Container Platform upgrades. Additionally, Virtual IP address (VIP) is used as an external IP address for each Kubernetes cluster. VIPs are configured using VIP pools. You can obtain this IP address from the same or a different subnet and you must ensure that it is not part of a DHCP pool.

Static and DHCP IP Address Requirements

The following table summarizes the static and DHCP IP address requirements for the Cisco Container Platform components.

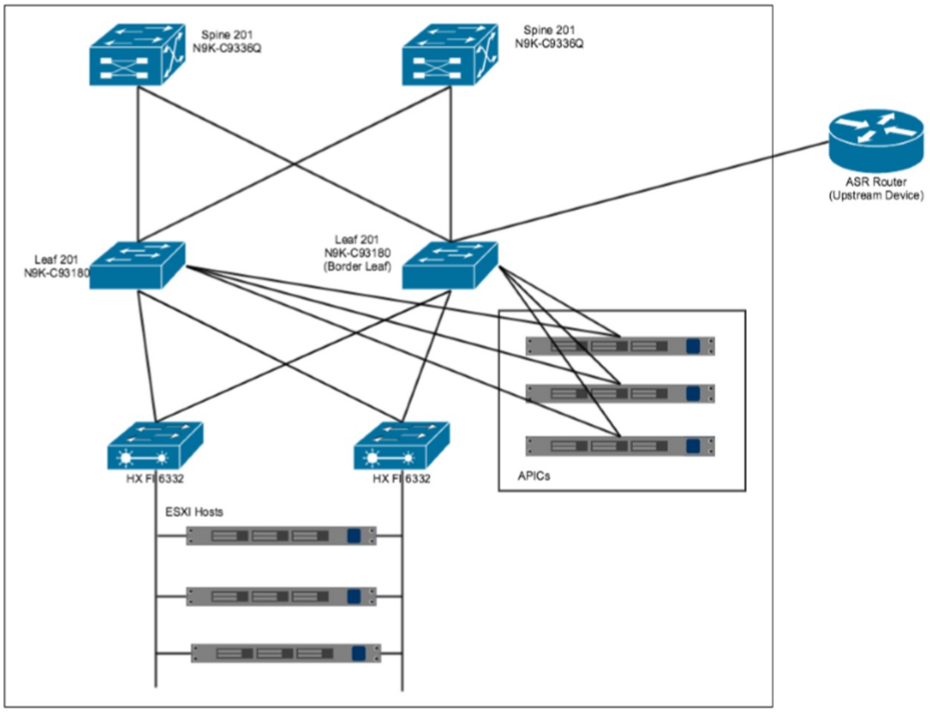
Component	Static IP	DHCP IP
Cisco Container Platform web interface	1	4
Installer VM	-	1
Tenant clusters	2 + Number of load balancer VIPs desired for applications	1 + Number of workers

ACI Integration Requirements

Cisco ACI enables you to group your application into End Point Groups (EPGs), define policies for the EPGs, and then deploy network policies on the ACI fabric. The policy enforcement is implemented using the spine and leaf architecture of the ACI fabric.

The following figure shows the components of a Cisco Container Platform ACI integrated network topology.

Figure 4: Cisco Container Platform ACI Integrated Network Topology



The main components of the network topology are as follows:

- **ACI Fabric** includes two spine nodes, two leaf nodes, and three APIC controllers. You can choose the number of the spine and leaf nodes and APIC controllers as per your network requirement.
- **HyperFlex Fabric Interconnect (FI)** includes two fabric interconnect switches connected between the ESXi hosts and the ACI leaf switches.
- **ESXi Hosts** includes a UCS server such as UCS C220 M4.
- **ASR router** is connected to an ACI border leaf for external internet access.

APIC Controller Requirements

If you are using ACI, ensure that you have configured the following settings on the APIC controller:

- Assign a port number other than 4094 for Infra VLAN as 4094 is reserved for provisioning HyperFlex fabric interconnect
- Create a common tenant
- Create a Virtual Route Forwarder (VRF) in the common tenant
- Create at least one L3OUT
- Create an Access Entity Profile (AEP) for the ACI tenant physical domain
- Create an AEP for L3OUT
- Create a Virtual Machine Manager (VMM) domain which connects to vSphere

For more information on configuring an APIC controller, refer to the latest ACI documentation.

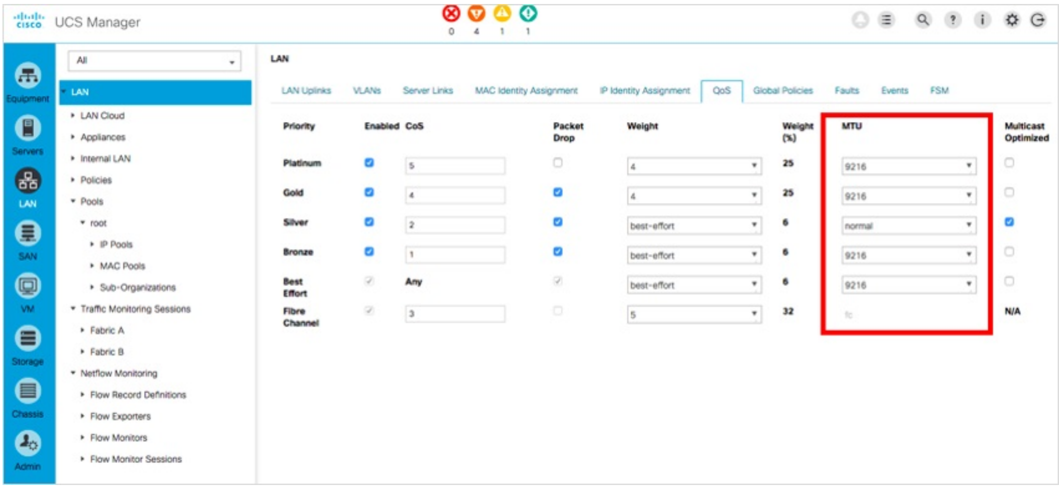
HyperFlex FI Requirements

Ensure that you have configured the following settings on HyperFlex FI:

- Configure QOS
  1. From the left pane, click **LAN**.
  2. From the right pane, click the **QoS** tab, and then configure QoS.

Note Using the **MTU** configuration, you must set the priority that is associated with the QoS policy of the vNIC template.

Figure 5: QoS Tab



Priority	Enabled	CoS	Packet Drop	Weight	Weight (%)	MTU	Multicast Optimized
Platinum	☑	5	□	4	25	9216	□
Gold	☑	4	☑	4	25	9216	□
Silver	☑	2	☑	best-effort	6	normal	☑
Bronze	☑	1	☑	best-effort	6	9216	□
Best Effort	☑	Any	☑	best-effort	6	9216	□
Fibre Channel	☑	3	□	5	32	fc	N/A

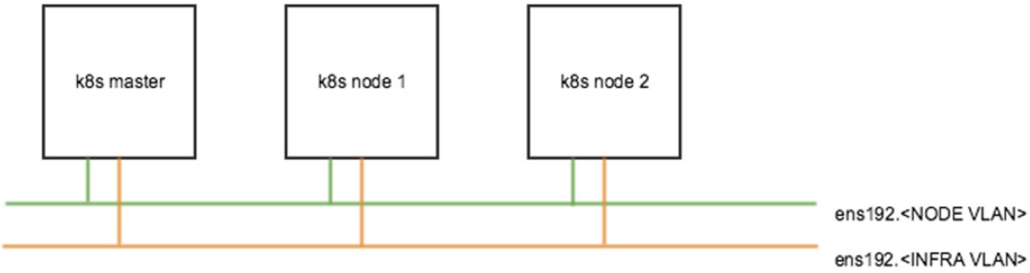
- Ensure that the tenant VLAN is allowed

Once Cisco Container Platform Control Plane and management node networking are configured, you can access the HyperFlex cluster on vSphere and install Cisco Container Platform. Each time you create a tenant cluster, the ACI constructs such as L3OUT, VRF, and AEP stored in the common tenant cluster are reused.

Tenant Cluster with ACI Deployment

With an ACI deployment, each tenant cluster is required to have its own routable subnet. The node VLAN, pod subnet, and multicast subnet range should not overlap between clusters. Cisco Container Platform ensures that the VLAN and subnet do not overlap.

Unlike other CNI, an ACI tenant cluster requires two VLAN subinterfaces, one for the Node VLAN, and another for the Infra VLAN. As shown in the following figure, Cisco Container Platform assigns unique Node VLAN IDs. You need to assign a unique Infra VLAN ID for clusters during cluster creation.



For more information on creating tenant clusters, refer to the *Creating Kubernetes Clusters* section of the *Cisco Container Platform User Guide*.

For more information on the ACI and CNI plugin, refer to the latest documentation on Cisco ACI and Kubernetes Integration.

Getting Cisco Container Platform Software

This chapter contains the following topics:

Downloading the Software

Before you begin the installation, you need to download the required software assets.

Step 1 Go to the Product Support Page of Cisco Container Platform.

Step 2 Under **Support Documentation And Software**, click **Download Software**.

The **Software Download** page appears displaying the latest release assets.

Step 3 Log in using your Cisco username and password that is associated with a valid service contract.

Step 4 Download the Installer and Tenant images.

Unpacking the Software

Step 1 Browse to the directory where you have downloaded the software.

Step 2 Open the Shell command prompt and extract each `tar.gz` file.

Example

```
$ tar -zxvf kcp-vm-$VERSION.tar.gz
kcp-vm-$VERSION/
kcp-vm-$VERSION/ee.pem
kcp-vm-$VERSION/ccp_image_signing_release_v1_pubkey.der
kcp-vm-$VERSION/root_ca.pem
kcp-vm-$VERSION/kcp-vm-$VERSION.ova.signature
kcp-vm-$VERSION/kcp-vm-$VERSION.ova
kcp-vm-$VERSION/verify
kcp-vm-$VERSION/sub_ca.pem
kcp-vm-$VERSION/README
```

The `.ova` file contains the Cisco Container Platform image.

Verifying the Software

Before you begin

Ensure that your system has python 3.5.2 or later and OpenSSL installed.

Step 1 Browse to the directory where you have unpacked the software.

Step 2 Open the Shell command prompt and run the script to verify the software.

Note You must run the verification steps for each release image.

Example

```
$ ./verify --type release --signature kcp-vm-$VERSION.ova.signature --image kcp-vm-$VERSION.ova
Verifying sha512 hash of ./root_ca.pem
Successfully verfied sha512 hash of ./root_ca.pem
Verifying sha512 hash of ./sub_ca.pem
Successfully verfied sha512 hash of ./sub_ca.pem
Verifying root(./root_ca.pem) and subca(./sub_ca.pem)
Successfully verified root and subca.
Verifying cert(./ee.pem) against root(./root_ca.pem) and subca(./sub_ca.pem)
Successfully verified end entity cert.
Extracting pubkey(kcp-vm-$VERSION/ee.pubkey) from ./ee.pem
Successfully extrated public key to kcp-vm-$VERSION/ee.pubkey.
Verifying signature(kcp-vm-$VERSION.ova.signature) of kcp-vm-$VERSION.ova using
kcp-vm-$VERSION/ee.pubkey
Successfully verified signature.
```