Cisco HyperFlex and Red Hat OCP for Hybrid Cloud

Design and Deployment Guide

Published: December 2022

CISCO VALIDATED DESIGN

In partnership with:

Red Hat

About the Cisco Validated Design Program

The Cisco Validated Design (CVD) program consists of systems and solutions designed, tested, and documented to facilitate faster, more reliable, and more predictable customer deployments. For more information, go to: http://www.cisco.com/go/designzone.

Executive Summary

Hybrid cloud has become the de facto deployment and operating model in most Enterprises. In a study conducted by 451 Research across 2500 organizations from around the globe, 82% of the IT decision makers responded that they are already using a hybrid cloud model. Cloud computing from hyper-scalers such as Amazon Web Services (AWS), Microsoft Azure and Google Cloud offer limitless scale and flexibility, but it also comes with increasingly high costs, at times higher risk, leaving Enterprises with less control over their business-critical applications and data. As a result, Enterprises are adopting a hybrid strategy that allows them to optimally use both on-prem and public cloud infrastructure to meet their computing needs.

Hybrid cloud model enables Enterprises to:

- Leverage public cloud for specific use cases, for example, to meet short-term spikes in demand or for disaster recovery (DR). An Enterprise can minimize their CAPEX investment by not having to maintain under-utilized on-prem resources for these scenarios. However, a hybrid cloud DR strategy that requires the movement of data back to the Enterprise could get very costly as cloud providers charge considerably more for moving the data out of the cloud than for moving data into the cloud.
- Benefit from higher availability inherent in the hybrid cloud model. The Enterprise’s data center is now distributed across different infrastructures in different geographical locations, one managed by the Enterprise and the other by the cloud provider. As such, in most cases and if designed properly, a failure in one location should only impact that location.
- Accelerate innovation through increased agility as Enterprises can quickly spin up environments in the public cloud to start their development efforts and still have the option to deploy the application on-prem for testing or production where it might be easier to integrate into existing tools and processes. It also allows them to retain control of their data.
- Flexibility to select an optimal infrastructure and location that best meets their business needs. Each organization will have unique costs, compliance, security, performance, and other requirements and it helps to have more options.

Some of the common Enterprise use cases for hybrid cloud are:

- **Enabling cloud-native environments anywhere,** either on-prem or public cloud, with consistent life cycle management across a hybrid infrastructure environment. Enterprises need this to accelerate their application modernization efforts and for developing new applications. In production, the hybrid model enables them to deploy some applications in the cloud while keeping others on-prem, or host applications in both environments for redundancy, load-balancing etc.
- **Development and Test (Dev/Test)** where multiple teams in an application’s build/release cycle need multiple infrastructure environments for development, testing, production etc. For example, organizations may start their initial development in the public cloud where they can quickly spin up an environment, but then will deploy that application into production on-prem where they can easily access backend data, tooling, and other resources .
- **Backup and recovery** where the application resides either on-prem or distributed across both on-prem and cloud, but the data is backed up in the cloud. Recovery in this case can be to on-prem and/or cloud depending on the application.
- **Cloud bursting** or data center extension where an application scales into the cloud to meet peak demands or to enhance the on-prem application using Machine Learning or other data-intensive computations running in the cloud.

The solution presented in this document will address the first two use cases and deliver a foundational cloud-native hybrid-cloud infrastructure with operational simplicity and ease. It will enable developers and operators to quickly deploy cloud-native workloads anywhere with consistent operational experience across both environments. The solution is built using Cisco HyperFlex, Cisco Intersight, Amazon Web Services (AWS), and Red Hat OpenShift Container Platform (OCP). The solution also uses Cisco HyperFlex Container Storage Interface (CSI) to provide persistent storage for stateful Kubernetes (K8S) workloads running on Cisco HyperFlex, and Cisco Intersight Workload Optimizer (IWO) to ensure application performance, optimize resource usage, and manage cloud costs in a hybrid cloud deployment. The on-prem infrastructure deployment is automated using Red Hat Ansible to provide Infrastructure as Code (IaC) that can be integrated into existing CI/CD pipelines or other automation to accelerate deployments.

Solution Overview

This chapter contains the following:

- Audience
- Purpose of this Document
- What's New in this Release?
- Solution Summary

Hybrid architectures are complex and challenging, and ad-hoc deployments that often occur organically in many Enterprises, only add to that challenge, making it more risky and costly for Enterprises. To address this, Enterprises need a hybrid cloud strategy and architecture that can deliver an agile, responsive infrastructure to its users to power new applications and innovations. The solution presented here can serve as a foundational hybrid-cloud reference architecture for Enterprises, with the simplicity, agility, and operational consistency that IT and Dev-Ops need.

Audience

The intended audience for this document includes, but is not limited to, sales engineers, field consultants, professional services, IT teams, partners, and customers who are working on or interested in designing and deploying Cisco's Hybrid Cloud solutions.

Purpose of this Document

This document is a Cisco Validated Design (CVD) for a Cisco hybrid cloud infrastructure solution for cloud-native workloads. The document provides the end-to-end design with detailed procedures for implementing the solution across a Cisco data center and public cloud. Hardware and software components used to validate the solution in Cisco's internal labs are also provided.

What's New in this Release?

This is the initial release of Cisco's hybrid cloud solution with Cisco HyperFlex and Red Hat OCP. At a high level, the solution delivers a simple, flexible, and scalable infrastructure for an Enterprise's cloud-native efforts, enabling workloads to be deployed anywhere from on-prem to a public cloud. The solution supports the following hybrid cloud use cases:

- Enable cloud-native environments anywhere with consistent management
- Development and Test

Other capabilities in this release of the solution are:

- Support for Cisco HyperFlex CSI 1.2.5
- Support for Cisco HyperFlex release 5.0(2a)
- Support for Red Hat OpenShift Container Platform 4.10
- Cisco Intersight for consistent, centralized operations across a hybrid environment
- Red Hat Hybrid Cloud Console and OpenShift cluster console for consistent, centralized K8S management across a hybrid environment
- Application performance monitoring with cloud cost management and resource optimization using Cisco Intersight Workload Optimizer (IWO) to ensure the performance of containerized applications in a hybrid deployment.
- IaC using Red Hat Ansible for the automated deployment of on-prem compute, storage, and networking

- Automated Install of Red Hat OCP on HyperFlex Virtual Server Infrastructure (VSI) and on AWS EC2 instances using Installer Provisioned Infrastructure (IPI) method

Solution Summary

This solution provides a foundational reference architecture for a hybrid cloud infrastructure solution. The solution enables Enterprises to deploy and develop cloud-native applications anywhere, with consistent management and operational experience for both developers and IT Operations/Dev-Ops teams. The solution also supports two common hybrid cloud use cases as outlined earlier.

A hybrid cloud, by definition, is a cloud-computing architecture consisting of at least one on-prem location, a public cloud, and a secure network that interconnects the two locations. This solution delivers a hybrid cloud using a combination of Cisco, Red Hat, and AWS products and technologies as outlined below.

- **Cisco HyperFlex** standard clusters provide Enterprise-class software-defined compute, storage, and server networking for the on-prem Enterprise data center. HyperFlex delivers operational simplicity and agility without compromising scale, performance, or flexibility.
- **Cisco Intersight** provides cloud-based infrastructure management with centralized visibility and operations for all HyperFlex, Cisco UCS, and supported third-party infra that an Enterprise has located anywhere across the globe, from edge locations to Enterprise data centers. The SaaS delivery model enables IT teams to benefit from the continuous delivery of innovations and features without having to life cycle manage the management platform. Integration of vCenter, AWS, and OCP nodes in Intersight enables full-stack visibility, monitoring, and resource optimization.
- **Cisco Networking** using on-prem CSR 1000v and AWS Transit Gateways provides secure hybrid cloud connectivity.
- **Red Hat OpenShift Container Platform** provides a highly secure, Enterprise-class container orchestration platform with development and operational tools that simplify cloud-native efforts. OCP also delivers a consistent operational experience across both on-prem and public cloud.
- **Red Hat Hybrid Cloud Console** provides cloud-based centralized management of OCP clusters distributed across on-prem and public clouds in a hybrid deployment. The application OCP clusters in the solution, hosted on HyperFlex VSI and AWS, are also deployed from the Red Hat Hybrid Cloud Console.
- **Cisco HyperFlex CSI** provides persistent container storage for cloud-native workloads hosted on Cisco HyperFlex using the underlying HyperFlex storage. It is Red Hat certified and can be quickly deployed on a cluster using a simple click-button install.
- **VMware vSphere** provides the virtualization on HyperFlex infrastructure. OCP clusters are deployed on VMs on HyperFlex and EC2 instances on AWS.
- **Cisco Intersight Workload Optimizer** ensures application performance with real-time analytics and automated decision-making that can adapt to changes real-time through actions that move workloads, add compute and memory resources etc. IWO continuously monitors components and resources to optimize resource usage and manage cloud-costs with full-stack visibility across a hybrid deployment.
- **Infrastructure as Code** using Red Hat Ansible automates the deployment of on-prem hyperconverged infrastructure, and networking to speed up deployment and for integration into existing Enterprise automation and/or CI/CD pipelines.

The end-to-end solution was validated in Cisco's internal labs with Cisco and partner recommended best practices in place.

Technology Overview

This chapter contains the following:

- Cisco HyperFlex
- Cisco Intersight
- Red Hat OpenShift Container Platform (OCP)
- Amazon Web Services (AWS)
- Cisco Intersight Workload Optimizer (IWO)
- Red Hat Ansible

The above components are interconnected across a hybrid environment and integrated using best practices from both Cisco and Red Hat to deliver an Enterprise-class hybrid-cloud solution for cloud-native applications and workloads. The solution also uses Cisco networking solutions, Cisco Application Centric Infrastructure (ACI) and IPsec VPN to provide on-prem and hybrid cloud connectivity, respectively. The solution provides simplicity, agility, flexibility, scalability, and performance to meet Enterprise needs with consistent operations and management across a hybrid environment. The upcoming sections provide a summary of the key features and capabilities available in these components.

Cisco HyperFlex

HyperFlex is a highly flexible, resilient, and scalable hyper-converged virtual server infrastructure (VSI) platform capable of supporting an organization's most mission critical and performance-intensive workloads, both traditional and cloud-native. HyperFlex systems with Intel® Xeon® Scalable processors is certified as an Intel® Select Solution for Hybrid Cloud deployments. HyperFlex provides software-defined infrastructure with software-defined computing and storage enabled by Cisco Unified Computing System (Cisco UCS) servers and Cisco HyperFlex Data Platform (HXDP) software. HyperFlex systems can be deployed and managed from the cloud using Cisco Intersight to quickly deliver software-defined compute and storage to any location within the Enterprise. Cisco Intersight offers centralized management of all HyperFlex systems (and Cisco UCS) with a comprehensive set of features and capabilities to simplify and ease operations.

HyperFlex is engineered data integrity, reliability, and availability to ensure business continuity and data protection. HyperFlex provides Enterprise-class shared storage with data services such as:

- **Replication** based on the replication factor configured, stripes and replicates data across nodes in the cluster to protect against single or multi-component failures.
- **Native replication** replicates data to other clusters for backup or disaster-recovery purposes. Each node participates in the data transfer, minimizing the performance impact.
- **Synchronous replication** in HyperFlex stretch clusters maintains data in two locations at the same time to enable failover from one location to the other with zero Recovery Point Objective (RPO) and very short Recovery Time Objective (RTO).
- **Data deduplication** is always on to reduce storage capacity which on virtualized clusters with virtual machines running multiple instances of an operating system, often the same, results in large amounts of replicated data.
- **Data compression** further reduces storage requirements, lowering costs. HyperFlex uses a log-structured file system that is designed to store variable-sized blocks which internal fragmentation.
- **Thin provisioning** allows large data volumes to be created without requiring actual storage until it is needed so that storage capacity isn't being reserved without being used.

- **Fast, space-efficient clones** rapidly replicate storage volumes so that virtual machines can be replicated simply through metadata operations, with actual data copied only for write operations.
- **Data protection API** enables enterprise backup tools to access data volumes for consistent per-VM backup operations.

Cisco HyperFlex Data Platform (HXDP)

The foundation for Cisco HyperFlex systems is the Cisco HyperFlex Data Platform software that runs on each node in a HyperFlex cluster. HyperFlex Data Platform is a purpose-built, high-performance, log-structured, scale-out file system with enterprise-class storage features. The HXDP software runs on Cisco HyperFlex (HX-series) nodes to create a highly available cluster. Each HyperFlex server/node includes an HyperFlex Data Platform controller or Storage Controller Virtual Machine (SCVM) that implements the scale-out and distributed file system using internal flash-based SSDs, NVMe storage, or a combination of NVMe, flash-based SSDs and high-capacity HDDs to store data. The distributed controller design prevents the controller from becoming a bottleneck, unlike many Enterprise storage systems. Each controller assumes direct control of all drives on a node via PCI passthrough and communicates with other controllers over 10 or 40 GbE to aggregate all available capacity across all nodes into a single pool of storage. As nodes are added, the cluster scales linearly to deliver computing, storage capacity, and I/O performance.

Each HyperFlex node also run **IOvisor** as a software VIB (SCVM client) inside the hypervisor to enable access to the datastores presented by the HyperFlex storage cluster. HyperFlex provides both high-performance and resiliency as the stored data is distributed across all nodes and all nodes participate in serving I/O.

HyperFlex provides several hardware configuration options. Enterprises can customize the HyperFlex server configuration as needed to meet business and workload requirements.

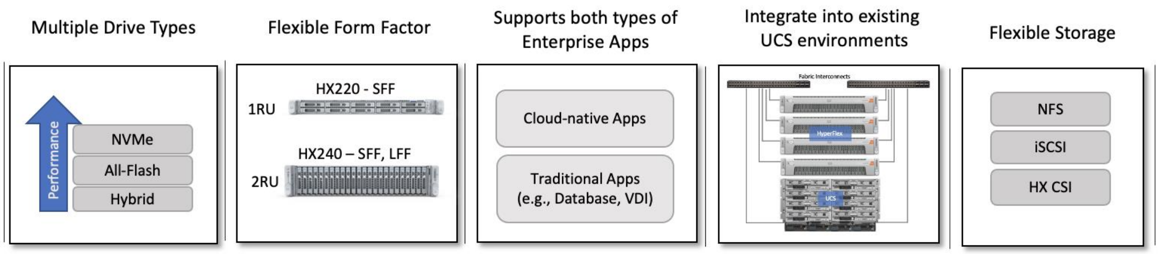
For more details on HyperFlex Data Platform software architecture and design, see: https://www.cisco.com/c/en/us/products/collateral/hyperconverged-infrastructure/hyperflex-hx-series/white-paper-c11-736814.html

HyperFlex Design Options

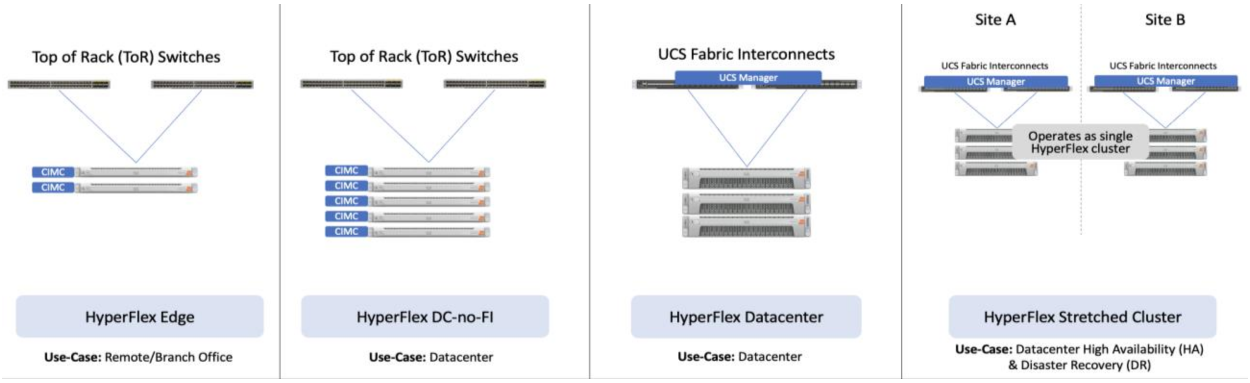
HyperFlex offers enterprise-class features and performance with distributed, scale-out storage and operational simplicity for an Enterprise’s cloud-native efforts in a hybrid environment. Some additional factors that make HyperFlex particularly well-suited as a hybrid cloud platform are:

- Flexibility

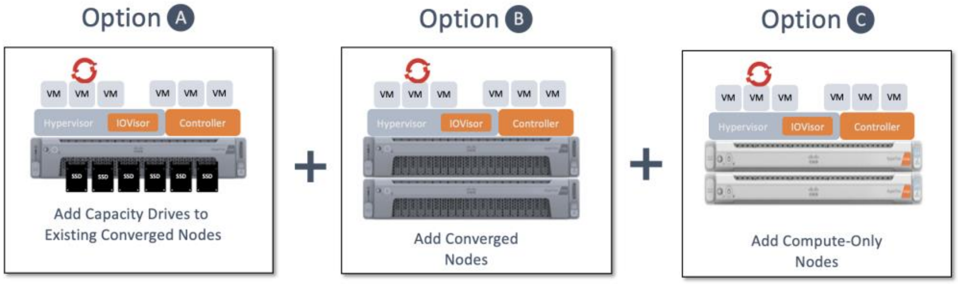
  HyperFlex supports modern and traditional business-critical Enterprise applications with GPU acceleration, flexible choice of processors (Intel, AMD), disk capacity and type, network bandwidth and interface type, form factor, and storage. Virtual Machines and containers running on HyperFlex can leverage HyperFlex storage using NFS and HX CSI, respectively. HyperFlex can also provide iSCSI storage to external iSCSI clients such as Windows servers, Linux servers, and so on. HyperFlex can integrate into existing environments; VMs running on HyperFlex can mount existing Enterprise storage using NFS, iSCSI, or FC. HyperFlex systems can also connect to existing Cisco UCS FIs with other Cisco UCS server platforms.



HyperFlex is also a very flexible, adaptable, and scalable platform that can be deployed anywhere from Enterprise data centers to edge locations with centralized deployment and management from Cisco Intersight. It is available in multiple configuration options such as HX edge and stretched cluster to support different Enterprise use cases.

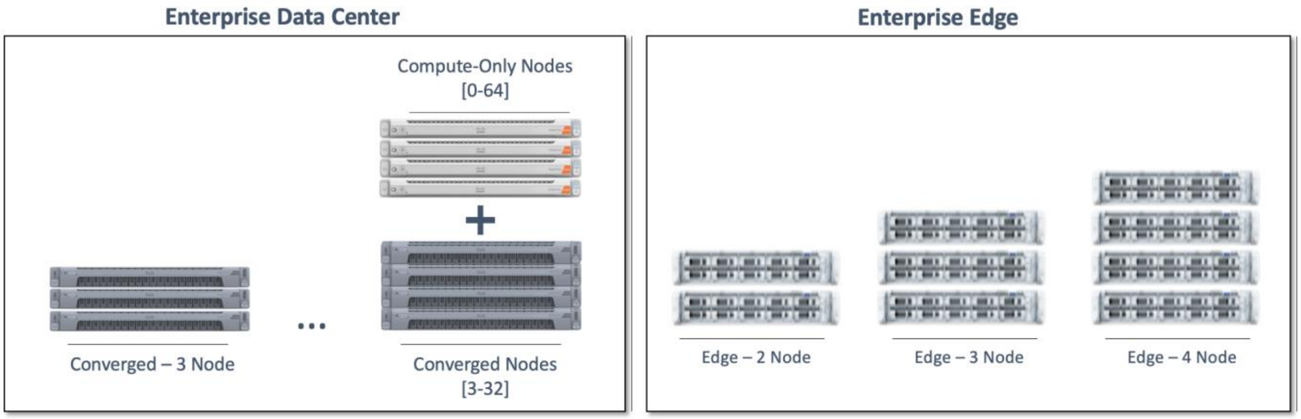


HyperFlex with support for multiple drive (Hybrid, All-Flash, NVMe) and node types (HX220, H240) can independently scale compute and storage. The cluster can also be expanded incrementally, one node at a time, with linear performance.



- Scalability

  At the time of writing this document, s HyperFlex cluster can start with as few as 2-nodes in a HyperFlex edge cluster and expand up to 96 nodes in a single cluster using a combination of converged and compute-only nodes. The flexibility to independently scale compute and storage allows Enterprises to tune the cluster performance to meet workload requirements using a combination of CPU, memory, GPU acceleration, and storage. A 4-node, 2RU HyperFlex system with 15TB NVMe drives and 16 x 256GB DIMMs can provide over a petabyte of storage capacity and 4TB of memory, respectively.



The HyperFlex platforms also offer multiple network interface options with speeds of 1-, 10-, and 25-Gbps on the edge platforms and 10-, 25-, 40-, and 100-Gbps on core data center platforms.

- Security

  Security is an integral part of HyperFlex with several security features as outlined below:

  - **Data-at-Rest-Encryption** with self-encrypting drives (SED) with Enterprise key management software using local passphrase or remote key management.
  - **Native Software Encryption** of filesystem built into HyperFlex with key management via Intersight
  - **Security Baseline System Checks** is part of HyperFlex Health Check capabilities in Intersight
  - **Secure CLI/Shell** reduces attack surface by limiting commands and access via SSH to 'admin' user only.
  - **Secure Boot** verifies the trustworthiness of ESXi hypervisor by chaining each stage of boot with a hardware root of trust provided by the Cisco Trust Anchor Module
  - **Security Hardened APIs** with support for Secure Technical Implementation Guides (STIGs) for all vSphere components

For more information, see: Cisco HyperFlex - Hyperconverged Infrastructure (HCI) on cisco.com

Cisco UCS Servers

A standard HyperFlex cluster requires a minimum of three HX-Series converged nodes with compute and storage. Data is replicated across at least two of these nodes, and a third node is required for continuous operation in the event of a single-node failure. Each node that has disk storage is equipped with at least one high-performance SSD drive for data caching and rapid acknowledgment of write requests. Each node also is equipped with additional disks, up to the platform's physical limit, for long term storage and capacity.

HyperFlex servers come in multiple form-factors (1RU, 2RU) with a range of Intel and AMD processor, memory, and drive options. For a complete list of server options and specifications, see: https://www.cisco.com/c/en/us/products/hyperconverged-infrastructure/hyperflex-hx-series/index.html#~models

The servers in the solution are equipped Cisco VIC 1387 MLOM Interface cards.

The mLOM slot is used to install a Cisco VIC without consuming a PCIe slot, which provides greater I/O expandability. It incorporates next-generation converged network adapter (CNA) technology from Cisco, providing investment protection for future feature releases. The card enables a policy-based, stateless, agile server infrastructure that can present up to 256 PCIe standards-compliant interfaces to the host, each dynamically configured as either a network interface card (NICs) or host bus adapter (HBA). The personality of the interfaces is set programmatically using the service profile associated with the server. The number, type (NIC or HBA), identity (MAC address and World Wide Name [WWN]), failover policy, adapter settings, bandwidth, and quality-of-service (QoS) policies of the PCIe interfaces are all specified using the service profile.

The Cisco UCS VIC 1387 Card is a dual-port Enhanced Quad Small Form-Factor Pluggable (QSFP+) 40-Gbps Ethernet, and Fibre Channel over Ethernet (FCoE)-capable PCI Express (PCIe) modular LAN-on-motherboard (mLOM) adapter installed in the Cisco UCS HX-Series Rack Servers. The Cisco UCS VIC 1387 is used in conjunction with the Cisco UCS 6332 or 6332-16UP model Fabric Interconnects.

Figure 1. Cisco VIC 1387 mLOM Card

Cisco UCS Fabric Interconnect

The Cisco UCS Fabric Interconnect (FI) is a core part of the Cisco Unified Computing System, providing both network connectivity and management capabilities for the system. Depending on the model chosen, the Cisco UCS Fabric Interconnect offers line-rate, low-latency, lossless 10 Gigabit, 25 Gigabit, 40 Gigabit, and 100 Gigabit Ethernet connectivity. Cisco UCS Fabric Interconnects provide the management and communication backbone for Cisco UCS and HyperFlex series Servers. All servers and chassis, and therefore all blades, attached to the Cisco UCS Fabric Interconnects become part of a single, highly available management domain, and provides LAN connectivity to all servers within a Cisco UCS domain. The product family supports Cisco low-latency, lossless Ethernet unified network fabric capabilities, which increase the reliability, efficiency, and scalability of Ethernet networks. The Fabric Interconnect supports multiple traffic classes over the Ethernet fabric from the servers to the uplinks.

There are several models of Cisco UCS Fabric Interconnects. A pair of Cisco UCS 6332 Fabric Interconnects are used in this design.

Cisco UCS 6332 Fabric Interconnect

The Cisco UCS 6332 Fabric Interconnect is a one-rack-unit (1RU) 40 Gigabit Ethernet and FCoE switch offering up to 2560 Gbps of throughput. The switch has 32 40-Gbps fixed Ethernet and FCoE ports. Up to 24 of the ports can be reconfigured as 4x10Gbps breakout ports, providing up to 96 10-Gbps ports, although Cisco HyperFlex nodes must use a 40GbE VIC adapter in order to connect to a Cisco UCS 6300 Series Fabric Interconnect.

Figure 2. Cisco UCS 6332 Fabric Interconnect

Other models of Cisco UCS Fabric Interconnects are the Cisco UCS 6500 and 6400 series in addition to the Cisco UCS 6300 series. Table 1 lists some key differences between these three Cisco UCS FI series.

Table 1. Cisco UCS Fabric Interconnect Models

Cisco UCS Fabric Interconnects	UCS 6500 Series	UCS 6400 Series	UCS 6300 Series
Port Density	36	54 or 108	32-40GbE -or- 24 x 40GbE and 16 x 1- and 10GbE
Port Speeds	1/10/25/40/100-Gbps Ethernet or FCoE	1/10/25/40/100-Gbps Ethernet or FCoE	1/10/40-Gbps
Throughput	Up to 7.42 Tbps	- 6454: Up to 3.82 Tbps - 64108: Up to 7.42 Tbps	Up to 2.56-Tbps

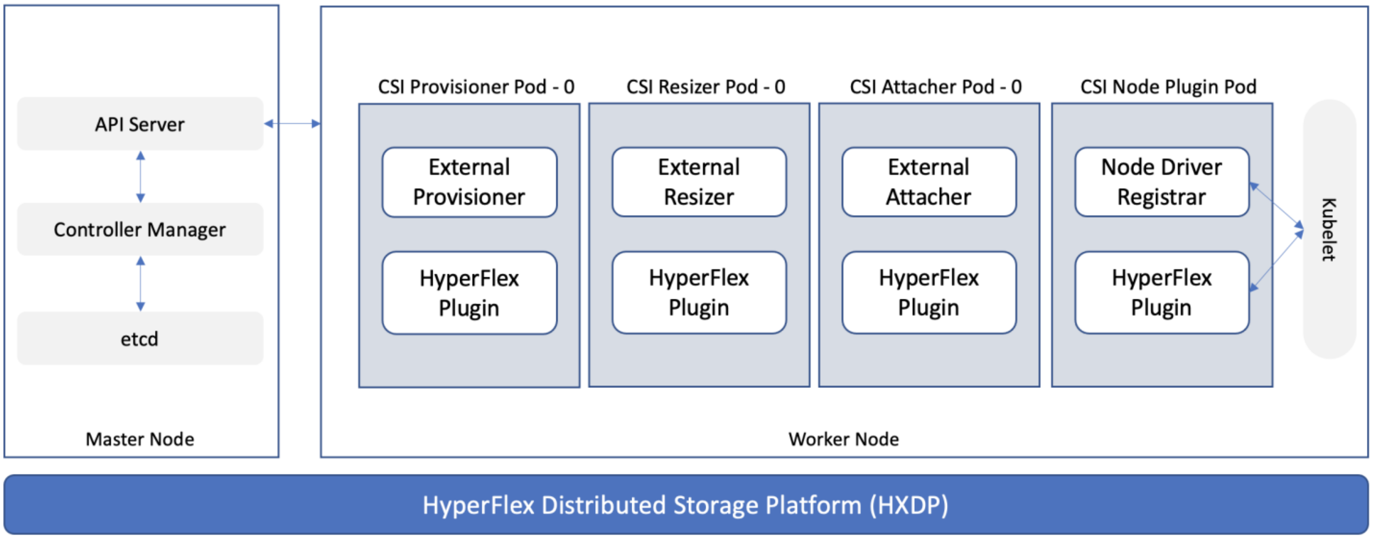
Cisco HyperFlex Container Storage Interface (CSI)

Cisco HyperFlex CSI is an out-of-tree container-based Kubernetes storage plugin that enables stateful container workloads to dynamically request persistent storage from the underlying Hyperflex storage. Developers and operators leverage their Cisco HyperFlex to meet both Kubernetes compute and storage needs with minimal additional administrative overhead. With Cisco HyperFlex CSI, the performance, efficiency, and resiliency of Cisco HyperFlex can now be extended to stateful Kubernetes applications. HyperFlex CSI is Red Hat certified and available as an operator from Red Hat's Operator Hub. The Cisco HyperFlex CSI operator is developed and published using a family of tools and capabilities (Operator SDK, Operator Lifecycle manager, Operator registry) – see Operator Framework for more information. The Cisco HyperFlex CSI Operator, along with the driver is published on Red Hat's registry (Quay.io) and made available to Kubernetes developers and operators via the embedded Operator Hub on the OCP cluster console.

To deploy and consume HyperFlex storage, the HyperFlex CSI uses native Kubernetes objects and primitives (for example, Storage Classes) that Kubernetes users are already familiar with, making it easier to adopt and use. CSI enables orchestration of the entire Persistent Volume object lifecycle to be offloaded and managed by Cisco HyperFlex by initiating a storage request using standard Kubernetes Persistent Volume Claims (PVC). The CSI plugin can be quickly deployed by simply searching for it in Red Hat's Operator Hub and then clicking 'Install' to deploy it. Once the plugin and driver are deployed and setup, any stateful application running on HyperFlex can request and use storage as needed.

The Cisco HyperFlex CSI is deployed in Pods and containers on the Red Hat OCP cluster where it is deployed. The different HyperFlex CSI components deployed on a cluster are shown in Figure 3.

Figure 3. Cisco HyperFlex CSI Architecture



On the HyperFlex side, the internal distributed storage file system is presented using Internet Small Computer Systems Interface (iSCSI) protocol to applications and other external entities. The CSI Plugin leverages the HyperFlex iSCSI capability to dynamically create iSCSI volumes and present them to the Red Hat OCP cluster to be utilized by Kubernetes Pods for persistent storage. To enable OCP pods to mount HyperFlex storage as iSCSI volumes, a dedicated iSCSI storage network is utilized that routes traffic from OCP cluster in the HyperFlex VM network to the iSCSI storage data network.

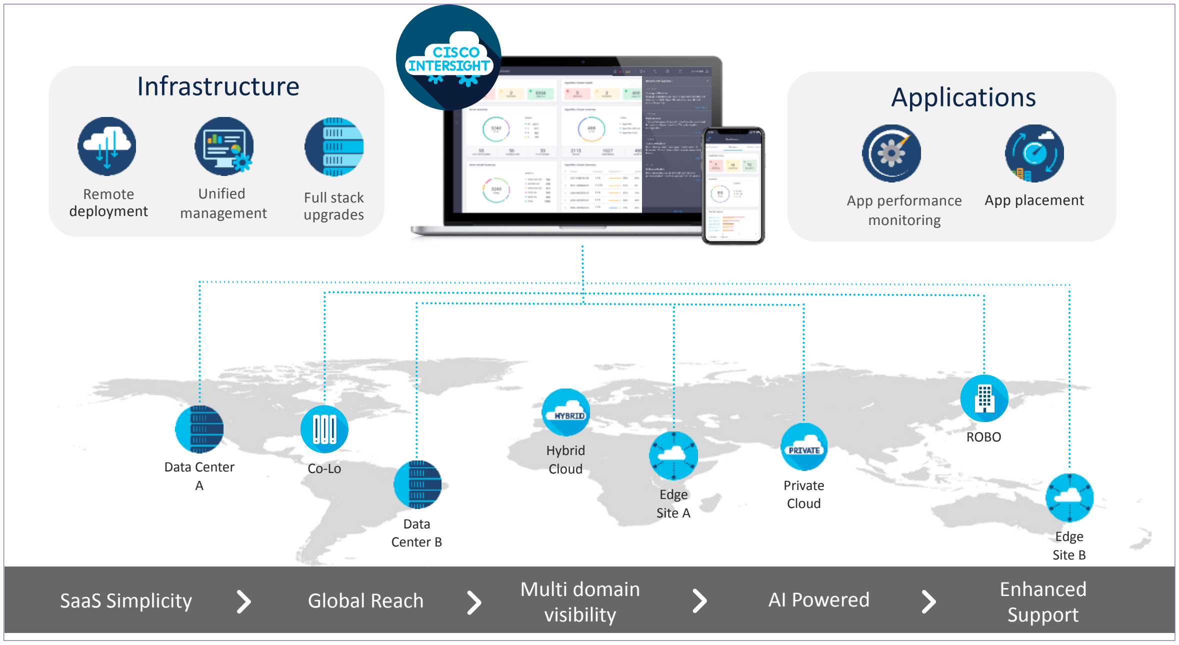
At the time of this document, Cisco HyperFlex CSI supports the following features:

- Support for CSI Spec 1.2 APIs
- Kubernetes 1.18, 1.19, 1.20, 1.21 support
- Kubernetes Cluster multi tenancy target masking using dedicated initiator group
- Dynamic creation and deletion of volumes
- Dynamic volume attach and detach
- Block access support
- CHAP support for iSCSI sessions
- Clone volume (when source volume is from the same Datastore)
- PV support with different filesystems (Ext4, Ext3, XFS)
- Volume space statistics reporting per CSI specs
- Multi-writer support (ReadWriteMany) for Block Mode only
- Volume resize support for block mode volumes and ext3, ext4 filesystem volumes (expansion)
- Software encrypted volumes (requires HXDP 5.0(2a) or higher)
- Volume snapshots (requires HXDP 5.0(2a) or higher)
- CHAP protection for volumes (requires HXDP 5.0(2a) or higher)

For more information, see Cisco Hyperflex CSI Operator page on catalog.redhat.com.

Cisco Intersight

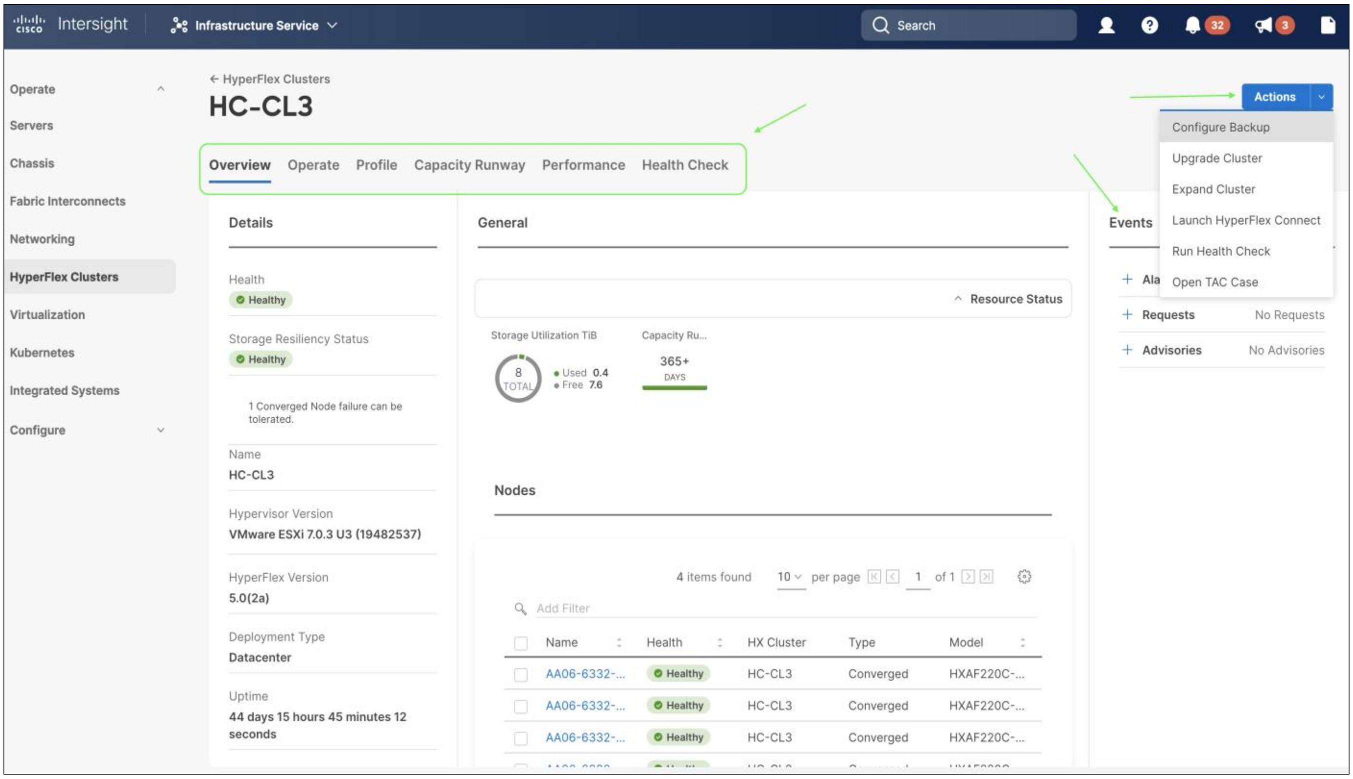
As applications and data become more distributed from core data center and edge locations to public clouds, a centralized management platform is essential. IT agility will be struggle without a consolidated view of the infrastructure resources and centralized operations. Cisco Intersight provides a cloud-hosted, management and analytics platform for all HyperFlex, Cisco UCS and other supported third-party infrastructure across the globe. It provides an efficient way of deploying, managing, and upgrading infrastructure in the data center, ROBO, edge, and co-location environments.



Cisco Intersight provides:

- **No Impact Transition:** Embedded connector (Cisco HyperFlex, Cisco UCS) will allow customers to start consuming benefits without forklift upgrade
- **SaaS/Subscription Model:** SaaS model provides for centralized, cloud-scale management and operations across hundreds of sites around the globe without the administrative overhead of managing the platform.
- **Enhanced Support Experi**ence: Hosted platform allows Cisco to address issues platform-wide and experience extends into TAC supported platforms
- **Unified Management**: Single pane of glass, consistent operations model, and experience for managing all systems and solutions
- **Programmability:** End to end programmability with native API, SDK's and popular DevOps toolsets will enable customers to consume natively.
- **Single point of automation:** Automation using Ansible, Terraform and other tools can be done through Intersight for all systems it manages.
- **Recommendation Engine:** Our approach of visibility, insight and action powered by machine intelligence and analytics provide real-time recommendations with agility and scale. Embedded recommendation platform with insights sourced from across Cisco install base and tailored to each customer

In this solution, Cisco Intersight unifies and simplifies the hybrid cloud operations of Cisco HyperFlex clusters wherever they are deployed. The life cycle management capabilities that Intersight offers specifically for HyperFlex systems are shown below.



HC-CL3

Details

Health: Healthy

Storage Resiliency Status: Healthy

1 Converged Node failure can be tolerated.

Name: HC-CL3

Hypervisor Version: VMware ESXi 7.0.3 U3 (19482537)

HyperFlex Version: 5.0(2a)

Deployment Type: Datacenter

Uptime: 44 days 15 hours 45 minutes 12 seconds

General

Used 0.4
Free 7.6

Nodes

4 items found 10 per page 1 of 1

Add Filter

Name	Health	HX Cluster	Type	Model
AA06-6332-...	Healthy	HC-CL3	Converged	HXAF220C-...
AA06-6332-...	Healthy	HC-CL3	Converged	HXAF220C-...
AA06-6332-...	Healthy	HC-CL3	Converged	HXAF220C-...

- Deploy HyperFlex standard/data center cluster including DC-no-FI configurations, and edge clusters
- Full-stack upgrade of a HyperFlex cluster from HyperFlex/Cisco UCS server firmware, HXDP software to VMware vSphere
- Monitor the status of all actions initiated via Intersight (for example, HyperFlex install)
- Get security alerts, software advisories with recommendations, alarms, and other notifications
- Detailed hardware and software component inventory for each system, including hosts, virtual machines, data stores, disks, and encryptions key management.
- Storage management through features such as create, delete, mount, and unmount data stores
- Tools for Hardware Compatible Checks (HCL), capacity planning, monitoring storage performance (IOPs, throughput latency), and health checks.
- Launch vKVM, Cisco UCS Manager, HyperFlex Connect to access individual cluster management tools.
- Provision backup, manage encryptions key, open support cases that will be populated with serial numbers of servers in the clusters, and expand clusters
- Customizable dashboard with pre-configured widgets to view the information that is of interest to the administrator

For more information, go to the Cisco Intersight product page on cisco.com.

Red Hat OpenShift Container Platform (OCP)

Enterprises are modernizing their applications using scalable, cloud-native architectures and technologies such as containers and micro-services that can run anywhere, from private to public to hybrid clouds. Kubernetes (K8s) is an open-source orchestration system that automates the deployment, scaling, and management of containerized applications. Kubernetes is a project within Cloud Native Computing Foundation (CNCF); it is the de facto industry standard for container orchestration. Kubernetes projects have over 3 million contributions from a large open-source community that includes companies such as Red Hat and Cisco. Red Hat is also one of the top contributors to the Kubernetes project.

Red Hat OpenShift

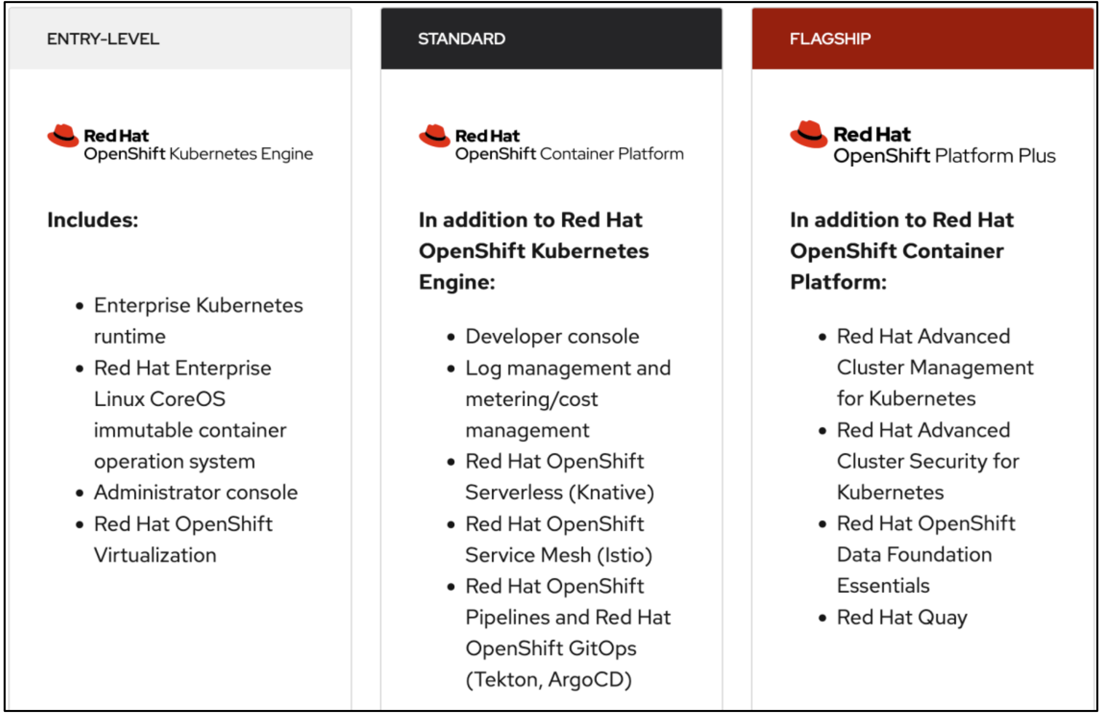
Red Hat OpenShift is a commercially packaged product derived from the upstream open-source Kubernetes. OpenShift goes beyond standard Kubernetes to provide a to provide a broader cloud-native ecosystem of tools and extensions that make it easier to design, develop, test, and operate a cloud-native environment. OpenShift provides extensions such as built-in security, a user-friendly dashboard, integrated image registries, an embedded operator hub, and integrated CI/CD pipelines. It is a secure Enterprise platform with certified third-party integrations, testing, and support from Red Hat.

Red Hat OpenShift is a suite of three products:

- OpenShift Kubernetes Engine (OKE)
- OpenShift Container Platform (OCP)
- OpenShift Container Platform Plus

Each product includes an Enterprise-class container orchestration system bundled with different value-added components as shown in Figure 4.

Figure 4. Red Hat OpenShift Product Suite



ENTRY-LEVEL	STANDARD	FLAGSHIP
Red Hat OpenShift Kubernetes Engine	Red Hat OpenShift Container Platform	Red Hat OpenShift Platform Plus
Includes:	**In addition to Red Hat OpenShift Kubernetes Engine:**	**In addition to Red Hat OpenShift Container Platform:**
• Enterprise Kubernetes runtime • Red Hat Enterprise Linux CoreOS immutable container operation system • Administrator console • Red Hat OpenShift Virtualization	• Developer console • Log management and metering/cost management • Red Hat OpenShift Serverless (Knative) • Red Hat OpenShift Service Mesh (Istio) • Red Hat OpenShift Pipelines and Red Hat OpenShift GitOps (Tekton, ArgoCD)	• Red Hat Advanced Cluster Management for Kubernetes • Red Hat Advanced Cluster Security for Kubernetes • Red Hat OpenShift Data Foundation Essentials • Red Hat Quay

Note: This solution uses Red Hat OpenShift Container Platform, but Enterprises can also Red Hat OpenShift Container Platform Plus to leverage advanced capabilities.

Red Hat OpenShift Container Platform

Red Hat OCP is a secure, enterprise-class, Kubernetes platform for building, deploying, and managing cloud-native environments and applications across a hybrid cloud. OCP provides enterprise-grade Kubernetes but with developer-focused (for example, a developer console, service mesh) and operational (for example, log and cost management, CI/CD pipelines, GitOps) capabilities and tools to support an Enterprise's cloud-native efforts. Red Hat OCP uses Red Hat Enterprise Linux CoreOS (RHCOS) as a foundation for running and managing containers using container technologies such as **CRI-O** and **runc**. OCP is designed to provide consistent developer and operator experience across a hybrid environment.

Some of the capabilities in Red Hat OCP include:

- **Automated deployment** of OCP clusters on-prem (bare metal, VMware vSphere, Red Hat OpenStack® Platform, Red Hat Virtualization) and in public clouds (Amazon Web Services, Google Cloud Platform, IBM Cloud, Microsoft Azure).
- **Automated upgrades** of OCP clusters with over-the-air upgrades using web console or OpenShift CLI (**oc**)
- **Add services with push-button ease** – Once a cluster is deployed, Red Hat OpenShift uses Kubernetes Operators to deploy additional capabilities and services on the cluster. Red Hat and community supported operators are available in the embedded Operator Hub and can be deployed with the click of a button.
- **Multi-cluster management** using Red Hat's cloud-based Hybrid Cloud Console or enterprise-managed Advance Cluster Management (ACM) provides a consolidated view of all clusters, with the ability to easily access and use other K8s technologies and services. OCP clusters can also be individually managed using a web-based cluster console or APIs.
- **Persistent storage support** – OCP provides support for a broad range of eco-system storage partners including the HyperFlex storage used in this solution.
- **Scalability** – OCP can scale to thousands of instances across hundreds of nodes in seconds as needed.
- **Automate** container and application builds, deployments, scaling, cluster management, and more with ease.
- **Self-service provisioning** – Developers can quickly and easily create applications on demand from the tools they use most, while operations retain full control over the entire environment.
- **Source-to-image deployment** – OCP provides a toolkit and workflow for producing ready-to-run images by injecting source code into a container and letting the container prepare that source code for execution.

For more information, see: Red Hat OpenShift Container Platform product page on redhat.com.

Red Hat Hybrid Cloud Console

Red Hat Hybrid Cloud Console is a centralized, SaaS-based management console for managing multiple OCP clusters. It is used in this solution to provide consistent container management across a hybrid environment. The SaaS model enables Enterprises to develop, deploy, and innovate faster across multiple infrastructures and quickly take advantage of new capabilities without the overhead of managing the tool. The console gives Enterprises more control and visibility as environments grow and scale. The Hybrid Cloud Console also provides tools to proactively address issues, open and manage support cases, manage cloud costs, subscriptions, etc.

For more information, see: Red Hat Hybrid Cloud Console product page on redhat.com

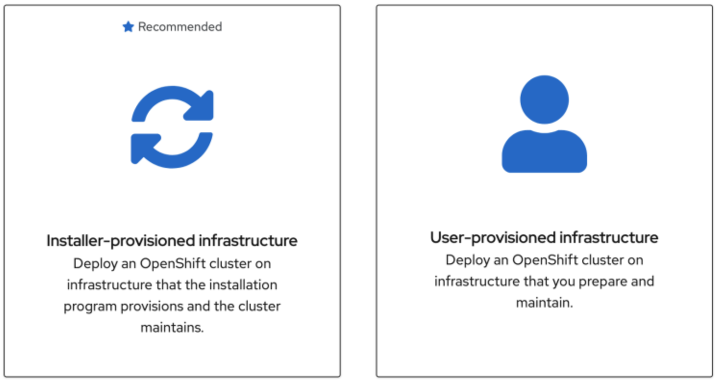
Consumption Models

Red Hat OpenShift is available as a managed service by Red Hat and the cloud provider or as self-managed service where the Enterprise manages and maintains the OCP cluster. Unlike other managed Kubernetes services, including ones offered by the public cloud providers, Red Hat OCP managed and hosted on the same public provider environments provides support for the full-stack rather than just Kubernetes, with consistent experience across the hybrid environment. Red Hat OpenShift is a complete, production-ready application platform with additional services such as CI/CD pipelines, monitoring, security, registry, service mesh etc. included on top of upstream Kubernetes. The managed cloud-hosted services include Red Hat OpenShift

Service on AWS, Microsoft Azure Red Hat OpenShift, Red Hat OpenShift Dedicated on Google Cloud or AWS, and Red Hat OpenShift on IBM Cloud.

Installation Options

The Red Hat OpenShift Container Platform installer can install and deploy a cluster using either Installer-Provisioned infrastructure (IPI) or the User-Provisioned infrastructure (UPI) methods.

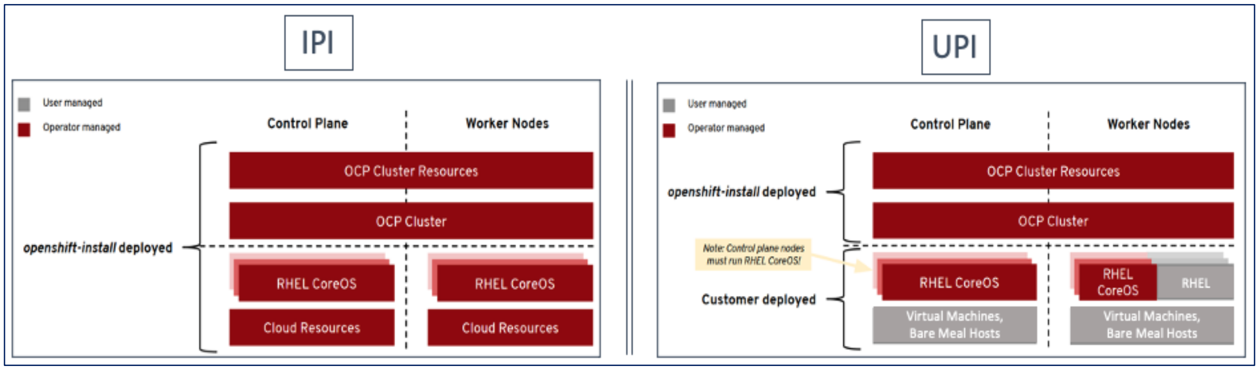


★ Recommended

Installer-provisioned infrastructure
Deploy an OpenShift cluster on infrastructure that the installation program provisions and the cluster maintains.

User-provisioned infrastructure
Deploy an OpenShift cluster on infrastructure that you prepare and maintain.

The IPI installation is an opinionated, prescriptive approach but it is the quickest way to deploy a cluster. The install implements best practices and is fully automated. The install takes approximately 45minutes and can be deployed with minimal understanding of the underlying infrastructure. In the UPI method, the administrator is responsible for loading the operating system (RHCOS, RHEL) and bringing up the different nodes (for example, bootstrap, master, worker) in an OCP cluster.

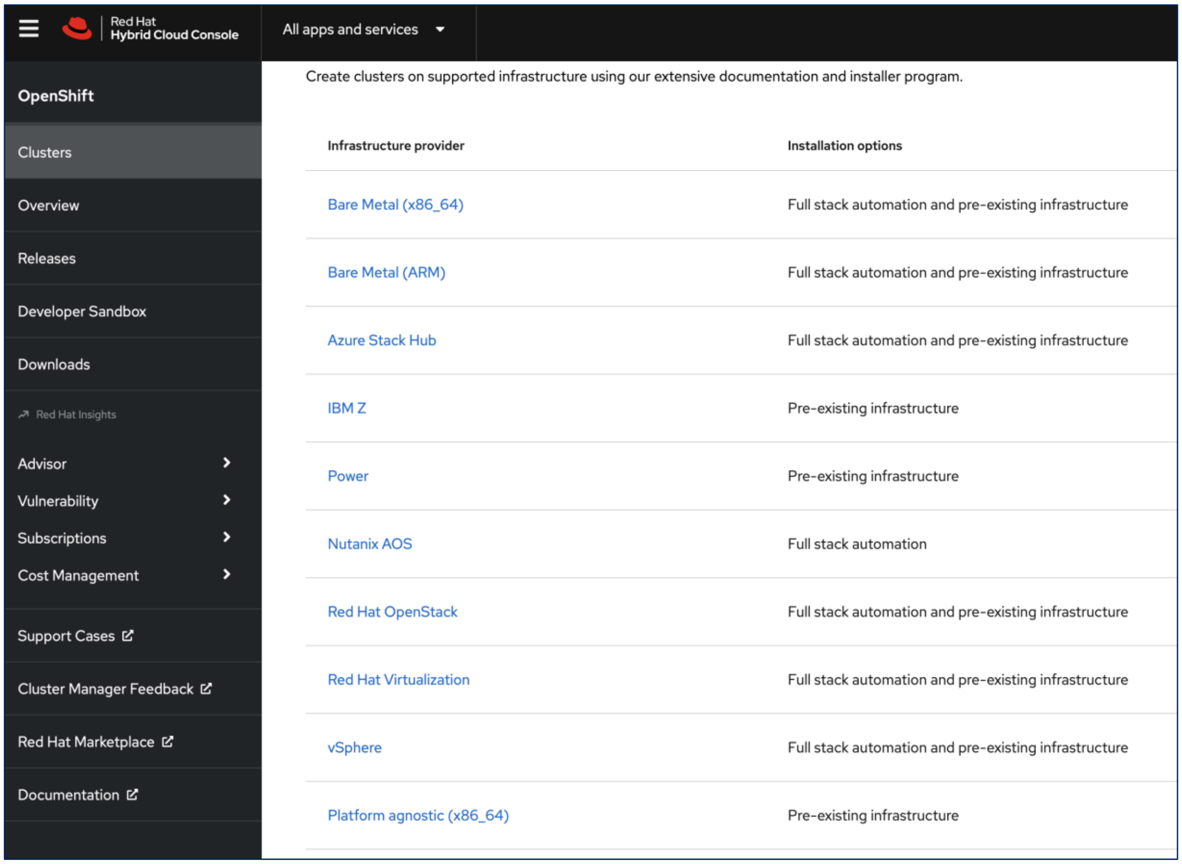


In on-prem environments, a third option, Assisted Installer, is also supported where the user provisions and maintains the infrastructure but with step-by-step guidance from the installer.

Both installation mechanisms offer either the default install or customizations. For example, IPI Install on AWS can deploy a cluster in a restricted or private network, existing virtual private cloud (VPC), government, or top-secret region, with network customizations or a fully custom install where the installer still provisions the infrastructure.

The Red Hat OCP installer is flexible and provide several options for deploying a cluster in a variety of infrastructure environments. The Hybrid Cloud Console supported environments at the time of writing of this document are:

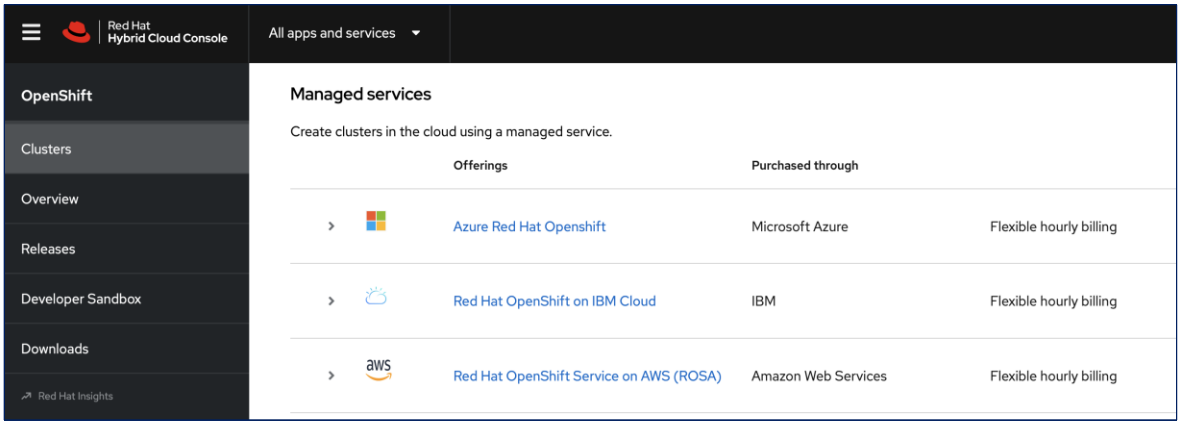
- On-prem (Bare Metal, Virtual, Private cloud)



Create clusters on supported infrastructure using our extensive documentation and installer program.

Infrastructure provider	Installation options
Bare Metal (x86_64)	Full stack automation and pre-existing infrastructure
Bare Metal (ARM)	Full stack automation and pre-existing infrastructure
Azure Stack Hub	Full stack automation and pre-existing infrastructure
IBM Z	Pre-existing infrastructure
Power	Pre-existing infrastructure
Nutanix AOS	Full stack automation
Red Hat OpenStack	Full stack automation and pre-existing infrastructure
Red Hat Virtualization	Full stack automation and pre-existing infrastructure
vSphere	Full stack automation and pre-existing infrastructure
Platform agnostic (x86_64)	Pre-existing infrastructure

- Public Cloud – Managed service

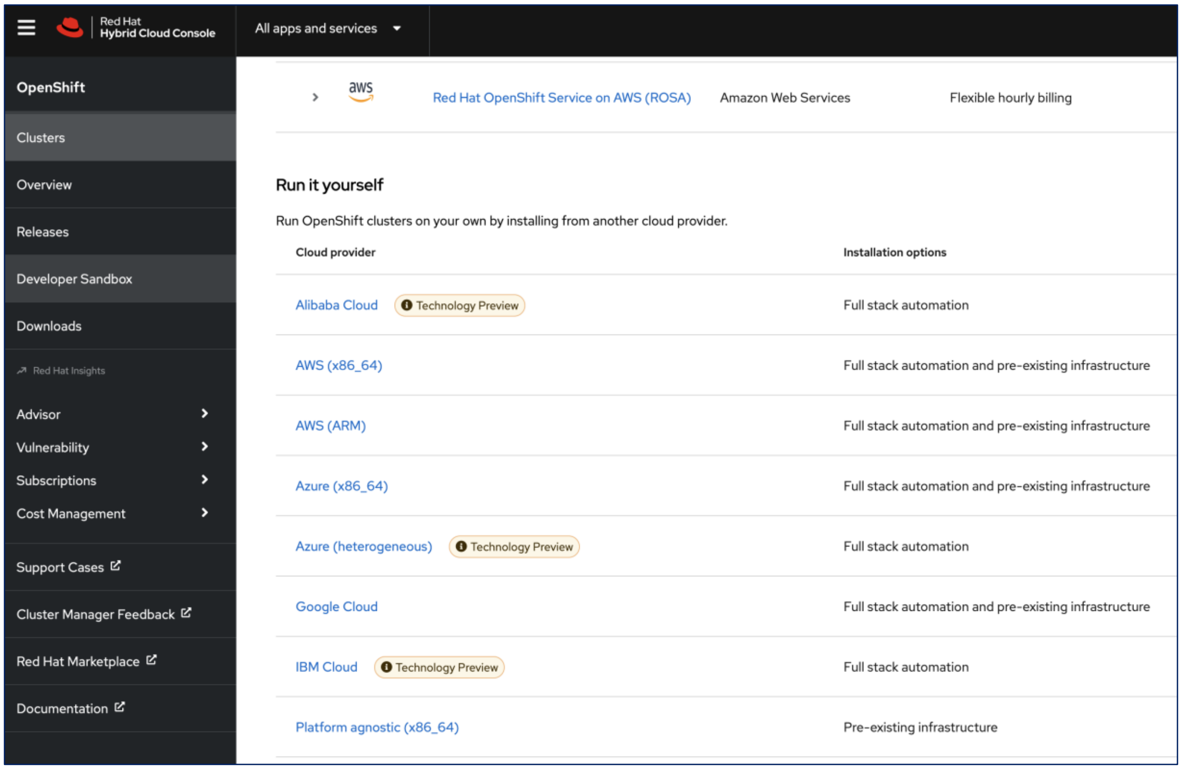


Managed services

Create clusters in the cloud using a managed service.

Offerings	Purchased through	
Azure Red Hat Openshift	Microsoft Azure	Flexible hourly billing
Red Hat OpenShift on IBM Cloud	IBM	Flexible hourly billing
Red Hat OpenShift Service on AWS (ROSA)	Amazon Web Services	Flexible hourly billing

- Public Cloud – Self-managed



Red Hat OpenShift Service on AWS (ROSA)	Amazon Web Services	Flexible hourly billing

Run it yourself

Run OpenShift clusters on your own by installing from another cloud provider.

Cloud provider	Installation options
Alibaba Cloud Technology Preview	Full stack automation
AWS (x86_64)	Full stack automation and pre-existing infrastructure
AWS (ARM)	Full stack automation and pre-existing infrastructure
Azure (x86_64)	Full stack automation and pre-existing infrastructure
Azure (heterogeneous) Technology Preview	Full stack automation
Google Cloud	Full stack automation and pre-existing infrastructure
IBM Cloud Technology Preview	Full stack automation
Platform agnostic (x86_64)	Pre-existing infrastructure

Note: For a complete list of all environments supported using the IPI and UPI installation methods, please refer to the Red Hat OCP documentation for a given release.

Red Hat Enterprise Linux CoreOS (RHCOS)

RHCOS is a light-weight, container-specific operating system, specifically designed for running container workloads. It is based on the secure, enterprise-grade Red Hat Enterprise Linux (RHEL). RHCOS is the default operating system on all Red Hat OCP cluster nodes, and the only operating system supported on control/master nodes. The recommended IPI installation method deploys RHCOS on all cluster machines, including compute/worker nodes though RHCOS and RHEL are supported. RHCOS is tightly controlled, allowing only a few system settings to be modified using the Ignition configuration files. RHCOS is designed to deploy an OCP cluster with minimal user configuration and once the cluster is deployed, the cluster will fully manage the RHCOS subsystem.

RHCOS includes:

- Ignition – for initial bootup configuration and disk related tasks on OCP cluster nodes
- CRI-O – Container Engine running on OCP cluster nodes
- Kubelet – Kubernetes service running on OCP cluster nodes
- Set of container tools

Ignition serves as a first boot system configuration utility for initially bringing up and configuring the nodes in the OCP cluster. It also creates and formats disk partitions, writes files, creates file systems and directories, configures users etc. During a cluster install, the control/master nodes get their configuration file from the temporary bootstrap machine used during install, and the worker nodes get theirs from the control/master nodes. Ignition is designed to initialize systems and all subsequent configuration done using the Machine Config Operator in OCP.

CRI-O is a stable, standards-based, lightweight container engine for Kubernetes that runs and manages the containers. CRI-O implements the Kubernetes Container Runtime Interface (CRI) for running Open Container Initiative (OCI) compliant runtimes. OCP uses the default OCI runtime, **runc** as the container runtime. CRI-O has a small footprint and a smaller attack surface, making it more secure. OCP uses the default OCI runtime - **runc**. CRI-O, along with RHCOS provides capabilities such as starting, stopping, and restarting containers. CRI-O is an independent project in Cloud Native Computing Foundation (CNCF) that also supports and maintains Kubernetes.

Kubelet is a Kubernetes service running on every worker node in the cluster. It communicates with the control plane components and processes requests for running, stopping, and managing container workloads.

Container Tools: RHCOS also includes a set of container tools (for example, **podman, skopeo, crictl**) for managing containers and container images such as start, stop, run list, remove containers and copy, authentication, sign images. RHCOS uses the **rpm-ostree** system to pull, extract, and write container images to disk for cluster updates.

RHCOS is deployed using configurations in ignition files. The OCP installer creates the Ignition configuration files necessary to deploy the OCP cluster with RHCOS. The configuration is based on the user provided responses to the installer. These files and images are downloaded and installed on the underlying infrastructure by the installer when using the IPI method. However, when using the UPI method, the user must download the RHCOS images, generate the Ignition configuration files and use them to provision the cluster machines.

Amazon Web Services (AWS)

AWS provides a flexible application computing environment for deploying cloud-native applications. Red Hat OCP cluster on AWS can accelerate application development and delivery by providing a consistent experience for developers and operators across both on-prem and public cloud. AWS is globally available, enabling Enterprises to extend their enterprise deployments to a variety of AWS regions as needed. Red Hat OCP cluster nodes can also be distributed across multiple AWS Availability Zones (AZ) to ensure availability. OCP is also supported on other cloud providers such as Google Cloud Platform, Microsoft Azure, and IBM Cloud. OCP is available as a managed service on AWS, Red Hat OCP Service on AWS (ROSA) and as self-managed. This solution uses the self-managed service and the IPI installation method that was used on-prem. The automated IPI installation method uses several AWS services such as Route 53, DHCP, load balancers, Virtual Private Cloud (VPC) and EC2 instances that are deployed or used as a part of the installation process. Transit Gateways (TGW) attached to the VPC provide connectivity to on-prem resources and services, including K8s clusters and application workloads.

A VPC in AWS provides an isolated virtual networking environment on a shared infrastructure where users can deploy resources to support applications workloads. Enterprises can deploy VPCs in AWS cloud and connect them directly to the on-prem datacenter to enable connectivity between applications, services, and resources in each environment. One mechanism for enabling this connectivity is to use a Site-to-Site VPN to establish an IPsec VPN tunnel between the two locations.

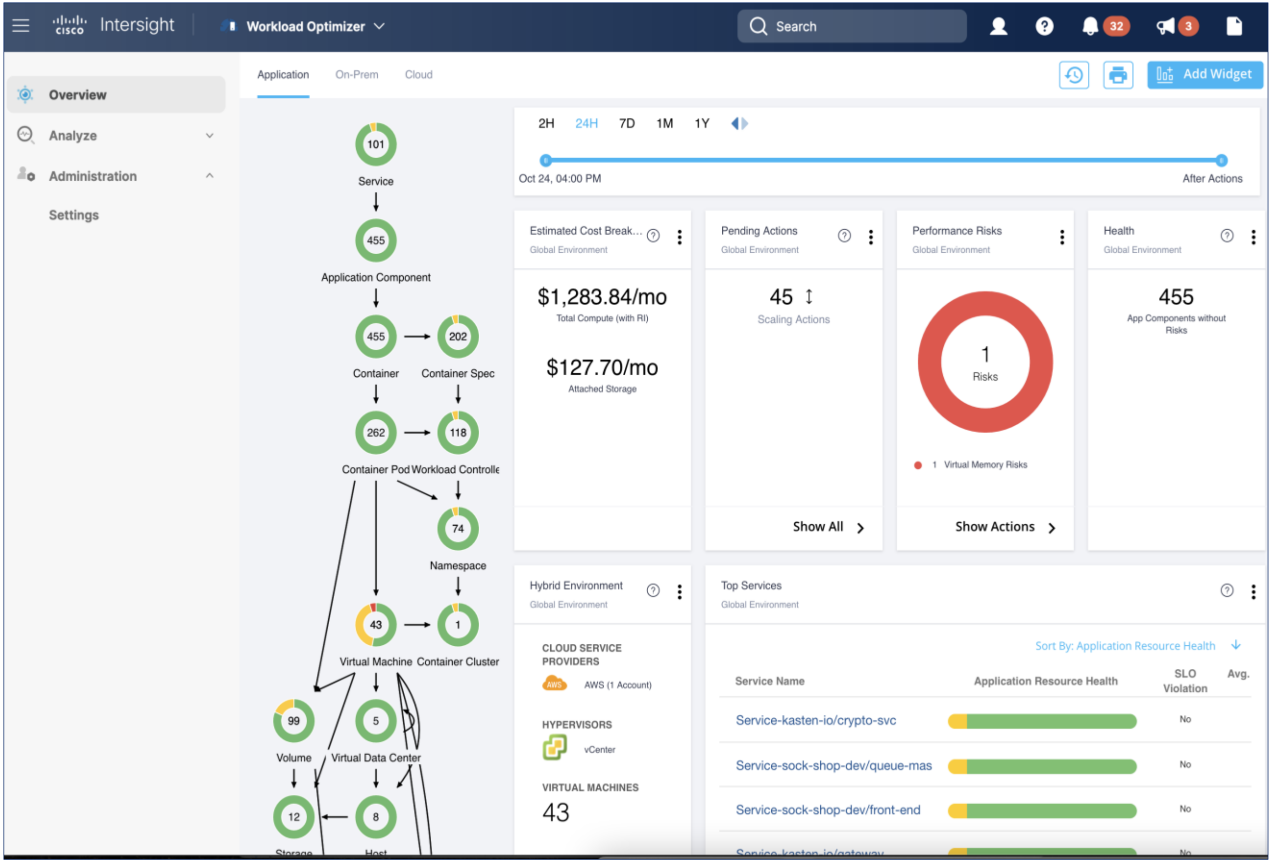
Cisco Intersight Workload Optimizer (IWO)

Cisco IWO provides resource management for cloud-native workloads in a hybrid cloud deployment. It takes a top-down approach to monitoring all resources in a hybrid cloud deployment to ensure that applications are healthy, and the environment is operating in an optimal manner. IWO uses real-time AI-powered analytics to continuously monitor and analyze application workloads and their resource consumption. IWO uses the historical and real-time data to make actionable recommendations (for example, provision additional resources) that can be automatically implemented to ensure application performance before users and services are impacted. IWO provides a cost analysis with each recommendation so that administrators can make better decisions, one that minimizes cloud costs, prevents sprawl, and generally uses resources more efficiently. IWO recommendations can be implemented to consolidate workloads to minimize infrastructure needs. Cisco IWO for planning purposes, to model future workload growth scenarios and to estimate the additional infrastructure that your organization will need and when.

To perform application resource management, Cisco IWO discovers software and hardware components running in your environment and creates an inter-dependency map of all components from an application perspective. It then monitors and analyzes the resources to optimize resource usage while assuring application performance. The inter-dependency map includes all layers of the infrastructure, from the containers and VMs down to the underlying compute, storage, and network infrastructure.

To evaluate and monitor the resource usage of applications, Cisco IWO collects telemetry data from both on-prem and public cloud components. The starting point for this data collection are the targets claimed in Intersight. Intersight supports a broad-range of Cisco and third-party products to provide a holistic top-down view of all infrastructure entities or components for a given application or in each location (on-prem, public). For cloud-native environments, Cisco IWO discovers and monitors the following container specific entities: Service, Application Components, Container, Container Pod, Container Spec, Workload Controller, Container Cluster, Namespace, Virtual Machine, and Volume. Figure 5 shows the inter-dependency map based on the discovered entities from an application perspective, widgets that show the estimated cost breakdown, pending IWO recommendations and actions, performance risks, overall health of the application and the top application services.

Figure 5. Cisco IWO – Application View



Service Name	Application Resource Health	SLO Violation	Avg.
Service-kasten-io/crypto-svc		No	
Service-sock-shop-dev/queue-mas		No	
Service-sock-shop-dev/front-end		No	
Service-kasten-io/gateway		No	

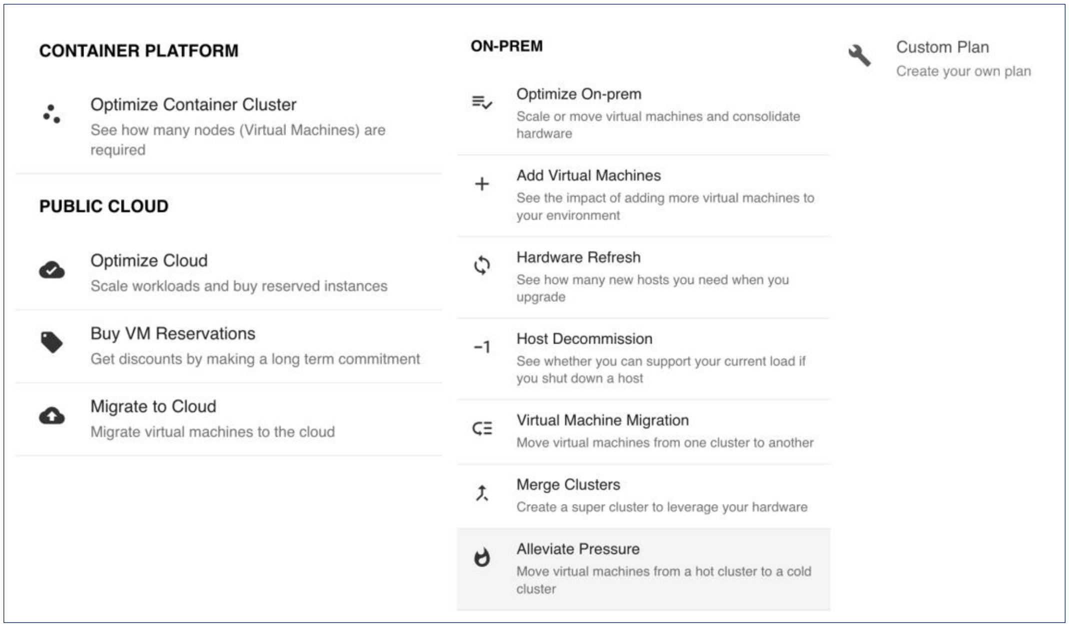
You can also use IWO to run simulations for planning purposes and placement recommendations. You can then schedule and implement the changes at your convenience. Intersight Workload Optimizer supports the following action modes:

- **Recommend** – Recommend the action so a user can execute it from the domain manager rather than from IWO
- **Manual** – Recommend the action, and provide an option to execute that action through the IWO Web GUI interface
- **Automatic** – Execute the action automatically without user involvement. IWO waits five minutes between actions.

Action modes specify the degree of automation that should be used for implementing the actions. The action modes in a policy are used to set the business rule. Some of the action types and actions supported by IWO are:

- **Types**: Placement, Scaling, RI optimization, configuration, start/buy stop, delete
- **Actions**: Provision, Start, Resize, Buy YRI, reconfigure, move/suspend/delete

IWO also provides several pre-defined plans to analyze and optimize various aspects of the deployment. Some of the plans available are:



CONTAINER PLATFORM

Optimize Container Cluster
See how many nodes (Virtual Machines) are required

PUBLIC CLOUD

Optimize Cloud
Scale workloads and buy reserved instances

Buy VM Reservations
Get discounts by making a long term commitment

Migrate to Cloud
Migrate virtual machines to the cloud

ON-PREM

Optimize On-prem
Scale or move virtual machines and consolidate hardware

Add Virtual Machines
See the impact of adding more virtual machines to your environment

Hardware Refresh
See how many new hosts you need when you upgrade

Host Decommission
See whether you can support your current load if you shut down a host

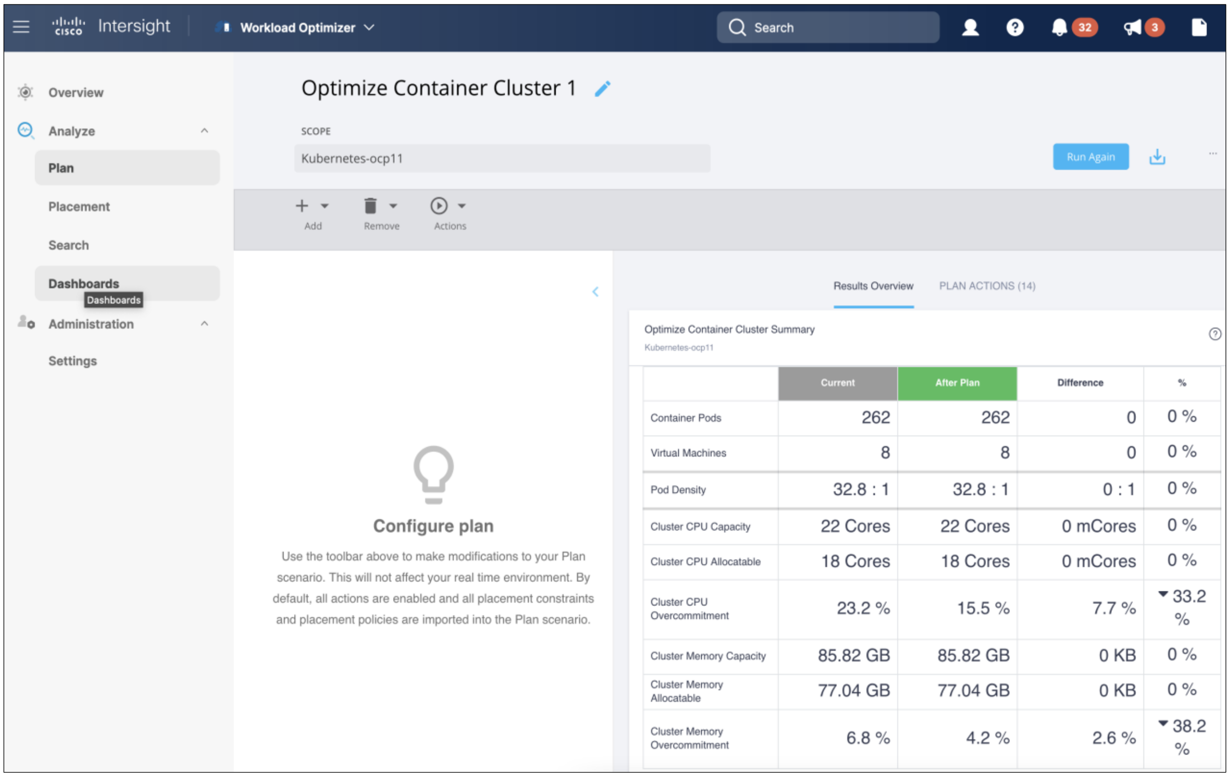
Virtual Machine Migration
Move virtual machines from one cluster to another

Merge Clusters
Create a super cluster to leverage your hardware

Alleviate Pressure
Move virtual machines from a hot cluster to a cold cluster

Custom Plan
Create your own plan

The figure below shows the results of a full container optimization plan that was executed and the actions that IWO recommended as a result.



Optimize Container Cluster 1

SCOPE

Kubernetes-ocp11

Configure plan

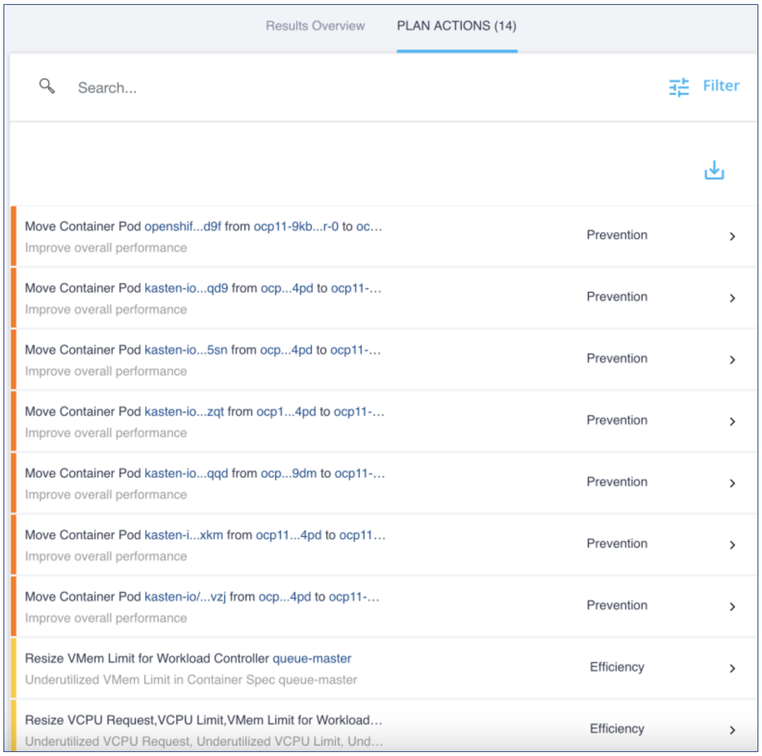
Use the toolbar above to make modifications to your Plan scenario. This will not affect your real time environment. By default, all actions are enabled and all placement constraints and placement policies are imported into the Plan scenario.

Optimize Container Cluster Summary

Kubernetes-ocp11

	Current	After Plan	Difference	%
Container Pods	262	262	0	0 %
Virtual Machines	8	8	0	0 %
Pod Density	32.8 : 1	32.8 : 1	0 : 1	0 %
Cluster CPU Capacity	22 Cores	22 Cores	0 mCores	0 %
Cluster CPU Allocatable	18 Cores	18 Cores	0 mCores	0 %
Cluster CPU Overcommitment	23.2 %	15.5 %	7.7 %	▾ 33.2 %
Cluster Memory Capacity	85.82 GB	85.82 GB	0 KB	0 %
Cluster Memory Allocatable	77.04 GB	77.04 GB	0 KB	0 %
Cluster Memory Overcommitment	6.8 %	4.2 %	2.6 %	▾ 38.2 %

Some of the actions recommended by the execution of this plan are provided below as an example.



Action	Category
Move Container Pod openshif...d9f from ocp11-9kb...r-0 to oc… Improve overall performance	Prevention
Move Container Pod kasten-io...qd9 from ocp...4pd to ocp11-… Improve overall performance	Prevention
Move Container Pod kasten-io...5sn from ocp...4pd to ocp11-… Improve overall performance	Prevention
Move Container Pod kasten-io...zqt from ocp1...4pd to ocp11-… Improve overall performance	Prevention
Move Container Pod kasten-io...qqd from ocp...9dm to ocp11-… Improve overall performance	Prevention
Move Container Pod kasten-i...xkm from ocp11...4pd to ocp11… Improve overall performance	Prevention
Move Container Pod kasten-io/...vzj from ocp...4pd to ocp11-… Improve overall performance	Prevention
Resize VMem Limit for Workload Controller queue-master Underutilized VMem Limit in Container Spec queue-master	Efficiency
Resize VCPU Request,VCPU Limit,VMem Limit for Workload… Underutilized VCPU Request, Underutilized VCPU Limit, Und…	Efficiency

Note: The plan was executed on an under-utilized OCP cluster in Cisco's internal labs so many of the recommendations are to reduce the vCPU or vMem that the containers can use.

Cisco IWO also provides these recommendations without running plan based on its ongoing monitoring and analysis. These can be viewed selecting and browsing to the entity directly from the inter-dependency map.

For more information, see: Cisco Intersight Workload Optimizer product page on cisco.com

Red Hat Ansible

Ansible is an open-source tool for Infrastructure as Code (IaC). Ansible is also used for configuration management and application software deployment. Ansible is designed to be agentless, secure, and simple. Ansible available in Red Hat's Ansible Automation Platform is part of a suite of tools supported by Red Hat. Ansible manages endpoints and infrastructure components in an inventory file, formatted in YAML or INI. The inventory file can be a static file populated by an administrator or dynamically updated. Passwords and other sensitive data can be encrypted using Ansible Vault. Ansible uses playbooks to orchestrate the provisioning. Playbooks are written in human readable YAML format that is easy to understand. Ansible playbooks are executed against a subset of components in the inventory file. From a control machine, Ansible uses SSH or Windows Remote Management to remotely configure and provision target devices in the inventory based on the playbook tasks.

Ansible is used to provision Cisco HyperFlex VSI and Cisco ACI fabric infrastructure in the solution. Ansible also provides a robust container and Kubernetes management, including Red Hat OpenShift Container Platform that Enterprises can leverage for their automation efforts.

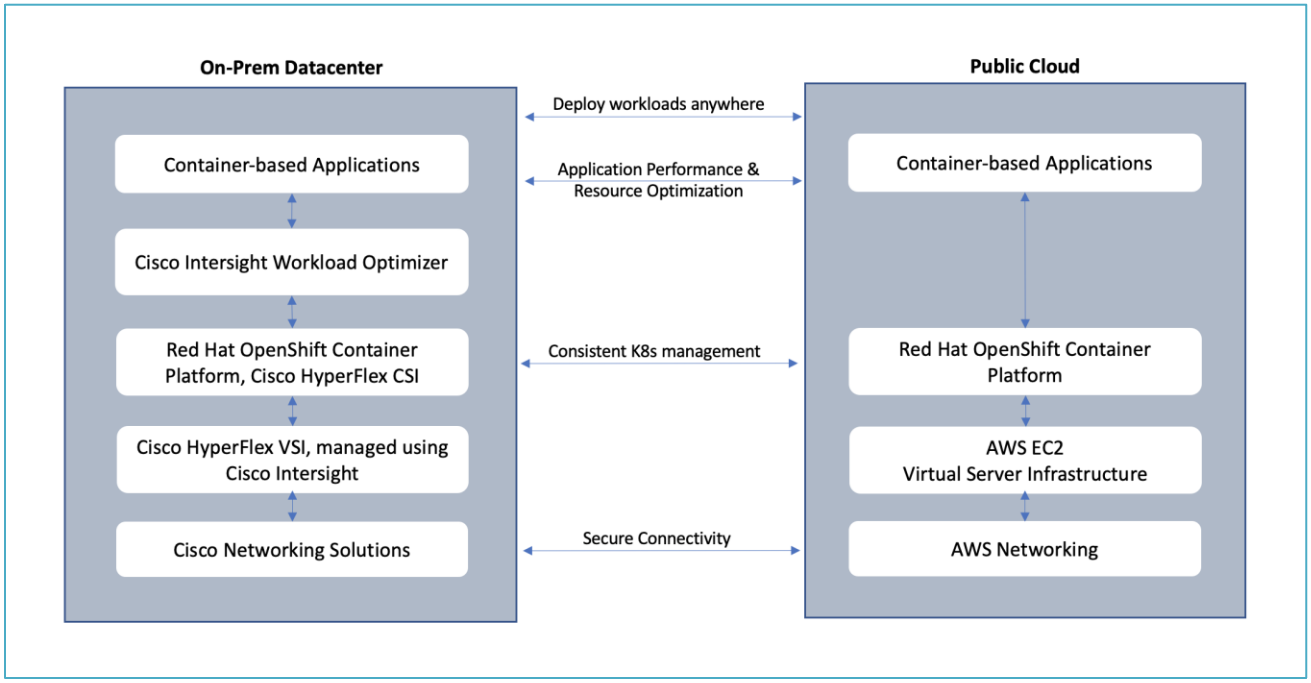
Solution Design

This chapter contains the following:

- Overview
- Solution Topology
- Design Requirements
- Design Details
- Requirements

Overview

At a high level, the hybrid cloud infrastructure design in this solution consists of an on-prem datacenter, public cloud infrastructure, and a secure network interconnecting the two environments, as shown below:


On-Prem Datacenter
Container-based Applications
Cisco Intersight Workload Optimizer
Red Hat OpenShift Container Platform, Cisco HyperFlex CSI
Cisco HyperFlex VSI, managed using Cisco Intersight
Cisco Networking Solutions
Deploy workloads anywhere
Application Performance & Resource Optimization
Consistent K8s management
Secure Connectivity
Public Cloud
Container-based Applications
Red Hat OpenShift Container Platform
AWS EC2 Virtual Server Infrastructure
AWS Networking

Virtual Server Infrastructure (VSI)

The on-prem Virtual Server Infrastructure in the solution consists of:

- Four (4) node HyperFlex standard (or HyperFlex Datacenter) cluster as an application cluster for running cloud-native workloads on a Red Hat OCP cluster. The cluster is deployed and managed from the cloud using Cisco Intersight.
- Four (4) node HyperFlex standard cluster as a management cluster for hosting services and management components to support the application cluster. The cluster is deployed and managed from the cloud using Cisco Intersight. The services deployed include VMware vCenter managing the application cluster, DNS, DHCP and OCP Installer workstation. The management cluster can also host a management OCP cluster to run services and other components. For example, Red Hat's Advanced Cluster Manager requires a seed OCP cluster to run on before it can be used for multi-cluster management (see Red Hat OCP in Technology Review section)

The public Virtual Server Infrastructure in the solution consists of:

- Amazon Elastic Compute Cloud (Amazon EC2) instances in the Amazon Web Services (AWS) Cloud provide the virtual server infrastructure in the public cloud. Red Hat OCP clusters are deployed on EC2 instances in a Virtual Private Cloud (VPC). The OCP cluster in the VPC are distributed across 3 x Availability Zones (AZ) in the region for resiliency.

Network Connectivity

- Two redundant IPsec VPN connections provide secure connectivity between the cloud-native environments. The VPN connections are between 2 x CSR1000v routers on-prem and transit gateway routers in the public cloud.

Kubernetes Infrastructure

- Red Hat OCP cluster(s) provide a Kubernetes environment for cloud-native applications and use cases. The clusters are deployed from the cloud on HyperFlex VSI and on AWS EC2 instances using Red Hat Hybrid Cloud
- HyperFlex CSI provides persistent storage for stateful workloads hosted on HyperFlex VSI

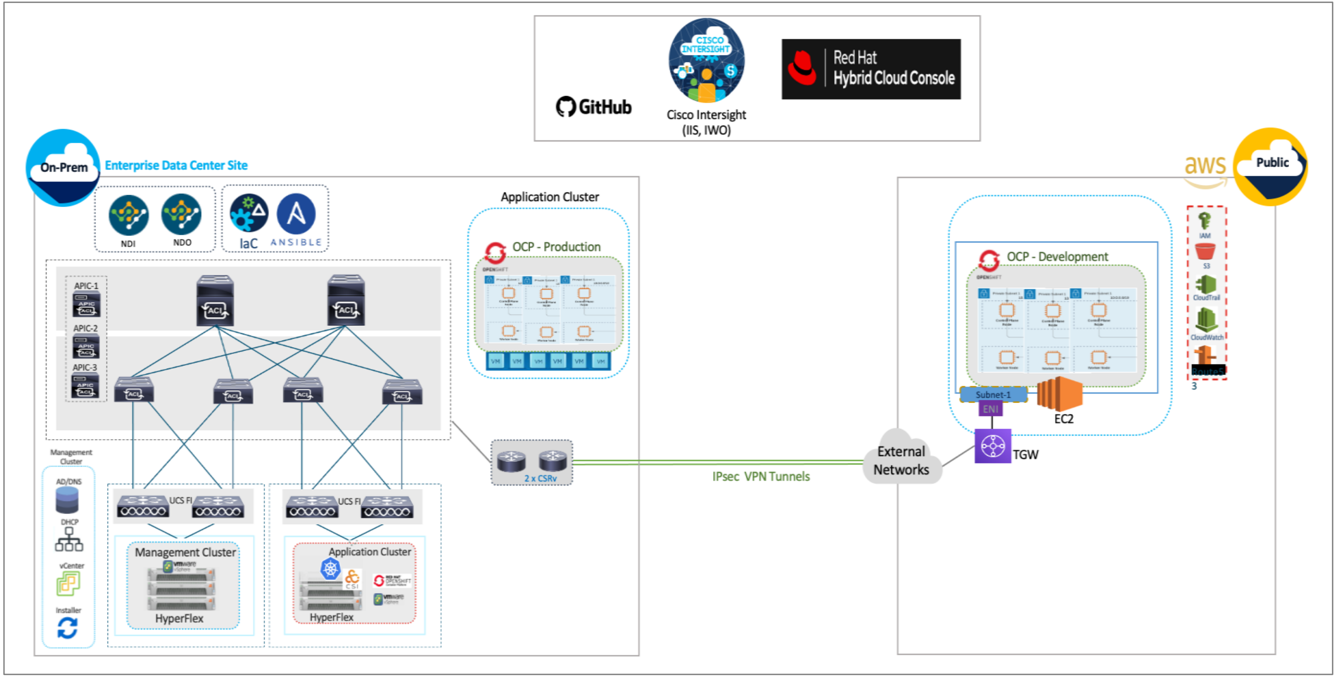
Application Performance

- Cisco Intersight Workload Optimizer ensure application performance. IWO also provides resource monitoring, optimization, and cost management across this environment.

Solution Topology

Figure 6 illustrates the end-to-end solution that was designed, built, and validated in Cisco internal labs.

Figure 6. Solution Topology



Design Requirements

Hybrid Cloud deployments give Enterprises complete flexibility in selecting an optimal location for their workloads based on performance, cost, compliance, and other factors. Some of the design requirements that this hybrid cloud solution addresses are outlined below.

- Operational simplicity and agility with the flexibility to deploy and manage workloads anywhere. The on-prem infrastructure that the Enterprise manages, must be easy to deploy and manage without compromising functionality, scale, or performance.
- The infrastructure must also be available as code for integration into existing Enterprise automation or CI/CD pipelines.
- The solution must continuously monitor and optimize the hybrid environment to ensure application performance and manage cloud costs.

The overall solution also addresses the following high-level design goals:

- Resilient design across all layers of the infrastructure with no single point of failure
- Scalable design with the ability to independently scale compute, storage, and networking as needed
- Modular design with the ability to upgrade or replace components and sub-systems as needed
- Flexible design across all layers of the solution that includes sub-system design, individual components used, and storage configuration and connectivity options
- Operational agility and simplicity through best-of-breed products, SaaS operations, automation, and orchestration tools
- Incorporates technology and product best practices for the different components in the solution

The design also addresses two commonly seen use cases in hybrid cloud use deployments:

- Enable cloud-native deployments anywhere, from on-prem to public cloud, while maintaining a consistent management experience
- Dev/Test for cloud-native workloads where organizations have multiple teams in a CI/CD pipeline that need separate environments, both on-prem and in the public cloud. Development teams may start the work in the cloud, but the application is staged and deployed into production on-prem.

Design Details

The Cisco Validated Design (CVD) in this document provides a foundational infrastructure architecture for a secure, scalable, enterprise-class hybrid cloud solution to run cloud-native workloads. The main aspects of the design are:

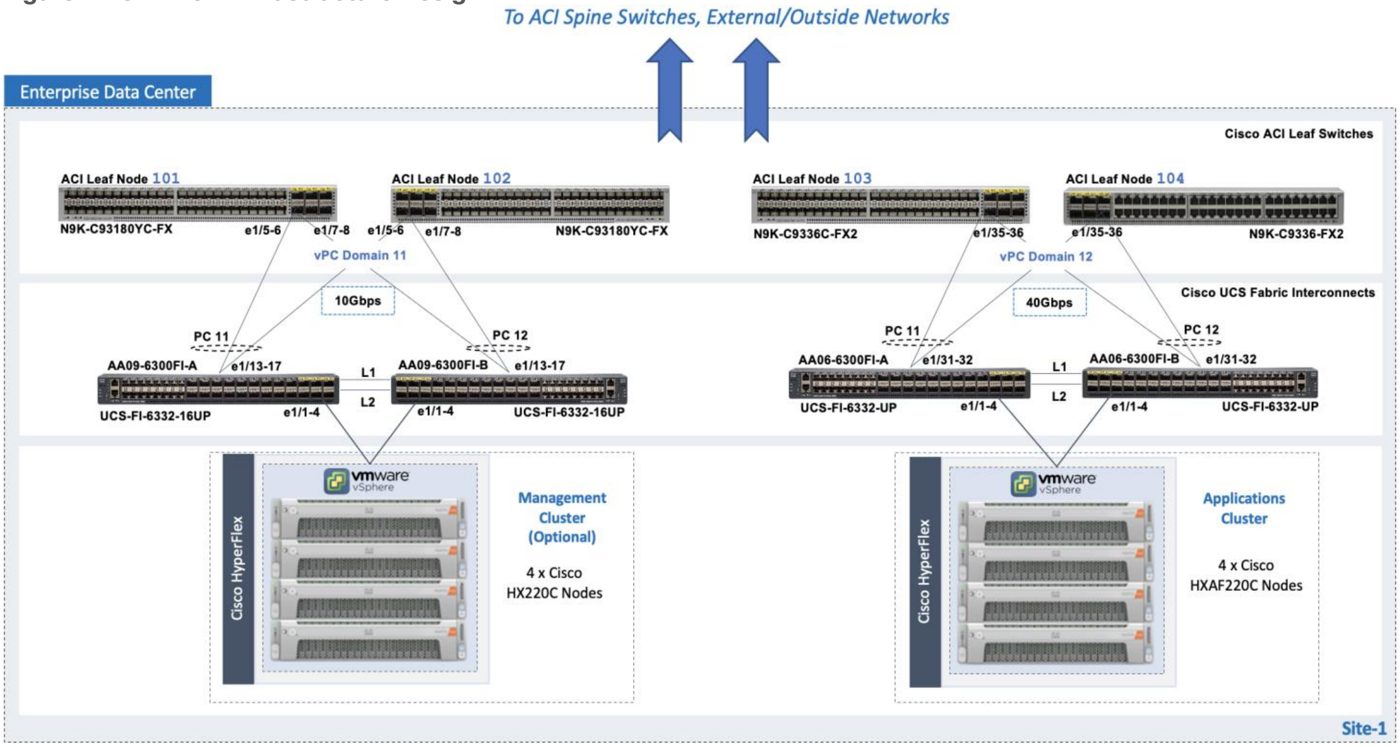
- On-prem infrastructure (compute, storage, virtualization, and network) design
- Infrastructure as Code (IaC)
- Operational ease and consistency as the hybrid model expand to other Enterprise locations
- Secure hybrid cloud connectivity
- Cloud-native Infrastructure design (on-prem, public cloud)
- Persistent Storage for stateful, cloud-native workloads
- Operational ease with consistent cloud-native development and operational experience regardless of the environment
- Application resource monitoring, visibility, and cost management across on-prem and public cloud
- Ensuring application performance in any environment

Infrastructure Design

For cloud-native efforts, Enterprise IT teams need the ability to quickly deploy and manage the infrastructure in the Enterprise data center. The infra in the on-prem location, unlike the public cloud, is typically deployed and managed by the Enterprise. Because of this, IT and DevOps teams need infrastructure that is simple and easy to deploy and operate without compromising performance, scale, or functionality. The data center infrastructure design outlined in this section offers just that.

Figure 7 shows the on-prem infrastructure design in this solution.

Figure 7. On-Prem Infrastructure Design



Cisco HyperFlex

Cisco HyperFlex provides the on-prem virtual server infrastructure in the solution. HyperFlex delivers software-define compute, storage, and virtualization as a fully-integrated product, enabling IT teams to spend less time on infrastructure management and more time on other business initiatives. The design uses HyperFlex clusters, an application cluster for running application workloads, and a management cluster for hosting infrastructure management and other services. The management HyperFlex cluster is optional in the design as the services running on the management cluster can also be deployed on an Enterprise's existing infrastructure. Both are HyperFlex standard (data center) cluster with 4 nodes in each cluster.

The nodes in the cluster connect to a Cisco UCS domain consisting of a pair of Cisco UCS 6300 Series Fabric Interconnects. The management and application clusters attach to different Cisco UCS domains in this solution, but they could also connect to the same UCS domain if needed. A single UCS domain consisting of a pair of Fabric Interconnects can support multiple HyperFlex clusters (and Cisco UCS systems). The exact number depends on the number of servers in a given cluster and the port density of the Fabric Interconnect model chosen.

The data center fabric in the solution is a Cisco ACI fabric. The HyperFlex clusters connect to the ACI fabric through the Cisco UCS 6300 Fabric Interconnects in their UCS domain. Some HyperFlex systems (for example, DC-no-FI, Edge) can also connect directly to the leaf switches in the ACI fabric. However, connecting to Cisco UCS Fabric Interconnects provides some benefits. It serves as an aggregation point for a group of servers with unified management. The two Cisco UCS Fabric Interconnects in a UCS domain form a highly available, unified, low-latency switching fabric capable of handling I/O traffic from hundreds of servers. It provides connectivity between the servers and systems that connect to it and connectivity to the upstream data center fabric to connect to other UCS domains and networks. Cisco UCS Manager (UCSM) provides unified management for all HyperFlex and Cisco UCS systems in that Cisco UCS domain. It runs as embedded software on the Fabric Interconnects. The design can use centralized (via Intersight) and local management (using Cisco UCSM), which can be helpful in troubleshooting situations. The unified fabric can also provide Quality-of-Service (QoS) for the different types of traffic traversing the fabric, including HyperFlex infrastructure (management, storage data, vMotion, replication) and application traffic.

The HyperFlex UCS domains connect to Nexus 9000 series leaf switches in the ACI fabric for northbound connectivity to other networks within and outside the Enterprise and intra-cluster connectivity when traffic needs to go from one Fabric Interconnect (FI-A) to the other (FI-B) within the same domain. In a HyperFlex cluster, this typically happens in failure situations. Each Fabric Interconnect connects to a pair of leaf switches for redundancy. For the application cluster, the uplink connections from the Cisco UCS 6300 Fabric Interconnects are:

- 2 x 40GbE links from FI-A, one to each switch in the leaf switch pair
- 2 x 40GbE links from FI-B, one to each switch in the leaf switch pair

The multiple uplinks provide higher bandwidth and resiliency. Cisco UCS FI and Nexus switches support 802.3ad standards for aggregating links into a port channel (PC) using Link Aggregation Protocol (LACP). The 40Gbps links on the FI side are bundled using a port channel, while the links on the Nexus 9000 leaf switch pair are bundled using a virtual port channel (vPC). The Fabric Interconnects use two port channels to connect to the ACI leaf switch pair - one from each FI. The 2 x 40GbE links on each FI connect to different Nexus switches in the leaf switch pair. In the reverse direction, the ACI leaf switches pair uses two vPCs to connect to the Fabric Interconnects - one to each FI. vPC enables links from two switches to be bundled such that it appears as a “single logical" port channel to a third device (in this case, FI). The vPC design provides higher aggregate bandwidth with both link and node-level redundancy. In this design, the total uplink bandwidth for the application UCS domain, is 160Gbps (40Gbps per link x 2 uplinks per FI x 2FI). You can also add more links to the PC/vPC bundle to increase the uplink bandwidth. The Fabric Interconnects can also use higher speed links to connect to the upstream Nexus switches. The Cisco UCS 6300 series Fabric Interconnects used in this solution support 10G and 40G, but other hardware models are available that support 25G and 100G connectivity. The HyperFlex management cluster is also connected similarly, using a PC/vPC design but uses 4 x 10GbE links for uplink connectivity. The FI uplinks to the ACI fabric operate as trunks, carrying traffic from multiple 802.1Q VLAN IDs to the ACI fabric. The uplinks trunk VLAN traffic for HyperFlex infrastructure (in-band management, vMotion, storage data) and application networks. The VLANs are also configured on the individual virtual NIC (vNIC) templates going to each server in the HX cluster.

The HyperFlex nodes in a cluster are dual-homed to the Fabric Interconnects in the UCS domain for high availability. Each server in the application cluster uses a VIC 1387 adapter with two 40Gbps uplink ports to connect to each FI, resulting in two redundant paths, one through each fabric (FI-A, FI-B). The two uplinks provide each server with 2x40Gbps of uplink bandwidth and redundancy in the event of a failure.

HyperFlex uses an automated installation process to deliver a production-ready HyperFlex VSI cluster in less than an hour. Either an Installer VM (OVF) or Cisco Intersight can be used to initiate the deployment. Intersight provides a simple workflow/wizard to gather input from the user, validate and begin the installation process. HyperFlex systems use a best-practice configuration combined with user-provided inputs to generate a cluster profile. The installer uses this profile to deploy and configure the cluster. The profile can also be cloned and re-used (with minimal changes) to deploy additional clusters as needed. HyperFlex systems typically undergo a factory install process that pre-installs some of the firmware and software required (for example, server firmware, VMware ESXi software), which makes the customer site installation and setup much quicker and easier.

The Cisco UCS Fabric Interconnects (when used) must be claimed as a target for Intersight to discover the servers in the cluster. Intersight and other components will also require connectivity to multiple networks to deploy the cluster. The Cisco ACI fabric provides this connectivity – see the Data Center Fabric – Cisco ACI section for additional details.

The two HyperFlex clusters in the solution are deployed and managed using Cisco Intersight. The installation results in a complete virtual server infrastructure that is ready for deploying a Red Hat OCP cluster and cloud-native workloads. The virtual server infrastructure is managed using VMware vCenter running on the management HyperFlex cluster.

Virtual Networking Design

The automated HyperFlex installation process deploys a VMware vSphere cluster with a pre-defined virtual networking design. The design is identical on all ESXi hosts in the cluster. The design uses four VMware virtual switches (vSwitch) for different types of traffic. The virtual switches deployed by the automated installation process are:

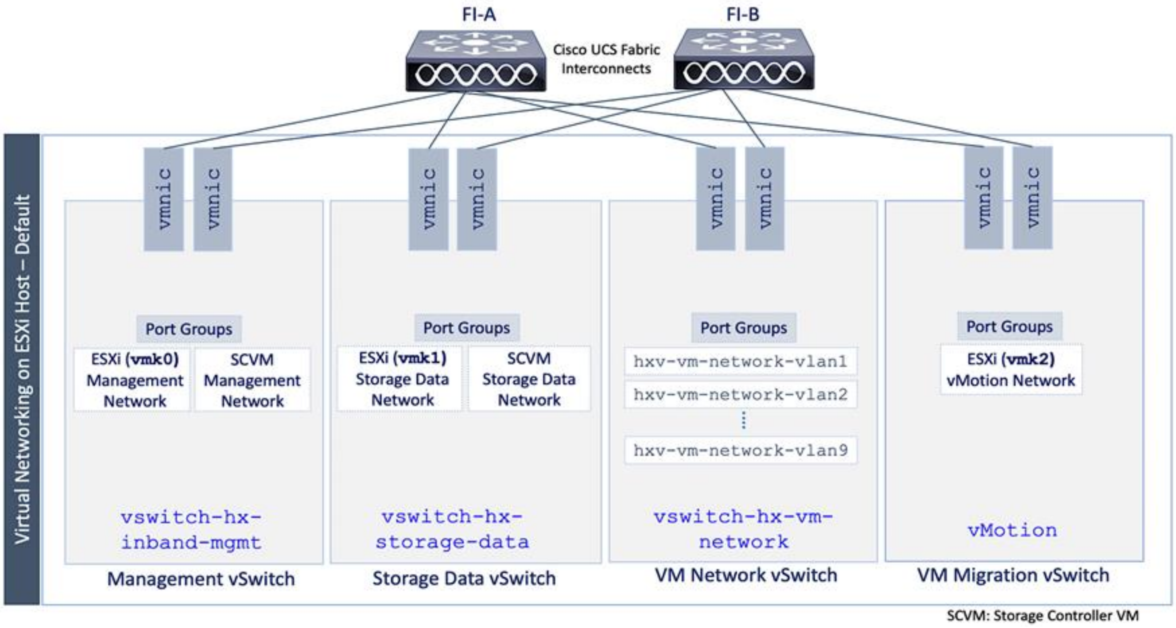
- **vswitch-hx-inband-mgmt**: This is the default ESXi vSwitch0 that is renamed by the ESXi kickstart file as part of the automated installation process. The switch has two uplinks, active on fabric A and standby on fabric B – by default, jumbo frames are not enabled on these uplinks. The installer deploys multiple port groups on this vSwitch for ESXi management, storage controller VM management and replication (if enabled). The management interfaces include ESXi management interface, SCVM management and replication interface, and a roaming management cluster IP (one per cluster).
- **vswitch-hx-storage-data**: This vSwitch has two uplinks, active on fabric B and standby on fabric A – by default, jumbo frames are enabled on these uplinks. The installer deploys multiple port groups on this vSwitch for ESXi and storage controller VM storage-data networks. The storage-data interfaces include ESXi host, SCVM storage interface, and a roaming storage cluster IP (one per cluster).
- **vswitch-hx-vm-network**: This vSwitch has two uplinks, active on both fabrics A and B – by default, jumbo frames are not enabled on these uplinks. However, in this design, it has been reconfigured for jumbo frames through Cisco UCS Manager. The VLANs associated with the above port-groups are all tagged VLANs (not native VLANs) in Cisco UCS vNIC templates. Therefore, these VLANs are also explicitly configured in ESXi/vSphere.
- **vmotion**: This vSwitch has two uplinks, active on fabric A and standby on fabric B – by default, jumbo frames are enabled on these uplinks. The IP addresses of the VMkernel ports (vmk2) are configured using a post-install script. The VLANs associated with the above port-groups are all tagged VLANs (not native VLANs) in Cisco UCS vNIC templates. Therefore, these VLANs are also explicitly configured in ESXi/vSphere.

Enterprises can migrate the VM network virtual switch to a VMware distributed virtual switch (vDS), but the management and storage-data should stay on the installer-deployed VMware vSwitches. The VMware vDS can be optionally deployed and managed by Cisco ACI using the Virtual Machine Manager (VMM) integration feature. With VMM integration, ACI can dynamically provision port-groups on the vDS when corresponding endpoint groups (or VLANs) are provisioned in the fabric.

The installer deploys four virtual switches on each ESXi host in the cluster, with two uplinks or VMware virtual NICs (vmnics) per vSwitch. The vmnics at the hypervisor level map to virtual NICs (vNICs) on the Cisco UCS VIC adapter deployed in each server. The HyperFlex installation process configures the service-profiles in Cisco UCS Manager to create the different vNICs and vmnics that are used by the virtual switches running on the ESXi hosts in the HyperFlex VSI cluster.

Figure 8 shows the **default** virtual networking deployed on the ESXi hosts in a cluster.

Figure 8. Virtual Networking on HyperFlex Nodes - Default



The port groups on each vSwitch correspond to tagged VLANs in the server's vNIC template. The automated installation provisions the vNIC templates and VLANs on the server side and maps them to port groups on the VMware vSwitch. The vMotion vSwitch port groups and VLAN configuration is done post-deployment using a post-install script.

The virtual machine port groups and VLANs can be provisioned during the initial install or post-deployment using the post-install script. You can also ACI VMM integration to dynamically provision the ESXI networking as needed to roll out new applications and services. However, the ACI VMM integration will not provision the VLANs in the Cisco UCS Fabric Interconnect so this will have to be provisioned separately.

The Red Hat OCP clusters deployed in this solution uses the default VMware vSwitch for VM networks, on a VM network VLAN deployed by the installer.

Data Center Fabric - Cisco ACI

The data center fabric in the solution is a Cisco ACI fabric. ACI provides connectivity to the different components in the solution (HyperFlex server, Cisco UCS Fabric Interconnects, VMware vCenter, DNS, DHCP, NTP) and other networks, including external connectivity to Cisco Intersight and Red Hat Hybrid Cloud Console. The design separates all infrastructure connectivity necessary for deploying and maintaining the HyperFlex VSI cluster to a separate ACI tenant (**HXV-Foundation**). This tenant is responsible for providing the following HyperFlex infrastructure connectivity:

- **Management**: A HyperFlex cluster requires in-band management connectivity to ESXi hosts and storage controller virtual machines (SCVM) in the cluster. The roaming management cluster IP and other endpoints must be accessible to administrators, tools, and other entities outside the fabric. VMware vCenter managing the VSI cluster also needs access to the ESXi management interfaces.
- **Storage-data**: A HyperFlex cluster requires storage-data connectivity to ESXi hosts and storage controller virtual machines in the cluster. The roaming storage-data cluster IP and other endpoints must be accessible from the data center fabric.
- **VMware vMotion:** To enable vMotion of guest VMs running on the application HyperFlex cluster, the vMotion network must be reachable from VMware vCenter running on the management cluster.

- **iSCSI Storage-data**: The HyperFlex CSI used in this design uses iSCSI. A dedicated network is created for the iSCSI storage data traffic.

Multiple HyperFlex clusters connected to the same ACI fabric can use the same management and vMotion infrastructure networks, but the storage-data traffic should be on a dedicated network, one for each cluster. The ACI fabric also provides the following infrastructure connectivity:

- Internet Access for reachability to Cisco Intersight. The UCS domain must be claimed as a target in Intersight before it can be used to deploy the HyperFlex VSI cluster.
- Reachability to networks and services within the Enterprise, both within and outside the ACI fabric.

For the on-prem Red Hat OCP cluster and application workloads running on the HyperFlex VSI, the design uses a dedicated Application Tenant (for example, HC-Tenant1) to provide connectivity for the following HyperFlex network:

- **VM network(s)**: Applications networks deployed on the HyperFlex application cluster will need connectivity to various networks and services depending on the application requirements. In this solution, the Red Hat OCP is deployed on Guest VM networks and will require reachability to various internal and external entities required to install and operate the OCP cluster.
- **Persistent Storage**: For Red Hat OCP clusters to access persistent storage on HyperFlex using the HyperFlex CSI plugin, the ACI fabric will need to route traffic from the OCP cluster network (VM network) to the HyperFlex iSCSI storage-data network.

The ACI tenancy design will separate the infrastructure and application traffic. Multiple application tenants can be deployed as needed. For Red Hat OCP clusters and cloud native workloads, the ACI fabric also provides the following connectivity:

- Internet Access for deploying the Red Hat OCP cluster on HyperFlex VSI. Cloud-native applications running on the cluster will most likely need Internet access as well.
- Reachability to internal Enterprise networks and services, both within and outside the ACI fabric.

In the ACI architecture, ACI constructs (Tenants, Application profiles, Bridge domains, EPGs etc.) define and enable the connectivity through the fabric. To meet the infrastructure connectivity requirements outlined above, EPGs and other ACI constructs are defined in the **HXV-Foundation** tenant to enable this connectivity. The infrastructure VLAN networks provisioned by the HyperFlex installer are then mapped to end-point groups in ACI to enable forwarding between endpoints in the same network and to other networks. The same is done for application connectivity, but within the application tenant (**HC-Tenant1**). The ACI constructs to enable this forwarding are provided in the Solution Deployment section (Table 5).

To enable connectivity through the fabric, the ACI fabric must also provide access layer connectivity to the UCS domains and HyperFlex servers. The access layer connectivity includes:

- Physical connectivity to the UCS domains that HyperFlex clusters connect to. The PC/vPC design and connectivity are described in the Detailed Design - Cisco HyperFlex section.
- Access Layer configuration to enable connectivity to/from the Fabric Interconnects and HyperFlex servers in the UCS domain. In ACI, fabric access policies represent the configuration for connecting to access layer devices. To enable this connectivity, policies are first created and then applied to the leaf switch interfaces that connect to the access layer devices.

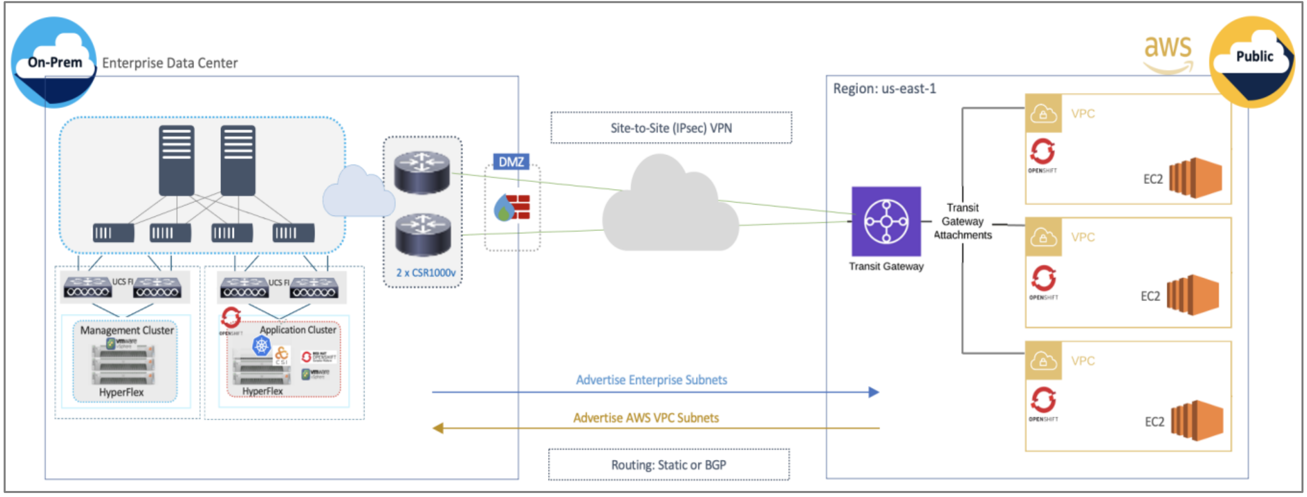
The specific policies and configuration used to enable the access layer connectivity to the Cisco UCS Fabric Interconnects and HyperFlex servers for the application cluster in this solution are provided in the Solution Deployment section (Table 4). The policies and profiles will create virtual port channels on the ACI leaf switch pair and enable access layer connectivity to the Cisco UCS Fabric Interconnects in the UCS domain.

The Cisco ACI fabric, together with Cisco UCS FIs provide the connectivity necessary to bring a HyperFlex cluster online and maintain the cluster post-deployment for hosting application workloads. The on-prem virtual server infrastructure is now ready for deploying a cloud-native environment on-prem using Red Hat OpenShift Container platform.

Hybrid Cloud Connectivity

Cisco offers multiple options for hybrid cloud connectivity between an on-prem location and public cloud. Options include Cisco ACI or VXLAN EVPN fabrics with Nexus Dashboard Orchestrator and Cloud Controller to orchestrate and extend the on-prem networking into the public cloud, SD-WAN, IPsec VPN, and dedicated connections. The hybrid cloud connectivity in this solution is established over the Internet using IPsec VPN tunnels. The IPsec VPN tunnel enables applications, services, and other components to securely access networks and entities in the other location. IPsec VPN tunnels are also referred to as Site-to-Site VPNs. IPsec VPN establishes encrypted tunnels across the Internet to provide secure connectivity for Enterprise traffic in a hybrid deployment. The IPsec VPN tunnels are established between Transit Gateways (TGW) in AWS and a pair of Cisco Cloud Services 1000 series router (CSR1kv). TGWs in AWS serve as transit hubs that can be used to consolidate the hybrid cloud connectivity from multiple VPCs by adding an attachment in each. Alternatively, you can also Virtual Private Gateways (VGW) from each VPC to establish IPsec VPN tunnels to the Enterprise data center, but you can only attach one VPC to this connection. When interconnecting locations in this manner, there should not be any overlapping addressing otherwise you could have routing issues. Figure 9 shows the hybrid cloud connectivity used in this solution. The AWS transit gateway establishes two tunnels to each customer/enterprise gateway (CSR1kv) for redundancy. The design uses two CSRs for higher availability to provide a total for 4 IPsec VPN tunnels between locations. The 4 tunnels will require 4 public IPs to establish the tunnels across the Internet. Enterprise firewalls must be provisioned to allow IPsec protocols and traffic from the AWS side tunnel addresses. By default, AWS expect the enterprise gateway to initiate the Internet Key Exchange (IKE) negotiation process to bring up the tunnel by the by generating traffic. It is recommended that AWS is configured to initiate the IKE negotiation to prevent the tunnel from going down when there is a lull in the traffic flow.

Figure 9. Hybrid Cloud Connectivity Design



When using Transit Gateways, there are some service limits in terms of MTU, routes, routing protocols, bandwidth, etc. that you should be aware of. Please review the AWS documentation for details on the VPN limits: https://docs.aws.amazon.com/vpn/latest/s2svpn/vpn-limits.html.

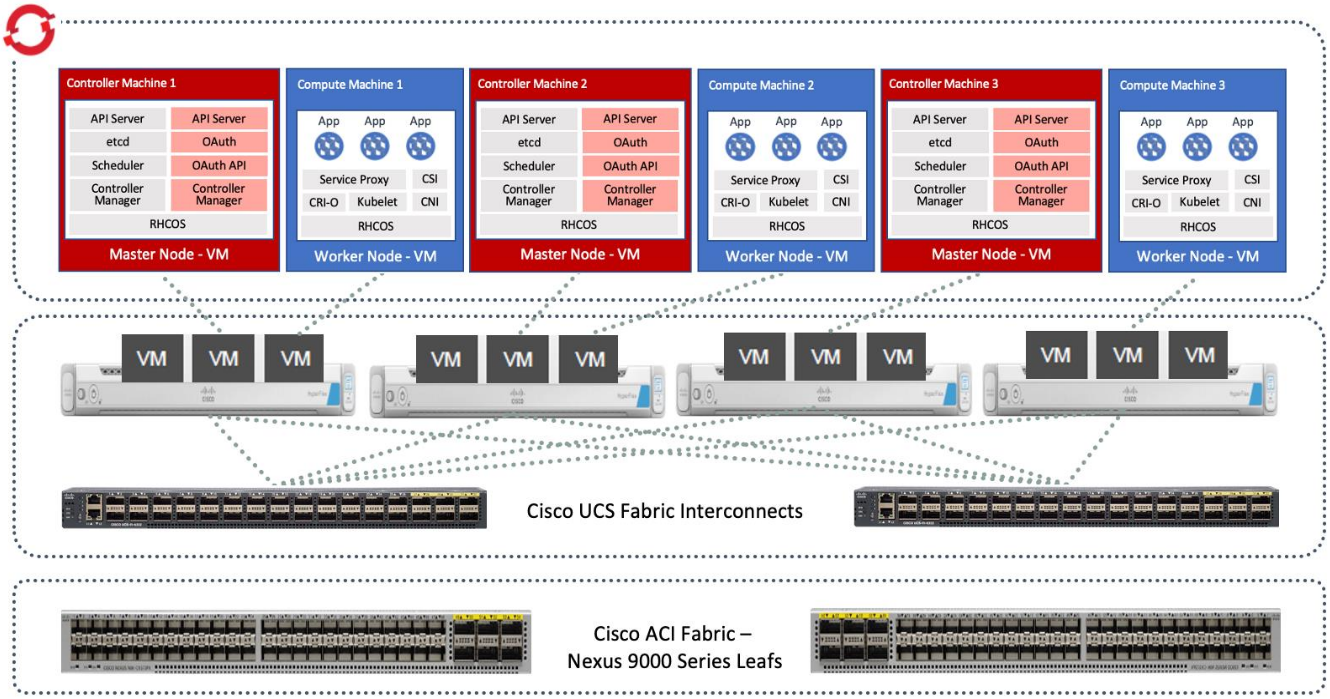
Kubernetes Infrastructure Design

Red Hat OCP provides secure, enterprise-class Kubernetes environment for developing, deploying, and managing cloud-native applications. The Red Hat Hybrid Cloud Console provides centralized, SaaS-based management of OCP clusters deployed anywhere, from on-prem to public cloud. Hybrid Cloud Console provides access to the latest installer package and other tools necessary for securely accessing and managing the cluster for each environment (for example, bare metal, VMware vSphere, AWS, Azure, GCP) and consumption (managed, unmanaged) model that OCP supports. In this solution, Hybrid Cloud Console is used to deploy OCP clusters in both on-prem and public using the Red Hat recommended Installer Provisioned Installation (IPI) method. This method deploys the full infrastructure using an opinionated, prescriptive approach with best-practices implemented. This is the fastest way to deploy an OCP cluster and OCP continues to manage the infrastructure components post-deployment. The automated OCP installer for each infrastructure environment collects minimal information from the user (for example, on-prem datastore). It does offer some customizations (for example, on AWS - deploy in an existing VPC on AWS) but for more extensive infrastructure customizations, the User Provisioned Installation (UPI) method is available. However, UPI does increase the operational burden on the IT/DevOps teams unlike IPI where OCP manages the infrastructure components.

On-Prem - Red Hat OCP on HyperFlex VSI

The Red Hat OCP cluster in the solution is deployed on HyperFlex VSI in the Enterprise data center as shown in Figure 10.

Figure 10. Application Cluster - HyperFlex VSI with Red Hat OCP

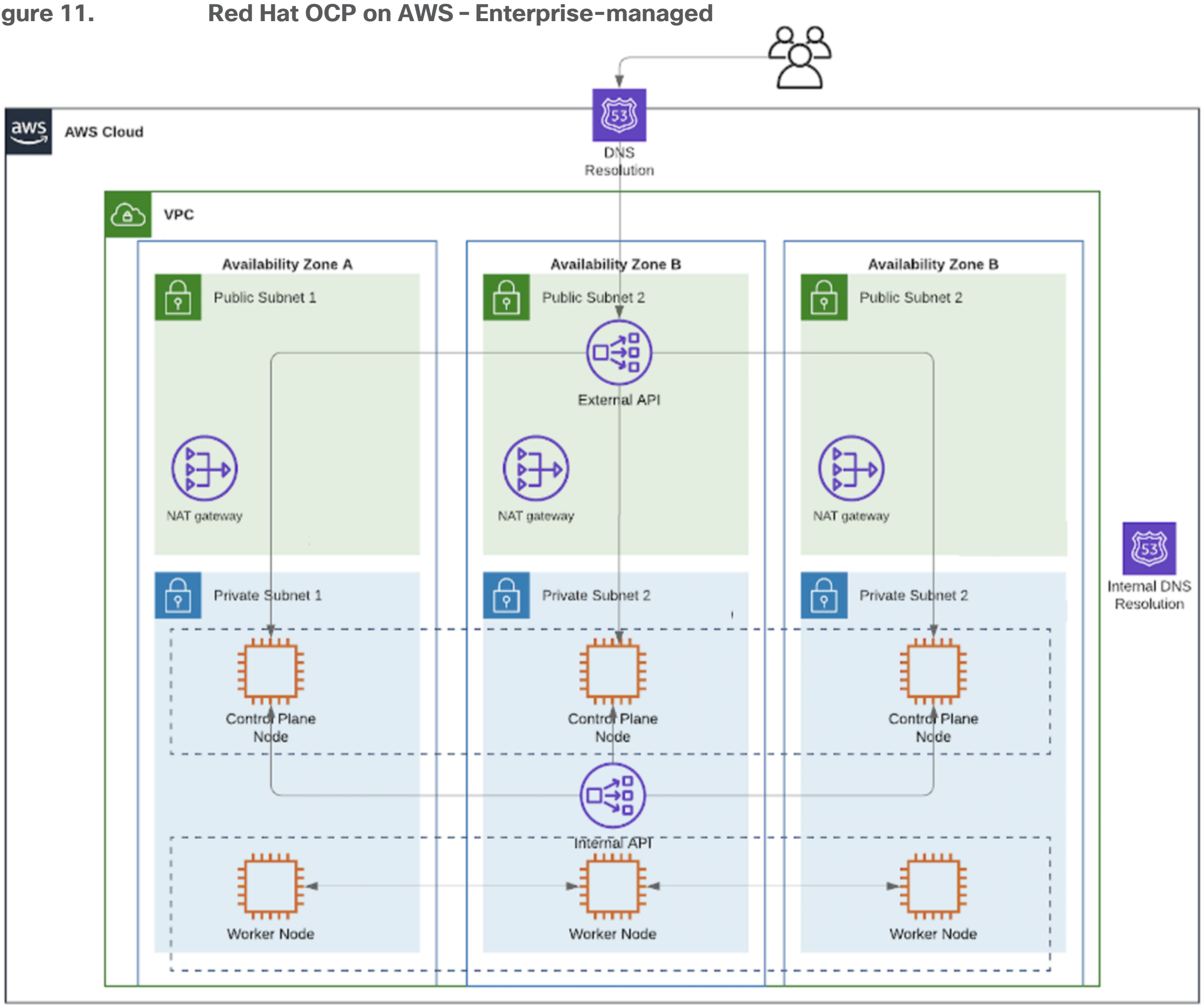


The nodes in the K8s cluster are VMs running on the Application HyperFlex VSI cluster, managed by VMware vCenter running on the management HyperFlex VSI cluster. The default IPI install deploys 3 x control/master and 3 x compute/worker nodes (VMs) - additional worker nodes can be added as needed. To isolate applications from infrastructure and management, OCP cluster and HyperFlex VSI are deployed on separate tenants in the Cisco ACI fabric. The same tenant can be used for additional OCP clusters or on new tenants can be added as needed. The OCP cluster is deployed on a VMware vSwitch VM network that was provisioned as a part of the HyperFlex VSI install. Post-install, the control/master and compute/worker nodes in the cluster are distributed across the ESXi hosts in the HyperFlex VSI cluster using VMware VM affinity rules. Each controller node runs on a separate physical ESXi host in the 4-node HyperFlex VSI cluster. Red Hat recommends three control nodes in all production deployments and therefore, a minimum of 3-node, but ideally a 4-node HyperFlex VSI cluster should be used for OCP deployments. VM affinity rules are also used to distribute the compute/worker nodes across the available ESXi hosts.

Public Cloud – Red Hat OCP on AWS

The Red Hat OCP cluster in AWS is deployed in **us-east-1** region. The AWS OCP cluster is Enterprise-managed and deployed from the Hybrid Cloud Console using the same IPI method as on-prem. The deployment is automated by the OCP installer. The cluster design, deployed in us-east-1 region, is shown in Figure 11.

Figure 11. **Red Hat OCP on AWS – Enterprise-managed**



When using the default installation, the installer deploys the cluster in a new dedicated Virtual Private Cloud (VPC) with 6 EC2 instances – 3 x control/master and 3 x compute/worker nodes running on Elastic Block Store (EBS) volumes. The default configuration of the EC2 instances is provided in Table 2.

Table 2. Default Configuration – EC2 Instances

Component	EC2 Instance Type	EBS Volume	Snapshots	Availability Zones (us-east-1 region)
Master/Control Node-1	`m6i.xlarge`	Type: gp3, Size: 120GiB, IOPS:3000, Throughout:125	Enabled	`us-east-1a`
Master/Control Node-2	`m6i.xlarge`	Type: gp3, Size: 120GiB, IOPS:3000, Throughout:125	Enabled	`us-east-1b`
Master/Control Node-3	`m6i.xlarge`	Type: gp3, Size: 120GiB, IOPS:3000, Throughout:125	Enabled	`us-east-1c`
Worker/Compute Node-1	`m6i.large`	Type: gp3, Size: 120GiB, IOPS:3000, Throughout:125	Enabled	`us-east-1a`
Worker/Compute Node-2	`m6i.large`	Type: gp3, Size: 120GiB, IOPS:3000, Throughout:125	Enabled	`us-east-1b`
Worker/Compute Node-3	`m6i.large`	Type: gp3, Size: 120GiB, IOPS:3000, Throughout:125	Enabled	`us-east-1c`

The cluster nodes are distributed across different availability zones (AZ) – the number of availability zones used depends on the number of AZs supported in each region. At the time of the writing of this document, the **us-east-1** region supports 6 availability zones (**us-east-1a** through **us-east-1f**). The default installation in this region uses three AZs to distribute the control and worker nodes for high availability. The cluster is also deployed with 3 public subnets and 3 private subnets (with NAT gateways) to support cloud-native applications with front-end and back-end components and services.

In both environments, the automated installer and tools are downloaded to an installer workstation in each environment. The installation process is then initiated from this installer with access to the infrastructure environment where the cluster will be deployed (HyperFlex VSI – Application Tenant, AWS – Account/Tenant VPC). The installation wizard gathers cluster-specific information from the user for each environment. Post-deployment, the Hybrid Cloud Console is used to access the cluster console for post-install activities and management. The AWS IPI installer does allow some customizations, for example, deploying with network customizations or using an existing VPC.

The automated installer also deploys other AWS infrastructure resources and configurations (for example, security rules, networking, DNS) as needed. OCP uses AWS service, Route 53, for DNS resolution. Internal DNS resolution within the cluster and for external access to the cluster are both provided by Route 53. To deploy Red Hat OCP, a dedicated public hosted zone must be defined in Route 53. A hosted zone is a container for DNS records. Each record provides information about how to route traffic to the domain and the sub-domains under it. The hosted zone is also the base domain. The zone must be authoritative for that domain. Red Hat OCP also uses private hosted zones – the installer will provision these during installation for DNS resolution within the cluster.

All AWS resources deployed by the installer will be tagged using a unique key (for example, kubernetes.io/cluster/ocp11-qvb4h) for each cluster deployed which makes it easy to find and manage cluster resources.

HyperFlex CSI

OCP Cluster console provides direct access to Red Hat Operator Hub where several community and Red Hat certified operators are available. Operators are an integral part of Red Hat OCP. It is the preferred way to package, deploy and manage K8s control plane services that the K8s cluster and users of the cluster can use. The embedded Operator Hub enables administrators and developers to quickly discover and deploy services to a given cluster. The HyperFlex CSI operator is Red Hat certified and available on the operator hub. HyperFlex CSI is used in this solution to provide persistent storage for the stateful workloads running on the OCP cluster. It enables applications to dynamically request and provision storage from the underlying HyperFlex storage using native K8S objects (for example, **Persistent Volume, Persistent Volume Claim, Storage Class**) that K8s administrators and operations teams are familiar with. HyperFlex CSI enables Red Hat OCP environments to leverage HyperFlex storage for cloud-native efforts with HyperFlex handling the entire life cycle management and orchestration of the persistent volume or storage.

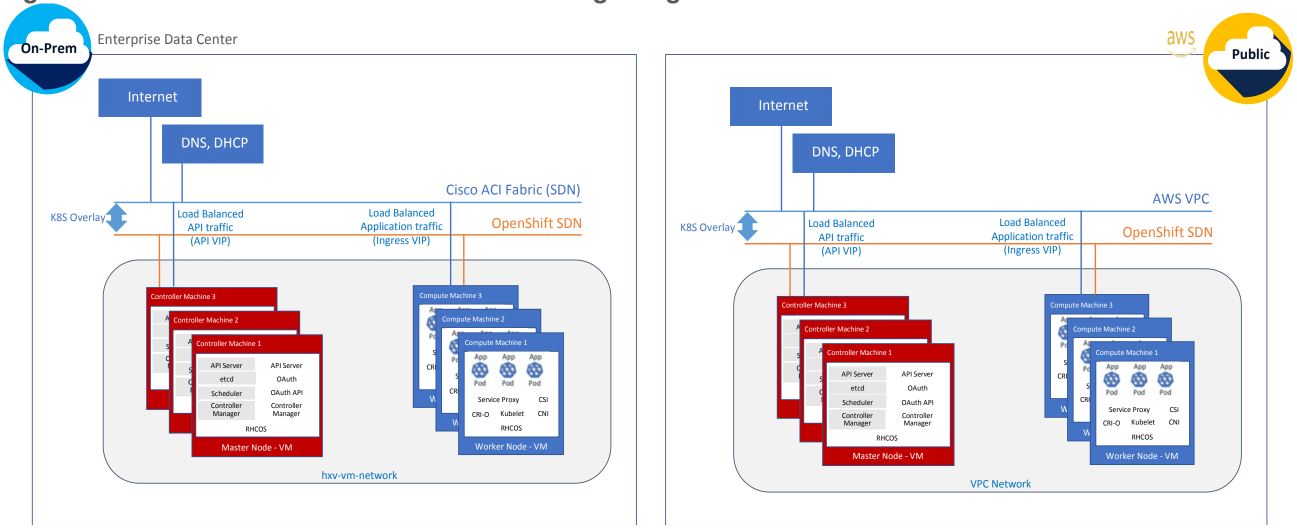
The HyperFlex storage is typically presented by the HyperFlex Distributed File System (HXDP) software running on the HyperFlex servers as a NFS file system to external clients. HyperFlex, in more recent releases, also supports presenting HyperFlex storage as raw, block-based iSCSI storage. HyperFlex CSI utilizes this capability to provision HyperFlex storage as iSCSI volumes and present them to the K8s clients when requested. To enable this, the application HyperFlex VSI cluster must be provisioned for an iSCSI network for clients to request storage from the HyperFlex system. The HyperFlex installation process deploys a storage-data network for the intra-cluster storage operations and for presenting that storage as NFS datastores to ESXi for use by guest virtual machines. However, for iSCSI, a separate network is provisioned but using the same Cisco UCS vNIC template and VMware vSwitch as the installer provisioned one for the NFS storage data. Additional iSCSI port groups are also created on the HyperFlex storage-data vSwitch for the new iSCSI VLAN network.

OpenShift Networking

Networking is critical for a highly distributed application environment such as Red Hat OCP. Kubernetes and OCP must enable different types of communications, securely, both within a cluster and to entities outside the cluster. Within a cluster, containers within Kubernetes Pods need connectivity to other Pods and Services within a given K8s cluster. The cluster and the applications running on the cluster will also need to be accessible from outside the cluster. Similar to CSI for storage, Kubernetes uses Container Networking Interface (CNI) plugins to manage network and security within a cluster. The CNI plugins are available on Red Hat's Operator hub from multiple vendors including one for Cisco ACI. Red Hat OCP, by default, uses OVN-Kubernetes which is based on Open Virtual Network (OVN) and Open vSwitch (OVS). OVN complements OVS to enable network overlays or logical networks on physical networks for Pod-to-Pod communication in Kubernetes. OCP takes a software-defined approach to enable Pod-to-Pod or intra-cluster networking using OVN and OVS. Enterprises can also choose to deploy other CNI solutions available on the Red Hat Operator Hub. This solution uses the native CNI provided in OCP, such as OVN-Kubernetes.

By default, Kubernetes (and OCP) allocates each pod an internal cluster-wide IP address that it can use for Pod-to-Pod communication. Within a Pod, all containers behave as if they're on the same logical host and communicate with each other using local-host, using the ports assigned to the containers. All containers within a Pod can communicate with each other using the Pod network.

Figure 12. **Red Hat OCP Virtual Networking Design- Internal and External**



For communication outside the cluster, OCP provides services (node ports, load balancers) and API resources (Ingress, Route) to expose an application or a service outside cluster so that users can securely access the application or service running on the OCP cluster. API resources, Ingress and Routes are used in this solution to expose the application deployed in the OCP cluster.

Requirements

Table 3 lists the hardware components used for validating the solution.

Table 3. Solution Components - Hardware

Component	Hardware
Hyperconverged Servers	4 x Cisco HXAF220C-M5SX for application workload cluster (additional 4 for an optional management cluster)
Fabric Interconnects	Two (2) Cisco UCS 6332 Fabric Interconnects
Data Center Fabric	Cisco ACI Fabric with 3 x APIC, 2 x Spine switches (N9k-C9332C), 2 x Leaf switches (N9K-C9336C-FX2), additional Leaf switches for optional management cluster.
External Gateways	2 x CSR 1000vs for IPsec VPN to AWS

Table 4 lists the software components and the versions used for validating the solution.

Table 4. Solution Components - Software

Component	Software
Cisco HyperFlex Data Platform Software	5.0(2a)
VMware vSphere	7.0.3(U3)
Red Hat OCP	4.9 or higher
Cisco HyperFlex CSI	1.2.5 or higher
Cisco UCS Firmware	4.2(1m)

Component	Software
Cisco ACI Software	5.2(4d)
SaaS - Cisco Intersight	N/A
SaaS – Red Hat Hybrid Cloud Console	N/A
SaaS – Cisco Intersight Workload Optimizer	N/A (Cisco IWO Collector on OCP cluster: v1.20)
Public Cloud - AWS	N/A

Solution Deployment

This chapter contains the following:

- Deployment Overview
- Provision Cisco ACI Fabric
- Deploy HyperFlex Virtual Server Infrastructure
- Deploy Red Hat OpenShift Container Platform (On-Prem)
- Deploy HyperFlex CSI
- Deploy Red Hat OpenShift Container Platform (Public Cloud)
- Enable Secure Hybrid Cloud Connectivity
- Enable Cisco Intersight Workload Optimizer (On-Prem)
- Enable Cisco Intersight Workload Optimizer (Public Cloud)

This chapter describes the solution deployment with step-by-step procedures for implementing and managing the solution.

Deployment Overview

At a high-level, the solution deployment can be split into the following areas:

- Deploy on-prem infrastructure
- Deploy public cloud infrastructure
- Enable secure interconnectivity between on-prem and public cloud infrastructure

On-Prem Infrastructure

In the on-prem data center, Enterprises typically manage their own compute, storage, and networking. To accelerate application efforts, we want to deploy this infrastructure as quickly as possible. In cloud-native environments, it is also critical to provide infrastructure as code for integration into CI/CD and other automation that an Enterprise has.

The deployment of the on-prem infrastructure consists of:

- Deploy networking to enable reachability for remaining components in the infrastructure stack. Red Hat Ansible provisions the network infrastructure as code.
- Deploy compute, storage, virtualization, and networking for server/virtual machines. The VSI cluster serves as an Applications cluster for hosting cloud-native workloads. Red Hat Ansible automates the hyperconverged infrastructure deployment using Cisco Intersight APIs to claim, provision and deploy a complete virtual server infrastructure platform.
- Deploy Red Hat OpenShift Container Platform to enable a cloud-native application development environment
- Deploy HyperFlex Container Storage Interface as persistent storage for cloud-native workloads
- Enable resource optimization of on-prem resources using Intersight Workload Manager to ensure application performance.

Public Cloud Infrastructure

Since the public cloud provider manages the compute, storage and networking infrastructure, the deployment of the public cloud infrastructure is limited to:

- Deploy Red Hat OpenShift Container Platform. The automated installer for Red Hat OCP will take care of deploying the infrastructure resources required to host the OCP cluster (for example, AWS VPC, EC2 instances).
- Enable resource optimization of public cloud resources using Intersight Workload Manager to ensure application performance and manage cloud costs.

Secure Hybrid Cloud Connectivity Infrastructure

To securely interconnect the two environments in the hybrid cloud deployment involves the following:

- Provision public cloud for secure connectivity to the on-prem location
- Provision on-prem infrastructure for secure connectivity to the public location

Provision Cisco ACI Fabric

This section describes the deployment of the on-prem network infrastructure in this hybrid cloud solution. The on-prem network in this solution is a Cisco ACI fabric managed using Cisco APIC. The deployment is automated using Red Hat Ansible playbooks available in the Cisco UCS Solutions GitHub repository. The automation will focus on the day-2 networking required to support the infrastructure and workloads in the solution.

Assumptions

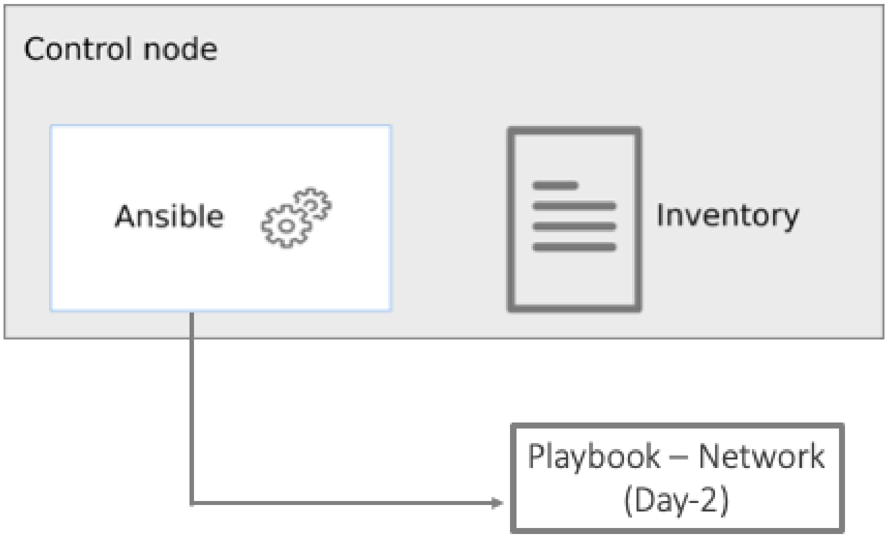
The deployment procedures outlined in this section makes the following assumptions:

- Enterprise has an existing Cisco ACI fabric - the on-prem hybrid cloud infrastructure in this solution will connect into this fabric.
- Day-0 and day-1 provisioning of the above ACI fabric is complete, with internal and external connectivity to services in place – for example, Internet access and services such as DNS, DHCP.
- Management infrastructure with the necessary connectivity is available for hosting an automation workstation. The workstation will need reachability to the Internet and Cisco APIC cluster managing the ACI fabric.

Prerequisites

The prerequisites for provisioning the network are:

- The Cisco ACI Fabric leaf switches that the application HyperFlex cluster and Cisco UCS Fabric Interconnects connect to should be deployed and provisioned to be part of the ACI fabric.
- Application HyperFlex cluster servers and the Cisco UCS Fabric Interconnects should ideally be physically cabled, powered, and connected to the ACI fabric so that connectivity can be verified when the network provisioning is complete.
- Ansible control node or workstation for executing the Ansible playbooks. The playbooks automate the day-2 provisioning of the networking required to bring the cloud-native infrastructure up.



Setup Information

Table 5 lists the configuration parameters.

Table 5. Cisco ACI - Configuration Parameters

Variable	Variable Name	Value	Additional Info
Cisco APIC cluster	-	172.26.163.120	<Login Credentials>
Git Hub Repo	-	https://github.com/ucs-compute-solutions/Hybrid-Cloud/tree/main/CVD_HC-OCP-HXFI Ansible scripts for provisioning the ACI fabric is in the **network** directory.	
Other		All variables required to execute the Ansible scripts are defined in the file below, located in a separate directory called **inventory** in the above repo: inventory > group_vars > cisco_dc_fabric	

Deployment Steps

This section describes the procedures for configuring the day-2 networking required to connect and deploy the application HyperFlex VSI cluster. The application cluster connects to a pair of Cisco UCS Fabric Interconnects that is dual-homed to a pair of leaf switches in the data center fabric.

Procedure 1. Prerequisite – Setup an Ansible Control Node running MacOS

Note: The Ansible workstation is running MacOS in this setup.

- To install on other operating systems: https://docs.ansible.com/ansible/latest/installation_guide/installation_distros.html

For additional information, see the Ansible Installation Guide: https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html#

Step 1. Ansible control node requires Python 3.8 or higher. Verify if it is already installed.

```
$ python3 -V
```

```
Python 3.9.13
```

Step 2. If Python is not installed or needs to be upgrade, use the commands below to install it.

```
$ brew install python3
```

-OR-

```
$ brew upgrade python3
```

Step 3. Verify if you have the Python package manager (**pip**). The above python install should automatically install **pip**.

```
$ python3 -m pip -V
pip 22.3.1 from /usr/local/lib/python3.9/site-packages/pip (python 3.9)
```

Step 4. If **pip** is not installed or needs to be upgraded, use the following commands

```
$ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
$ python3 get-pip.py
```

-OR-

```
$ pip3 install --upgrade pip
```

Step 5. (Optional) Create Virtual Environment (venv) using Python and activate it for use.

```
$ python3 -m venv <venv_name>
$ python3 -m venv venv1
$ source ./venv1/bin/activate
```

To deactivate: `deactivate`

Create aliases (example): `alias switchto_venv=‘source ./venv1/bin/activate’`

Step 6. Install Ansible on workstation in virtual environment (optional); other useful commands also provided

```
(venv1)$ pip install ansible                # not necessary to specify python version in venv
(venv1)$ which ansible
(venv1)$ ansible --version
(venv1)$ ansible -h
(venv1)$ pip install --upgrade ansible
```

Step 7. Verify the path and version of python is what you want Ansible to use

```
(venv1)$ ansible --version
ansible [core 2.13.6]
  config file = None
  configured module search path = ['/Users/asharma/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /Users/asharma/CISCO-AUTOMATION-PROJECTS/venv1/lib/python3.9/site-packages/ansible
  ansible collection location = /Users/asharma/.ansible/collections:/usr/share/ansible/collections
  executable location = /Users/asharma/CISCO-AUTOMATION-PROJECTS/venv1/bin/ansible
  python version = 3.9.13 (main, May 24 2022, 21:28:31) [Clang 13.1.6 (clang-1316.0.21.2)]
  jinja version = 3.1.2
  libyaml = True
```

Step 8. Install GIT. It might already be installed on MacOS through other tools. Otherwise install git as follows:

```
(venv1) $ brew install git                # not necessary to execute this in venv
```

Procedure 2. Clone Git Hub repository for network provisioning

To access the Ansible playbooks in the GitHub repository (repo), clone the Git Hub repo as outlined below. The cloning will create a completely new copy of the repo in the location specified on the Ansible workstation. The repo is located at: https://github.com/ucs-compute-solutions/Hybrid-Cloud directory.

Step 1. On the Ansible workstation, use a terminal console or command-line tool to create a directory for the project. The GitHub repo will be cloned to a sub-directory in this directory. For example, if the directory is **CISCO-AUTOMATION-PROJECTS**, the repo will be cloned within this directory under a sub-directory.

Step 2. Navigate to the newly created directory from the terminal window and execute the following command:

```
git clone https://github.com/ucs-compute-solutions/Hybrid-Cloud.git
```

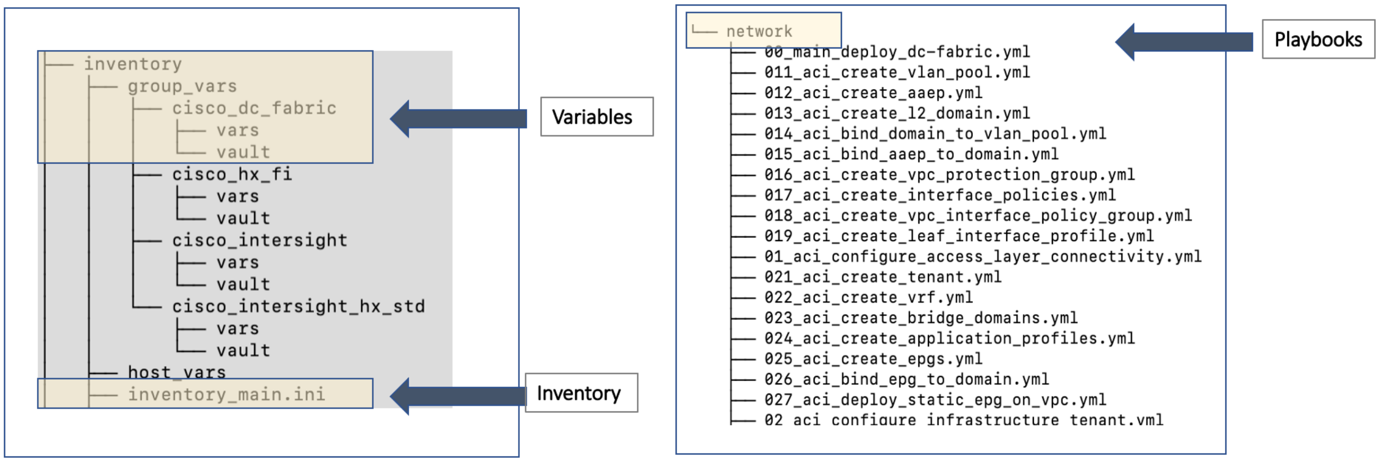
Step 3. Navigate to the sub-directory (**CVD_HC-OCP-HXFI**), followed by the network directory below that (**network**).

Step 4. (Optional) Switch to the Python virtual environment using the command provided in the **Setup Ansible Control Node** deployment procedure earlier in the document.

Procedure 3. Review the Ansible files for provisioning the network

Ansible uses inventory files (**inventory.ini, inventory.yml**), variables files (**group_vars, host_vars**), and playbooks to automate the provisioning. The inventory files are the targets of the automation, in this case, the Cisco APICs managing the Cisco ACI Fabric. It will have information to connect to the target device. The variables files contain the configuration parameters. The inventory files and variable files will need to be modified for each environment.

Step 1. Review inventory, variables, and playbooks for provisioning the network. The Ansible playbooks for provisioning the network infrastructure and the associated inventory and variables are highlighted in the figure below.



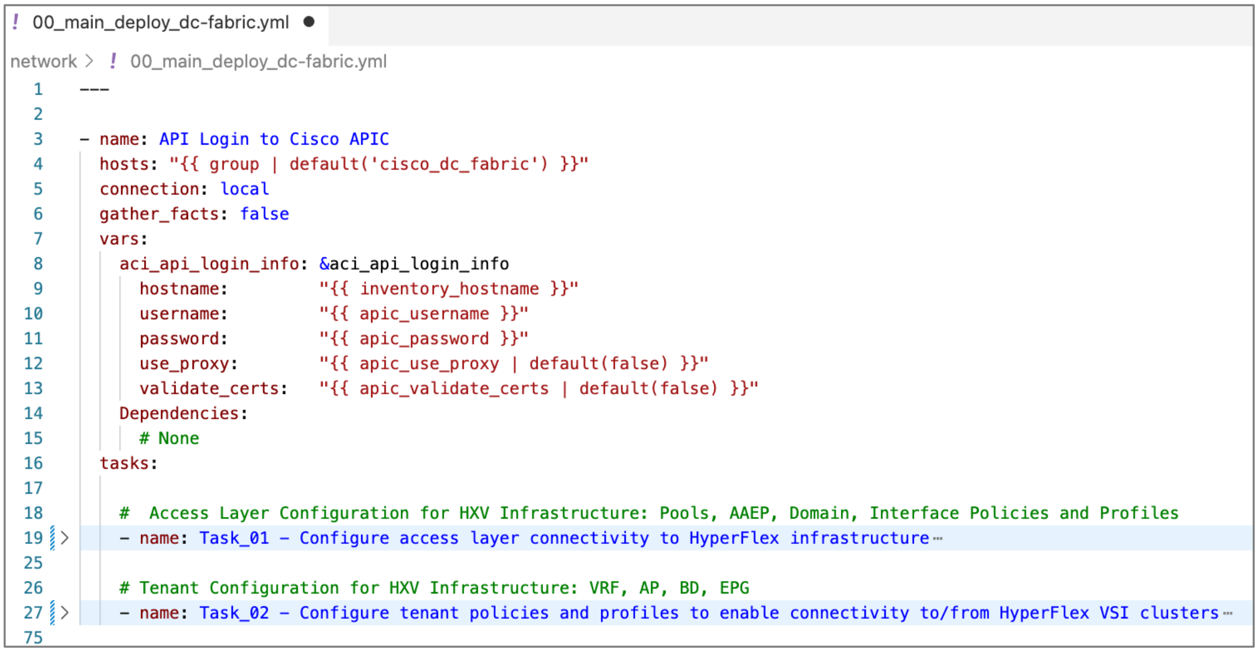
```
├── inventory
│   ├── group_vars
│   │   ├── cisco_dc_fabric
│   │   │   ├── vars
│   │   │   └── vault
│   │   ├── cisco_hx_fi
│   │   │   ├── vars
│   │   │   └── vault
│   │   ├── cisco_intersight
│   │   │   ├── vars
│   │   │   └── vault
│   │   └── cisco_intersight_hx_std
│   │       ├── vars
│   │       └── vault
│   ├── host_vars
│   ├── inventory_main.ini
```

```
└── network
    ├── 00_main_deploy_dc-fabric.yml
    ├── 011_aci_create_vlan_pool.yml
    ├── 012_aci_create_aaep.yml
    ├── 013_aci_create_l2_domain.yml
    ├── 014_aci_bind_domain_to_vlan_pool.yml
    ├── 015_aci_bind_aaep_to_domain.yml
    ├── 016_aci_create_vpc_protection_group.yml
    ├── 017_aci_create_interface_policies.yml
    ├── 018_aci_create_vpc_interface_policy_group.yml
    ├── 019_aci_create_leaf_interface_profile.yml
    ├── 01_aci_configure_access_layer_connectivity.yml
    ├── 021_aci_create_tenant.yml
    ├── 022_aci_create_vrf.yml
    ├── 023_aci_create_bridge_domains.yml
    ├── 024_aci_create_application_profiles.yml
    ├── 025_aci_create_epgs.yml
    ├── 026_aci_bind_epg_to_domain.yml
    ├── 027_aci_deploy_static_epg_on_vpc.yml
    ├── 02 aci configure infrastructure tenant.yml
```

Note: The playbooks are in the 'network' sub-directory under 'CVD_HC-OCP-HXFI.' The variables and inventory are in the same 'inventory' sub-directory under 'CVD_HC-OCP-HXFI,' in 'group_vars' directory and 'inventory_main.ini' file, respectively.

Step 2. Review the main playbook file for network provisioning. Note that at a high-level, the main playbook consists of

- Authenticating with Cisco APIC to provision the ACI fabric. Authentication information is encrypted using Ansible vault and referenced in the variable files and in the playbook as shown below.
- Configuring the access layer connectivity to Cisco UCS FI and HyperFlex servers to provision and bring up the VPC/PC configuration between Cisco UCS Fabric Interconnects and Nexus 9000 series leaf switches.
- Configuring the ACI Tenant constructs and policies (Tenant, VRFs, Bridge Domains, Application profiles, EPGs, Contracts) that enable connectivity through the ACI fabric for the application HyperFlex VSI cluster connected to it.


! 00_main_deploy_dc-fabric.yml ●
network > ! 00_main_deploy_dc-fabric.yml

```yaml
---

- name: API Login to Cisco APIC
  hosts: "{{ group | default('cisco_dc_fabric') }}"
  connection: local
  gather_facts: false
  vars:
    aci_api_login_info: &aci_api_login_info
      hostname:         "{{ inventory_hostname }}"
      username:         "{{ apic_username }}"
      password:         "{{ apic_password }}"
      use_proxy:        "{{ apic_use_proxy | default(false) }}"
      validate_certs:   "{{ apic_validate_certs | default(false) }}"
    Dependencies:
      # None
  tasks:

    #  Access Layer Configuration for HXV Infrastructure: Pools, AAEP, Domain, Interface Policies and Profiles
    - name: Task_01 - Configure access layer connectivity to HyperFlex infrastructure⋯

    # Tenant Configuration for HXV Infrastructure: VRF, AP, BD, EPG
    - name: Task_02 - Configure tenant policies and profiles to enable connectivity to/from HyperFlex VSI clusters⋯

```

Note: The playbook tasks are named to reflect their hierarchy within the within the overall collection of playbooks. The main task is numbered as 00_ , with the two tasks below it as 01_ and 02_ . The sub-tasks within each of these are then numbered starting with 011_ and 021_ respectively.

Procedure 4. Modify the variables files for executing the Ansible playbooks

To execute the playbook file, the Ansible variables file is populated with the parameters that will be used to configure the network fabric. The configuration variable names, and the parameters used in this solution are provided in Table 6. You can use either a text editor or an IDE environment (for example, Microsoft Visual Studio Code) to edit the variables file.

Note: The variables and parameters in the table below, is not a comprehensive list – it shows the parameters that are environment specific that the Enterprise administrator must provide and few additional ones.

Table 6. Access Layer Configuration Parameters

Variable	Variable Name	Value	Additional Info
VLAN	hxv-inband-mgmt	1171	HyperFlex iband-management – SCVM, ESXi
VLAN	hxv-cl3-storage-data	1273	HyperFlex storage data – SCVM, ESXi
VLAN	hxv-vmotion	1371	HyperFlex vMotion
VLAN	hxv-vm-network	1521-1530	
VLAN	hxv-cl3-iscsi-a/b	3013, 3023	
VPC	protection_group_id	12	VPC ID
Leaf switch ID	switch_1_id	103	VPC Leaf switch pair

Variable	Variable Name	Value	Additional Info
Leaf switch ID	switch_2_id	104	VPC Leaf switch pair
Interface Policies	cdp_policy_name	CDP_Enabled	
	link_level_policy_name	40Gbps-Link	
	lldp_policy_name	LLDP-Enabled	
	port_channel_policy_name	LACP-Active	
	l2_interface_policy_name	VLAN-Scope-Local	
	stp_interface_policy_name	BPDU-FG-Enabled	
Leaf Selector	HXV-UCS_FI-Leaf_103-104	103,104	switch_1_id, switch_2_id
Leaf Access Port Selectors	HXV-FI_p1_1	1/1	
	HXV-FI_p1_1	1/2	

Table 7 lists the ACI tenant networking constructs that enable forwarding through the fabric.

Table 7. ACI Constructs

Variable	Variable Name	Value	Additional Info
Tenant Name	Tenant	HXV-Foundation	
VRF Name	VRF	HXV-Foundation_VRF	
Bridge Domain	HXV-IB-MGMT_BD		HyperFlex iband-management
Gateway IP address	gateway	10.1.171.254	
Gateway Netmask	mask	255.255.255.0	
Bridge Domain	HXV-CL3-StorData_BD		HyperFlex storage-data
Bridge Domain	HXV-vMotion_BD		HyperFlex vMotion
Bridge Domain	HXV-CL3-iSCSI_BD		HyperFlex iSCSI (for HX CSI)
Application Profile	HXV-IB-MGMT_AP		HyperFlex iband-management
	HXV-CL3-StorData_AP		HyperFlex storage-data
	HXV-vMotion_AP		HyperFlex vMotion
	HXV-CL3-iSCSI_AP		HyperFlex iSCSI (for HX CSI)
EPG	HXV-IB-MGMT_EPG		HyperFlex iband-management
	HXV-CL3-StorData_EPG		HyperFlex storage-data
	HXV-vMotion_EPG		HyperFlex vMotion

Variable	Variable Name	Value	Additional Info
	HXV-CL3-iSCSI_EPG		HyperFlex iSCSI (for HX CSI)
Contracts	Allow-Common-L3Out_Contract		Existing contract – allows access to any network outside the ACI fabric – Internet, other Enterprise internal networks
	Allow-IB-MGMT_Contract		Allows access to HyperFlex and ESXi hosts in the cluster
	Allow-INFRA-MGMT_Contract		Allows access to VMware vCenter and other infrastructure management services running on the Management cluster

Note: The above tables only show a subset of the configuration parameters required to enable forwarding of HyperFlex VSI traffic to other nodes in the cluster, to the Internet, to out-of-band networks etc. For a complete list, see variables file in GitHub repository

Procedure 5. Execute the main playbook for provisioning the network infrastructure

Step 1. Identify the main playbook file in the **network** sub-directory. This main YAML file have a name that starts with '00_'.

Step 2. Identify the path to the inventory file relative to where you will run the playbook

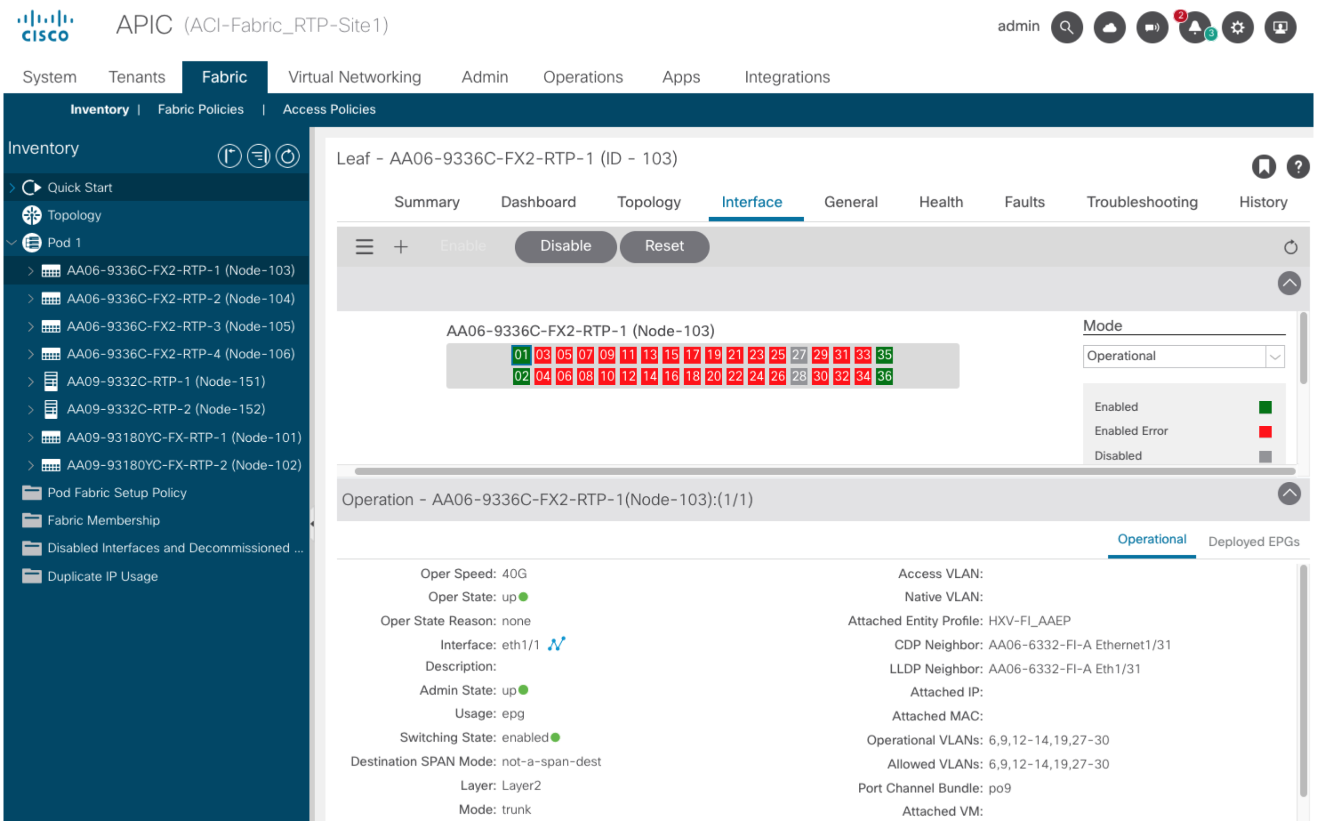
Step 3. Run the main playbook to configure networking: **ansible-playbook** <playbook-name> **-i** <inventory_file_name>

```
(venv1)CVD_HC-OCP-HXFI $ ansible-playbook network/00_main_deploy_dc-fabric.yml -i
inventory/inventory_main.ini
```

Procedure 6. Verify the deployed configuration

Step 1. Login to web GUI of the Cisco APIC cluster managing the ACI fabric.

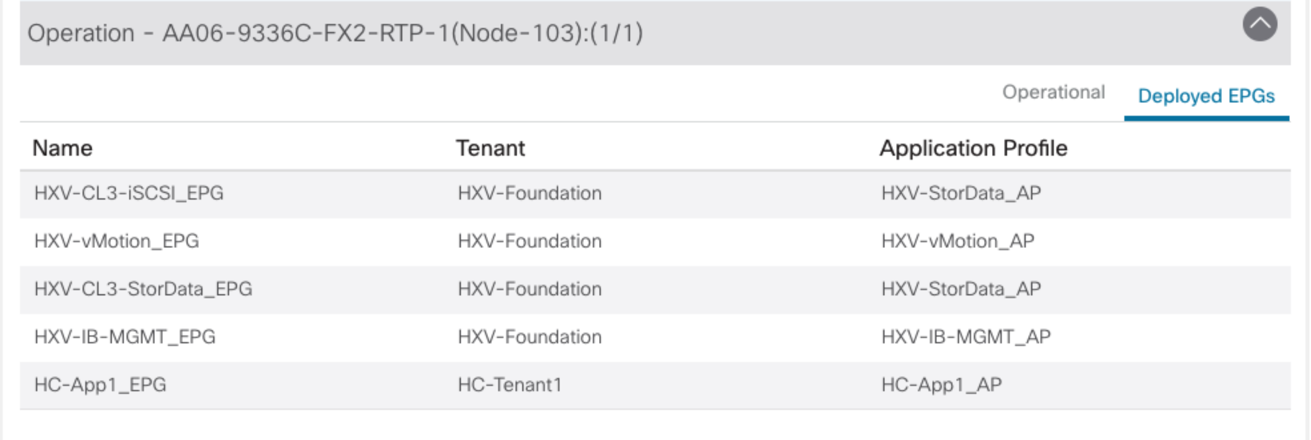
Step 2. To verify the access layer setup to Cisco UCS Fabric Interconnects and Application HyperFlex cluster in the ACI Pod (in this case, Pod1) it connects to. Navigate to **Fabric > Inventory > Pod 1** and select the leaf switch and interface for the connection. Verify that it is up and operational with correct LLDP/CDP neighbor. Ignore the VLAN info as they ACI internal VLANs. You can also view the EPGs deployed on the interface.


APIC (ACI-Fabric_RTP-Site1)
admin
System
Tenants
Fabric
Virtual Networking
Admin
Operations
Apps
Integrations
Inventory | Fabric Policies | Access Policies
Inventory
Quick Start
Topology
Pod 1
AA06-9336C-FX2-RTP-1 (Node-103)
AA06-9336C-FX2-RTP-2 (Node-104)
AA06-9336C-FX2-RTP-3 (Node-105)
AA06-9336C-FX2-RTP-4 (Node-106)
AA09-9332C-RTP-1 (Node-151)
AA09-9332C-RTP-2 (Node-152)
AA09-93180YC-FX-RTP-1 (Node-101)
AA09-93180YC-FX-RTP-2 (Node-102)
Pod Fabric Setup Policy
Fabric Membership
Disabled Interfaces and Decommissioned ...
Duplicate IP Usage
Leaf - AA06-9336C-FX2-RTP-1 (ID - 103)
Summary
Dashboard
Topology
Interface
General
Health
Faults
Troubleshooting
History
Enable
Disable
Reset
AA06-9336C-FX2-RTP-1 (Node-103)
01 03 05 07 09 11 13 15 17 19 21 23 25 27 29 31 33 35
02 04 06 08 10 12 14 16 18 20 22 24 26 28 30 32 34 36
Mode
Operational
Enabled 🟩
Enabled Error 🟥
Disabled ▨
Operation - AA06-9336C-FX2-RTP-1(Node-103):(1/1)
Operational
Deployed EPGs

Oper Speed: 40G
Oper State: up ●
Oper State Reason: none
Interface: eth1/1
Description:
Admin State: up ●
Usage: epg
Switching State: enabled ●
Destination SPAN Mode: not-a-span-dest
Layer: Layer2
Mode: trunk

Access VLAN:
Native VLAN:
Attached Entity Profile: HXV-FI_AAEP
CDP Neighbor: AA06-6332-FI-A Ethernet1/31
LLDP Neighbor: AA06-6332-FI-A Eth1/31
Attached IP:
Attached MAC:
Operational VLANs: 6,9,12-14,19,27-30
Allowed VLANs: 6,9,12-14,19,27-30
Port Channel Bundle: po9
Attached VM:

Step 3. Select the **Deployed EPGs** tab to see the EPGs deployed on the interface. Repeat for all interfaces on this leaf switch and the second switch in the pair.



Name	Tenant	Application Profile
HXV-CL3-iSCSI_EPG	HXV-Foundation	HXV-StorData_AP
HXV-vMotion_EPG	HXV-Foundation	HXV-vMotion_AP
HXV-CL3-StorData_EPG	HXV-Foundation	HXV-StorData_AP
HXV-IB-MGMT_EPG	HXV-Foundation	HXV-IB-MGMT_AP
HC-App1_EPG	HC-Tenant1	HC-App1_AP

Step 4. You can also verify that there are no faults in the access layer policies by starting with the switch profile that includes all the policies associated with a subset of leaf interfaces on a specific leaf. Navigate to **Fabric > Access Policies > Switches > Leaf Switches > Profiles** and find the profile that defines the access layer configuration to the Cisco UCS FI and HyperFlex. Verify that there are no faults. Use this as a starting point to verify that there are no faults in all the policies and selector profiles that this profile uses. From the profile, you can click on a policy to navigate to it.

Step 5. To verify the configuration and status of the ACI constructs that enable connectivity through the fabric (Tenant, VRF, Application Profile, BD, EPG, Contracts, and so on), navigate to **Tenants > HXV-Foundation** and verify that there are no faults or errors for each of the above constructs.

Step 6. Verify the contracts by testing reachability to the Internet, VMware vCenter managing the vSphere cluster and other Enterprise internal networks (for example, out-of-band management interface on Fabric Interconnects and KVM IP of servers)

The network is now ready for deploying the HyperFlex VSI infrastructure as outlined in the next section.

Deploy HyperFlex Virtual Server Infrastructure

This section describes the deployment of the on-prem virtual server infrastructure in this hybrid cloud solution. The on-prem infrastructure in this solution is a HyperFlex VSI running VMware vSphere, deployed, and managed using Cisco Intersight. The deployment is automated using Red Hat Ansible playbooks available in the GitHub repository for Cisco UCS Solutions. The automation will focus on day-0 and day-1 deployment of a new HyperFlex VSI cluster for hosting cloud-native applications.

Assumptions

The deployment outlined in this section assumes the following:

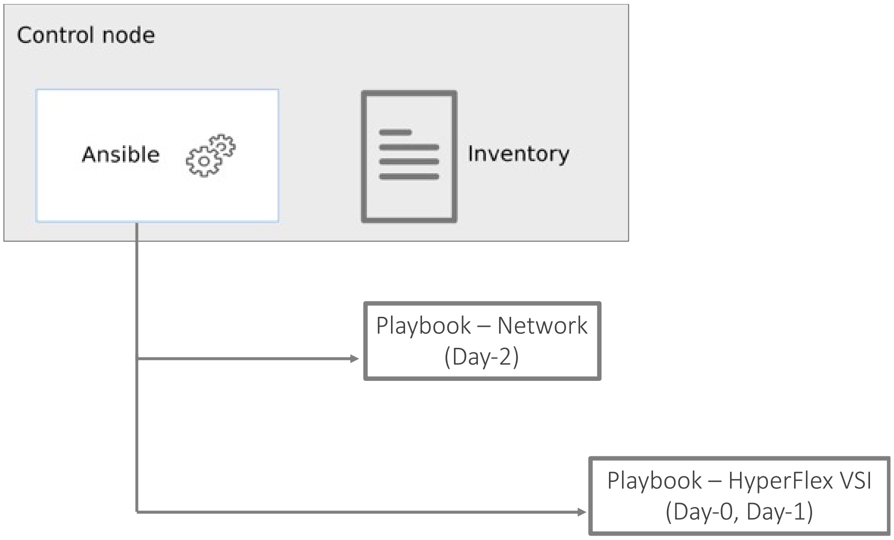
- Enterprise has an existing management HyperFlex cluster in place to host infrastructure and management components required in this solution.
- For purposes of this document, the management cluster is a HyperFlex VSI cluster. The deployment of this cluster will not be covered in this document. It is connected to the same Cisco ACI fabric through a different Cisco UCS domain. It will be in the same **HXV-Foundation** tenant and share the same in-band management and vMotion network as the application HyperFlex cluster discussed here.
- For accessing the infrastructure services hosted on the management cluster, ACI fabric exposes a contract (**Allow-INFRA-MGMT_Contract**) that is used by the application VSI cluster to enable reachability between the two. The contract enables VMware vCenter running on the management cluster to manage the application VSI cluster. It also hosts the DNS, DHCP, Red Hat OCP installer for deploying the Application OCP cluster.

The deployment procedures outlined in this section will therefore focus on deploying the HyperFlex "Application" or workload cluster for hosting cloud-native applications.

Prerequisites

The prerequisites for the HyperFlex VSI deployment are:

- Cisco data center fabric is provisioned to provide the infrastructure connectivity required to deploy a HyperFlex cluster.
- Network services (DNS, NTP) and VMware vCenter are available and reachable from the Application HyperFlex cluster and Cisco Intersight.
- Application HyperFlex server and Cisco UCS Fabric Interconnects are physically cabled, powered, and connected to the ACI fabric leaf switch pair.
- Ansible control node or workstation for executing the Ansible playbooks.


Control node
Ansible
Inventory
Playbook – Network (Day-2)
Playbook – HyperFlex VSI (Day-0, Day-1)

Setup Information

Table 8 lists the installation parameters.

Table 8. HyperFlex – Installation Parameters

Variable	Variable Name	Value	Additional Info
Cisco UCS Manager IP and Login Credentials	-	192.168.171.192	<Login Credentials>
Intersight Account & Login	-	<collect>	
Cisco UCS Manager FI – Claim Code	-	<collect>	
Cisco UCS Manager FI – Device ID	-	<collect>	
Git Hub Repo	-	https://github.com/ucs-compute-solutions/Hybrid-Cloud/tree/main/CVD_HC-OCP-HXFI Ansible scripts for provisioning the HyperFlex VSI is in the **compute** directory.	
Other		All variables required to execute the Ansible scripts are defined in the files below, located in a separate directory called **inventory** in the above repo: **inventory > group_vars > cisco_hx_fi** **inventory > group_vars > cisco_intersight** **inventory > group_vars > cisco_intersight_hx_std_servers**	

Deployment Steps

This section describes the procedures for deploying an Application HyperFlex VSI cluster. The application cluster connects to a pair of Cisco UCS Fabric Interconnects that is dual-homed to a pair of leaf switches in the data center fabric. A built-in installer on Cisco Intersight will remotely deploy the cluster. The Ansible automation will provide the necessary configuration parameters and initiate the install of a HyperFlex VSI cluster using Cisco Intersight. The parameters provided are the same as the inputs to the installer wizard on Intersight GUI.

To deploy a HyperFlex cluster using Intersight, the Cisco UCS Fabric Interconnects that the HyperFlex servers connect to must be first claimed in Cisco Intersight. The Ansible script provided will use the Cisco UCS FI and Intersight information in the Ansible inventory file to:

- Connect to the out-of-band management interfaces on Cisco UCS FI
- Collect the **Claim Code** and **Device ID**
- Connect to Cisco Intersight
- Claim the Cisco UCS Fabric Interconnects as targets in Intersight

Once the FI is claimed, Intersight will discover the HyperFlex servers connected to it. The script will then use the server information to dynamically update the Ansible inventory file. A dummy inventory item must be provided as a placeholder in the inventory file to enable the dynamic inventory update. Ansible and Cisco Intersight will use the HyperFlex server information to deploy and provision the HyperFlex VSI cluster. The script also provides an option to (1) validate the provided configuration (without deploying) or (2) validate and deploy the configuration.

Once the install is kicked off, status of the install can be monitored directly from Intersight. The install itself will take some time (< 1 hour) to complete but most of it is unattended. If any failures occur, the install will stop, and you will have to address the issue and restart from Intersight. The script currently does not have the ability to restart a stalled install. If you have met all the pre-requisites outlined in the Cisco documentation and the pre-install validation checks were successful with no errors/warnings, then the install should complete successfully.

Note: The HyperFlex VSI install will implement best practices and deliver a fully-functional, hyper-converged virtual server infra platform that is ready for deploying applications, either traditional or cloud-native, making it simple and easy for IT and DevOps team to quickly spin up a new environment with compute and storage. The cluster can also be easily expanded, upgraded (full-stack) and generally managed post-deployment from Cisco Intersight.

Procedure 1. Prerequisite – Setup an Ansible Control Node running MacOS

The steps for configuring this and additional information is explained in the **Solution Deployment > Provision Network Infrastructure > Deployment Steps** section of this document.

Procedure 2. Clone Git Hub repository for HyperFlex VSI deployment

To access the Ansible playbooks in the GitHub repository (repo), clone the Git Hub repo as explained in the **Solution Deployment > Provision Network Infrastructure > Deployment Steps** section of this document. Skip this if you’ve already done this for accessing the Ansible network scripts.

The repo is located her: https://github.com/ucs-compute-solutions/Hybrid-Cloud/ in a sub-directory called CVD_HC-OCP-HXFI. The cloning will generate a local copy of the repo in GitHub. The HyperFlex VSI scripts will be in a sub-directory called ‘**compute**.’

Procedure 3. Create Intersight API Keys for API authentication

To use Ansible playbooks to deploy the cluster from Cisco Intersight, Ansible needs API access to Intersight. This access is enabled through Intersight API keys (**API Key ID**, **Secret Key**). The API Key ID is visible and available after the initial key creation. The secret key is an RSA Private Key, and it is only available at API Key creation so it should be saved in a secure location.

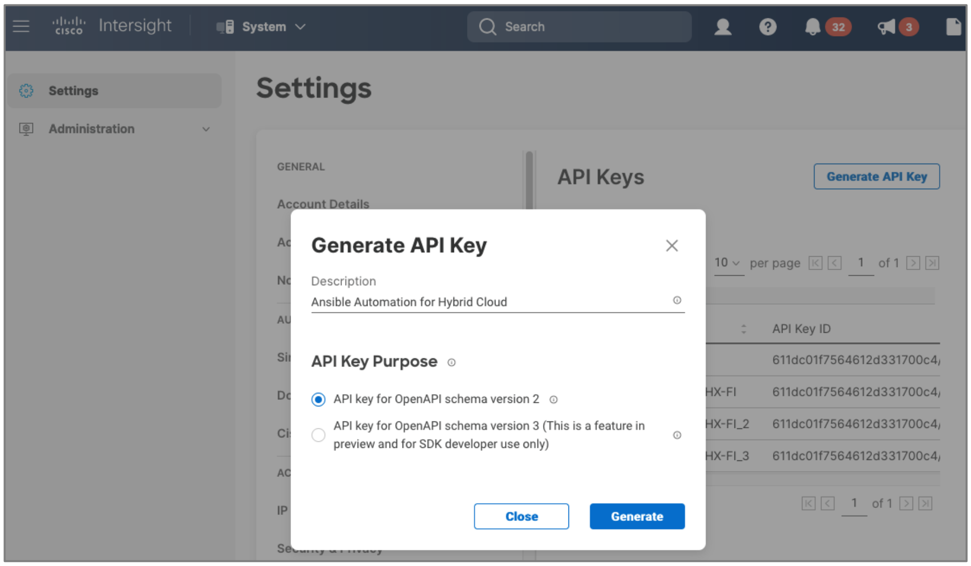
Step 1. Use a web browser and navigate to Intersight.com. Login using an admin account.

Step 2. Select **System** from the drop-down list in the top left-hand side of the Intersight GUI.

Step 3. Select and click **API Keys** in the left-hand navigation pane.

Step 4. Click **Generate API Key** button on the upper-right side of the window.

Step 5. In the **Generate API Key** pop-up window, enter a **Description** for the key, select the radio button for **API key for OpenAPI schema version2** radio button, and click **Generate**.



Generate API Key

Description
Ansible Automation for Hybrid Cloud

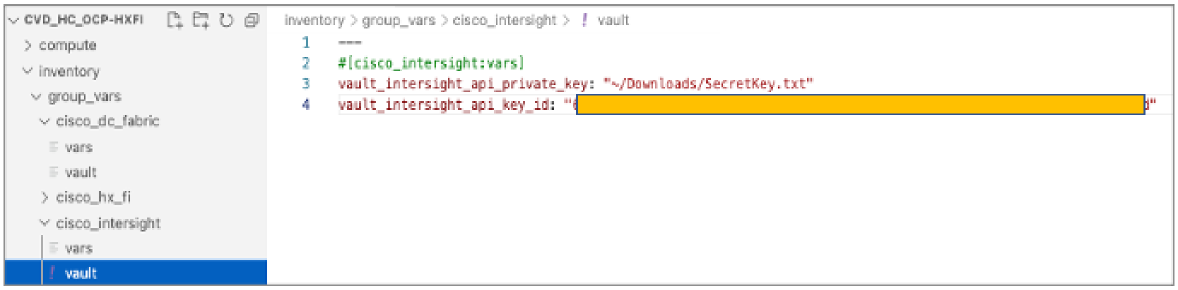
API Key Purpose

● API key for OpenAPI schema version 2

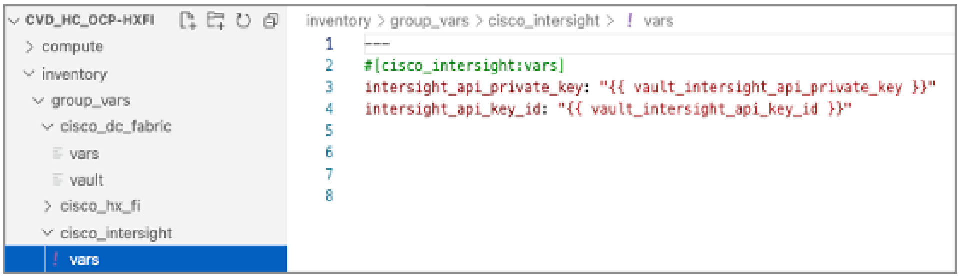
○ API key for OpenAPI schema version 3 (This is a feature in preview and for SDK developer use only)

Close Generate

Step 6. Click on the icon to copy the API Key ID and update the API Key ID variable int in the **inventory/group_vars/cisco_intersight/vault** file**.** The vault will be encrypted, and the Key ID will be referenced by a variable in the **inventory/group_vars/cisco_intersight/vars** file for use by the playbook tasks as shown below.



```
---
#[cisco_intersight:vars]
vault_intersight_api_private_key: "~/Downloads/SecretKey.txt"
vault_intersight_api_key_id: "[redacted]"
```



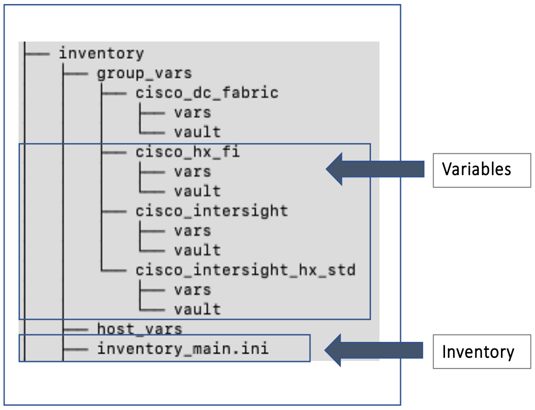
```
---
#[cisco_intersight:vars]
intersight_api_private_key: "{{ vault_intersight_api_private_key }}"
intersight_api_key_id: "{{ vault_intersight_api_key_id }}"
```

Step 7. Click on the icon to save the **Secret Key** to text file. Rename and save the file in a secure location. Update the private key variable in **inventory/group_vars/cisco_intersight/vault** file. The secret key will be referenced by a variable in the **inventory/group_vars/cisco_intersight/vars** file for use by the playbook tasks.

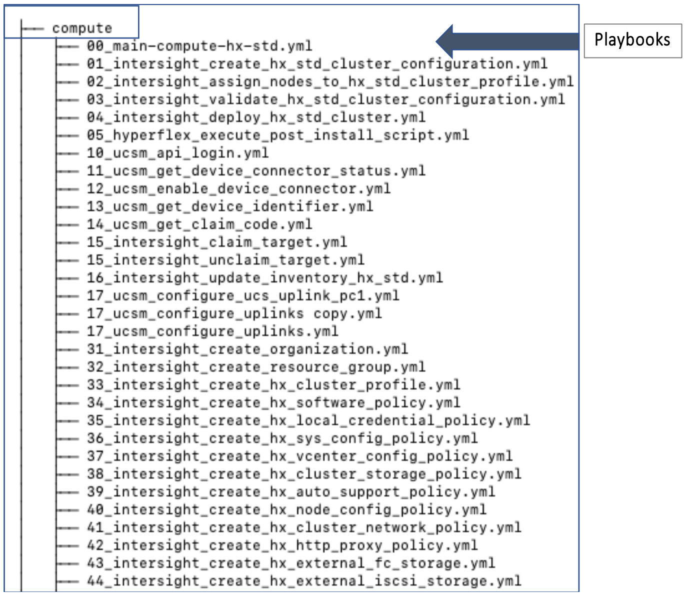
Procedure 4. Review the Ansible files for HyperFlex VSI deployment

Ansible uses inventory files (**inventory.ini, inventory.yml**), variables files (**group_vars, host_vars**), and playbooks or scripts to deploy a HyperFlex VSI cluster using the inventory and input variables provided. The inventory files are the targets of the automation which in this case are Cisco UCS FI (for claiming the servers) and Intersight (t deploy and manage the cluster). Inventory will provide information to connect to the target device. The variables files contain the configuration parameters. The inventory files and variable files will need to be modified for each environment and deployment.

Step 1. Review inventory, variables, and playbooks for deploying a HyperFlex VSI cluster. The Ansible playbooks and the associated inventory and variables are highlighted in the figure below.



```
├── inventory
│   ├── group_vars
│   │   ├── cisco_dc_fabric
│   │   │   ├── vars
│   │   │   └── vault
│   │   ├── cisco_hx_fi
│   │   │   ├── vars
│   │   │   └── vault
│   │   ├── cisco_intersight
│   │   │   ├── vars
│   │   │   └── vault
│   │   └── cisco_intersight_hx_std
│   │       ├── vars
│   │       └── vault
│   ├── host_vars
│   └── inventory_main.ini
```

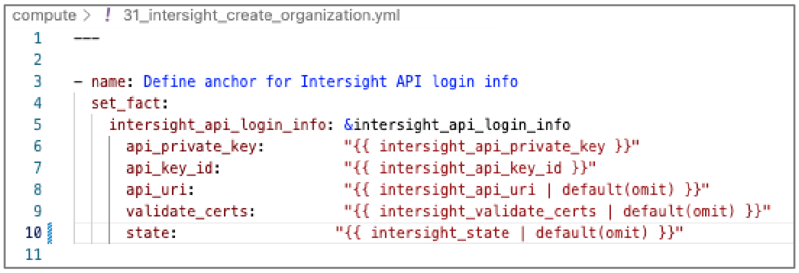


```
├── compute
│   ├── 00_main-compute-hx-std.yml
│   ├── 01_intersight_create_hx_std_cluster_configuration.yml
│   ├── 02_intersight_assign_nodes_to_hx_std_cluster_profile.yml
│   ├── 03_intersight_validate_hx_std_cluster_configuration.yml
│   ├── 04_intersight_deploy_hx_std_cluster.yml
│   ├── 05_hyperflex_execute_post_install_script.yml
│   ├── 10_ucsm_api_login.yml
│   ├── 11_ucsm_get_device_connector_status.yml
│   ├── 12_ucsm_enable_device_connector.yml
│   ├── 13_ucsm_get_device_identifier.yml
│   ├── 14_ucsm_get_claim_code.yml
│   ├── 15_intersight_claim_target.yml
│   ├── 15_intersight_unclaim_target.yml
│   ├── 16_intersight_update_inventory_hx_std.yml
│   ├── 17_ucsm_configure_ucs_uplink_pc1.yml
│   ├── 17_ucsm_configure_uplinks copy.yml
│   ├── 17_ucsm_configure_uplinks.yml
│   ├── 31_intersight_create_organization.yml
│   ├── 32_intersight_create_resource_group.yml
│   ├── 33_intersight_create_hx_cluster_profile.yml
│   ├── 34_intersight_create_hx_software_policy.yml
│   ├── 35_intersight_create_hx_local_credential_policy.yml
│   ├── 36_intersight_create_hx_sys_config_policy.yml
│   ├── 37_intersight_create_hx_vcenter_config_policy.yml
│   ├── 38_intersight_create_hx_cluster_storage_policy.yml
│   ├── 39_intersight_create_hx_auto_support_policy.yml
│   ├── 40_intersight_create_hx_node_config_policy.yml
│   ├── 41_intersight_create_hx_cluster_network_policy.yml
│   ├── 42_intersight_create_hx_http_proxy_policy.yml
│   ├── 43_intersight_create_hx_external_fc_storage.yml
│   ├── 44_intersight_create_hx_external_iscsi_storage.yml
```

The playbooks are in the '**compute**' sub-directory under 'CVD_HC-OCP-HXFI.' The variables and inventory are in the same '**inventory**' sub-directory under 'CVD_HC-OCP-HXFI,' in 'group_vars' directory and 'inventory_main.ini' file, respectively.

Step 2. Review the main playbook file for deploying a HyperFlex VSI cluster. Note that at a high-level, the main playbook consists of:

- Authenticating with Cisco UCS Manager and Cisco Intersight. Authentication information is encrypted using Ansible vault and referenced in the variable files and in the playbook as shown below.



compute > ! 31_intersight_create_organization.yml

```
---

- name: Define anchor for Intersight API login info
  set_fact:
    intersight_api_login_info: &intersight_api_login_info
      api_private_key:         "{{ intersight_api_private_key }}"
      api_key_id:              "{{ intersight_api_key_id }}"
      api_uri:                 "{{ intersight_api_uri | default(omit) }}"
      validate_certs:          "{{ intersight_validate_certs | default(omit) }}"
      state:                  "{{ intersight_state | default(omit) }}"

```

- Connect to Cisco UCS Manager and collect device claim info
- Connect to Cisco Intersight and claim device, update Ansible inventory with HyperFlex server info
- Validate and deploy HyperFlex VSI from Cisco Intersight

Step 3. The playbook tasks are named to reflect their hierarchy within the overall collection of playbooks. The main task is numbered as 00_ , with the first-level tasks below it as 01_ and 02_ . The sub-tasks or the second-level within each are then numbered starting with 011_ and 021_ etc.

Procedure 5. Modify the variables files for executing the Ansible playbooks

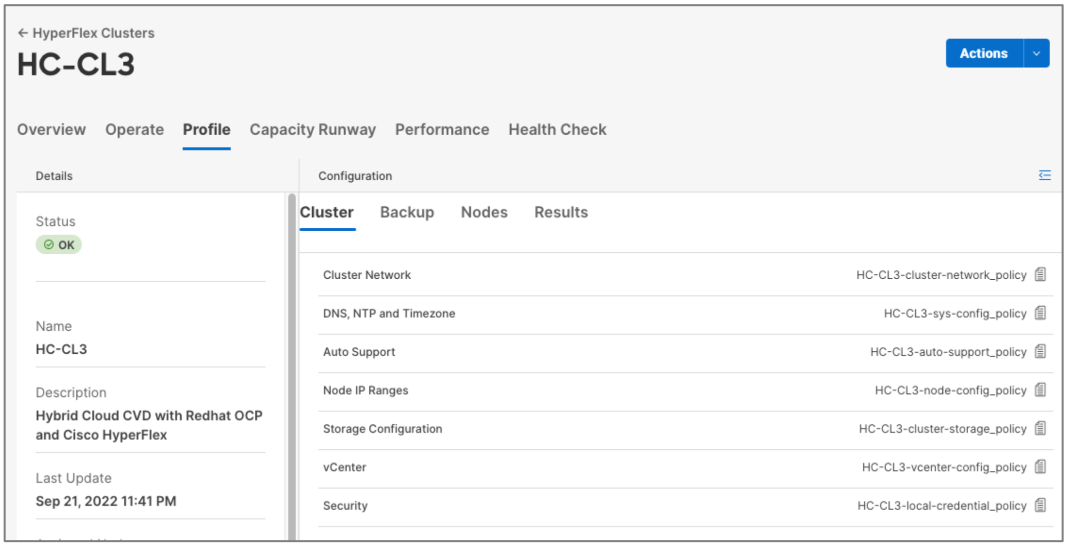
To execute the playbook file, the Ansible variables file is populated with the parameters that will be used to deploy the HyperFlex VSI cluster within the overall collection of playbooks. The configuration variable names, and the parameters used in this solution are provided in Table 9. You can use either a text editor or an IDE environment (for example, Microsoft Visual Studio Code) to update the variables file.

Table 9. HyperFlex VSI Cluster Configuration

Variable Type	Variable	Value	Additional Info
VLAN	hxv-inband-mgmt	1171	HyperFlex iband-management – SCVM, ESXi
VLAN	hxv-cl3-storage-data	1273	HyperFlex storage data – SCVM, ESXi
VLAN	hxv-vmotion	1371	HyperFlex vMotion
VLAN	hxv-vm-network	1521-1530	
KVM	Starting IP	192.168.171.111	
	Ending IP	192.168.171.114	
	Subnet Mask	255.255.255.0	
	Gateway	192.168.171.254	
Jumbo Frames	N/A	Yes	
DNS, NTP and Timezone	Timezone	America/New_York	
	DNS Suffix	hc.com	
	DNS Server (s)	192.168.171.240	
	NTP Server(s)	192.168.171.254	
Auto Support	Auto-Support	Yes	
	Send Service Ticket Notification to	hc-admin@xyz.corp	
Node IP Ranges	Management Network Starting IP	10.1.171.111	
	Management Network Ending IP	10.1.171.114	
	Management Network Subnet Mask	255.255.255.0	
	Management Network Gateway	10.1.171.254	
	SCVM Network Starting IP	10.1.171.161	

Variable Type	Variable	Value	Additional Info
	SCVM Network Ending IP	10.1.171.164	
Storage Configuration	VDI Optimization	No	
	Logical Availability Zones	No	
vCenter	vCenter Server FQDN or IP	vc1-1.hc.com	
	vCenter Username	administrator@vsphere.local	
	vCenter Password	···············	
	vCenter Datacenter Name	HC-App-Site1	
Security	Hypervisor Admin	root	
	The hypervisor on this node uses the factory default password	Yes	
	Hypervisor Password	···············	
	Controller VM Admin Password	···············	
Cluster Profile	Profile Name	HC-CL3	
	Profile Description		
	HyperFlex Management Platform	FI	
	HyperFlex Hypervisor Type	ESXi	
	HyperFlex Management IP	10.1.171.110	
	HyperFlex MAC Prefix	00:25:B5:03	
	HyperFlex WWXN Prefix	20:00:00:25:B5:03	
	HyperFlex Replication Factor	3	

The deployment will result in the following policies and profiles in Cisco Intersight.



← HyperFlex Clusters

HC-CL3

Details

Status
OK

Name
HC-CL3

Description
Hybrid Cloud CVD with Redhat OCP and Cisco HyperFlex

Last Update
Sep 21, 2022 11:41 PM

Configuration

Cluster Network	HC-CL3-cluster-network_policy
DNS, NTP and Timezone	HC-CL3-sys-config_policy
Auto Support	HC-CL3-auto-support_policy
Node IP Ranges	HC-CL3-node-config_policy
Storage Configuration	HC-CL3-cluster-storage_policy
vCenter	HC-CL3-vcenter-config_policy
Security	HC-CL3-local-credential_policy

Procedure 6. Execute the main playbook to deploy the HyperFlex VSI cluster

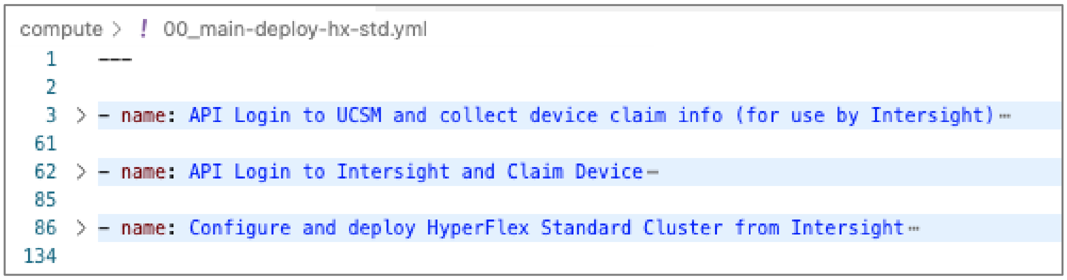
Step 1. Identify the main playbook YAML file in the **compute** sub-directory. This main playbook will have ‘00_’ in the name.

Step 2. Identify the path to the inventory file relative to where you will run the playbook

Step 3. Run the main playbook to deploy HyperFlex VSI: **ansible-playbook** <playbook-name> **-i** <inventory_file_name>

```
(venv1)CVD_HC-OCP-HXFI $ ansible-playbook compute/00_main_deploy_hx-std.yml -i
inventory/inventory_main.ini
```

Step 4. The high-level tasks in the main playbook for HyperFlex VSI is shown below:



compute > ! 00_main-deploy-hx-std.yml

```
---

> - name: API Login to UCSM and collect device claim info (for use by Intersight)⋯

> - name: API Login to Intersight and Claim Device⋯

> - name: Configure and deploy HyperFlex Standard Cluster from Intersight⋯
```

Step 5. At some point during the playbook execution, the script will prompt you for a deployment action i.e., **[1] Validate Only** (without deploying) or **[2] Validate and Deploy.**

Step 6. Choose Option [1] initially and monitor the validation from Cisco Intersight. Verify that there are no errors or warnings, otherwise address them before proceeding. Re-validate multiple times until you’re sure there are no errors. Instead of re-running the validation, you can also choose ‘Continue’ or ‘Retry’ in Intersight.

Step 7. When the validation is successful with no errors or warning, run the playbook again but using Option {2] this time.

Procedure 7. Monitor the validation and deployment from Cisco Intersight

Once the script kicks off the install with the configuration parameters provided, you should monitor the deployment from Cisco Intersight

Step 1. From Intersight, select **Infrastructure Service** from the drop-down list near the top left side of the GUI

Step 2. Select **Operate > HyperFlex Clusters** and select the Application cluster (in this case, HC-CL3) from the list.

Step 3. Navigate to **Overview > Events > Requests.** You should see the install that was just kicked off. Select and click on it to observe the validation and deployment process.

Step 4. The figure below shows the Intersight GUI during the install process. The Intersight GUI has since been refreshed so the UI will look different for your install, but the information provided during this process should be similar.

Procedure 8. Complete post-install tasks – run post-install script

When the installation is complete, a few post-install configuration and best-practices should be implemented using a post-install script. The script should be run before deploying any production workloads on the cluster. The script can:

- License the hosts in VMware vCenter
- Enable HA/DRS on the cluster in VMware vCenter
- Suppress SSH/Shell warnings in VMware vCenter
- Configure vMotion in VMware vCenter
- Configure additional guest VM networks and port-groups
- Perform HyperFlex Health check

Step 1. SSH into Cluster Management IP of the HyperFlex Cluster. Login using the 'admin' account and Storage Controller VM password provided during installation.

Step 2. Run the following command in the shell, and press enter: **hx_post_install**

Step 3. Select the first post_install workflow type – **1. New/Existing Cluster**.

Step 4. Enter the HyperFlex Storage Controller VM root password for the cluster (the one entered during installation).

Step 5. Enter the vCenter server username and password.

Step 6. Enter ESXi host root password (the one entered during installation).

Step 7. You must license the vSphere hosts through the script or complete this task in vCenter before continuing. You will need a valid license or HA/DRS in the next step will result in an error. Enter "n" if you have already deployed a license in vCenter.

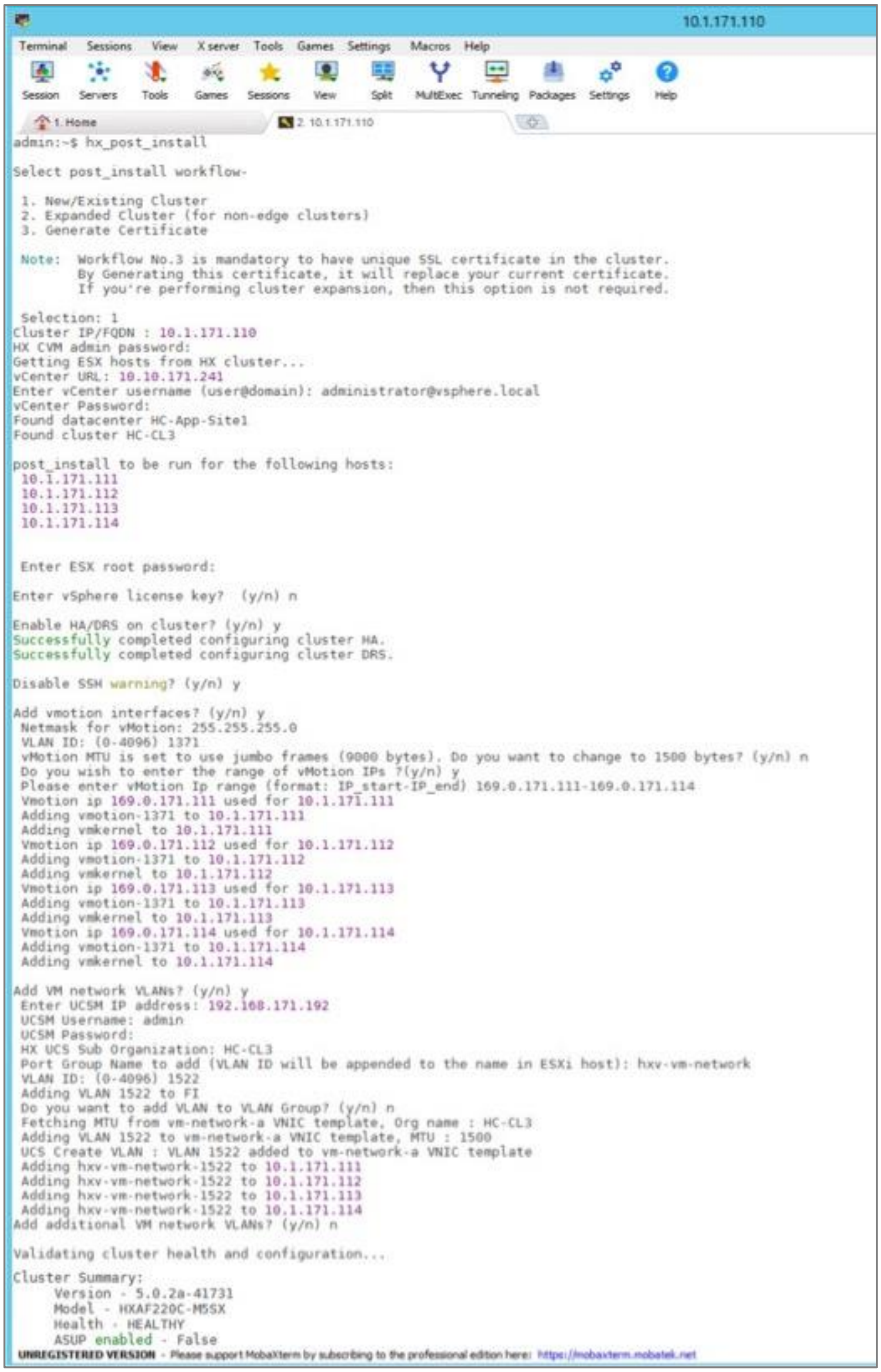
Step 8. Enter "**y**" to enable HA/DRS if you have the appropriate licensing to enable these features.

Step 9. Enter "**y**" to disable the ESXi hosts' SSH warning.

Step 10. Add the vMotion VMkernel interfaces to each node by entering "**y.**" Input the netmask, the vMotion VLAN ID, plus a starting and ending vMotion IP address range to be used by the hosts. The script will assign the addresses in sequential order.

Step 11. You can add more VM network port groups for guest VM traffic via the script. Enter "n" to skip this step. If desired, enter "y" and enter the information for the additional port groups and VLAN IDs. The VM network port groups will be created and added to the **vm-network** vSwitch. This step will add identical network configuration to all nodes in the cluster.

Step 12. The script will run a health check on the cluster and display cluster summary – confirm the cluster is healthy.


10.1.171.110
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
1. Home
2. 10.1.171.110

```
admin:~$ hx_post_install

Select post_install workflow-

 1. New/Existing Cluster
 2. Expanded Cluster (for non-edge clusters)
 3. Generate Certificate

 Note:  Workflow No.3 is mandatory to have unique SSL certificate in the cluster.
        By Generating this certificate, it will replace your current certificate.
        If you're performing cluster expansion, then this option is not required.

 Selection: 1
Cluster IP/FQDN : 10.1.171.110
HX CVM admin password:
Getting ESX hosts from HX cluster...
vCenter URL: 10.10.171.241
Enter vCenter username (user@domain): administrator@vsphere.local
vCenter Password:
Found datacenter HC-App-Site1
Found cluster HC-CL3

post_install to be run for the following hosts:
 10.1.171.111
 10.1.171.112
 10.1.171.113
 10.1.171.114


 Enter ESX root password:

Enter vSphere license key?  (y/n) n

Enable HA/DRS on cluster? (y/n) y
Successfully completed configuring cluster HA.
Successfully completed configuring cluster DRS.

Disable SSH warning? (y/n) y

Add vmotion interfaces? (y/n) y
 Netmask for vMotion: 255.255.255.0
 VLAN ID: (0-4096) 1371
 vMotion MTU is set to use jumbo frames (9000 bytes). Do you want to change to 1500 bytes? (y/n) n
 Do you wish to enter the range of vMotion IPs ?(y/n) y
 Please enter vMotion Ip range (format: IP_start-IP_end) 169.0.171.111-169.0.171.114
 Vmotion ip 169.0.171.111 used for 10.1.171.111
 Adding vmotion-1371 to 10.1.171.111
 Adding vmkernel to 10.1.171.111
 Vmotion ip 169.0.171.112 used for 10.1.171.112
 Adding vmotion-1371 to 10.1.171.112
 Adding vmkernel to 10.1.171.112
 Vmotion ip 169.0.171.113 used for 10.1.171.113
 Adding vmotion-1371 to 10.1.171.113
 Adding vmkernel to 10.1.171.113
 Vmotion ip 169.0.171.114 used for 10.1.171.114
 Adding vmotion-1371 to 10.1.171.114
 Adding vmkernel to 10.1.171.114

Add VM network VLANs? (y/n) y
 Enter UCSM IP address: 192.168.171.192
 UCSM Username: admin
 UCSM Password:
 HX UCS Sub Organization: HC-CL3
 Port Group Name to add (VLAN ID will be appended to the name in ESXi host): hxv-vm-network
 VLAN ID: (0-4096) 1522
 Adding VLAN 1522 to FI
 Do you want to add VLAN to VLAN Group? (y/n) n
 Fetching MTU from vm-network-a VNIC template, Org name : HC-CL3
 Adding VLAN 1522 to vm-network-a VNIC template, MTU : 1500
 UCS Create VLAN : VLAN 1522 added to vm-network-a VNIC template
 Adding hxv-vm-network-1522 to 10.1.171.111
 Adding hxv-vm-network-1522 to 10.1.171.112
 Adding hxv-vm-network-1522 to 10.1.171.113
 Adding hxv-vm-network-1522 to 10.1.171.114
Add additional VM network VLANs? (y/n) n

Validating cluster health and configuration...

Cluster Summary:
     Version - 5.0.2a-41731
     Model - HXAF220C-M5SX
     Health - HEALTHY
     ASUP enabled - False
 UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
```

Procedure 9. Post-install tasks – Create datastores

To use the cluster, at least one datastore should be created before a Red Hat OCP cluster can be deployed on the HyperFlex VSI cluster. This task can be completed from Cisco Intersight.

Step 1. Use a web browser and navigate to Intersight.com and login using an admin account.

Step 2. Select **Infrastructure Services** from the drop-down list in the top left side of the Intersight GUI.

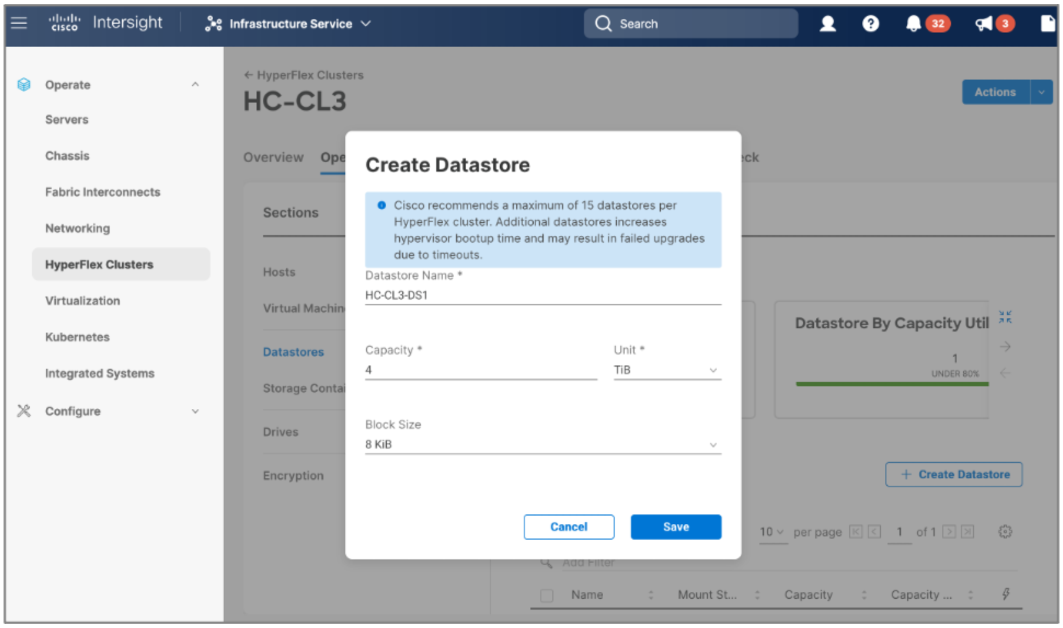
Step 3. Click **HyperFlex Clusters** in the left navigation bar

Step 4. Select the Application HyperFlex cluster (in this case, **HC-CL3**) from the list in the right window.

Step 5. Navigate to **Operate > Datastores**.

Step 6. Click on **Create Datastore** from the right window.

Step 7. In the **Create Datastore** pop-up window, specify a **Datastore Name** and size (**Capacity)**. For most applications, leave the **Block Size** at the default (8k). Only dedicated Virtual Desktop Infrastructure (VDI) environments should choose 4K Block Size.



Create Datastore

Cisco recommends a maximum of 15 datastores per HyperFlex cluster. Additional datastores increases hypervisor bootup time and may result in failed upgrades due to timeouts.

Datastore Name * HC-CL3-DS1

Capacity * 4

Unit * TiB

Block Size 8 KiB

Step 8. Click **Save**.

The HyperFlex VSI is now ready for deploying the Red Hat OpenShift Container Platform as outlined in the next section.

Deploy Red Hat OpenShift Container Platform (On-Prem)

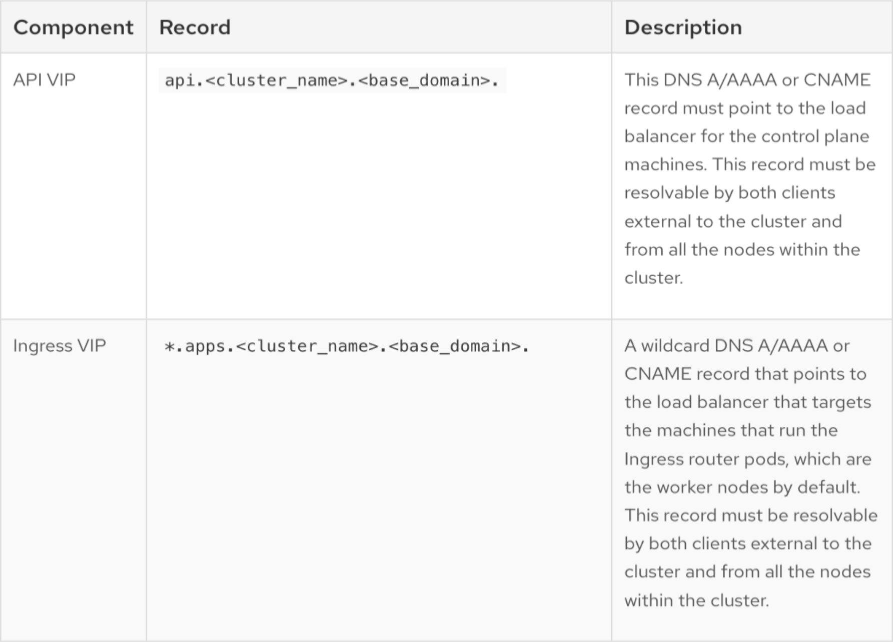
This section describes the deployment of the on-prem Kubernetes environment in this hybrid cloud solution. The Kubernetes environment deployed in the solution is Red Hat OpenShift Container Platform, deployed and managed from the cloud using Red Hat Hybrid Cloud Console. The cluster is deployed using the recommended installer-provisioned infrastructure (IPI) method. The installer will prompt the user for minimal values to collect information that is environment and deployment specific. Red Hat does support customizations, but the IPI is the fastest way to deploy a production-ready OCP cluster. Other methods are outside the scope of this document

Prerequisites

The prerequisites for deploying Red Hat OCP are:

- Management infrastructure for deploying an installer workstation. The installer will need access to the Internet, HyperFlex VSI cluster, and VMware vCenter managing the vSphere cluster. Installer is supported on Linux and MacOS. Post-deployment, the installer will be used for SSH access and other management functions.
- Application HyperFlex VSI running VMware vSphere to host the Red Hat OCP cluster. The cluster should be fully licensed with vSphere HA/DRS enabled to ensure availability of the virtual machines that will serve as control and worker nodes in the Red Hat OCP cluster. Also, the cluster must be provisioned for at least one datastore for the VMs deployed by OCP.
- A valid Red Hat account to login to Red Hat Hybrid Cloud Console. Hybrid Cloud Console is used to centrally deploy and manage the on-prem and public cloud OCP clusters in the Enterprise.

- VLAN and IP subnet for the Red Hat OCP cluster. OCP cluster is deployed on a guest VM network on HyperFlex VSI. All nodes in an OCP cluster must be in the same VLAN. The VLAN must be provisioned in the ACI fabric, Cisco UCS Fabric Interconnects (port-channel uplinks), HyperFlex server (vNIC templates), and VMware ESXi hosts. A guest VM network deployed during HyperFlex VSI install will be used for the Red Hat OCP cluster. If the guest VLAN is provisioned using the HyperFlex installer or the post-install scripts, it will take care of configuring the VLAN across the different component, but the Cisco ACI fabric will still need to be provisioned.
- The ACI fabric will be the default gateway for the IP subnet allocated to the guest VM network and OCP cluster. The OCP clusters will use the ACI fabric for reachability to destinations outside the cluster based on established contracts. The ACI fabric will provide Internet access which will provide reachability Red Hat Hybrid Cloud Console, quay.io, Operator Hub and other services in the cloud. It will also enable the default Telemetry service running on the cluster to automatically register with the Hybrid Cloud Console. Telemetry service also provides remote health monitoring. The ACI fabric will also provide reachability to network services (DNS, DHCP, NTP), VMware vCenter, and Installer workstation hosted on a management cluster in the ACI fabric. Post-install, the installer will be used for SSH and debugging purposes and therefore will continue to need access to the OCP cluster.
- Installer requires two static IP addresses: [1] **API** address to access the cluster API and [2] **Ingress** address for cluster ingress traffic.
- DNS – The following DNS records for the two static IP addresses must be in place prior to install.



Component	Record	Description
API VIP	`api.<cluster_name>.<base_domain>.`	This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.
Ingress VIP	`*.apps.<cluster_name>.<base_domain>.`	A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

- DHCP is required to provide IP addresses to the cluster nodes (control/master, compute/worker). The DHCP server must be available and provisioned with a DHCP scope for the OCP subnet before the OCP cluster installation can begin.
- NTP – OCP cluster nodes should be configured for NTP, ideally during DHCP process.
- SSH Access – To debug the installation and for disaster recovery and other post-install activities, the SSH public keys must be provided to the OCP installer to authenticate the access. The key will be passed to the nodes through the initial configuration (ignition) files. The nodes will add the keys to the **~/.ssh/authorized_keys** list for the **core** user to enable password-less authentication.

- VMware vCenter root CA certificates – To install the OCP cluster, the installer needs access to the VMware vCenter API. For this, the vCenter's root CA certificates must be added to the system trust on the OCP installer workstation.

Setup Information

Table 10 lists the installation parameters for the on-prem deployment.

Table 10. Red Hat OCP - Installation Parameters for on-prem deployment

Variable	Variable Name	Value	Additional Info
DNS Server IP	-	10.10.171.240	
Base DNS Domain		hc.com	
Red Hat OCP Cluster Name	-	ocp11.hc.com	
API VIP	api.ocp11.hc.com	10.171.11.252	
Ingress VIP	*.apps.ocp11.hc.com	10.171.11.253	
DHCP Server IP	-	10.10.171.240	
DHCP Scope	-	10.171.11.1 - 10.171.11.250	
NTP Server IP		192.168.171.254	
Default Gateway IP	-	10.171.11.254	
VMware vCenter IP (& Login)	vc1-1.hc.com	10.10.171.241	Login Credentials
vCenter Datacenter Name	-	HC-App-Site1	For use by OCP Installer
vSphere Cluster Name	-	HC-CL3	For use by OCP Installer
vSphere Datastore Name	-	HC-CL3-DS1	For use by OCP Installer
vSphere Virtual Network	Hxv-vm-network-1521	VLAN 1521	For use by OCP Installer

Deployment Steps

The section provides the procedures for deploying a Red Hat OCP cluster on a VMware vSphere HyperFlex cluster.

Procedure 1. Generate a key pair for SSH access to Red Hat OCP cluster

The commands you need are:

Commands

```
ssh-keygen -t rsa -N '' -f <path>/<file_name>
```

Commands

```
eval "$(ssh-agent -s)"
```

```
ssh-add <path>/<file_name>
```

Step 1. On the installer workstation running a Linux operating system, use the following command. You can generate the key using **rsa** or **edcsa** algorithm.

```
ssh-keygen -t rsa -N '' -f ~/.ssh/id_rsa
```

Step 1. Add the SSH private key identity to the SSH agent for your local user. If the **`ssh-agent`** process is not already running for your local user, start it as a background task:

```
eval "$(ssh-agent -s)"
```

Step 2. Add your SSH private key to the **`ssh-agent`** using the command:

```
ssh-add ~/.ssh/id_rsa
```

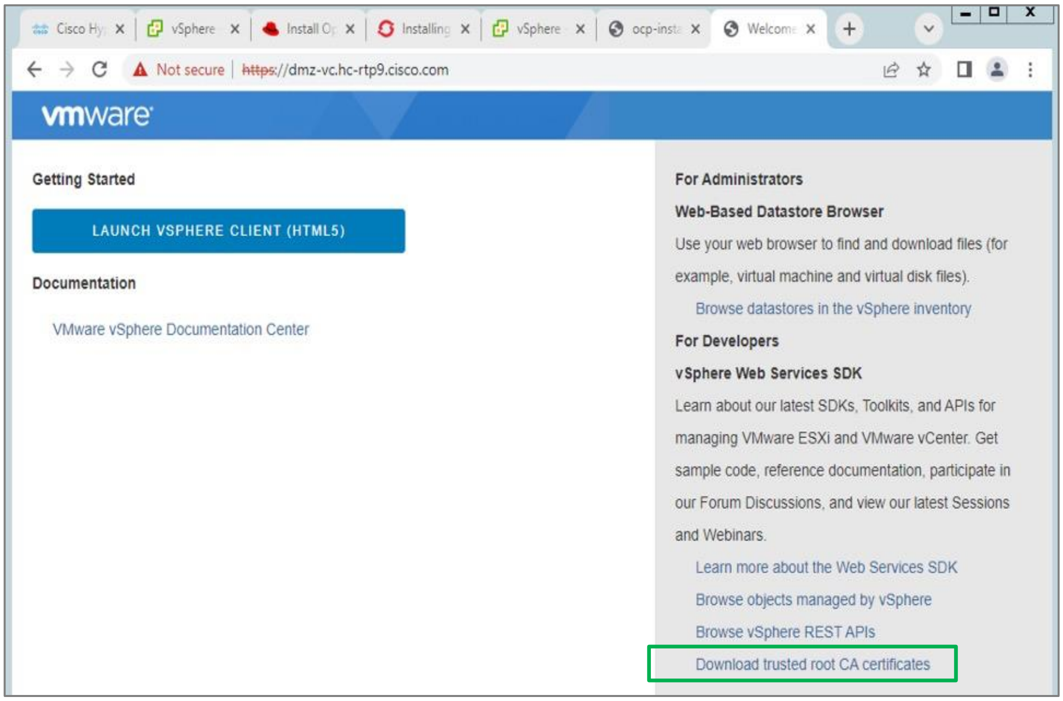
The above key is provided as input to the installer. Installer will add it to the ignition files that are used to do the initial configuration of the OCP nodes. Once the OCP cluster is deployed, you will be able to access the cluster as user '**core'** without the need for password.

Procedure 2. Download vCenter's root CA Certificates to OCP installer's system trust

Note: The installer needs API access to VMware vCenter to deploy the OCP cluster.

Step 1. Use a web browser to navigate to VMware vCenter. Login using an admin account.

Step 2. From the vCenter home page, select and click **Download trusted root CA certificates** from the bottom left side of the window. A **download.zip** file downloads.



Getting Started

Documentation

VMware vSphere Documentation Center

For Administrators

Web-Based Datastore Browser

Use your web browser to find and download files (for example, virtual machine and virtual disk files).

Browse datastores in the vSphere inventory

For Developers

vSphere Web Services SDK

Learn about our latest SDKs, Toolkits, and APIs for managing VMware ESXi and VMware vCenter. Get sample code, reference documentation, participate in our Forum Discussions, and view our latest Sessions and Webinars.

Learn more about the Web Services SDK

Browse objects managed by vSphere

Browse vSphere REST APIs

Download trusted root CA certificates

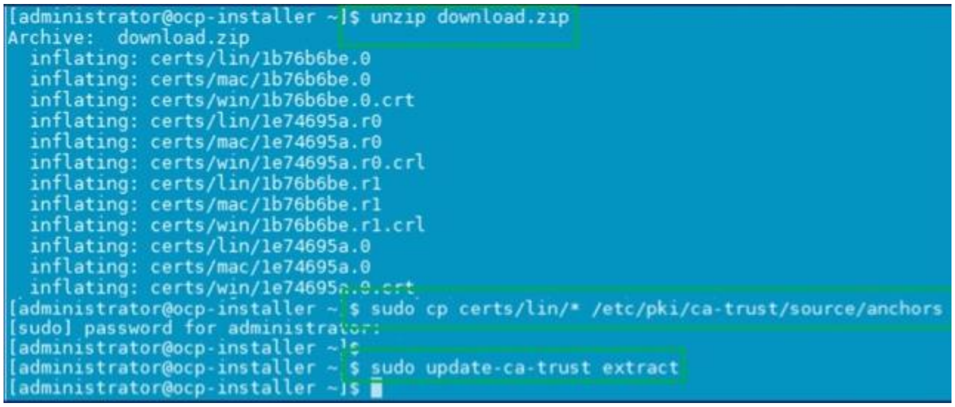
Step 3. Extract the vCenter root CA certificates from the downloaded file.

Step 4. Copy the certificates to the system trust for the operating system running on your workstation.

```
cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

Step 5. Update the system trust on your workstation.

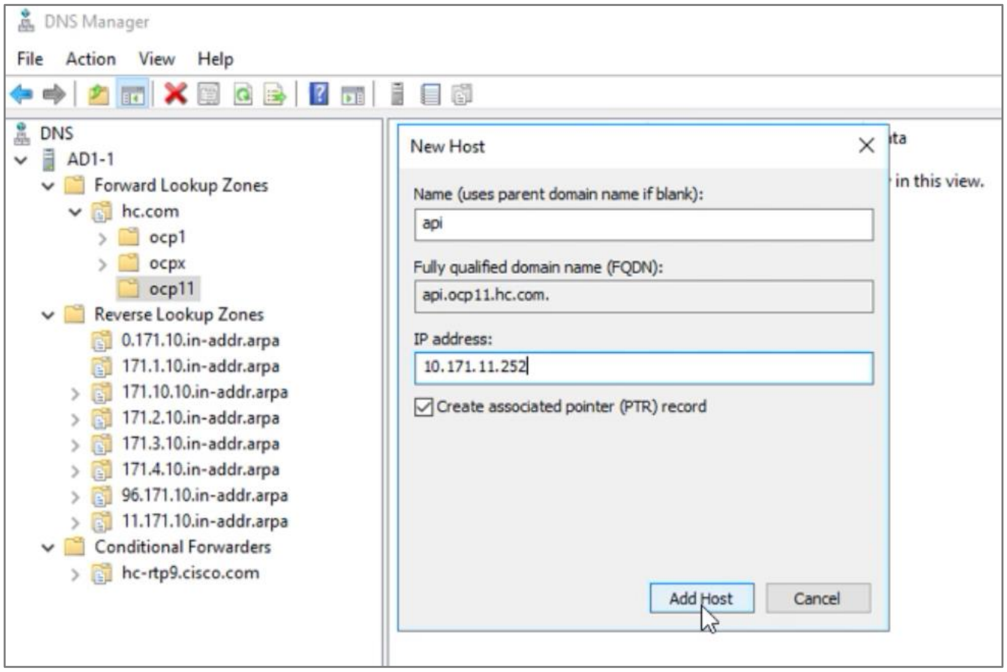
```
        update-ca-trust extract
```



```
[administrator@ocp-installer ~]$ unzip download.zip
Archive:  download.zip
  inflating: certs/lin/1b76b6be.0
  inflating: certs/mac/1b76b6be.0
  inflating: certs/win/1b76b6be.0.crt
  inflating: certs/lin/1e74695a.r0
  inflating: certs/mac/1e74695a.r0
  inflating: certs/win/1e74695a.r0.crl
  inflating: certs/lin/1b76b6be.r1
  inflating: certs/mac/1b76b6be.r1
  inflating: certs/win/1b76b6be.r1.crl
  inflating: certs/lin/1e74695a.0
  inflating: certs/mac/1e74695a.0
  inflating: certs/win/1e74695a.0.crt
[administrator@ocp-installer ~ $ sudo cp certs/lin/* /etc/pki/ca-trust/source/anchors
[sudo] password for administrator:
[administrator@ocp-installer ~]$
[administrator@ocp-installer ~ $ sudo update-ca-trust extract
[administrator@ocp-installer ~]$ 
```

Procedure 3. Create DNS records for the Red Hat OCP cluster

Step 1. Create the API and Ingress DNS records required to deploy the OCP cluster. The records on a windows DNS server are shown below.

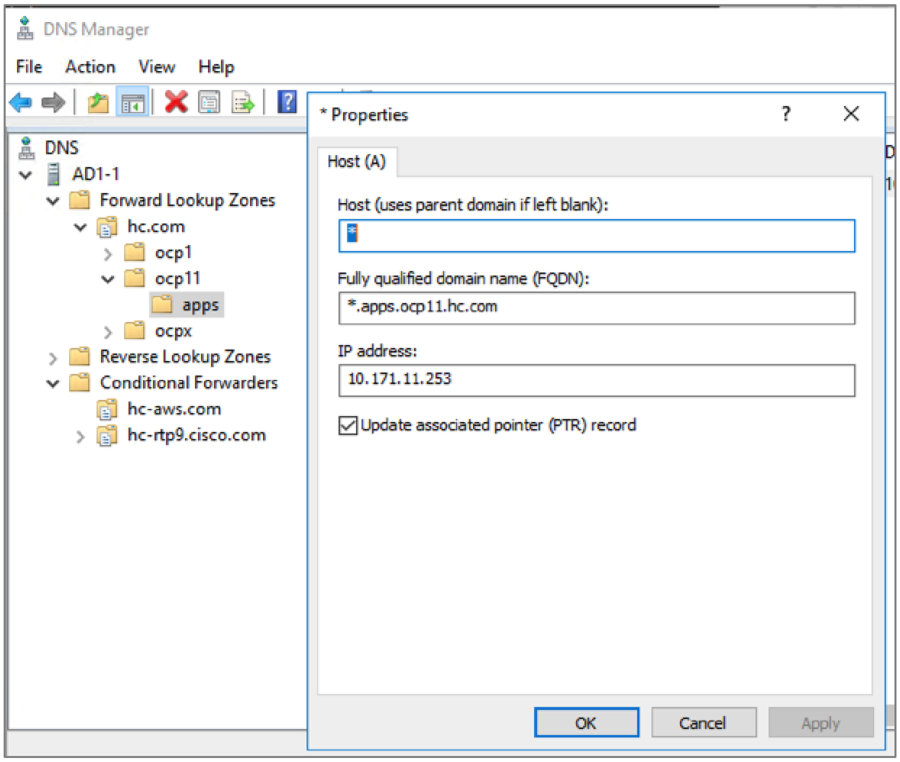


Name (uses parent domain name if blank): api

Fully qualified domain name (FQDN): api.ocp11.hc.com.

IP address: 10.171.11.252

☑ Create associated pointer (PTR) record



Host (uses parent domain if left blank): *

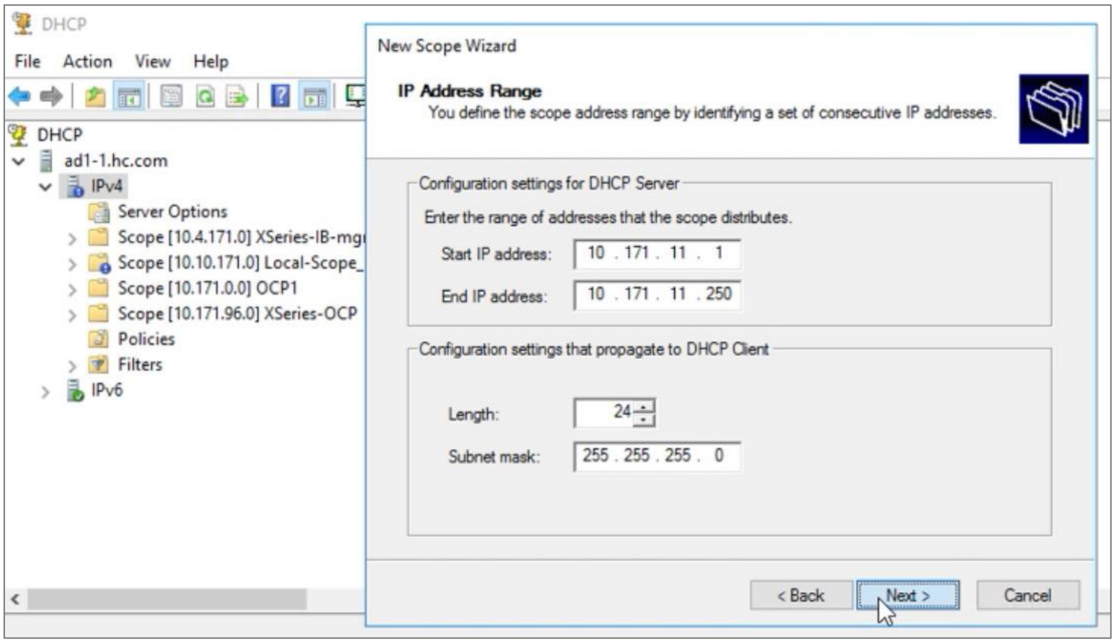
Fully qualified domain name (FQDN): *.apps.ocp11.hc.com

IP address: 10.171.11.253

☑ Update associated pointer (PTR) record

Procedure 4. Create a DHCP scope for the Red Hat OCP cluster

Step 1. Create a DHCP scope for the OCP cluster with scope options for Gateway, DNS server, base DNS domain and NTP. The DHCP configuration on a Windows DHCP server is shown below.



IP Address Range

You define the scope address range by identifying a set of consecutive IP addresses.

Configuration settings for DHCP Server

Enter the range of addresses that the scope distributes.

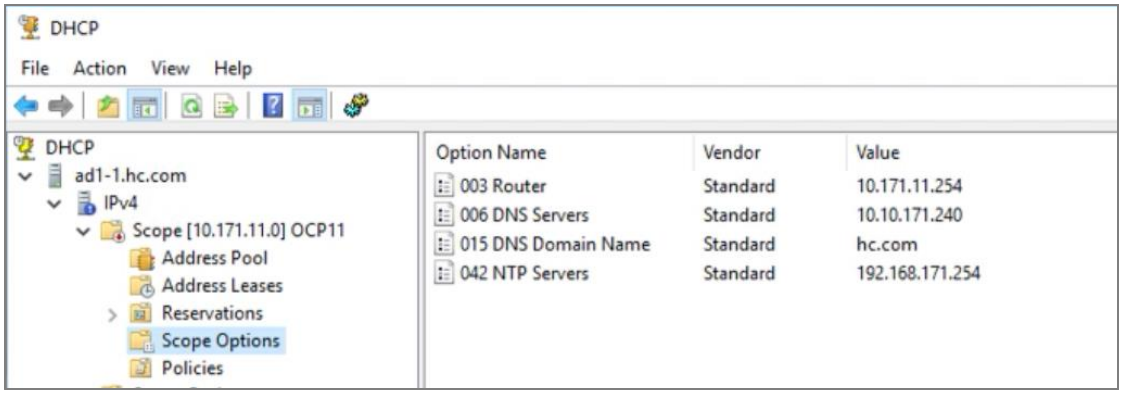
Start IP address: 10 . 171 . 11 . 1

End IP address: 10 . 171 . 11 . 250

Configuration settings that propagate to DHCP Client

Length: 24

Subnet mask: 255 . 255 . 255 . 0



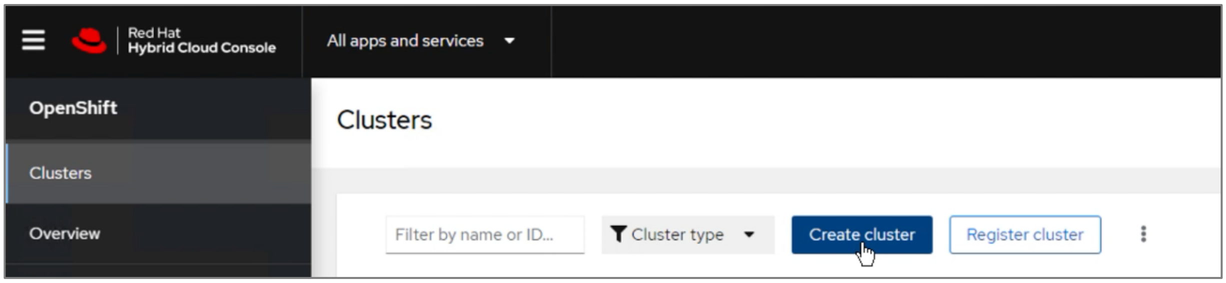
Option Name	Vendor	Value
003 Router	Standard	10.171.11.254
006 DNS Servers	Standard	10.10.171.240
015 DNS Domain Name	Standard	hc.com
042 NTP Servers	Standard	192.168.171.254

Procedure 5. Download the Red Hat OCP installer and other tools from Red Hat Hybrid Cloud Console

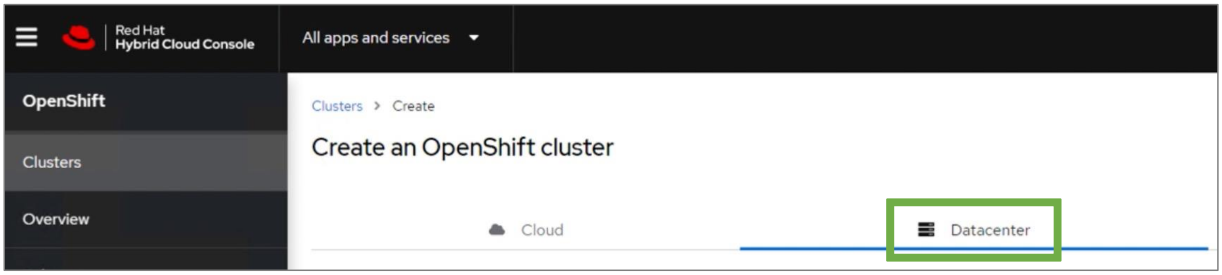
Step 1. Use a web browser and navigate to Red Hat Hybrid Cloud Console at console.redhat.com. Login to your Red hat account.

Step 2. From the left navigation pane, select and click on **OpenShift**.

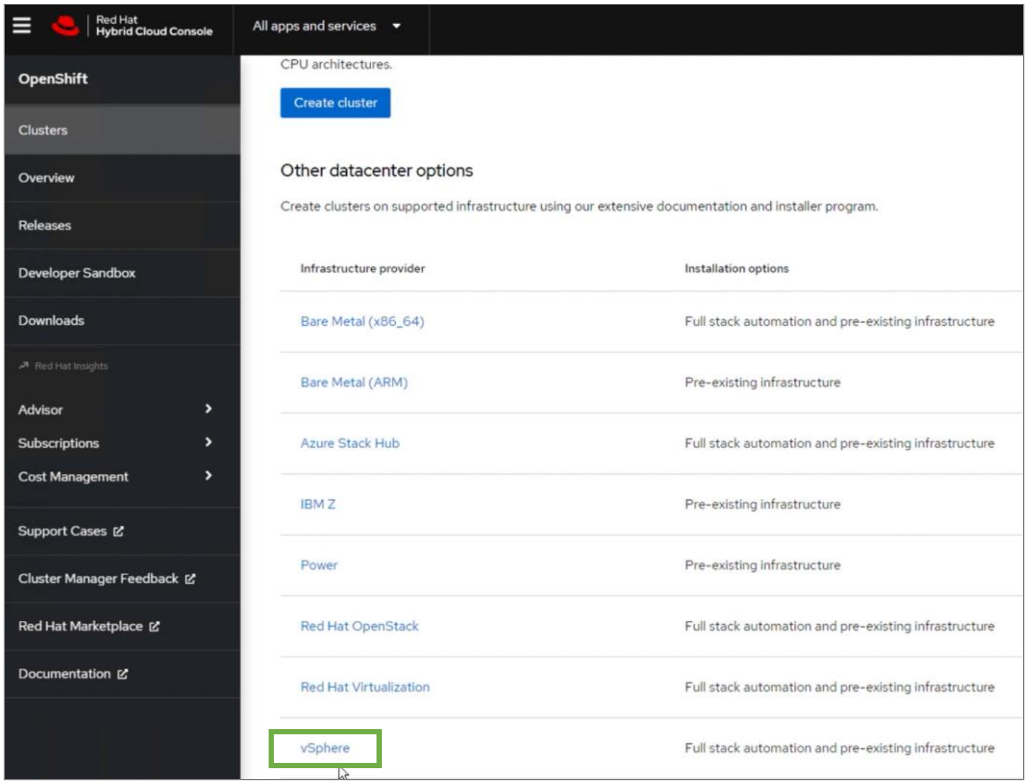
Step 3. Navigate to **Clusters** and click on **Create Cluster**.



Step 4. Navigate to the **Data Center** tab.



Step 5. Scroll down and click on VMware vSphere for the infrastructure provider.



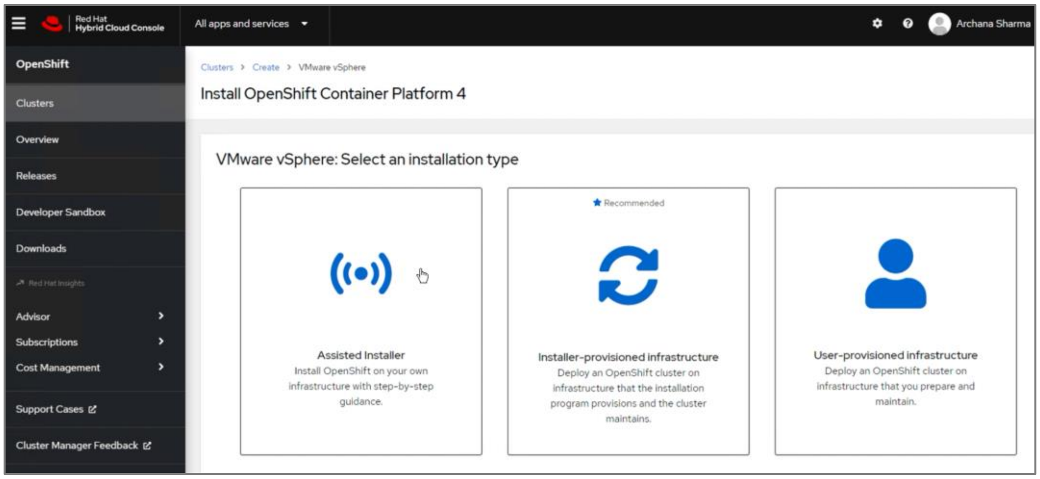
CPU architectures.

Other datacenter options

Create clusters on supported infrastructure using our extensive documentation and installer program.

Infrastructure provider	Installation options
Bare Metal (x86_64)	Full stack automation and pre-existing infrastructure
Bare Metal (ARM)	Pre-existing infrastructure
Azure Stack Hub	Full stack automation and pre-existing infrastructure
IBM Z	Pre-existing infrastructure
Power	Pre-existing infrastructure
Red Hat OpenStack	Full stack automation and pre-existing infrastructure
Red Hat Virtualization	Full stack automation and pre-existing infrastructure
vSphere	Full stack automation and pre-existing infrastructure

Step 6. Select the Installer-Provisioned Infrastructure (IPI) method.



Install OpenShift Container Platform 4

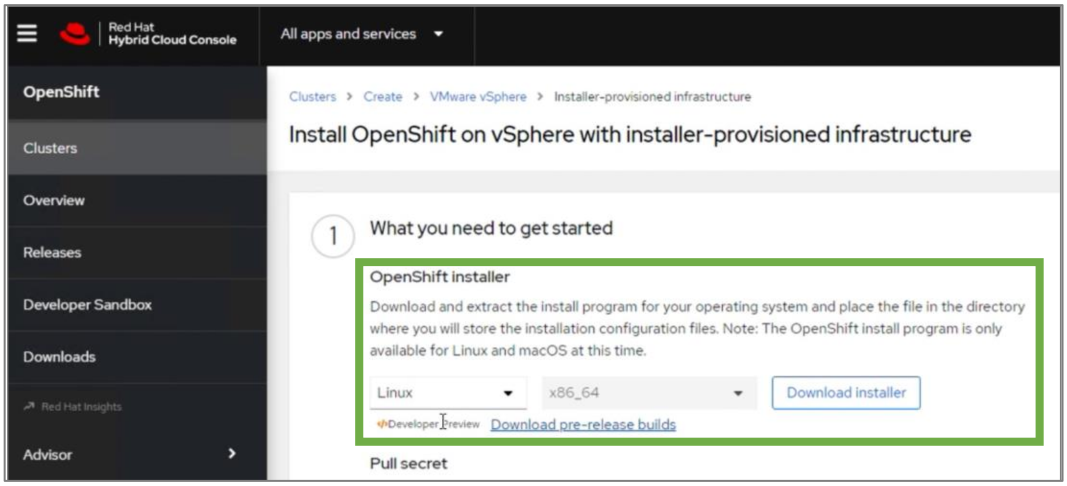
VMware vSphere: Select an installation type

Assisted Installer
Install OpenShift on your own infrastructure with step-by-step guidance.

Installer-provisioned infrastructure
Deploy an OpenShift cluster on infrastructure that the installation program provisions and the cluster maintains.

User-provisioned infrastructure
Deploy an OpenShift cluster on infrastructure that you prepare and maintain.

Step 7. From the VMware vSphere infrastructure page, download the installation program for the operating system running on the installer workstation.



Install OpenShift on vSphere with installer-provisioned infrastructure

1 What you need to get started

OpenShift installer

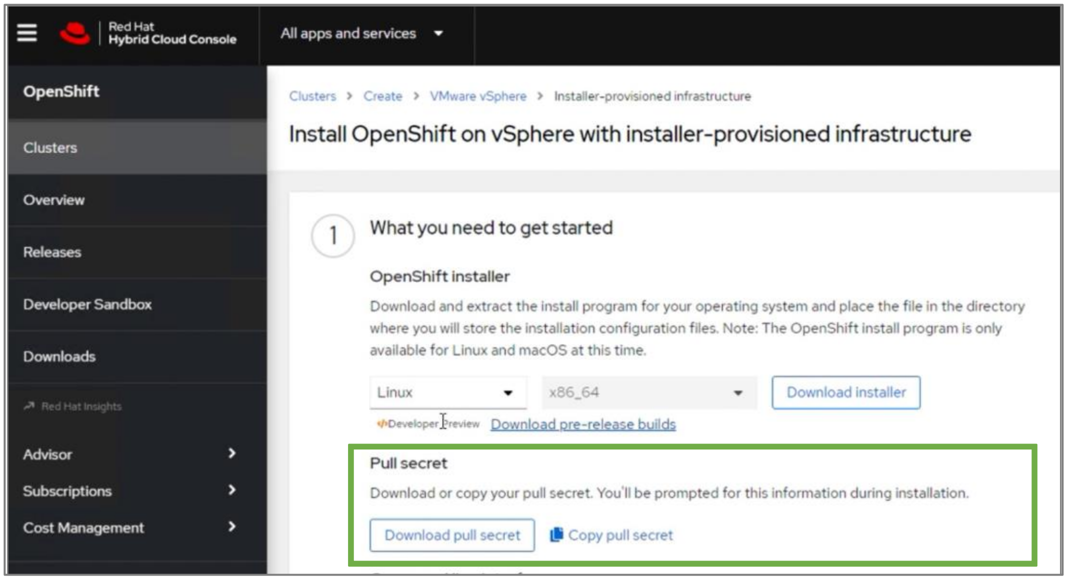
Download and extract the install program for your operating system and place the file in the directory where you will store the installation configuration files. Note: The OpenShift install program is only available for Linux and macOS at this time.

Step 8. On the installer workstation, create a directory for the cluster (in this case, **ocp11**) and move the installation package to this directory.

Step 9. Extract the installation package.

```
tar -xvf openshift-install-linux.tar.gz
```

Step 10. Copy the pull-secret and save it in a file in the same directory. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.



Install OpenShift on vSphere with installer-provisioned infrastructure

1 What you need to get started

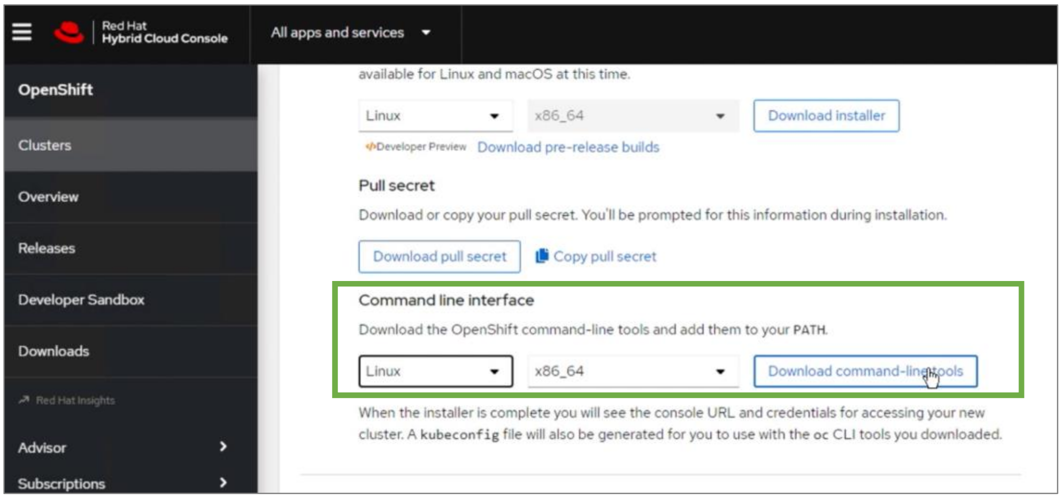
OpenShift installer

Download and extract the install program for your operating system and place the file in the directory where you will store the installation configuration files. Note: The OpenShift install program is only available for Linux and macOS at this time.

Pull secret

Download or copy your pull secret. You'll be prompted for this information during installation.

Step 11. Download the OpenShift CLI tools to the same directory.



available for Linux and macOS at this time.

Pull secret

Download or copy your pull secret. You'll be prompted for this information during installation.

Command line interface

Download the OpenShift command-line tools and add them to your PATH.

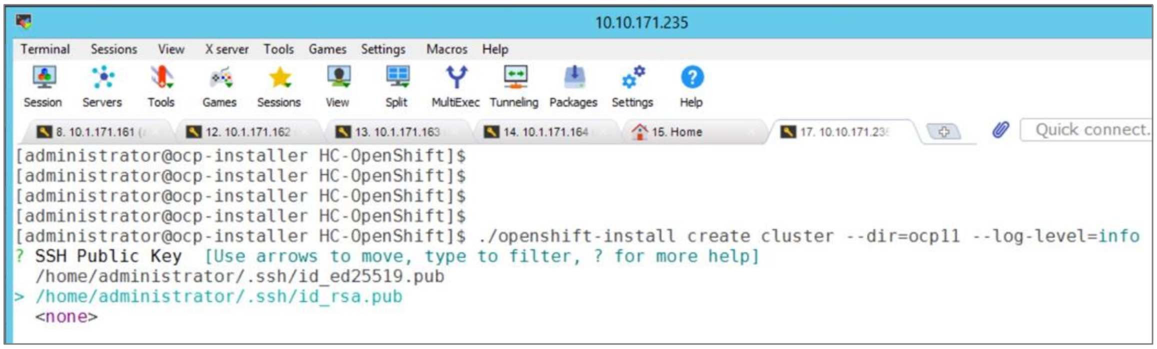
When the installer is complete you will see the console URL and credentials for accessing your new cluster. A kubeconfig file will also be generated for you to use with the oc CLI tools you downloaded.

Procedure 6. Install Red Hat OpenShift Container Platform

Step 1. Run this command to run the installer:

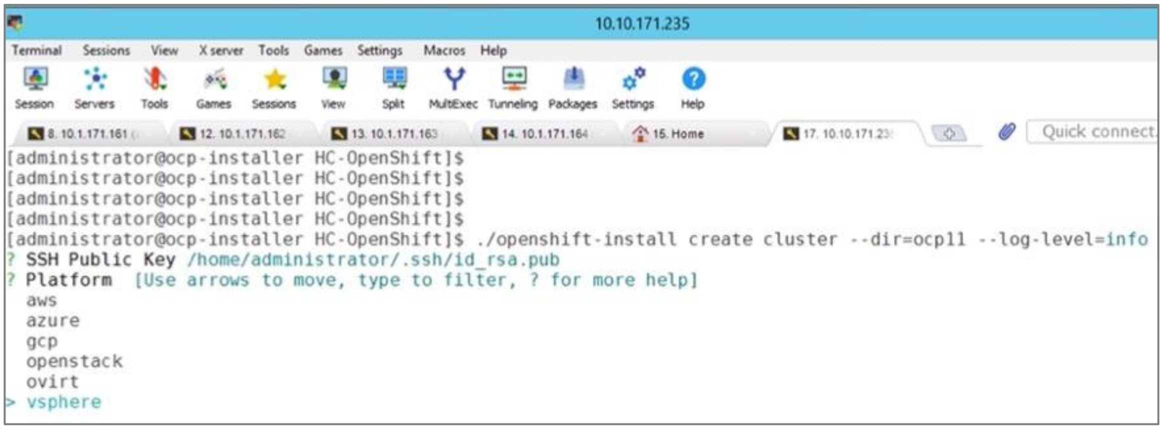
```
./openshift-install create cluster --dir=<installation_directory> --log-level=info
```

Step 2. Select the SSH public key to pass to the cluster nodes through the installation configuration (ignition) file.


10.10.171.235
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
8. 10.1.171.161
12. 10.1.171.162
13. 10.1.171.163
14. 10.1.171.164
15. Home
17. 10.10.171.23
Quick connect...

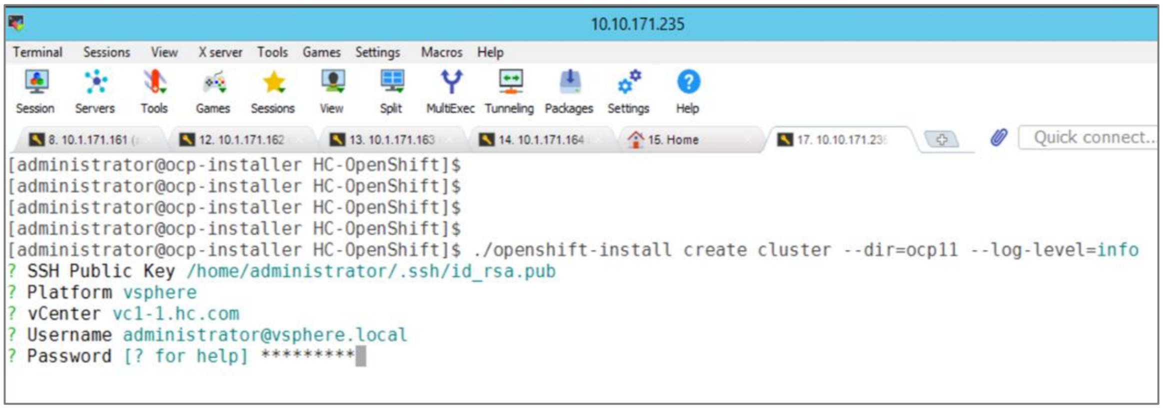
```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key  [Use arrows to move, type to filter, ? for more help]
  /home/administrator/.ssh/id_ed25519.pub
> /home/administrator/.ssh/id_rsa.pub
  <none>
```

Step 3. Select infrastructure environment.


10.10.171.235
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
8. 10.1.171.161
12. 10.1.171.162
13. 10.1.171.163
14. 10.1.171.164
15. Home
17. 10.10.171.23
Quick connect...

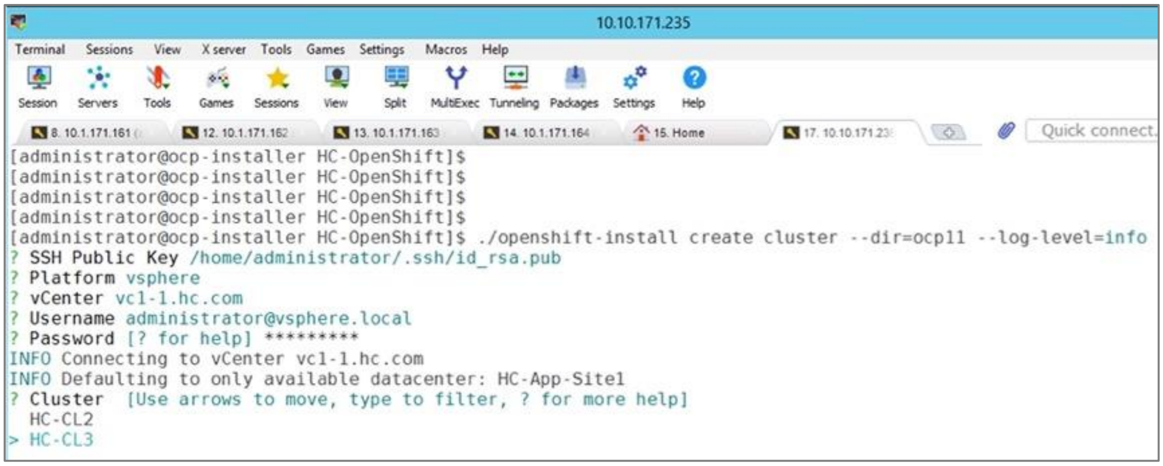
```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/administrator/.ssh/id_rsa.pub
? Platform  [Use arrows to move, type to filter, ? for more help]
  aws
  azure
  gcp
  openstack
  ovirt
> vsphere
```

Step 4. Provide VMware vCenter information.


10.10.171.235
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
15. Home
Quick connect...

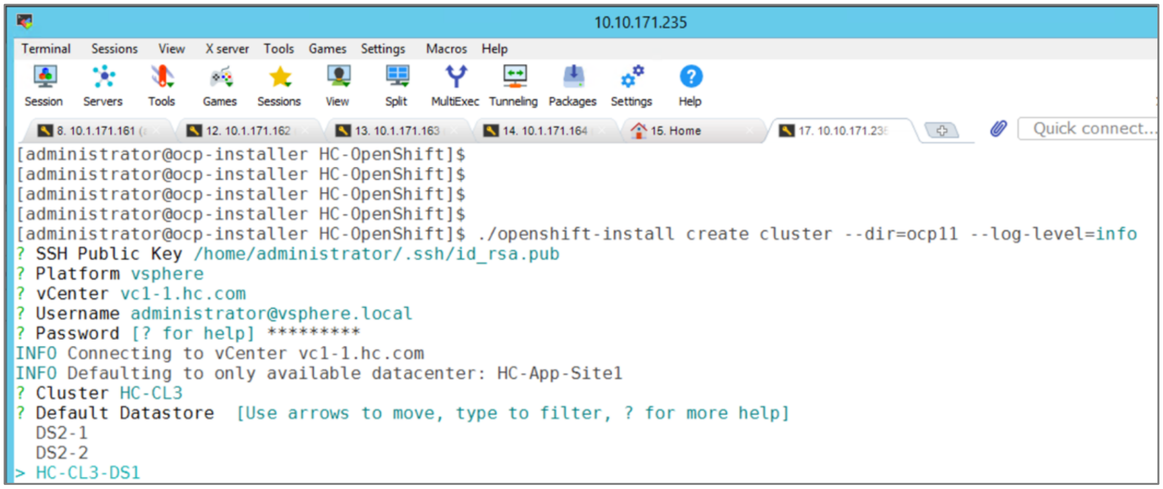
```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/administrator/.ssh/id_rsa.pub
? Platform vsphere
? vCenter vc1-1.hc.com
? Username administrator@vsphere.local
? Password [? for help] *********
```

Step 5. Select a VMware vSphere cluster to deploy the OCP cluster on.


10.10.171.235
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
15. Home
Quick connect...

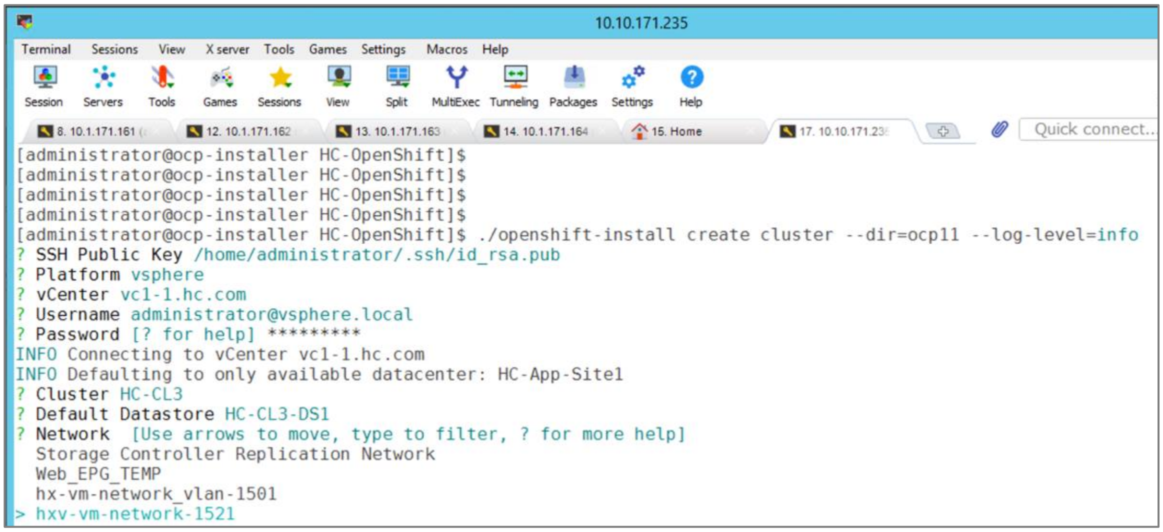
```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/administrator/.ssh/id_rsa.pub
? Platform vsphere
? vCenter vc1-1.hc.com
? Username administrator@vsphere.local
? Password [? for help] *********
INFO Connecting to vCenter vc1-1.hc.com
INFO Defaulting to only available datacenter: HC-App-Site1
? Cluster  [Use arrows to move, type to filter, ? for more help]
  HC-CL2
> HC-CL3
```

Step 6. Select a datastore to use.


10.10.171.235
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
15. Home
Quick connect...

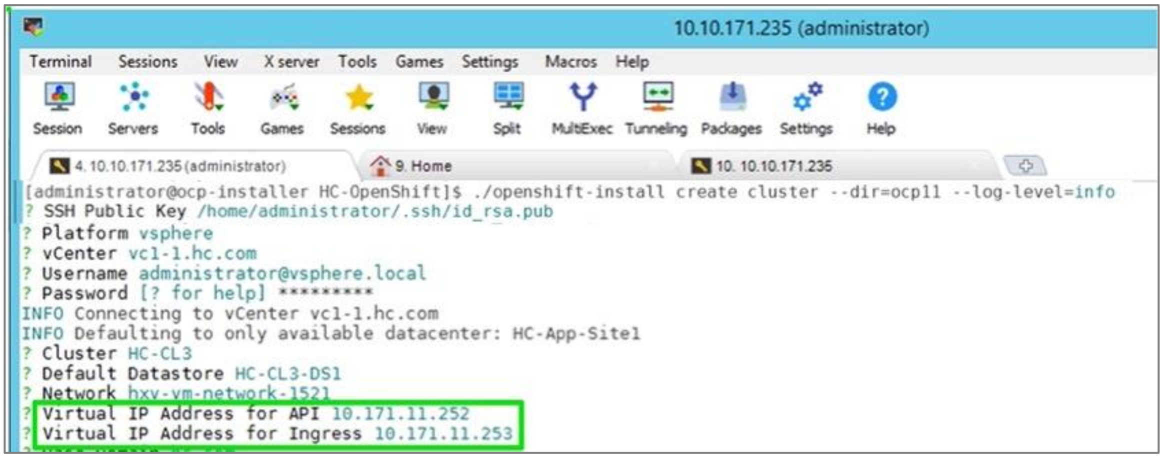
```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/administrator/.ssh/id_rsa.pub
? Platform vsphere
? vCenter vc1-1.hc.com
? Username administrator@vsphere.local
? Password [? for help] *********
INFO Connecting to vCenter vc1-1.hc.com
INFO Defaulting to only available datacenter: HC-App-Site1
? Cluster HC-CL3
? Default Datastore  [Use arrows to move, type to filter, ? for more help]
  DS2-1
  DS2-2
> HC-CL3-DS1
```

Step 7. Select a guest VM network for the OCP cluster.


10.10.171.235
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
8. 10.1.171.161
12. 10.1.171.162
13. 10.1.171.163
14. 10.1.171.164
15. Home
17. 10.10.171.23
Quick connect...

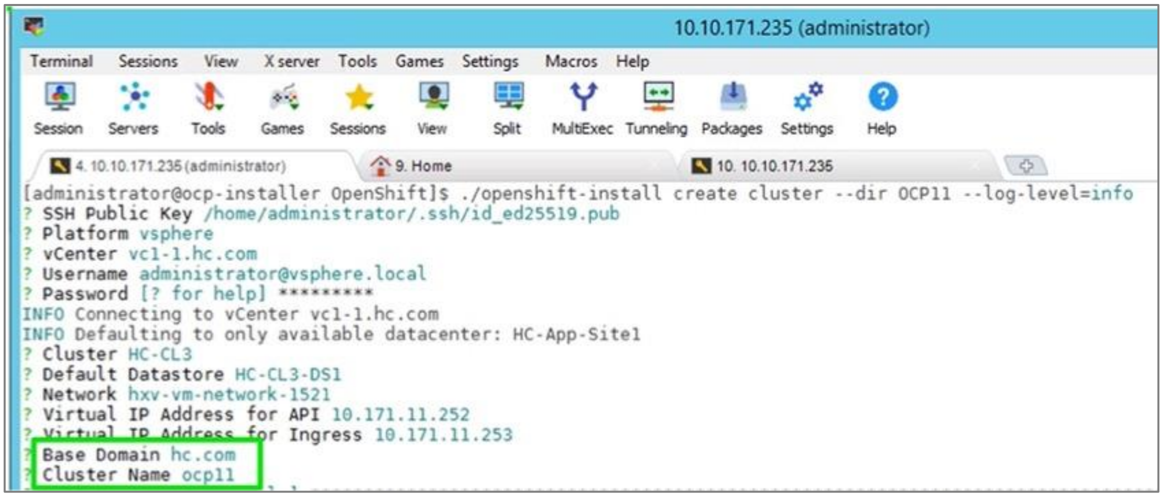
```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/administrator/.ssh/id_rsa.pub
? Platform vsphere
? vCenter vc1-1.hc.com
? Username administrator@vsphere.local
? Password [? for help] *********
INFO Connecting to vCenter vc1-1.hc.com
INFO Defaulting to only available datacenter: HC-App-Site1
? Cluster HC-CL3
? Default Datastore HC-CL3-DS1
? Network  [Use arrows to move, type to filter, ? for more help]
  Storage Controller Replication Network
  Web_EPG_TEMP
  hx-vm-network_vlan-1501
> hxv-vm-network-1521
```

Step 8. Provide a Virtual IP (VIP) for API and Ingress.


10.10.171.235 (administrator)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
4. 10.10.171.235 (administrator)
9. Home
10. 10.10.171.235

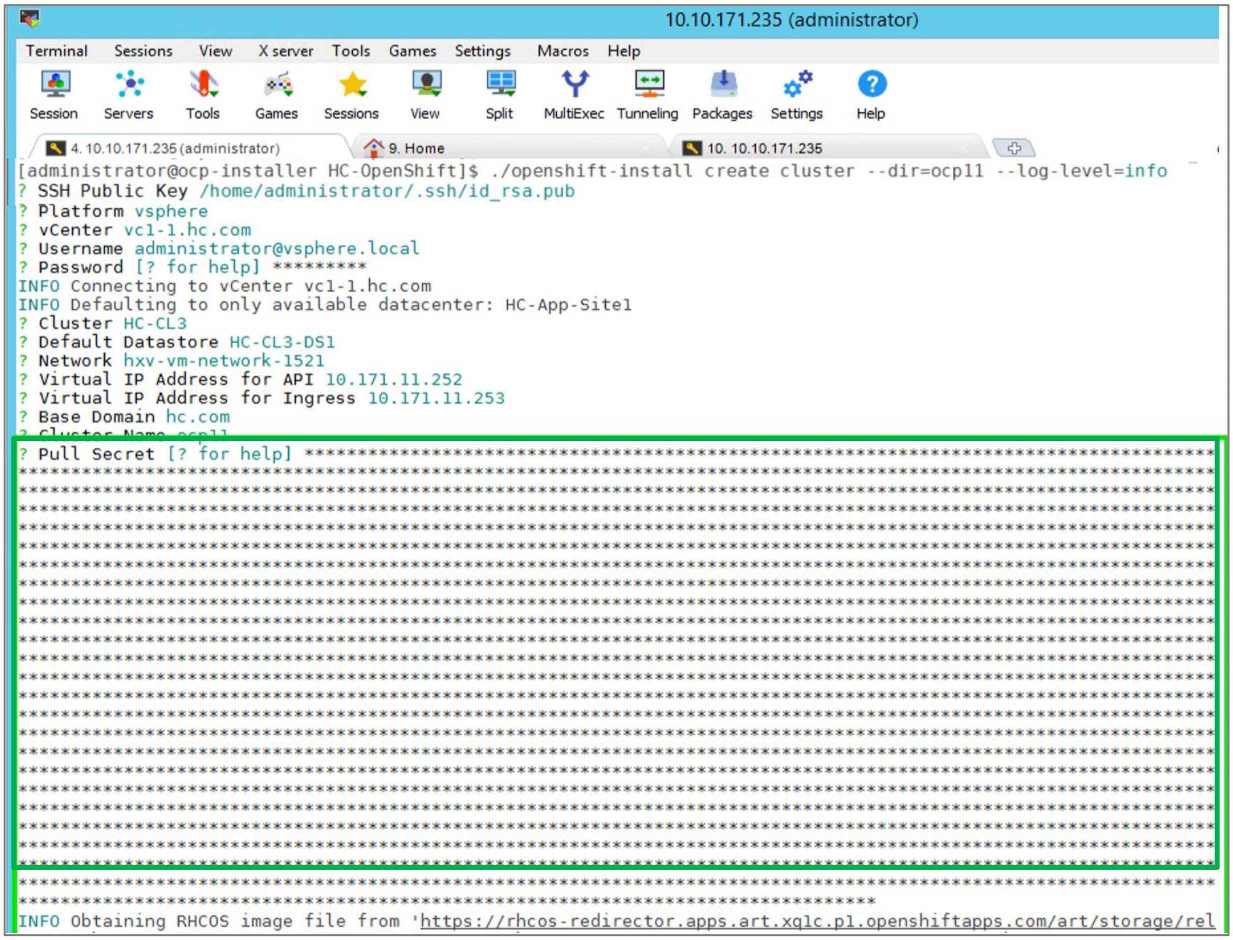
```
[administrator@ocp-installer HC-OpenShift]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/administrator/.ssh/id_rsa.pub
? Platform vsphere
? vCenter vc1-1.hc.com
? Username administrator@vsphere.local
? Password [? for help] *********
INFO Connecting to vCenter vc1-1.hc.com
INFO Defaulting to only available datacenter: HC-App-Site1
? Cluster HC-CL3
? Default Datastore HC-CL3-DS1
? Network hxv-vm-network-1521
? Virtual IP Address for API 10.171.11.252
? Virtual IP Address for Ingress 10.171.11.253
```

Step 9. Specify a base DNS domain and name for the OCP cluster.


10.10.171.235 (administrator)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
4. 10.10.171.235 (administrator)
9. Home
10. 10.10.171.235

```
[administrator@ocp-installer OpenShift]$ ./openshift-install create cluster --dir OCP11 --log-level=info
? SSH Public Key /home/administrator/.ssh/id_ed25519.pub
? Platform vsphere
? vCenter vc1-1.hc.com
? Username administrator@vsphere.local
? Password [? for help] *********
INFO Connecting to vCenter vc1-1.hc.com
INFO Defaulting to only available datacenter: HC-App-Site1
? Cluster HC-CL3
? Default Datastore HC-CL3-DS1
? Network hxv-vm-network-1521
? Virtual IP Address for API 10.171.11.252
? Virtual IP Address for Ingress 10.171.11.253
? Base Domain hc.com
? Cluster Name ocp11
```

Step 10. Paste the pull secret copied from the Hybrid Cloud Console.


10.10.171.235 (administrator)
Terminal Sessions View X server Tools Games Settings Macros Help
Session
Servers
Tools
Games
Sessions
View
Split
MultiExec
Tunneling
Packages
Settings
Help
4. 10.10.171.235 (administrator)
9. Home
10. 10.10.171.235

```
[administrator@ocp-installer HC-OpenShift]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/administrator/.ssh/id_rsa.pub
? Platform vsphere
? vCenter vc1-1.hc.com
? Username administrator@vsphere.local
? Password [? for help] *********
INFO Connecting to vCenter vc1-1.hc.com
INFO Defaulting to only available datacenter: HC-App-Site1
? Cluster HC-CL3
? Default Datastore HC-CL3-DS1
? Network hxv-vm-network-1521
? Virtual IP Address for API 10.171.11.252
? Virtual IP Address for Ingress 10.171.11.253
? Base Domain hc.com
? Cluster Name ocp11
? Pull Secret [? for help] ************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
***************************************************************************************************************
********************************************************************************
INFO Obtaining RHCOS image file from 'https://rhcos-redirector.apps.art.xq1c.p1.openshiftapps.com/art/storage/rel
```

Step 11. Once the pull-secret is provided, the installer will start the OCP deployment. It will take around 45min. Once the OCP cluster is deployed, there are a few post-install tasks that must be completed.

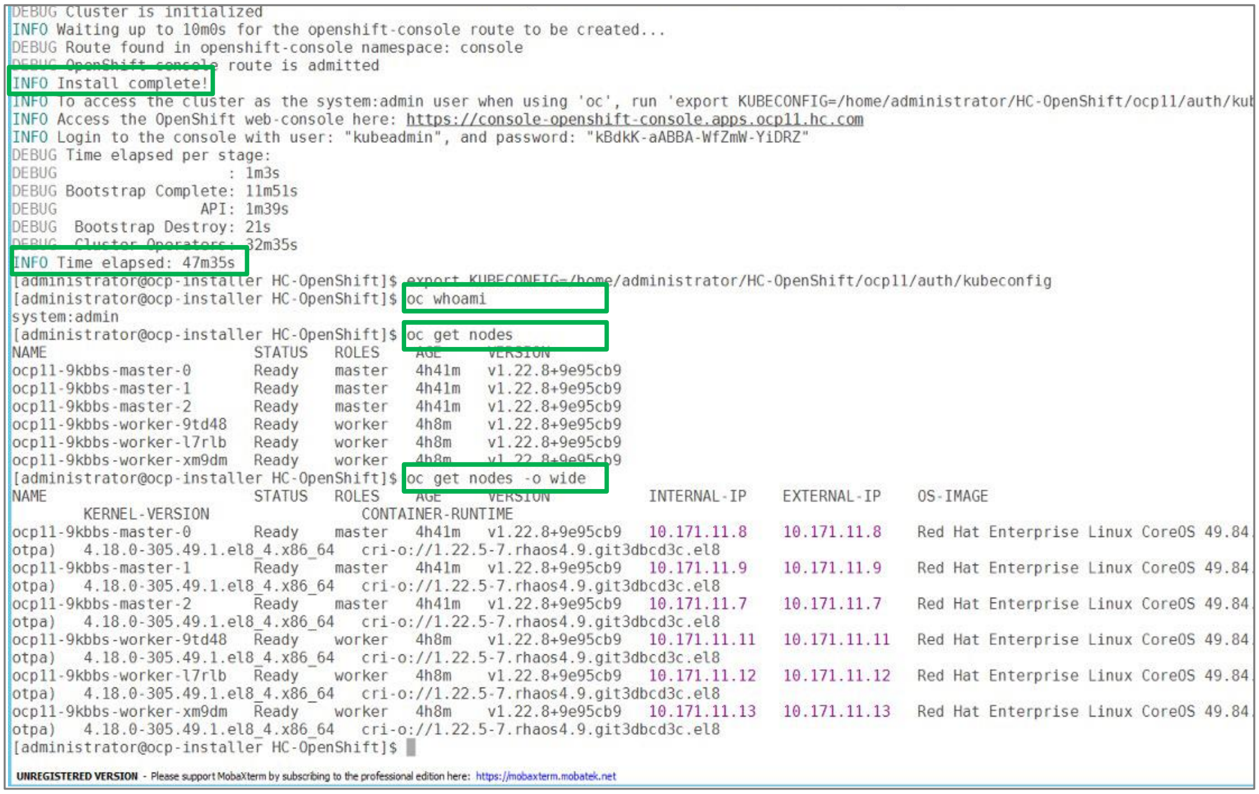
Note: Do **not** remove the installer or the files created by the installer. These will be necessary to delete the cluster.

Procedure 7. Post-install task – Verify access to new cluster

Note: When the cluster installation finished, complete the following post-install tasks to verify that you can access the new cluster using command line tools and web console.

Step 1. To verify access via the command line, execute the following commands from the workstation:

```
export KUBECONFIG=/root/openshift/<ocp_directory>/auth/kubeconfig
oc whoami
oc get nodes -o wide
```

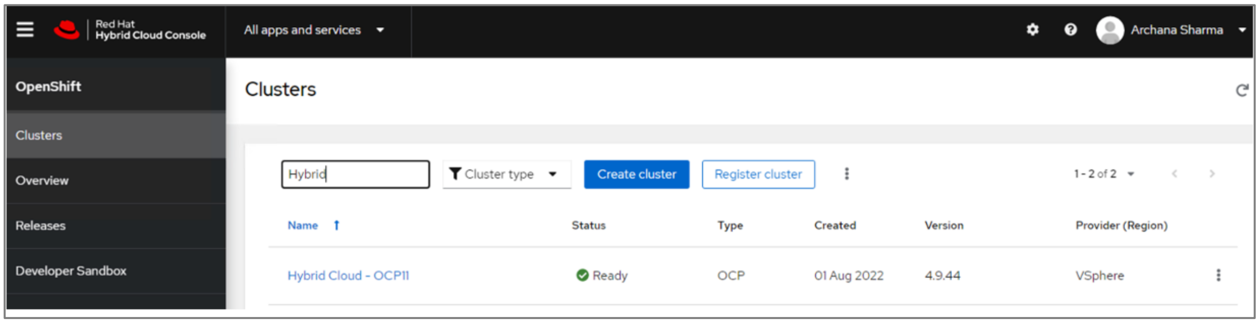


```
DEBUG Cluster is initialized
INFO Waiting up to 10m0s for the openshift-console route to be created...
DEBUG Route found in openshift-console namespace: console
DEBUG OpenShift console route is admitted
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/administrator/HC-OpenShift/ocp11/auth/kub
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.ocp11.hc.com
INFO Login to the console with user: "kubeadmin", and password: "kBdkK-aABBA-WfZmW-YiDRZ"
DEBUG Time elapsed per stage:
DEBUG                       : 1m3s
DEBUG Bootstrap Complete: 11m51s
DEBUG                  API: 1m39s
DEBUG  Bootstrap Destroy: 21s
DEBUG  Cluster Operators: 32m35s
INFO Time elapsed: 47m35s
[administrator@ocp-installer HC-OpenShift]$ export KUBECONFIG=/home/administrator/HC-OpenShift/ocp11/auth/kubeconfig
[administrator@ocp-installer HC-OpenShift]$ oc whoami
system:admin
[administrator@ocp-installer HC-OpenShift]$ oc get nodes
NAME                       STATUS   ROLES    AGE     VERSION
ocp11-9kbbs-master-0       Ready    master   4h41m   v1.22.8+9e95cb9
ocp11-9kbbs-master-1       Ready    master   4h41m   v1.22.8+9e95cb9
ocp11-9kbbs-master-2       Ready    master   4h41m   v1.22.8+9e95cb9
ocp11-9kbbs-worker-9td48   Ready    worker   4h8m    v1.22.8+9e95cb9
ocp11-9kbbs-worker-l7rlb   Ready    worker   4h8m    v1.22.8+9e95cb9
ocp11-9kbbs-worker-xm9dm   Ready    worker   4h8m    v1.22.8+9e95cb9
[administrator@ocp-installer HC-OpenShift]$ oc get nodes -o wide
NAME                       STATUS   ROLES    AGE     VERSION           INTERNAL-IP    EXTERNAL-IP    OS-IMAGE
        KERNEL-VERSION               CONTAINER-RUNTIME
ocp11-9kbbs-master-0       Ready    master   4h41m   v1.22.8+9e95cb9   10.171.11.8    10.171.11.8    Red Hat Enterprise Linux CoreOS 49.84
otpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o://1.22.5-7.rhaos4.9.git3dbcd3c.el8
ocp11-9kbbs-master-1       Ready    master   4h41m   v1.22.8+9e95cb9   10.171.11.9    10.171.11.9    Red Hat Enterprise Linux CoreOS 49.84
otpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o://1.22.5-7.rhaos4.9.git3dbcd3c.el8
ocp11-9kbbs-master-2       Ready    master   4h41m   v1.22.8+9e95cb9   10.171.11.7    10.171.11.7    Red Hat Enterprise Linux CoreOS 49.84
otpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o://1.22.5-7.rhaos4.9.git3dbcd3c.el8
ocp11-9kbbs-worker-9td48   Ready    worker   4h8m    v1.22.8+9e95cb9   10.171.11.11   10.171.11.11   Red Hat Enterprise Linux CoreOS 49.84
otpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o://1.22.5-7.rhaos4.9.git3dbcd3c.el8
ocp11-9kbbs-worker-l7rlb   Ready    worker   4h8m    v1.22.8+9e95cb9   10.171.11.12   10.171.11.12   Red Hat Enterprise Linux CoreOS 49.84
otpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o://1.22.5-7.rhaos4.9.git3dbcd3c.el8
ocp11-9kbbs-worker-xm9dm   Ready    worker   4h8m    v1.22.8+9e95cb9   10.171.11.13   10.171.11.13   Red Hat Enterprise Linux CoreOS 49.84
otpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o://1.22.5-7.rhaos4.9.git3dbcd3c.el8
[administrator@ocp-installer HC-OpenShift]$
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

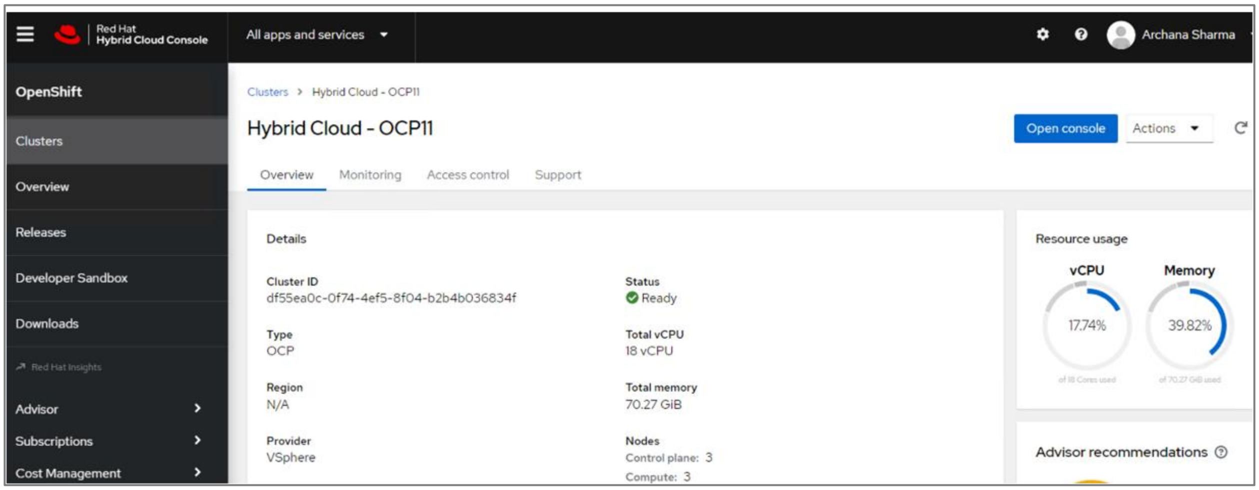
Step 2. To verify access via web console, open a web browser and navigate to the OpenShift Hybrid Cloud Console. Login using your Red Hat account. Navigate to **Clusters** and find the newly created OCP cluster. Select and click on the cluster name.

Note: If your cluster has access to the Internet, the Telemetry service will automatically register with the Hybrid Cloud Console, and you will see it in the cluster list.



Name	Status	Type	Created	Version	Provider (Region)
Hybrid Cloud - OCP11	Ready	OCP	01 Aug 2022	4.9.44	VSphere

Step 3. Click on **Open console** from the top-right corner and login to the cluster console as **kubeadm** user.



Hybrid Cloud - OCP11

Details

Cluster ID df55ea0c-0f74-4ef5-8f04-b2b4b036834f	Status Ready
Type OCP	Total vCPU 18 vCPU
Region N/A	Total memory 70.27 GiB
Provider VSphere	Nodes Control plane: 3 Compute: 3

Step 4. Verify status and health of cluster from OCP cluster console.

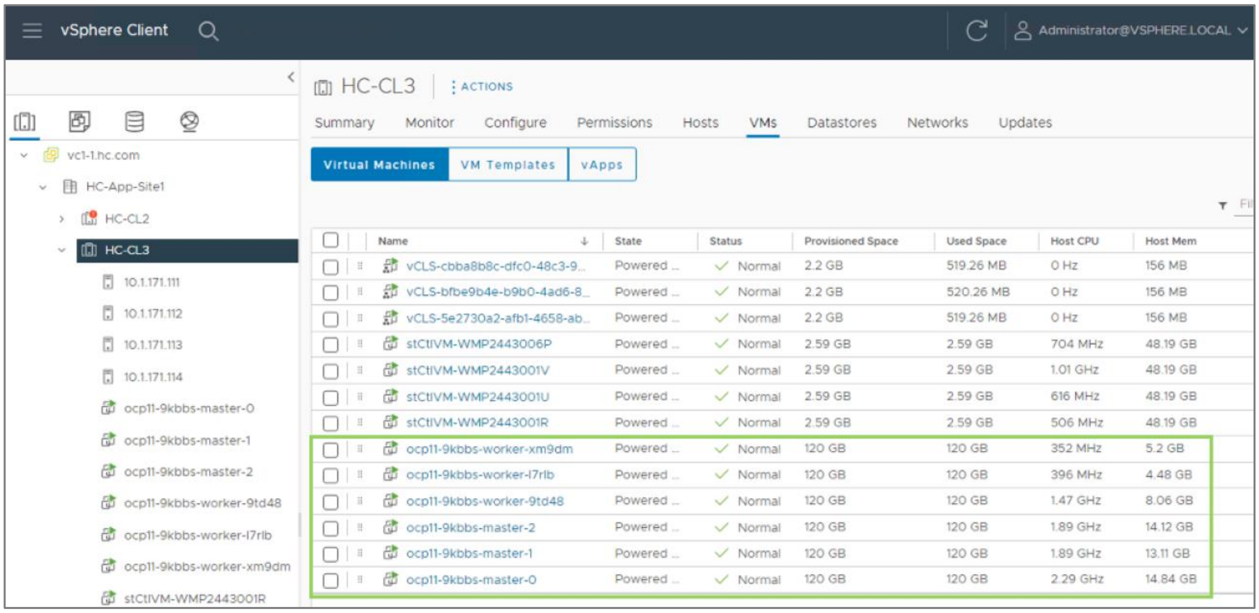
Procedure 8. Post-install task – Provision anti-affinity rules on VMware vCenter

Note: Red Hat recommends VMware anti-affinity rules to improve the availability of OpenShift Container Platform during maintenance or hardware issues.

This procedure will enable VMware anti-affinity rules to distribute OCP cluster nodes across all HyperFlex ESXi hosts in the cluster.

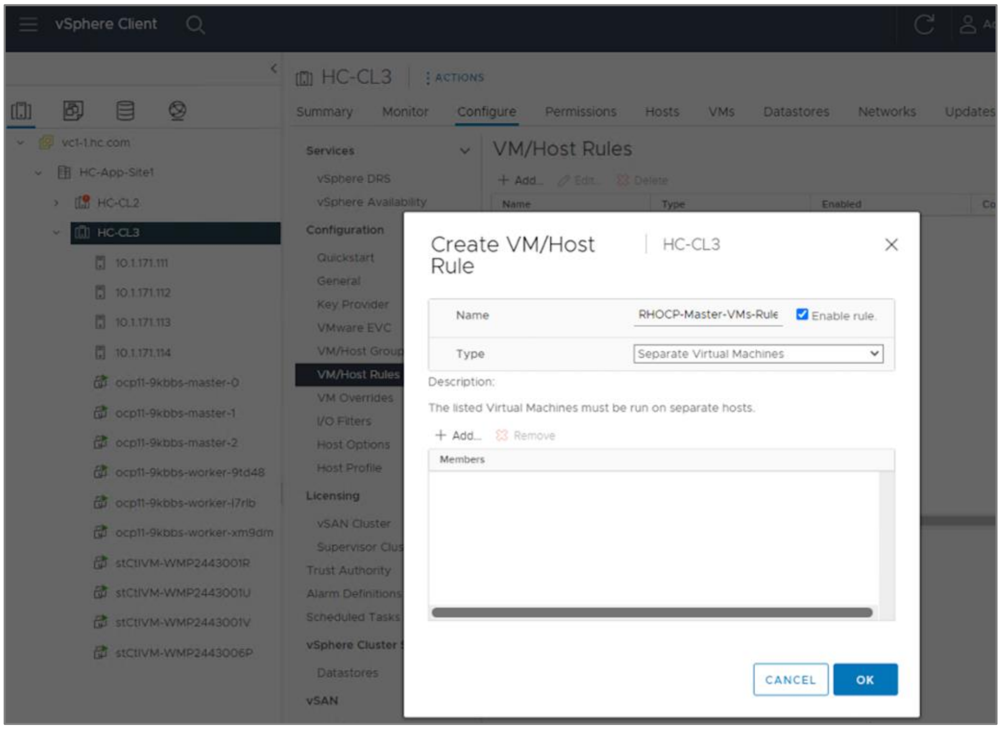
Step 1. Open a web browser and navigate to VMware vCenter managing the Application HyperFlex VSI cluster. Login as administrator.

Step 2. From **Hosts and Clusters** view, navigate to the Application VMware vSphere cluster. Identify the OCP cluster nodes.



Name	State	Status	Provisioned Space	Used Space	Host CPU	Host Mem
vCLS-cbba8b8c-dfc0-48c3-9...	Powered ...	✓ Normal	2.2 GB	519.26 MB	0 Hz	156 MB
vCLS-bfbe9b4e-b9b0-4ad6-8...	Powered ...	✓ Normal	2.2 GB	520.26 MB	0 Hz	156 MB
vCLS-5e2730a2-afb1-4658-ab...	Powered ...	✓ Normal	2.2 GB	519.26 MB	0 Hz	156 MB
stCtlVM-WMP2443006P	Powered ...	✓ Normal	2.59 GB	2.59 GB	704 MHz	48.19 GB
stCtlVM-WMP2443001V	Powered ...	✓ Normal	2.59 GB	2.59 GB	1.01 GHz	48.19 GB
stCtlVM-WMP2443001U	Powered ...	✓ Normal	2.59 GB	2.59 GB	616 MHz	48.19 GB
stCtlVM-WMP2443001R	Powered ...	✓ Normal	2.59 GB	2.59 GB	506 MHz	48.19 GB
ocp11-9kbbs-worker-xm9dm	Powered ...	✓ Normal	120 GB	120 GB	352 MHz	5.2 GB
ocp11-9kbbs-worker-l7rlb	Powered ...	✓ Normal	120 GB	120 GB	396 MHz	4.48 GB
ocp11-9kbbs-worker-9td48	Powered ...	✓ Normal	120 GB	120 GB	1.47 GHz	8.06 GB
ocp11-9kbbs-master-2	Powered ...	✓ Normal	120 GB	120 GB	1.89 GHz	14.12 GB
ocp11-9kbbs-master-1	Powered ...	✓ Normal	120 GB	120 GB	1.89 GHz	13.11 GB
ocp11-9kbbs-master-0	Powered ...	✓ Normal	120 GB	120 GB	2.29 GHz	14.84 GB

Step 3. Click on the **Configure** tab and select **VM/Host Rules** from the left navigation pane. Click **Add** in the right window.



Create VM/Host Rule | HC-CL3

Name: RHOCP-Master-VMs-Rule ☑ Enable rule.

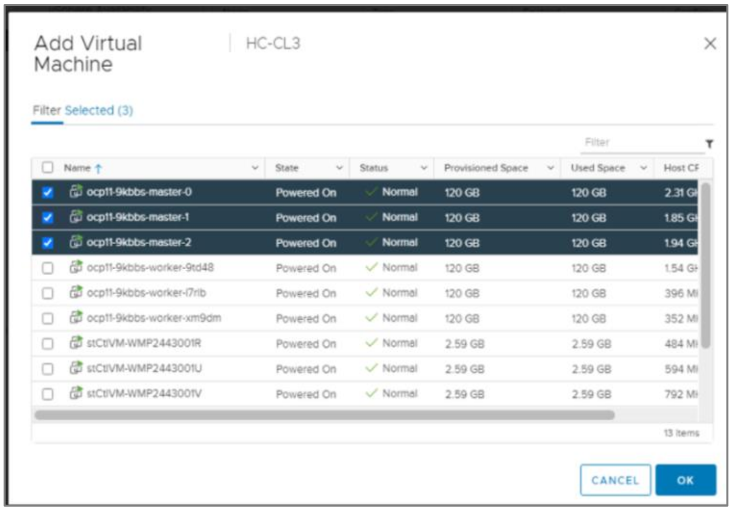
Type: Separate Virtual Machines

Description:

The listed Virtual Machines must be run on separate hosts.

Step 4. In the **Create VM/Host Rule** pop-up window, specify a name for control/master nodes rule. Set the **Type** as **Separate Virtual Machines**, then click **Add** to add control nodes that the rule applies to.

Step 5. In the **Add Virtual Machine** pop-up window, select the three OCP Control VMs from the list, then click **OK**.

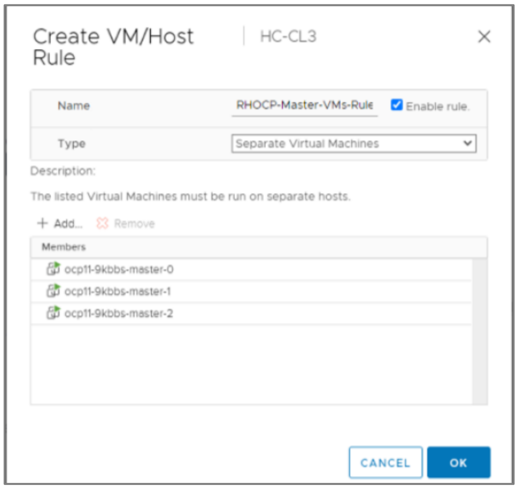


Add Virtual Machine | HC-CL3

	Name ↑	State	Status	Provisioned Space	Used Space	Host CF
☑	ocp11-9kbbs-master-0	Powered On	✓ Normal	120 GB	120 GB	2.31 G
☑	ocp11-9kbbs-master-1	Powered On	✓ Normal	120 GB	120 GB	1.85 G
☑	ocp11-9kbbs-master-2	Powered On	✓ Normal	120 GB	120 GB	1.94 G
□	ocp11-9kbbs-worker-9td48	Powered On	✓ Normal	120 GB	120 GB	1.54 G
□	ocp11-9kbbs-worker-i7rlb	Powered On	✓ Normal	120 GB	120 GB	396 M
□	ocp11-9kbbs-worker-xm9dm	Powered On	✓ Normal	120 GB	120 GB	352 M
□	stCtlVM-WMP2443001R	Powered On	✓ Normal	2.59 GB	2.59 GB	484 M
□	stCtlVM-WMP2443001U	Powered On	✓ Normal	2.59 GB	2.59 GB	594 M
□	stCtlVM-WMP2443001V	Powered On	✓ Normal	2.59 GB	2.59 GB	792 M

13 items

Step 6. Click **OK** to create the rule for OCP Control/Master VMs.



Create VM/Host Rule | HC-CL3

Name: RHOCP-Master-VMs-Rule ☑ Enable rule.

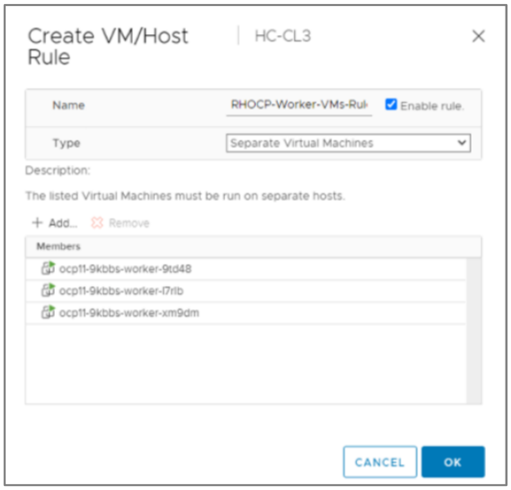
Type: Separate Virtual Machines

Description:

The listed Virtual Machines must be run on separate hosts.

Members
ocp11-9kbbs-master-0
ocp11-9kbbs-master-1
ocp11-9kbbs-master-2

Step 7. Repeat steps 3-6 to distribute the worker VMs across all available Cisco HyperFlex nodes.



Create VM/Host Rule | HC-CL3

Name: RHOCP-Worker-VMs-Rul ☑ Enable rule.

Type: Separate Virtual Machines

Description:

The listed Virtual Machines must be run on separate hosts.

Members
ocp11-9kbbs-worker-9td48
ocp11-9kbbs-worker-l7rlb
ocp11-9kbbs-worker-xm9dm

Deploy HyperFlex CSI

This section describes the deployment of a container storage interface for the cloud-native environment in this hybrid cloud solution. The solution uses a HyperFlex Container Storage Interface, deployed, and managed from Red Hat's Operator Hub. The HyperFlex CSI will be deployed on the Red Hat OCP cluster running on the Application HyperFlex VSI cluster.

Prerequisites

The prerequisites for deploying HyperFlex CSI on a Red Hat OCP cluster are:

- Provision Cisco ACI fabric to enable IP reachability to iSCSI networks that HyperFlex CSI uses. Red Hat OCP cluster and workloads are part of the Application Tenant (**HC-Tenant1**) , while the iSCSI network is in the HyperFlex Infrastructure tenant (**HXV-Foundation**). ACI fabric must therefore route traffic between these two networks and tenants.
- Configure additional iSCSI VLANs on HyperFlex for the HyperFlex CSI traffic to use.
- Enable (or verify) iSCSI daemon on the Red Hat OCP worker nodes

Setup Information

Table 11 lists the installation parameters for HyperFlex CSI.

Table 11. HyperFlex CSI – Installation Parameters

Variable	Variable Name	Value	Additional Info
iSCSI Network for HX CSI	Subnet	192.168.13.0/24	
iSCSI Gateway IP	Gateway	192.168.13.254	
iSCSI Cluster IP	iSCSI Storage IP	192.168.13.110	
iSCSI Host IP	IP Range	192.168.13.111 – 192.168.13.114	
iSCSI VLAN for HX CSI	VLAN ID	3013	
iSCSI VLAN for HX CSI	VLAN Name	hx-inband-iscsi-3013	
Cisco UCS Manager IP and Login Credentials		192.168.171.192	<Login Credentials>
Docker Registry Name		quay.io/hxcsiadmin	Public registry containing the HXCSI container images
HyperFlex Management Cluster IP		10.1.171.110	

Deployment Steps

The section provides the procedures for deploying HyperFlex CSI on a Red Hat OCP cluster running on a HyperFlex VSI cluster.

Procedure 1. Provision Cisco ACI fabric to enable IP reachability for HyperFlex CSI

The HyperFlex CSI uses iSCSI for communication between iSCSI clients on Red Hat OCP Pods and underlying HyperFlex storage. The OCP cluster and workloads are in a different VLANs and tenants. ACI fabric must therefore be configured for the following:

- Access layer connectivity to HyperFlex VSI in the UCS domain for iSCSI VLANs
- Forwarding of iSCSI VLAN traffic through the ACI fabric
- Enable reachability from the Red Hat OCP and application workload networks to the iSCSI network on HyperFlex cluster

The above connectivity is provisioned using the same Ansible playbooks that provisioned the ACI fabric for deploying the HyperFlex cluster. See the Provision Cisco ACI Fabric section under Solution Deployment for more details**.**

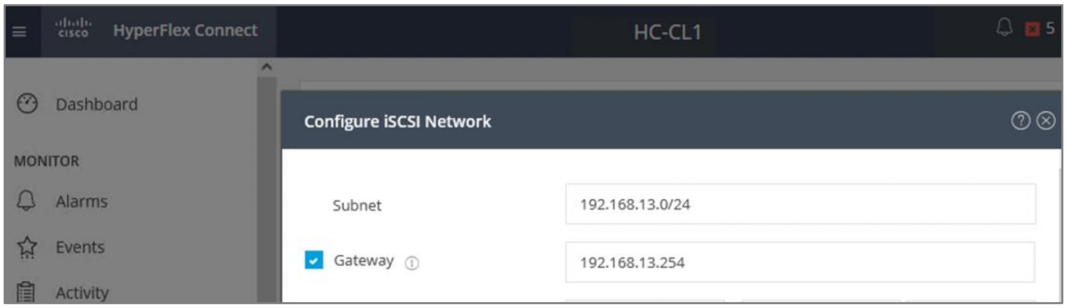
Procedure 2. Configure iSCSI networking on HyperFlex

This procedure enables you to provision iSCSI networking on HyperFlex used by HyperFlex CSI.

Step 1. Navigate using a web browser to the HyperFlex Cluster Management IP to access the Cisco HyperFlex HX Connect page. Login as admin.

Step 2. In the HyperFlex Connect webpage, click on **iSCSI** then click the link for **Configure Network**.

Step 3. In the Configure iSCSI Network pop-up window, specify the IP **Subnet** that iSCSI traffic should use. If the subnet is routable, check the box next to **Gateway**, and enter the gateway IP address.

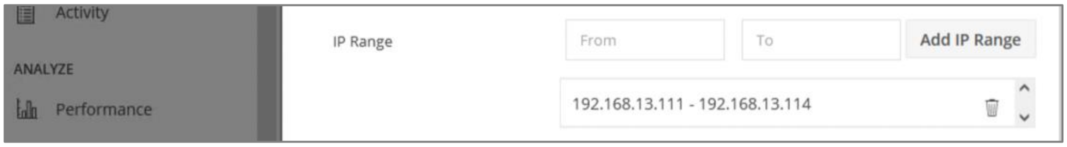

HC-CL1

Configure iSCSI Network

Subnet: 192.168.13.0/24

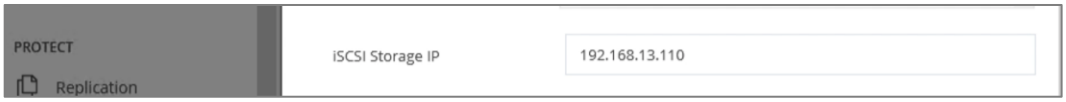
☑ Gateway: 192.168.13.254

Step 4. Specify the **IP Range** that the HyperFlex servers in the cluster should use for iSCSI. There should be enough IP addresses in the range to allocate one IP for each HyperFlex node in the cluster. Click **Add IP Range**.

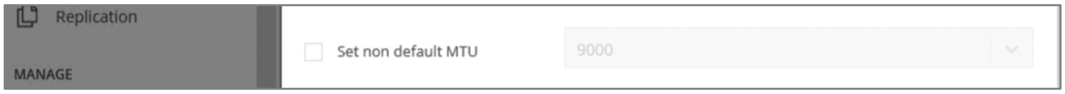

IP Range: From / To — Add IP Range

192.168.13.111 - 192.168.13.114

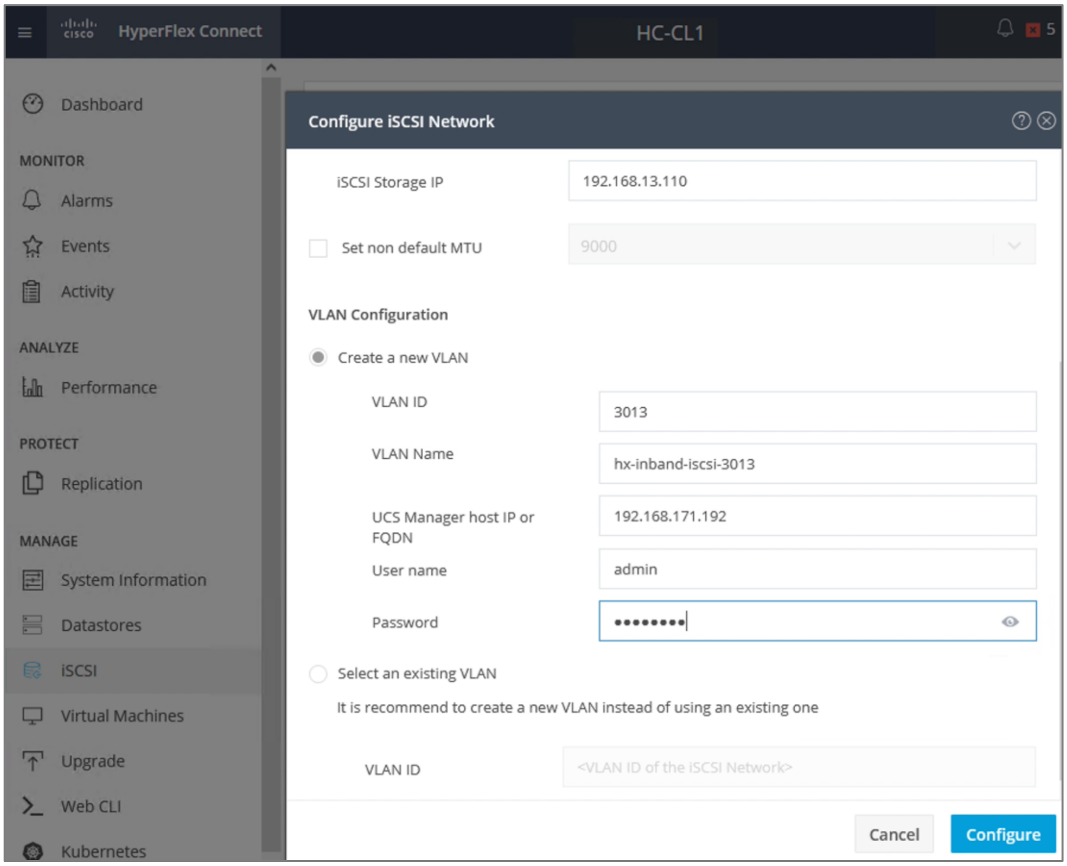
Step 5. Specify the **iSCSI Storage IP** for the HyperFlex cluster.


iSCSI Storage IP: 192.168.13.110

Step 6. Leave the check box to **Set non-default MTU** unchecked.


□ Set non default MTU: 9000

Step 7. Select the option to **Create a New VLAN**, then enter the **VLAN ID**, **VLAN name**, the **Cisco UCS Manager IP address**, **username**, and **password**. Alternatively, you can also choose to use an existing VLAN.



Configure iSCSI Network

iSCSI Storage IP: 192.168.13.110

□ Set non default MTU: 9000

VLAN Configuration

● Create a new VLAN

VLAN ID: 3013

VLAN Name: hx-inband-iscsi-3013

UCS Manager host IP or FQDN: 192.168.171.192

User name: admin

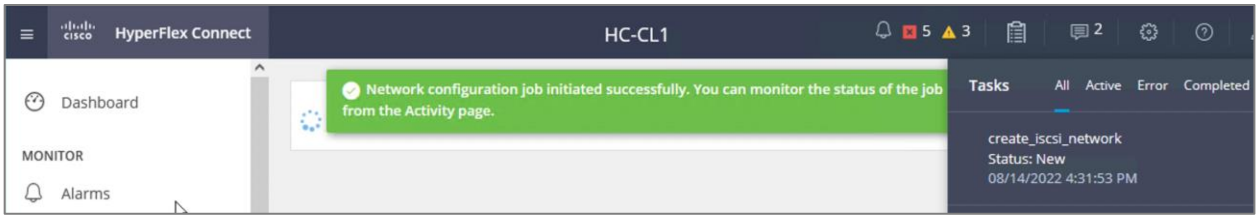
Password: ••••••••

○ Select an existing VLAN

It is recommend to create a new VLAN instead of using an existing one

VLAN ID: <VLAN ID of the iSCSI Network>

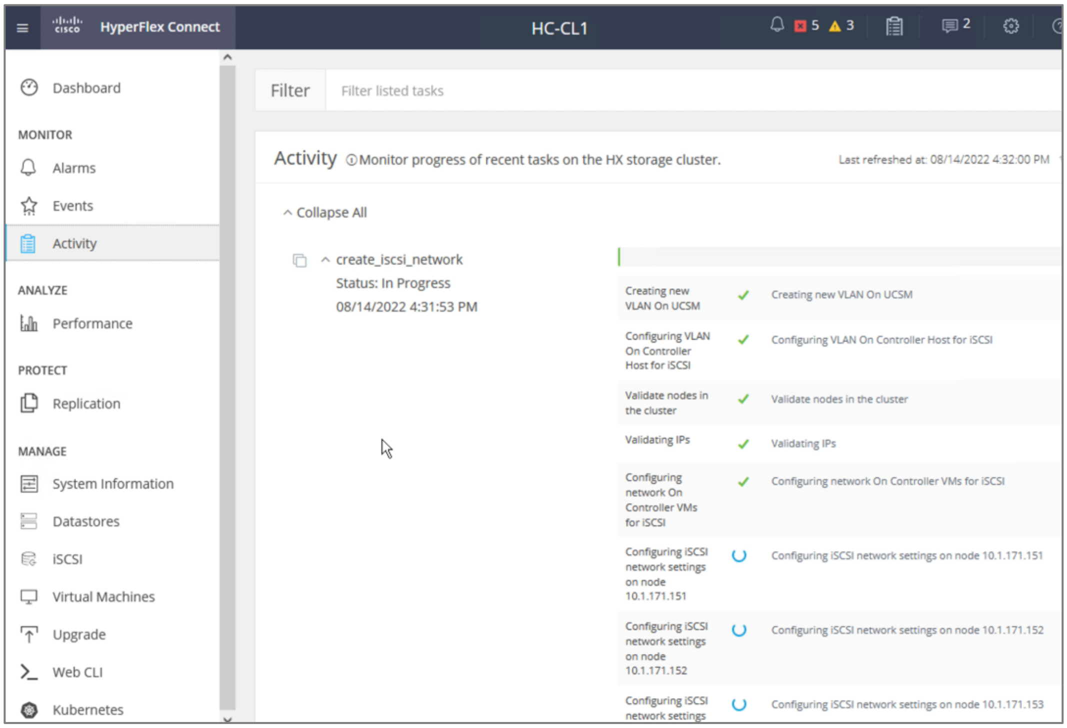
Step 8. Click **Configure.** This will kick of the configuration process that can be monitored by navigating to **Monitor > Activity** in the left navigation pane.



Network configuration job initiated successfully. You can monitor the status of the job from the Activity page.

create_iscsi_network
Status: New
08/14/2022 4:31:53 PM

Step 9. Navigate to **Activity** section to monitor the configuration.



Activity ⓘMonitor progress of recent tasks on the HX storage cluster.

Last refreshed at: 08/14/2022 4:32:00 PM

create_iscsi_network
Status: In Progress
08/14/2022 4:31:53 PM

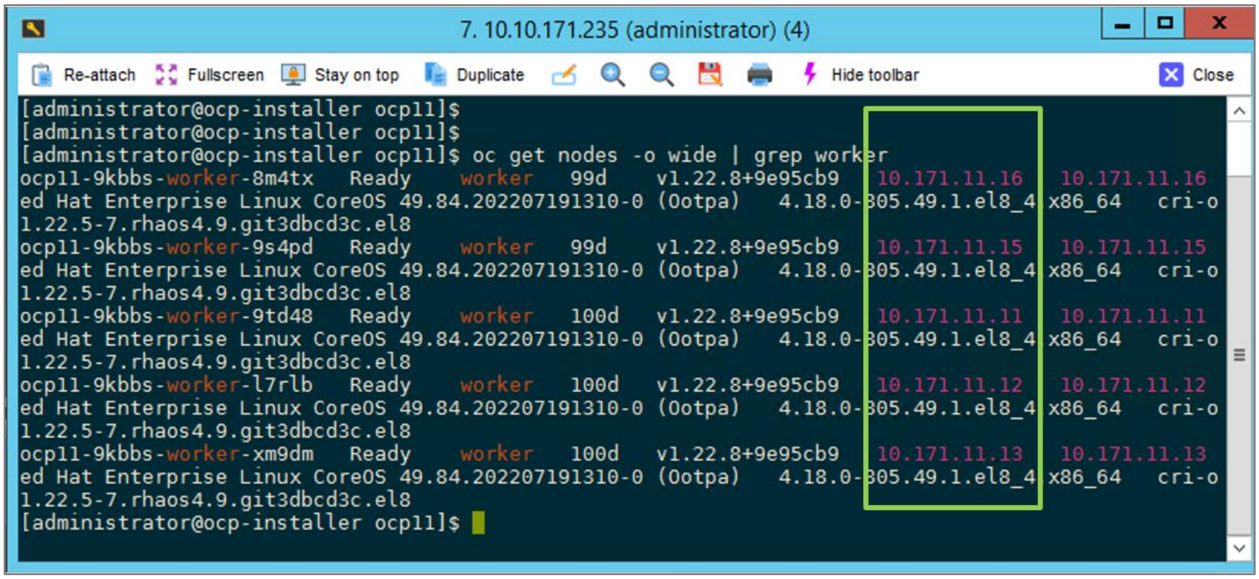
Creating new VLAN On UCSM	✓	Creating new VLAN On UCSM
Configuring VLAN On Controller Host for iSCSI	✓	Configuring VLAN On Controller Host for iSCSI
Validate nodes in the cluster	✓	Validate nodes in the cluster
Validating IPs	✓	Validating IPs
Configuring network On Controller VMs for iSCSI	✓	Configuring network On Controller VMs for iSCSI
Configuring iSCSI network settings on node 10.1.171.151	○	Configuring iSCSI network settings on node 10.1.171.151
Configuring iSCSI network settings on node 10.1.171.152	○	Configuring iSCSI network settings on node 10.1.171.152
Configuring iSCSI network settings	○	Configuring iSCSI network settings on node 10.1.171.153

Procedure 3. Enable (or verify) iSCSI daemon on Red Hat OCP worker nodes

For HyperFlex CSI to function correctly, the iSCSI daemon must be running on the OCP worker nodes. Complete the following steps to enable the iSCSI daemon. The commands that you will need for this are:

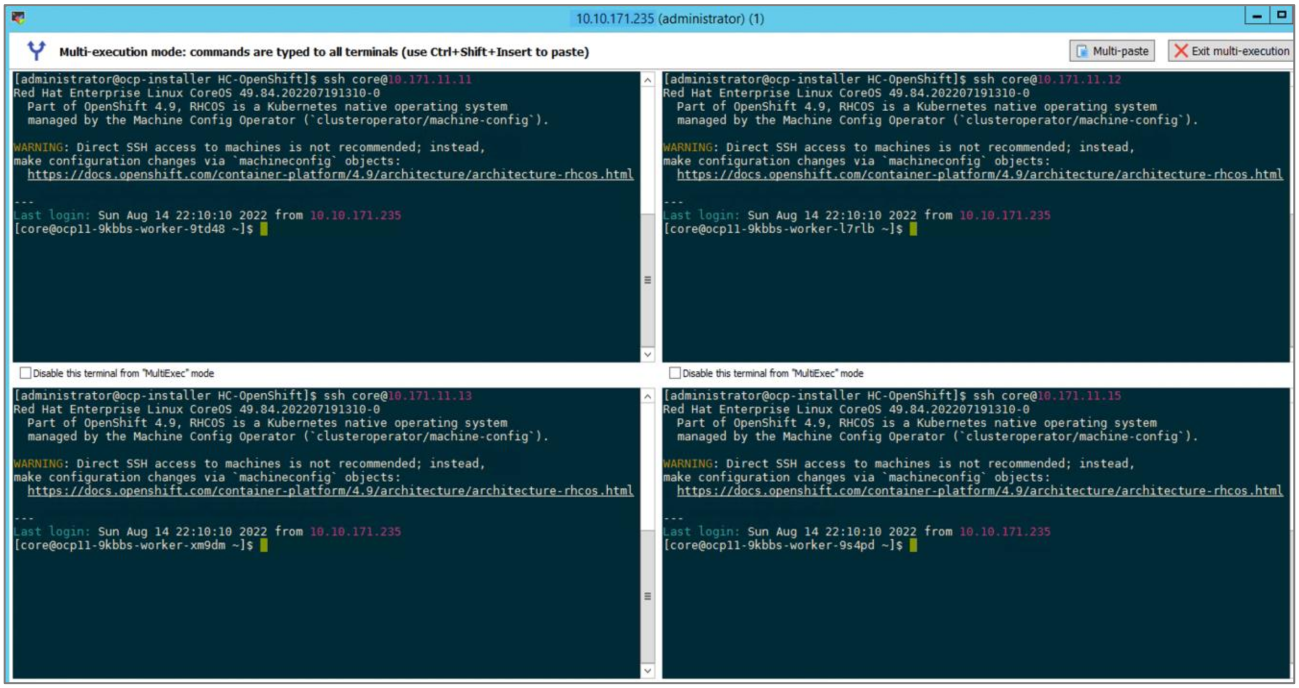
Commands
`systemctl status iscsid`
`systemctl enable iscsid`
`systemctl restart iscsid`
`oc get nodes -o wide | grep worker`

Step 1. To run the above commands, find the IP address of the worker nodes to connect to each node as user **'core.'**



```
[administrator@ocp-installer ocp11]$
[administrator@ocp-installer ocp11]$
[administrator@ocp-installer ocp11]$ oc get nodes -o wide | grep worker
ocp11-9kbbs-worker-8m4tx   Ready    worker   99d    v1.22.8+9e95cb9   10.171.11.16   10.171.11.16
ed Hat Enterprise Linux CoreOS 49.84.202207191310-0 (Ootpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o
1.22.5-7.rhaos4.9.git3dbcd3c.el8
ocp11-9kbbs-worker-9s4pd   Ready    worker   99d    v1.22.8+9e95cb9   10.171.11.15   10.171.11.15
ed Hat Enterprise Linux CoreOS 49.84.202207191310-0 (Ootpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o
1.22.5-7.rhaos4.9.git3dbcd3c.el8
ocp11-9kbbs-worker-9td48   Ready    worker   100d   v1.22.8+9e95cb9   10.171.11.11   10.171.11.11
ed Hat Enterprise Linux CoreOS 49.84.202207191310-0 (Ootpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o
1.22.5-7.rhaos4.9.git3dbcd3c.el8
ocp11-9kbbs-worker-l7rlb   Ready    worker   100d   v1.22.8+9e95cb9   10.171.11.12   10.171.11.12
ed Hat Enterprise Linux CoreOS 49.84.202207191310-0 (Ootpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o
1.22.5-7.rhaos4.9.git3dbcd3c.el8
ocp11-9kbbs-worker-xm9dm   Ready    worker   100d   v1.22.8+9e95cb9   10.171.11.13   10.171.11.13
ed Hat Enterprise Linux CoreOS 49.84.202207191310-0 (Ootpa)   4.18.0-305.49.1.el8_4.x86_64   cri-o
1.22.5-7.rhaos4.9.git3dbcd3c.el8
[administrator@ocp-installer ocp11]$
```

Step 1. SSH as user ‘**core’** to the worker nodes. No password is necessary due to the SSH keys we passed to the nodes during the installation process.


10.10.171.235 (administrator) (1)
Multi-execution mode: commands are typed to all terminals (use Ctrl+Shift+Insert to paste)
Multi-paste
Exit multi-execution
□ Disable this terminal from "MultiExec" mode
□ Disable this terminal from "MultiExec" mode

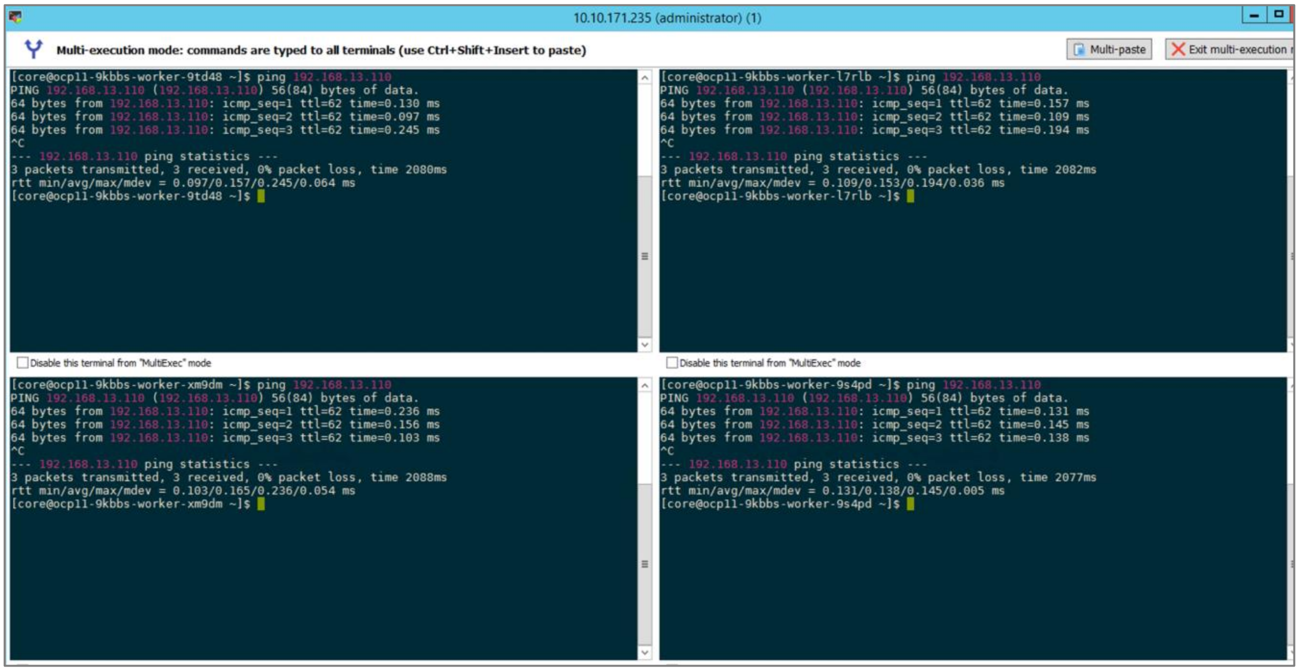
```
[administrator@ocp-installer HC-OpenShift]$ ssh core@10.171.11.11
Red Hat Enterprise Linux CoreOS 49.84.202207191310-0
  Part of OpenShift 4.9, RHCOS is a Kubernetes native operating system
  managed by the Machine Config Operator (`clusteroperator/machine-config`).

WARNING: Direct SSH access to machines is not recommended; instead,
make configuration changes via `machineconfig` objects:
  https://docs.openshift.com/container-platform/4.9/architecture/architecture-rhcos.html

---
Last login: Sun Aug 14 22:10:10 2022 from 10.10.171.235
[core@ocp11-9kbbs-worker-9td48 ~]$
```

```
[administrator@ocp-installer HC-OpenShift]$ ssh core@10.171.11.12
Red Hat Enterprise Linux CoreOS 49.84.202207191310-0
  Part of OpenShift 4.9, RHCOS is a Kubernetes native operating system
  managed by the Machine Config Operator (`clusteroperator/machine-config`).

WARNING: Direct SSH access to machines is not recommended; instead,
make configuration changes via `machineconfig` objects:
  https://docs.openshift.com/container-platform/4.9/architecture/architecture-rhcos.html

---
Last login: Sun Aug 14 22:10:10 2022 from 10.10.171.235
[core@ocp11-9kbbs-worker-l7rlb ~]$
```

```
[administrator@ocp-installer HC-OpenShift]$ ssh core@10.171.11.13
Red Hat Enterprise Linux CoreOS 49.84.202207191310-0
  Part of OpenShift 4.9, RHCOS is a Kubernetes native operating system
  managed by the Machine Config Operator (`clusteroperator/machine-config`).

WARNING: Direct SSH access to machines is not recommended; instead,
make configuration changes via `machineconfig` objects:
  https://docs.openshift.com/container-platform/4.9/architecture/architecture-rhcos.html

---
Last login: Sun Aug 14 22:10:10 2022 from 10.10.171.235
[core@ocp11-9kbbs-worker-xm9dm ~]$
```

```
[administrator@ocp-installer HC-OpenShift]$ ssh core@10.171.11.15
Red Hat Enterprise Linux CoreOS 49.84.202207191310-0
  Part of OpenShift 4.9, RHCOS is a Kubernetes native operating system
  managed by the Machine Config Operator (`clusteroperator/machine-config`).

WARNING: Direct SSH access to machines is not recommended; instead,
make configuration changes via `machineconfig` objects:
  https://docs.openshift.com/container-platform/4.9/architecture/architecture-rhcos.html

---
Last login: Sun Aug 14 22:10:10 2022 from 10.10.171.235
[core@ocp11-9kbbs-worker-9s4pd ~]$
```

Step 2. Verify the worker nodes can ping the iSCSI Cluster IP on HyperFlex nodes.



10.10.171.235 (administrator) (1)

Multi-execution mode: commands are typed to all terminals (use Ctrl+Shift+Insert to paste)

```
[core@ocp11-9kbbs-worker-9td48 ~]$ ping 192.168.13.110
PING 192.168.13.110 (192.168.13.110) 56(84) bytes of data.
64 bytes from 192.168.13.110: icmp_seq=1 ttl=62 time=0.130 ms
64 bytes from 192.168.13.110: icmp_seq=2 ttl=62 time=0.097 ms
64 bytes from 192.168.13.110: icmp_seq=3 ttl=62 time=0.245 ms
^C
--- 192.168.13.110 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2080ms
rtt min/avg/max/mdev = 0.097/0.157/0.245/0.064 ms
[core@ocp11-9kbbs-worker-9td48 ~]$
```

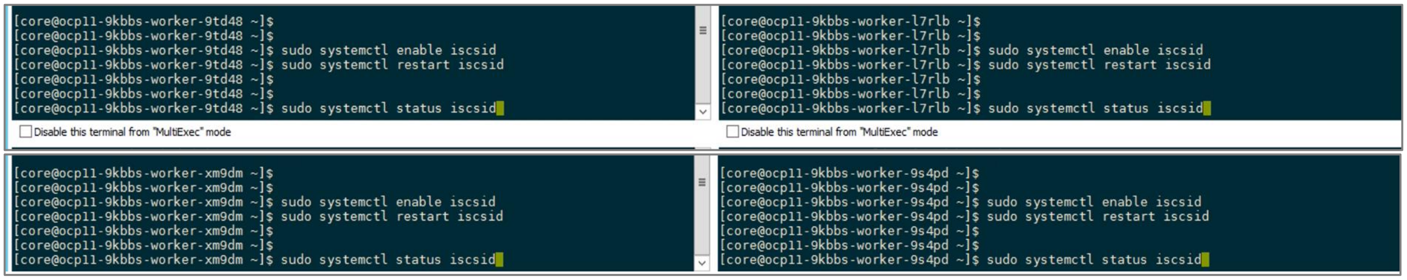
```
[core@ocp11-9kbbs-worker-l7rlb ~]$ ping 192.168.13.110
PING 192.168.13.110 (192.168.13.110) 56(84) bytes of data.
64 bytes from 192.168.13.110: icmp_seq=1 ttl=62 time=0.157 ms
64 bytes from 192.168.13.110: icmp_seq=2 ttl=62 time=0.109 ms
64 bytes from 192.168.13.110: icmp_seq=3 ttl=62 time=0.194 ms
^C
--- 192.168.13.110 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2082ms
rtt min/avg/max/mdev = 0.109/0.153/0.194/0.036 ms
[core@ocp11-9kbbs-worker-l7rlb ~]$
```

□ Disable this terminal from "MultiExec" mode
□ Disable this terminal from "MultiExec" mode

```
[core@ocp11-9kbbs-worker-xm9dm ~]$ ping 192.168.13.110
PING 192.168.13.110 (192.168.13.110) 56(84) bytes of data.
64 bytes from 192.168.13.110: icmp_seq=1 ttl=62 time=0.236 ms
64 bytes from 192.168.13.110: icmp_seq=2 ttl=62 time=0.156 ms
64 bytes from 192.168.13.110: icmp_seq=3 ttl=62 time=0.103 ms
^C
--- 192.168.13.110 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2088ms
rtt min/avg/max/mdev = 0.103/0.165/0.236/0.054 ms
[core@ocp11-9kbbs-worker-xm9dm ~]$
```

```
[core@ocp11-9kbbs-worker-9s4pd ~]$ ping 192.168.13.110
PING 192.168.13.110 (192.168.13.110) 56(84) bytes of data.
64 bytes from 192.168.13.110: icmp_seq=1 ttl=62 time=0.131 ms
64 bytes from 192.168.13.110: icmp_seq=2 ttl=62 time=0.145 ms
64 bytes from 192.168.13.110: icmp_seq=3 ttl=62 time=0.138 ms
^C
--- 192.168.13.110 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2077ms
rtt min/avg/max/mdev = 0.131/0.138/0.145/0.005 ms
[core@ocp11-9kbbs-worker-9s4pd ~]$
```

Step 3. Enable and restart the iSCSI daemon process.



```
[core@ocp11-9kbbs-worker-9td48 ~]$
[core@ocp11-9kbbs-worker-9td48 ~]$
[core@ocp11-9kbbs-worker-9td48 ~]$ sudo systemctl enable iscsid
[core@ocp11-9kbbs-worker-9td48 ~]$ sudo systemctl restart iscsid
[core@ocp11-9kbbs-worker-9td48 ~]$
[core@ocp11-9kbbs-worker-9td48 ~]$
[core@ocp11-9kbbs-worker-9td48 ~]$ sudo systemctl status iscsid
```

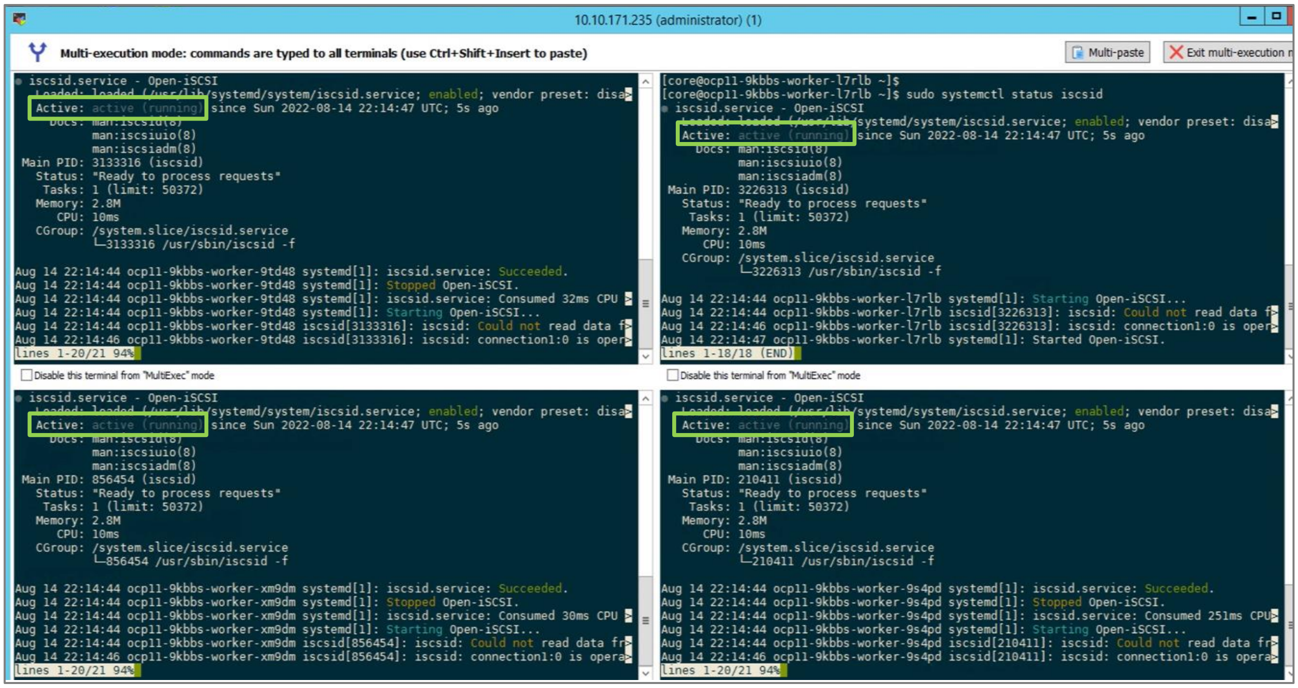
```
[core@ocp11-9kbbs-worker-l7rlb ~]$
[core@ocp11-9kbbs-worker-l7rlb ~]$
[core@ocp11-9kbbs-worker-l7rlb ~]$ sudo systemctl enable iscsid
[core@ocp11-9kbbs-worker-l7rlb ~]$ sudo systemctl restart iscsid
[core@ocp11-9kbbs-worker-l7rlb ~]$
[core@ocp11-9kbbs-worker-l7rlb ~]$
[core@ocp11-9kbbs-worker-l7rlb ~]$ sudo systemctl status iscsid
```

□ Disable this terminal from "MultiExec" mode
□ Disable this terminal from "MultiExec" mode

```
[core@ocp11-9kbbs-worker-xm9dm ~]$
[core@ocp11-9kbbs-worker-xm9dm ~]$
[core@ocp11-9kbbs-worker-xm9dm ~]$ sudo systemctl enable iscsid
[core@ocp11-9kbbs-worker-xm9dm ~]$ sudo systemctl restart iscsid
[core@ocp11-9kbbs-worker-xm9dm ~]$
[core@ocp11-9kbbs-worker-xm9dm ~]$
[core@ocp11-9kbbs-worker-xm9dm ~]$ sudo systemctl status iscsid
```

```
[core@ocp11-9kbbs-worker-9s4pd ~]$
[core@ocp11-9kbbs-worker-9s4pd ~]$
[core@ocp11-9kbbs-worker-9s4pd ~]$ sudo systemctl enable iscsid
[core@ocp11-9kbbs-worker-9s4pd ~]$ sudo systemctl restart iscsid
[core@ocp11-9kbbs-worker-9s4pd ~]$
[core@ocp11-9kbbs-worker-9s4pd ~]$
[core@ocp11-9kbbs-worker-9s4pd ~]$ sudo systemctl status iscsid
```

Step 4. Verify the status of the iSCSI daemon – it should be **Active** and **Running**.



10.10.171.235 (administrator) (1)

Multi-execution mode: commands are typed to all terminals (use Ctrl+Shift+Insert to paste)

```
● iscsid.service - Open-iSCSI
   Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled; vendor preset: disa>
   Active: active (running) since Sun 2022-08-14 22:14:47 UTC; 5s ago
     Docs: man:iscsid(8)
           man:iscsiuio(8)
           man:iscsiadm(8)
 Main PID: 3133316 (iscsid)
   Status: "Ready to process requests"
    Tasks: 1 (limit: 50372)
   Memory: 2.8M
      CPU: 10ms
   CGroup: /system.slice/iscsid.service
           └─3133316 /usr/sbin/iscsid -f

Aug 14 22:14:44 ocp11-9kbbs-worker-9td48 systemd[1]: iscsid.service: Succeeded.
Aug 14 22:14:44 ocp11-9kbbs-worker-9td48 systemd[1]: Stopped Open-iSCSI.
Aug 14 22:14:44 ocp11-9kbbs-worker-9td48 systemd[1]: iscsid.service: Consumed 32ms CPU >
Aug 14 22:14:44 ocp11-9kbbs-worker-9td48 systemd[1]: Starting Open-iSCSI...
Aug 14 22:14:44 ocp11-9kbbs-worker-9td48 iscsid[3133316]: iscsid: Could not read data f>
Aug 14 22:14:46 ocp11-9kbbs-worker-9td48 iscsid[3133316]: iscsid: connection1:0 is oper>
lines 1-20/21 94%
```

```
[core@ocp11-9kbbs-worker-l7rlb ~]$
[core@ocp11-9kbbs-worker-l7rlb ~]$ sudo systemctl status iscsid
● iscsid.service - Open-iSCSI
   Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled; vendor preset: disa>
   Active: active (running) since Sun 2022-08-14 22:14:47 UTC; 5s ago
     Docs: man:iscsid(8)
           man:iscsiuio(8)
           man:iscsiadm(8)
 Main PID: 3226313 (iscsid)
   Status: "Ready to process requests"
    Tasks: 1 (limit: 50372)
   Memory: 2.8M
      CPU: 10ms
   CGroup: /system.slice/iscsid.service
           └─3226313 /usr/sbin/iscsid -f

Aug 14 22:14:44 ocp11-9kbbs-worker-l7rlb systemd[1]: Starting Open-iSCSI...
Aug 14 22:14:44 ocp11-9kbbs-worker-l7rlb iscsid[3226313]: iscsid: Could not read data f>
Aug 14 22:14:46 ocp11-9kbbs-worker-l7rlb iscsid[3226313]: iscsid: connection1:0 is oper>
Aug 14 22:14:47 ocp11-9kbbs-worker-l7rlb systemd[1]: Started Open-iSCSI.
lines 1-18/18 (END)
```

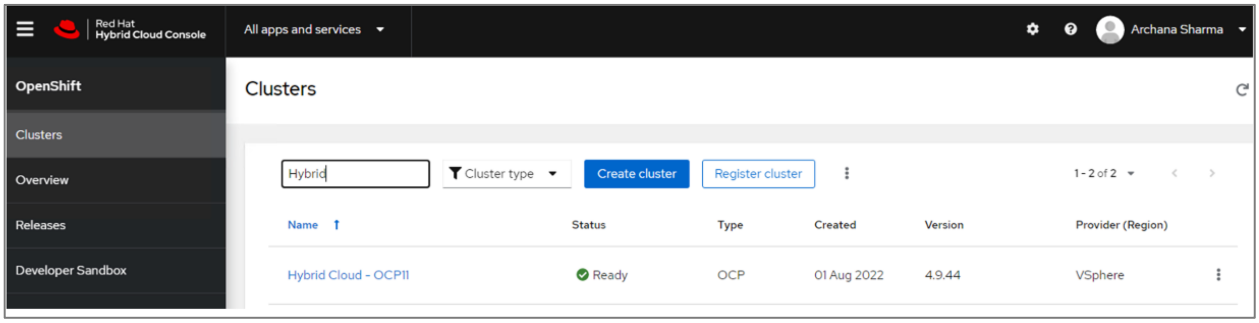
□ Disable this terminal from "MultiExec" mode
□ Disable this terminal from "MultiExec" mode

```
● iscsid.service - Open-iSCSI
   Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled; vendor preset: disa>
   Active: active (running) since Sun 2022-08-14 22:14:47 UTC; 5s ago
     Docs: man:iscsid(8)
           man:iscsiuio(8)
           man:iscsiadm(8)
 Main PID: 856454 (iscsid)
   Status: "Ready to process requests"
    Tasks: 1 (limit: 50372)
   Memory: 2.8M
      CPU: 10ms
   CGroup: /system.slice/iscsid.service
           └─856454 /usr/sbin/iscsid -f

Aug 14 22:14:44 ocp11-9kbbs-worker-xm9dm systemd[1]: iscsid.service: Succeeded.
Aug 14 22:14:44 ocp11-9kbbs-worker-xm9dm systemd[1]: Stopped Open-iSCSI.
Aug 14 22:14:44 ocp11-9kbbs-worker-xm9dm systemd[1]: iscsid.service: Consumed 30ms CPU >
Aug 14 22:14:44 ocp11-9kbbs-worker-xm9dm systemd[1]: Starting Open-iSCSI...
Aug 14 22:14:44 ocp11-9kbbs-worker-xm9dm iscsid[856454]: iscsid: Could not read data fr>
Aug 14 22:14:46 ocp11-9kbbs-worker-xm9dm iscsid[856454]: iscsid: connection1:0 is opera>
lines 1-20/21 94%
```

```
● iscsid.service - Open-iSCSI
   Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled; vendor preset: disa>
   Active: active (running) since Sun 2022-08-14 22:14:47 UTC; 5s ago
     Docs: man:iscsid(8)
           man:iscsiuio(8)
           man:iscsiadm(8)
 Main PID: 210411 (iscsid)
   Status: "Ready to process requests"
    Tasks: 1 (limit: 50372)
   Memory: 2.8M
      CPU: 10ms
   CGroup: /system.slice/iscsid.service
           └─210411 /usr/sbin/iscsid -f

Aug 14 22:14:44 ocp11-9kbbs-worker-9s4pd systemd[1]: iscsid.service: Succeeded.
Aug 14 22:14:44 ocp11-9kbbs-worker-9s4pd systemd[1]: Stopped Open-iSCSI.
Aug 14 22:14:44 ocp11-9kbbs-worker-9s4pd systemd[1]: iscsid.service: Consumed 251ms CPU>
Aug 14 22:14:44 ocp11-9kbbs-worker-9s4pd systemd[1]: Starting Open-iSCSI...
Aug 14 22:14:44 ocp11-9kbbs-worker-9s4pd iscsid[210411]: iscsid: Could not read data fr>
Aug 14 22:14:46 ocp11-9kbbs-worker-9s4pd iscsid[210411]: iscsid: connection1:0 is opera>
lines 1-20/21 94%
```

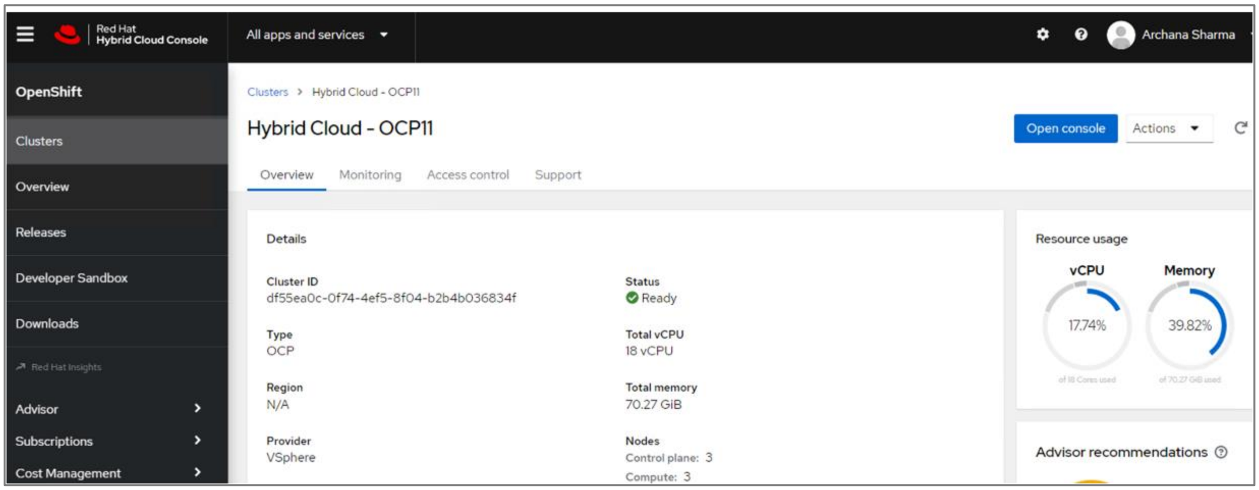
Procedure 4. Deploy HyperFlex CSI Operator from Red Hat Operator Hub

Step 1. Navigate using a web browser to the OpenShift Hybrid Cloud Console. Login using your Red Hat account. Navigate to Clusters and find the newly created OCP cluster. Select and click on the cluster name.



Name	Status	Type	Created	Version	Provider (Region)
Hybrid Cloud - OCP11	Ready	OCP	01 Aug 2022	4.9.44	VSphere

Step 2. Click on **Open console** from the top-right corner and login to the cluster console as **kubeadm** user.



Details

Cluster ID
df55ea0c-0f74-4ef5-8f04-b2b4b036834f

Status
Ready

Type
OCP

Total vCPU
18 vCPU

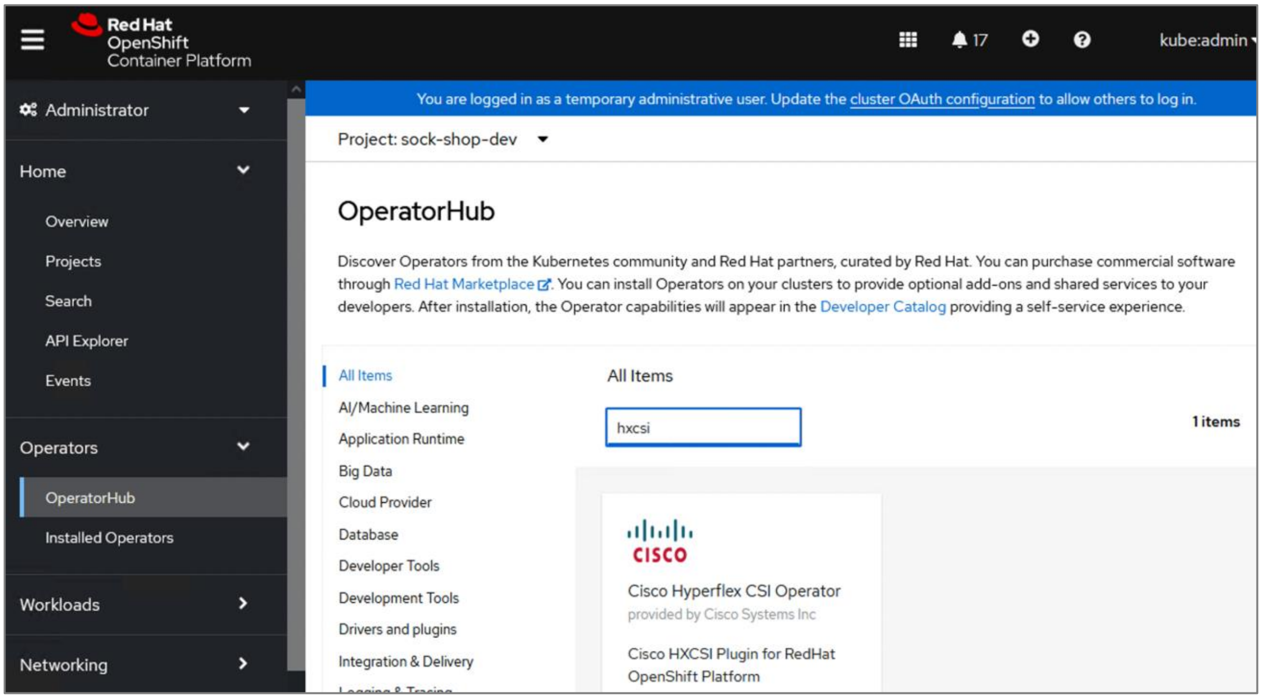
Region
N/A

Total memory
70.27 GiB

Provider
VSphere

Nodes
Control plane: 3
Compute: 3

Step 3. From the cluster console, navigate to **Operators > OperatorHub**. Search for **'hxcsi'** in the search box on the right-side window. Select and click on the one item (**Cisco HyperFlex CSI Operator**) that is listed.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

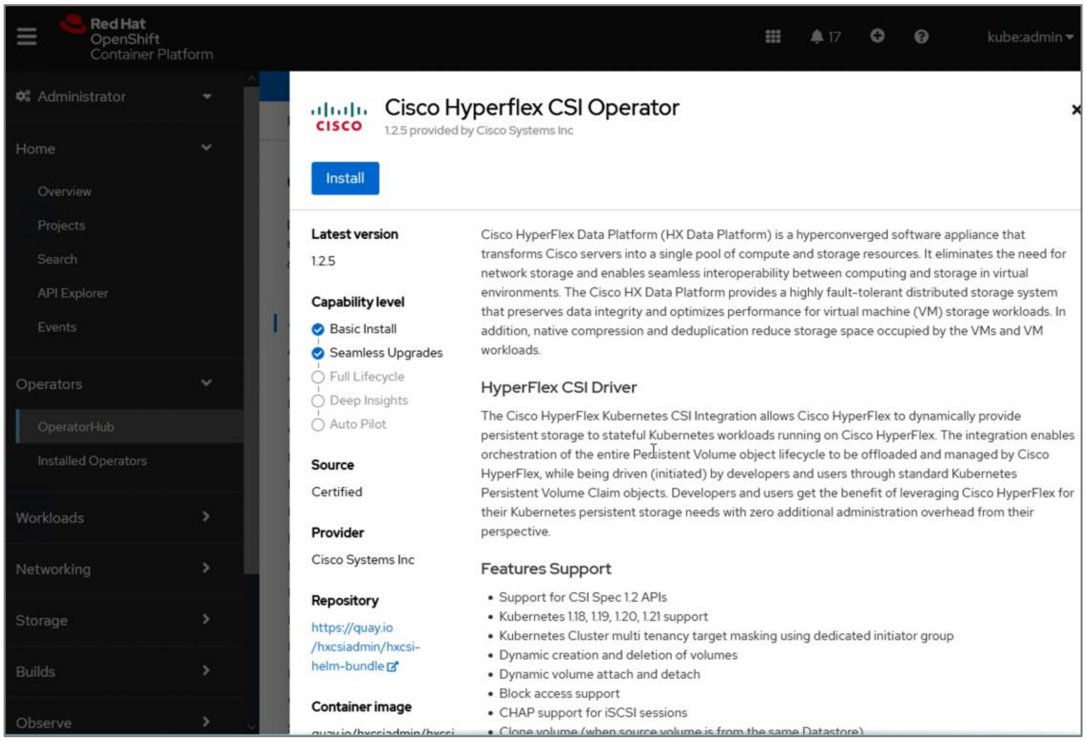
OperatorHub

Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. You can purchase commercial software through Red Hat Marketplace. You can install Operators on your clusters to provide optional add-ons and shared services to your developers. After installation, the Operator capabilities will appear in the Developer Catalog providing a self-service experience.

Cisco Hyperflex CSI Operator
provided by Cisco Systems Inc

Cisco HXCSI Plugin for RedHat OpenShift Platform

Step 4. In the Cisco HyperFlex CSI Operator pop-up window, click **Install**.



Cisco Hyperflex CSI Operator

1.2.5 provided by Cisco Systems Inc

Latest version
1.2.5

Capability level
- ● Basic Install
- ● Seamless Upgrades
- ○ Full Lifecycle
- ○ Deep Insights
- ○ Auto Pilot

Source
Certified

Provider
Cisco Systems Inc

Repository
https://quay.io/hxcsiadmin/hxcsi-helm-bundle

Container image
quay.io/hxcsiadmin/hxcsi-

Cisco HyperFlex Data Platform (HX Data Platform) is a hyperconverged software appliance that transforms Cisco servers into a single pool of compute and storage resources. It eliminates the need for network storage and enables seamless interoperability between computing and storage in virtual environments. The Cisco HX Data Platform provides a highly fault-tolerant distributed storage system that preserves data integrity and optimizes performance for virtual machine (VM) storage workloads. In addition, native compression and deduplication reduce storage space occupied by the VMs and VM workloads.

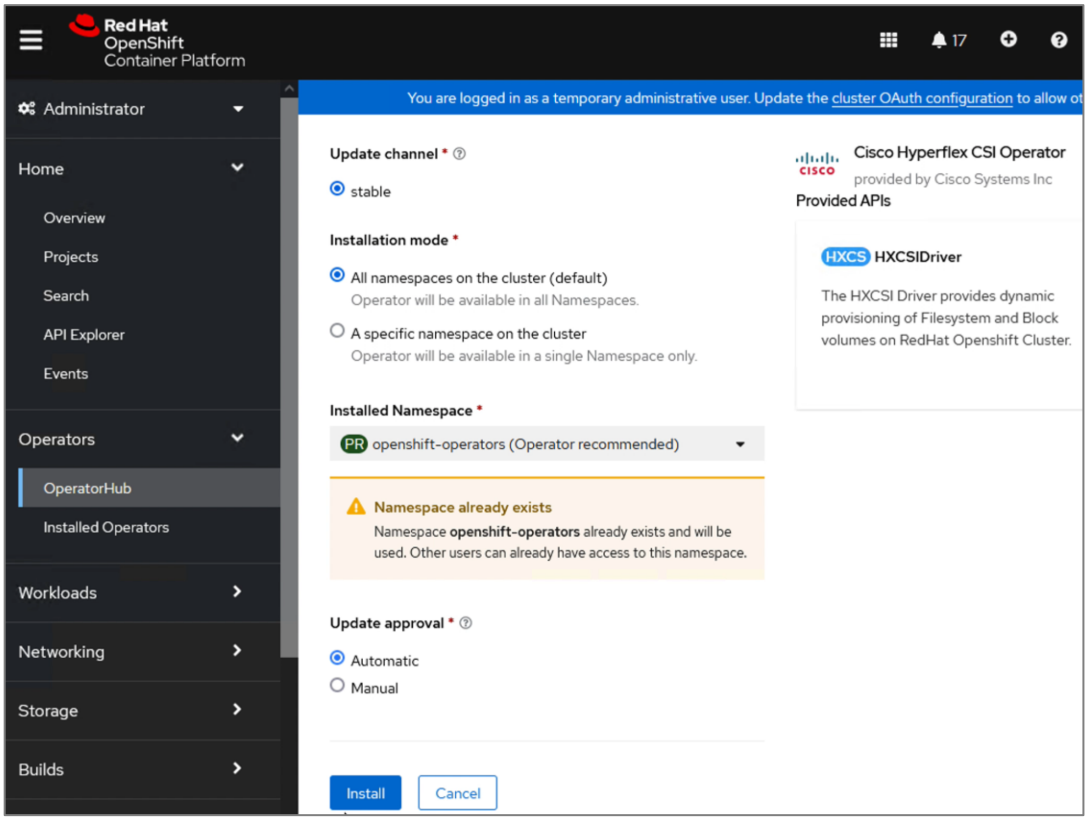
HyperFlex CSI Driver

The Cisco HyperFlex Kubernetes CSI Integration allows Cisco HyperFlex to dynamically provide persistent storage to stateful Kubernetes workloads running on Cisco HyperFlex. The integration enables orchestration of the entire Persistent Volume object lifecycle to be offloaded and managed by Cisco HyperFlex, while being driven (initiated) by developers and users through standard Kubernetes Persistent Volume Claim objects. Developers and users get the benefit of leveraging Cisco HyperFlex for their Kubernetes persistent storage needs with zero additional administration overhead from their perspective.

Features Support

- Support for CSI Spec 1.2 APIs
- Kubernetes 1.18, 1.19, 1.20, 1.21 support
- Kubernetes Cluster multi tenancy target masking using dedicated initiator group
- Dynamic creation and deletion of volumes
- Dynamic volume attach and detach
- Block access support
- CHAP support for iSCSI sessions
- Clone volume (when source volume is from the same Datastore)

Step 5. Select the defaults or change the options as needed and click **Install**.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow ot

Update channel *
- ● stable

Installation mode *
- ● All namespaces on the cluster (default)
  Operator will be available in all Namespaces.
- ○ A specific namespace on the cluster
  Operator will be available in a single Namespace only.

Installed Namespace *
PR openshift-operators (Operator recommended)

Namespace already exists
Namespace **openshift-operators** already exists and will be used. Other users can already have access to this namespace.

Update approval *
- ● Automatic
- ○ Manual

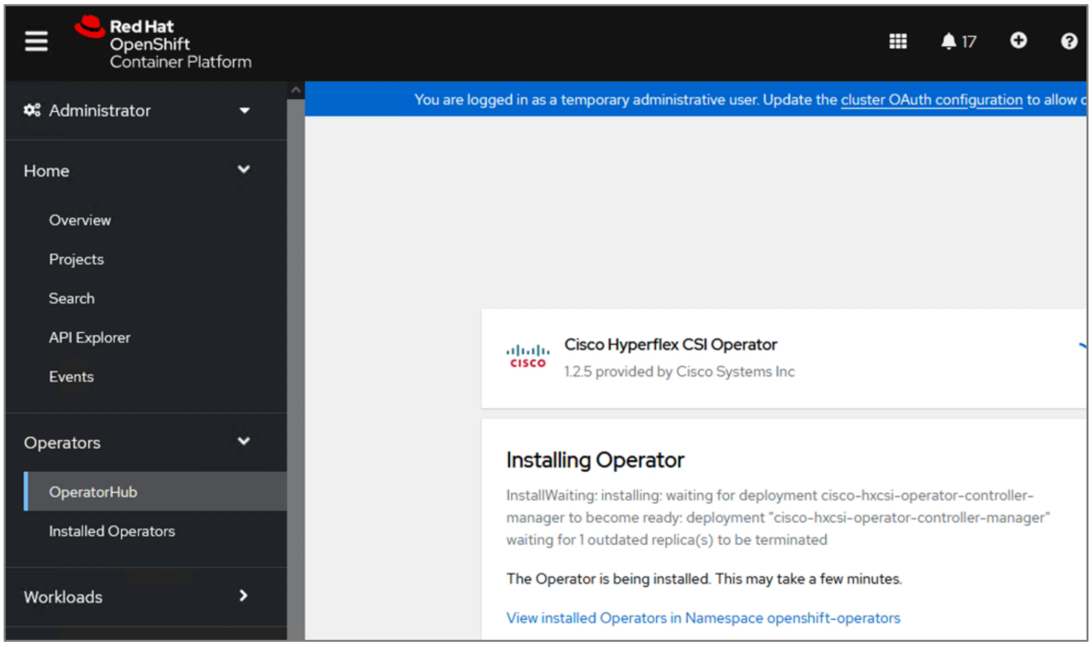
Cisco Hyperflex CSI Operator
provided by Cisco Systems Inc

Provided APIs

HXCS HXCSIDriver

The HXCSI Driver provides dynamic provisioning of Filesystem and Block volumes on RedHat Openshift Cluster.

Step 6. Verify the install is successful.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow c

Cisco Hyperflex CSI Operator

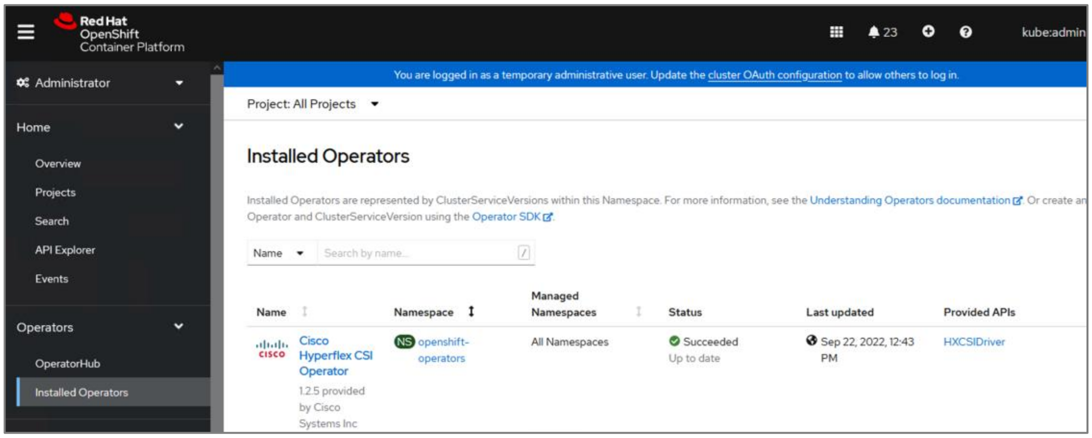
1.2.5 provided by Cisco Systems Inc

Installing Operator

InstallWaiting: installing: waiting for deployment cisco-hxcsi-operator-controller-manager to become ready: deployment "cisco-hxcsi-operator-controller-manager" waiting for 1 outdated replica(s) to be terminated

The Operator is being installed. This may take a few minutes.

View installed Operators in Namespace openshift-operators



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: All Projects

Installed Operators

Installed Operators are represented by ClusterServiceVersions within this Namespace. For more information, see the Understanding Operators documentation. Or create an Operator and ClusterServiceVersion using the Operator SDK.

Name | Search by name...

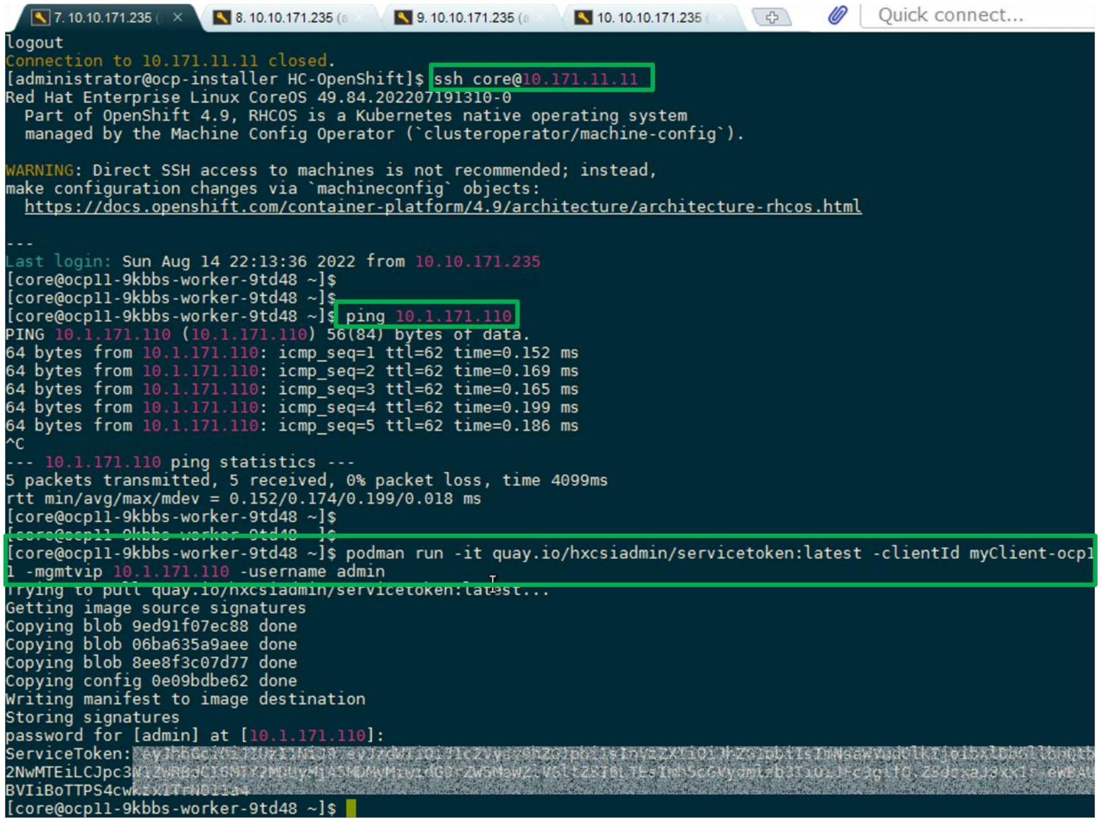
Name	Namespace	Managed Namespaces	Status	Last updated	Provided APIs
Cisco Hyperflex CSI Operator 1.2.5 provided by Cisco Systems Inc	NS openshift-operators	All Namespaces	Succeeded Up to date	Sep 22, 2022, 12:43 PM	HXCSIDriver

Procedure 5. Generate API Token for HyperFlex Cluster

This procedure enables you to generate an API Token for the HyperFlex cluster so that the CSI plugin can communicate with Cisco HyperFlex for storage related activities. The generated API token must be provided to the HyperFlex CSI plugin.

Step 1. Execute the following command to generate the API token for the CSI plugin:

```
podman run -it quay.io/hxcsiadmin/servicetoken:latest -clientId myClient-ocp11 -mgmtvip
<HyperFlex_ClusterIP> -username admin
```



```
logout
Connection to 10.171.11.11 closed.
[administrator@ocp-installer HC-OpenShift]$ ssh core@10.171.11.11
Red Hat Enterprise Linux CoreOS 49.84.202207191310-0
  Part of OpenShift 4.9, RHCOS is a Kubernetes native operating system
  managed by the Machine Config Operator (`clusteroperator/machine-config`).

WARNING: Direct SSH access to machines is not recommended; instead,
make configuration changes via `machineconfig` objects:
  https://docs.openshift.com/container-platform/4.9/architecture/architecture-rhcos.html

---
Last login: Sun Aug 14 22:13:36 2022 from 10.10.171.235
[core@ocp11-9kbbs-worker-9td48 ~]$
[core@ocp11-9kbbs-worker-9td48 ~]$
[core@ocp11-9kbbs-worker-9td48 ~]$ ping 10.1.171.110
PING 10.1.171.110 (10.1.171.110) 56(84) bytes of data.
64 bytes from 10.1.171.110: icmp_seq=1 ttl=62 time=0.152 ms
64 bytes from 10.1.171.110: icmp_seq=2 ttl=62 time=0.169 ms
64 bytes from 10.1.171.110: icmp_seq=3 ttl=62 time=0.165 ms
64 bytes from 10.1.171.110: icmp_seq=4 ttl=62 time=0.199 ms
64 bytes from 10.1.171.110: icmp_seq=5 ttl=62 time=0.186 ms
^C
--- 10.1.171.110 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4099ms
rtt min/avg/max/mdev = 0.152/0.174/0.199/0.018 ms
[core@ocp11-9kbbs-worker-9td48 ~]$
[core@ocp11-9kbbs-worker-9td48 ~]$
[core@ocp11-9kbbs-worker-9td48 ~]$ podman run -it quay.io/hxcsiadmin/servicetoken:latest -clientId myClient-ocp1
1 -mgmtvip 10.1.171.110 -username admin
Trying to pull quay.io/hxcsiadmin/servicetoken:latest...
Getting image source signatures
Copying blob 9ed91f07ec88 done
Copying blob 06ba635a9aee done
Copying blob 8ee8f3c07d77 done
Copying config 0e09bdbe62 done
Writing manifest to image destination
Storing signatures
password for [admin] at [10.1.171.110]:
ServiceToken:[redacted]
2NwMTEiLCJpc3[redacted]
BVIiBoTTPS4cw[redacted]
[core@ocp11-9kbbs-worker-9td48 ~]$
```

Step 2. Copy the output of the line beginning with **ServiceToken.** This is the raw token. Copy this token and the **client ID** used in the **'podman'** command for use in the next step.

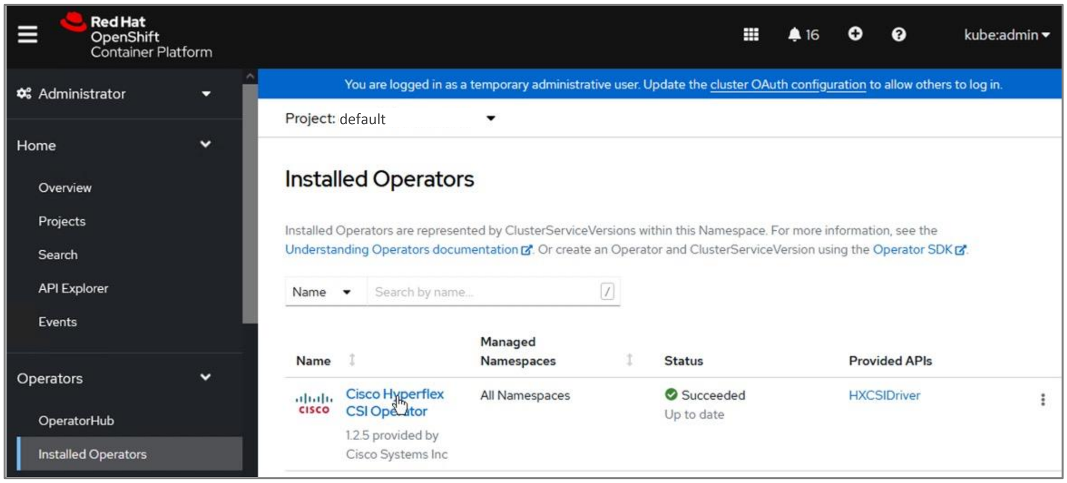
Procedure 6. Install HyperFlex CSI driver on Red Hat OCP cluster

To enable workloads running in Kubernetes Pods to request persistent storage, the HyperFlex CSI driver must be installed. The CSI driver creates CSI Pods that enable CSI functions.

Step 1. Navigate using a web browser to the OpenShift Hybrid Cloud Console. Login using your Red Hat account. Navigate to Clusters and find the newly created OCP cluster. Select and click on the cluster name.

Step 2. Click on **Open console** from the top-right corner and login to the cluster console as **kubeadm** user.

Step 3. Navigate **Operators > Installed Operators**. Select the default project from the drop-down list at the top of the window.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: default

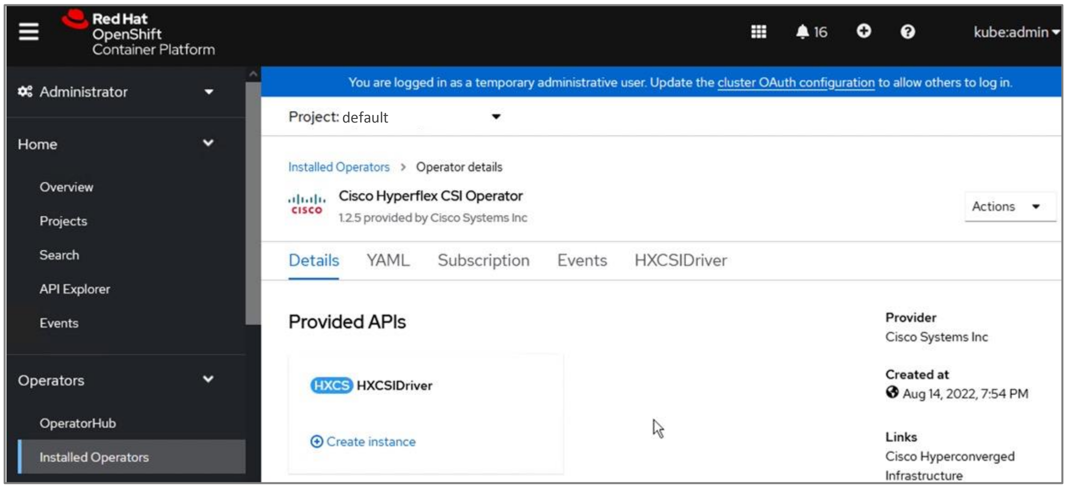
Installed Operators

Installed Operators are represented by ClusterServiceVersions within this Namespace. For more information, see the Understanding Operators documentation. Or create an Operator and ClusterServiceVersion using the Operator SDK.

Name ▾ Search by name...

Name	Managed Namespaces	Status	Provided APIs
Cisco Hyperflex CSI Operator 1.2.5 provided by Cisco Systems Inc	All Namespaces	✓ Succeeded Up to date	HXCSIDriver

Step 4. Select and click the **Cisco HyperFlex CSI Operator**.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: default

Cisco Hyperflex CSI Operator
1.2.5 provided by Cisco Systems Inc

Provided APIs

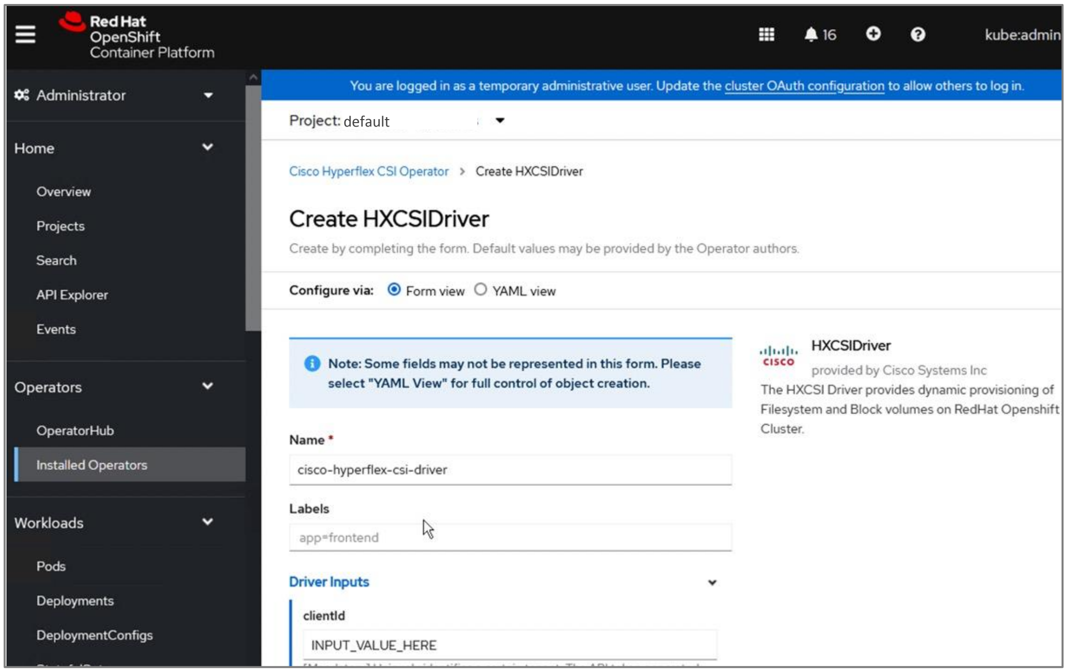
HXCS HXCSIDriver

Provider
Cisco Systems Inc

Created at
Aug 14, 2022, 7:54 PM

Links
Cisco Hyperconverged Infrastructure

Step 5. Click on **Create instance**.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: default

Create HXCSIDriver

Create by completing the form. Default values may be provided by the Operator authors.

Configure via: ● Form view ○ YAML view

Note: Some fields may not be represented in this form. Please select "YAML View" for full control of object creation.

HXCSIDriver
provided by Cisco Systems Inc
The HXCSI Driver provides dynamic provisioning of Filesystem and Block volumes on RedHat Openshift Cluster.

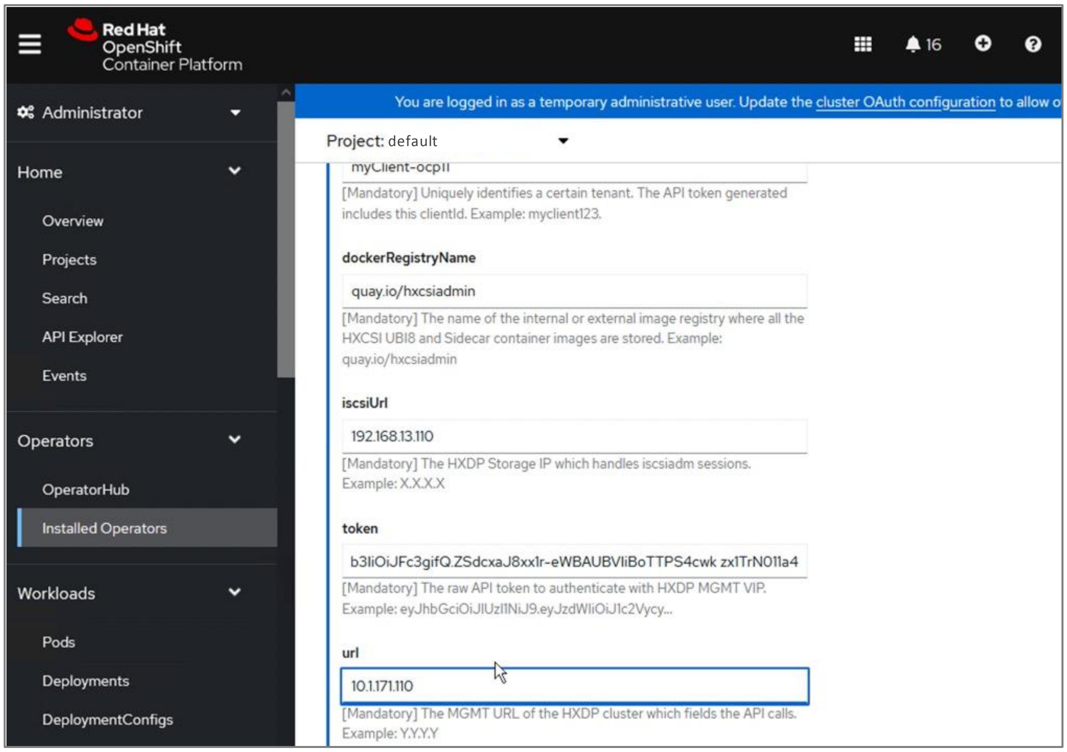
Name *
cisco-hyperflex-csi-driver

Labels
app=frontend

clientId
INPUT_VALUE_HERE

Step 6. Fill in the following information in the **Create HXCSIDriver** pop-up window.

- **clientId**: This must match the clientId used to collect the HyperFlex API
- **dockerRegistryName**: Enter the public or private registry containing the HXCSI container images (**quay.io/hxcsiadmin**).
- **iscsiUrl**: Enter the HyperFlex iSCSI Cluster IP address.
- **token**: Enter the raw token collected from HyperFlex using the **podman** command earlier.
- **url**: Enter the HyperFlex Management Cluster IP address.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow o

Project: default

myClient-ocpII

[Mandatory] Uniquely identifies a certain tenant. The API token generated includes this clientId. Example: myclient123.

dockerRegistryName

quay.io/hxcsiadmin

[Mandatory] The name of the internal or external image registry where all the HXCSI UBI8 and Sidecar container images are stored. Example: quay.io/hxcsiadmin

iscsiUrl

192.168.13.110

[Mandatory] The HXDP Storage IP which handles iscsiadm sessions. Example: X.X.X.X

token

b3liOiJFc3gifQ.ZSdcxaJ8xx1r-eWBAUBVliBoTTPS4cwk zx1TrN011a4

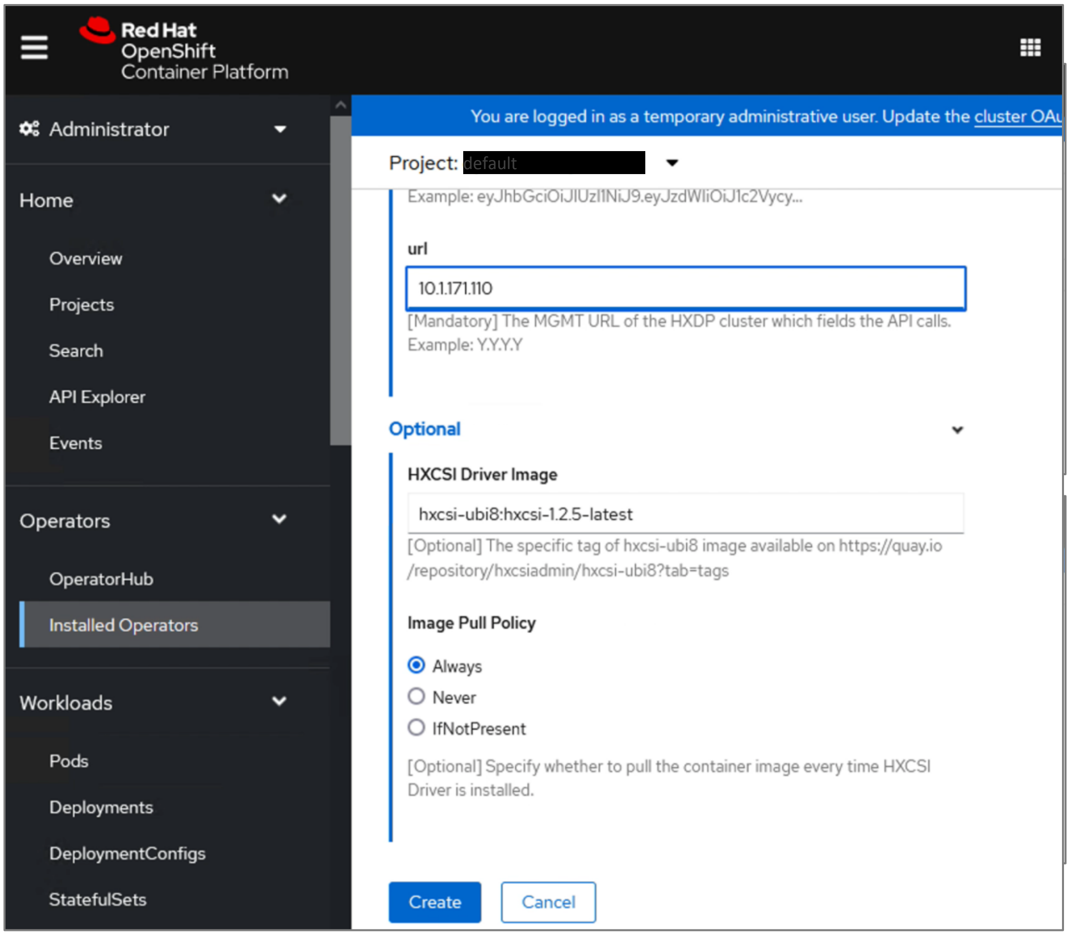
[Mandatory] The raw API token to authenticate with HXDP MGMT VIP. Example: eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2Vycy...

url

10.1.171.110

[Mandatory] The MGMT URL of the HXDP cluster which fields the API calls. Example: Y.Y.Y.Y

Step 7. Click **Create.**



You are logged in as a temporary administrative user. Update the cluster OAu

Project: [redacted]

Example: eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2Vycy...

url

10.1.171.110

[Mandatory] The MGMT URL of the HXDP cluster which fields the API calls. Example: Y.Y.Y.Y

Optional

HXCSI Driver Image

hxcsi-ubi8:hxcsi-1.2.5-latest

[Optional] The specific tag of hxcsi-ubi8 image available on https://quay.io /repository/hxcsiadmin/hxcsi-ubi8?tab=tags

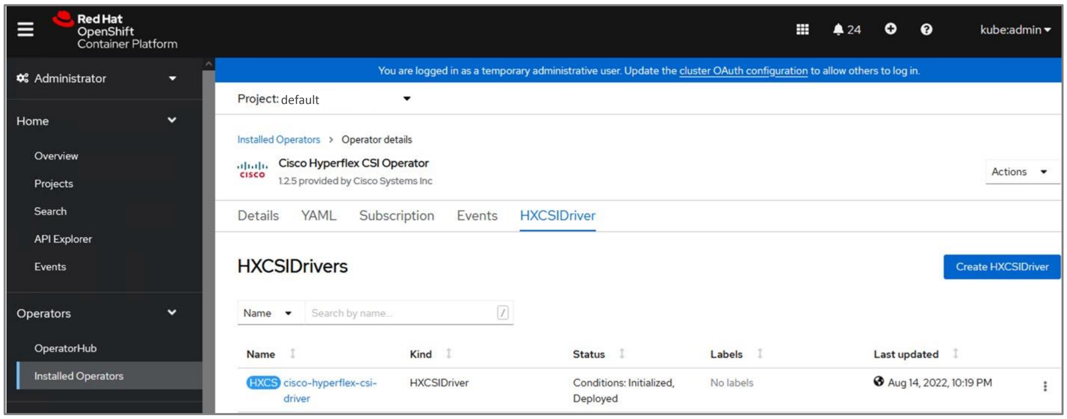
Image Pull Policy

● Always
○ Never
○ IfNotPresent

[Optional] Specify whether to pull the container image every time HXCSI Driver is installed.

Create Cancel

Step 8. Check the status of the driver – verify it is successfully deployed.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

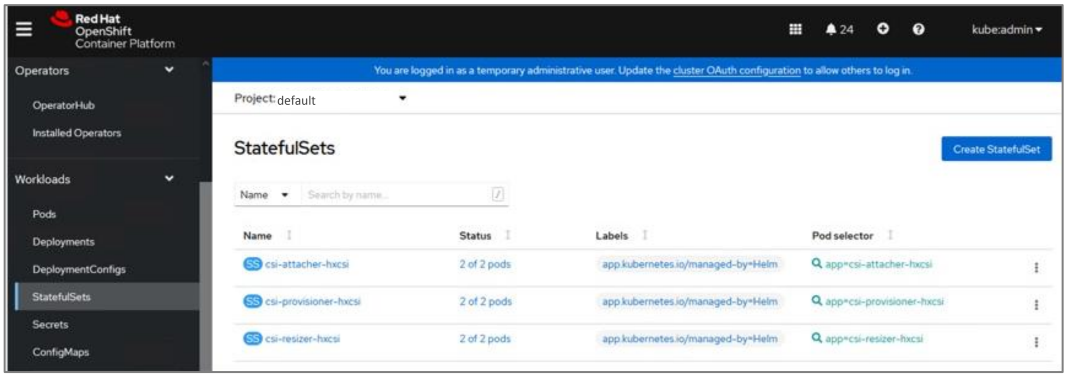
Project: default

Cisco Hyperflex CSI Operator
1.2.5 provided by Cisco Systems Inc

HXCSIDrivers

Name	Kind	Status	Labels	Last updated
HXCS cisco-hyperflex-csi-driver	HXCSIDriver	Conditions: Initialized, Deployed	No labels	Aug 14, 2022, 10:19 PM

Step 9. Verify the status of **StatefulSets** under **Workloads** – see **Status** of the Pods below.



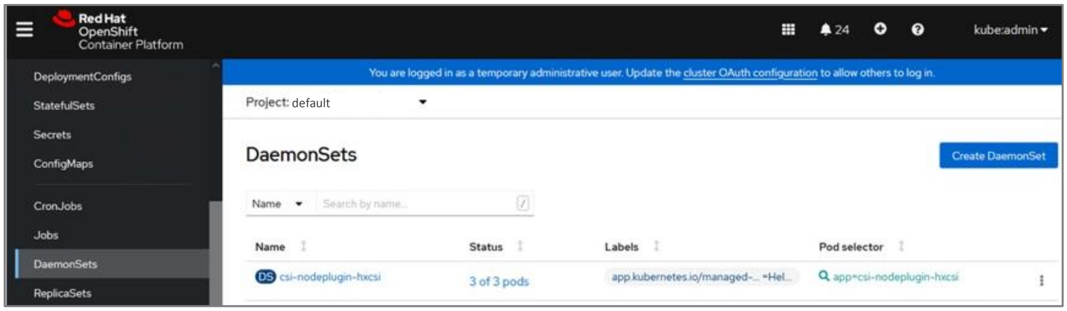
You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: default

StatefulSets

Name	Status	Labels	Pod selector
SS csi-attacher-hxcsi	2 of 2 pods	app.kubernetes.io/managed-by=Helm	app=csi-attacher-hxcsi
SS csi-provisioner-hxcsi	2 of 2 pods	app.kubernetes.io/managed-by=Helm	app=csi-provisioner-hxcsi
SS csi-resizer-hxcsi	2 of 2 pods	app.kubernetes.io/managed-by=Helm	app=csi-resizer-hxcsi

Step 10. Verify the status of **DaemonSets** under Workloads – see **Status** of the Pods below.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: default

DaemonSets

Name	Status	Labels	Pod selector
DS csi-nodeplugin-hxcsi	3 of 3 pods	app.kubernetes.io/managed-...=Hel...	app=csi-nodeplugin-hxcsi

The deployment of the on-prem Red Hat OCP cluster is now complete and ready for deploying cloud-native applications.

Procedure 7. Test and verify – Create Storage Class

With the HyperFlex CSI plugin deployed on the OCP cluster, complete the following steps to deploy a storage class to provision persistent volumes on Cisco Hyperflex Distributed Storage Filesystem. Storage Class links defines the name and size of the datastore within the Cisco HyperFlex cluster and specifies the filesystem format (default is **ext4**) to use.

Step 1. Log into the Red Hat OCP console for the cluster.

Step 2. From the left-hand navigation menu, select **Storage**, then click on **Storage Classes**.

Step 3. Click **Create Storage Class**, then **Edit YAML**.

Step 4. Modify the YAML file, then click **Create**.

Procedure 8. Test and verify – Create Persistent Volume Claim

Persistent Volume Claims specify a volume to be created and attached to a pod. The claim specifies the size of the volume, the type of volume, either filesystem or block, and if the volume is read/write accessible by only one pod or many pods at one time.

Step 1. Log into the Red Hat OCP console for the cluster.

Step 2. From the left-hand navigation menu, select **Storage**, then click on **Persistent Volume Claims**.

Step 3. Click **Create Persistent Volume Claim**, then **Edit YAML**.

Step 4. Modify the YAML file, then click **Create**.

Deploy Red Hat OpenShift Container Platform (Public Cloud)

With the deployment of the on-prem cloud-native environment complete, the first deployment step in the public cloud is to deploy Red Hat OCP since there is no underlying infrastructure that needs to be setup in public cloud. The OCP cluster is deployed and managed from the cloud using the same Red hat Hybrid Cloud Console that was used on-prem. The cluster will also be deployed using the same IPI installation method as on-prem. This will provide Enterprises developers and operators with a consistent environment from on-prem to public cloud.

Note: Red Hat supports multiple customization options, both for deployment and post-deployment. Custom deployments are outside the scope of this document.

Prerequisites

The prerequisites for the Red Hat OCP deployment in AWS are:

- A valid AWS account to deploy resources in AWS.
- AWS Account – IAM User: Create or use an IAM user with administrative privileges (not the root user) to deploy the OCP cluster. The administrative privileges can be removed post-installation.
- AWS Account – Service Limits: You may need to increase the default service limits on the account to deploy an OCP cluster. For example, in the AWS us-east-1 region with six availability zones, the default installation requires 6 Elastic IPs when the default is 5, so you will need to increase this limit. Alternatively, you can modify the default installation configuration file or pick a region with fewer availability zones.
- Installer workstation. The installer will need access to the Internet and to the VPC environment in the AWS account where the cluster is being deployed. Post-deployment, the installer will be used for SSH access and other management functions. The installer will need to be reachable from the Enterprise. The installer is an AWS EC2 instance.
- A valid Red Hat account to login to Red Hat Hybrid Cloud Console. Hybrid Cloud Console is used to centrally deploy and manage the on-prem and public cloud OCP clusters in the Enterprise.
- Internet Access for the Red Hat OCP cluster for accessing the Hybrid Cloud Console to manage the cluster, including subscriptions and telemetry. A newly deployed will also automatically register with Hybrid Cloud Console if it has Internet access and telemetry service enabled (default). OCP cluster also needs access to Quay.io to access packages for initial deployment and for upgrades.
- DNS – A dedicated base DNS domain or public hosted zone must be defined in Route 53 for deploying Red Hat OCP in AWS. Route 53 service in AWS provides DNS resolution for the OCP cluster and for external connections to the cluster. The zone must be authoritative for the domain.
- SSH Access – To debug the installation and for disaster recovery and other post-install activities, the SSH public keys must be provided to the OCP installer to authenticate the access. The key will be passed to

the nodes through the initial configuration (ignition) files. The nodes will add the keys to the **~/.ssh/authorized_keys** list for the **core** user to enable password-less authentication.

Setup Information

Table 12 lists the installation parameters for AWS.

Table 12. Red Hat OCP - Installation Parameters for AWS

Variable	Variable Name	Value	Additional Info
Base DNS Domain	Public Hosted Zone	hc-aws.com	
AWS Region	-	us-east-1	Region to deploy OCP cluster in
Red Hat OCP Cluster Name	-	ocp11.hc-aws.com	

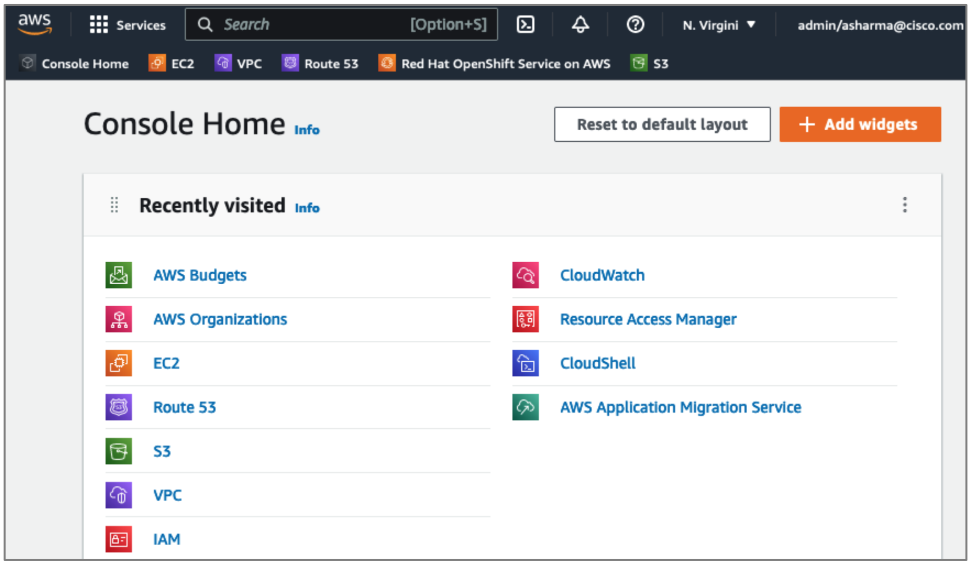
Deployment Steps

This section describes the procedures for the deployment of a Red Hat OCP environment in AWS.

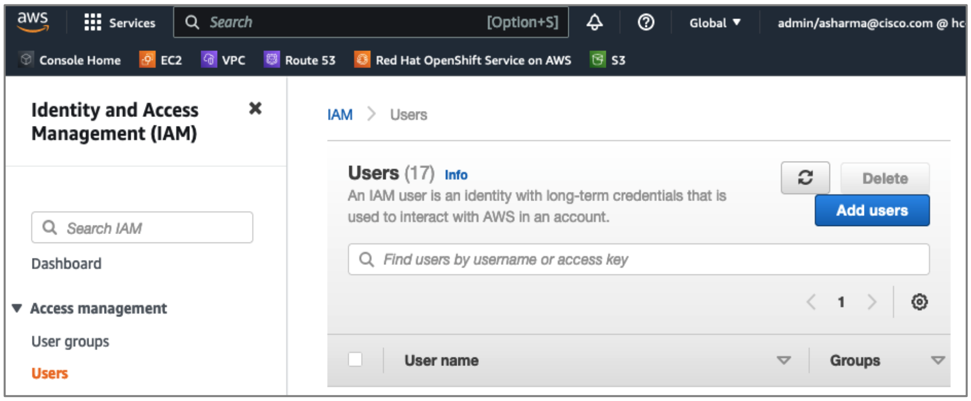
Procedure 1. Create an IAM user with administrator privileges

Note: Every AWS account has a root user account with all privileges, but it is a best-practice not to use this account other than for creating additional users or billing. For this reason, create a second IAM administrative user as outlined below.

Step 1. Use a web browser to navigate to **console.aws.amazon.com**. Login to your AWS account. Click on **IAM**.



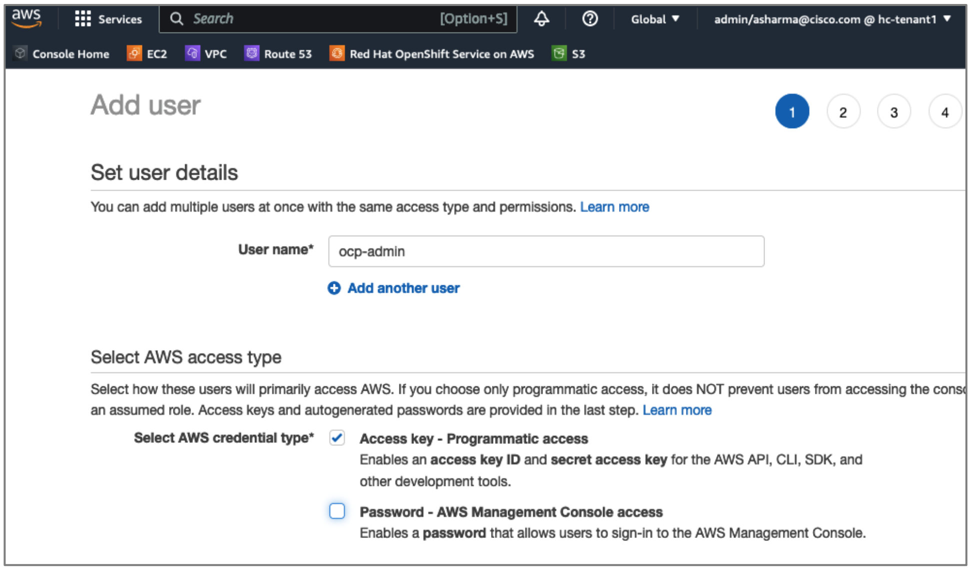
Step 2. From the left navigation pane, select **Users**.



Users (17) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

Step 3. From the right-window pane, click on **Add Users**. Specify a **username** (for example, **ocp-admin**). **Select AWS Credential type** as **Access key – Programmatic access** for OCP.



Add user

Set user details

You can add multiple users at once with the same access type and permissions. Learn more

User name* ocp-admin

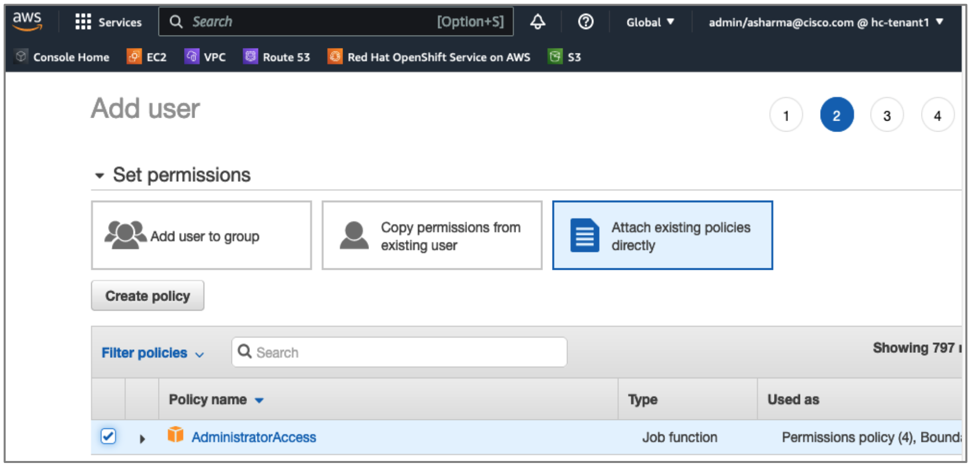
Select AWS access type

Select how these users will primarily access AWS. If you choose only programmatic access, it does NOT prevent users from accessing the cons
an assumed role. Access keys and autogenerated passwords are provided in the last step. Learn more

Select AWS credential type* ☑ **Access key - Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

☐ **Password - AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

Step 4. Scroll down and click on **Next:Permissions**. Select **Attach existing policies directly** and select the checkbox next to policy named: ‘**AdministratorAccess**.’


aws
Services
Search
[Option+S]
Global ▼
admin/asharma@cisco.com @ hc-tenant1 ▼
Console Home
EC2
VPC
Route 53
Red Hat OpenShift Service on AWS
S3
1
2
3
4
Add user to group
Copy permissions from existing user
Attach existing policies directly
Create policy
Filter policies
Search
Showing 797 r

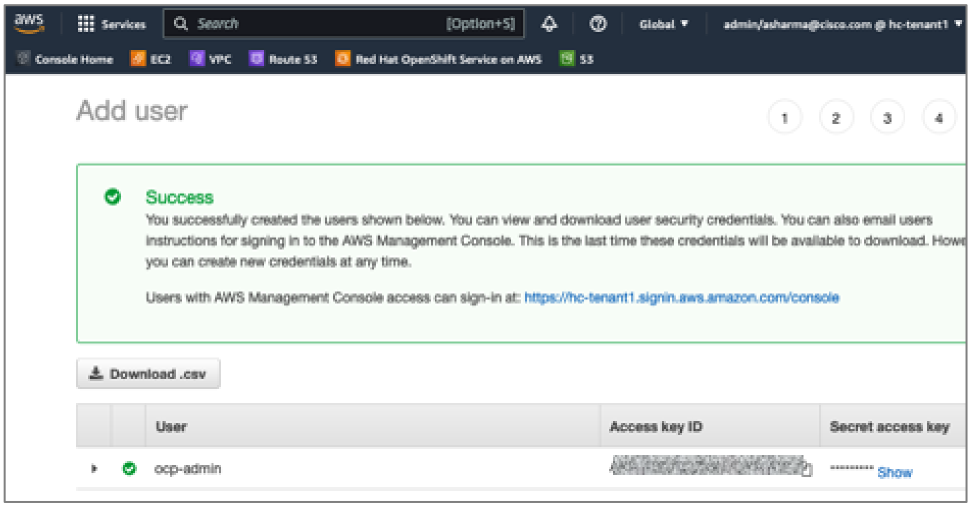
Add user

Set permissions

		Policy name	Type	Used as
☑	▸	AdministratorAccess	Job function	Permissions policy (4), Bound

Step 5. Scroll down and click on **Next:Tags**. Add optional tags. Click on **Next:Review** and click on **Create user.**

Step 6. Save the Access key id and Secret access key for use in a secure place.



Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. Howe you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: https://hc-tenant1.signin.aws.amazon.com/console

		User	Access key ID	Secret access key
▸	✓	ocp-admin	[redacted]	******** Show

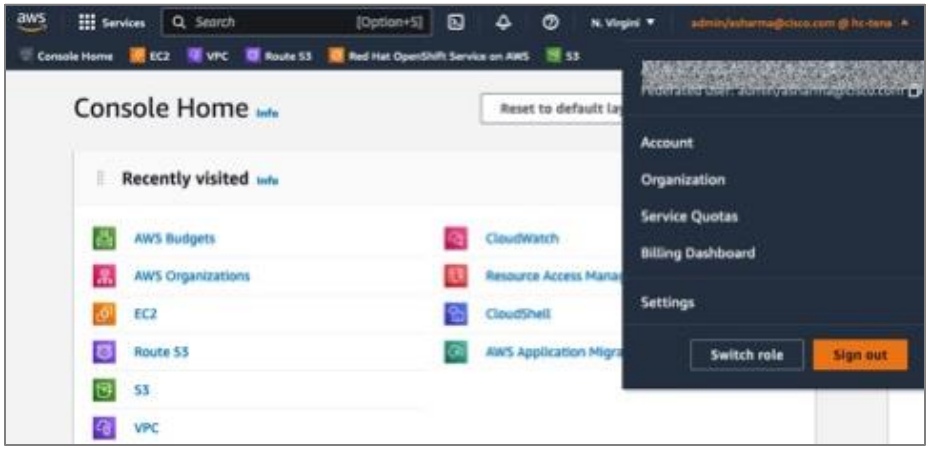
Procedure 2. Increase AWS Account Service Limits

The default limits for AWS services (for example, Instance limits, Elastic IPs) may need to be increased depending on where you're deploying the cluster. For example, in AWS region us-east-1, the Elastic IP limit will need to be increased from 5 to 6 for the default OCP installation to use the six availability zones in us-east-1.

You can follow a similar process to increase other service limits.

Step 1. Use a web browser to navigate to **console.aws.amazon.com.** Login using your AWS account.

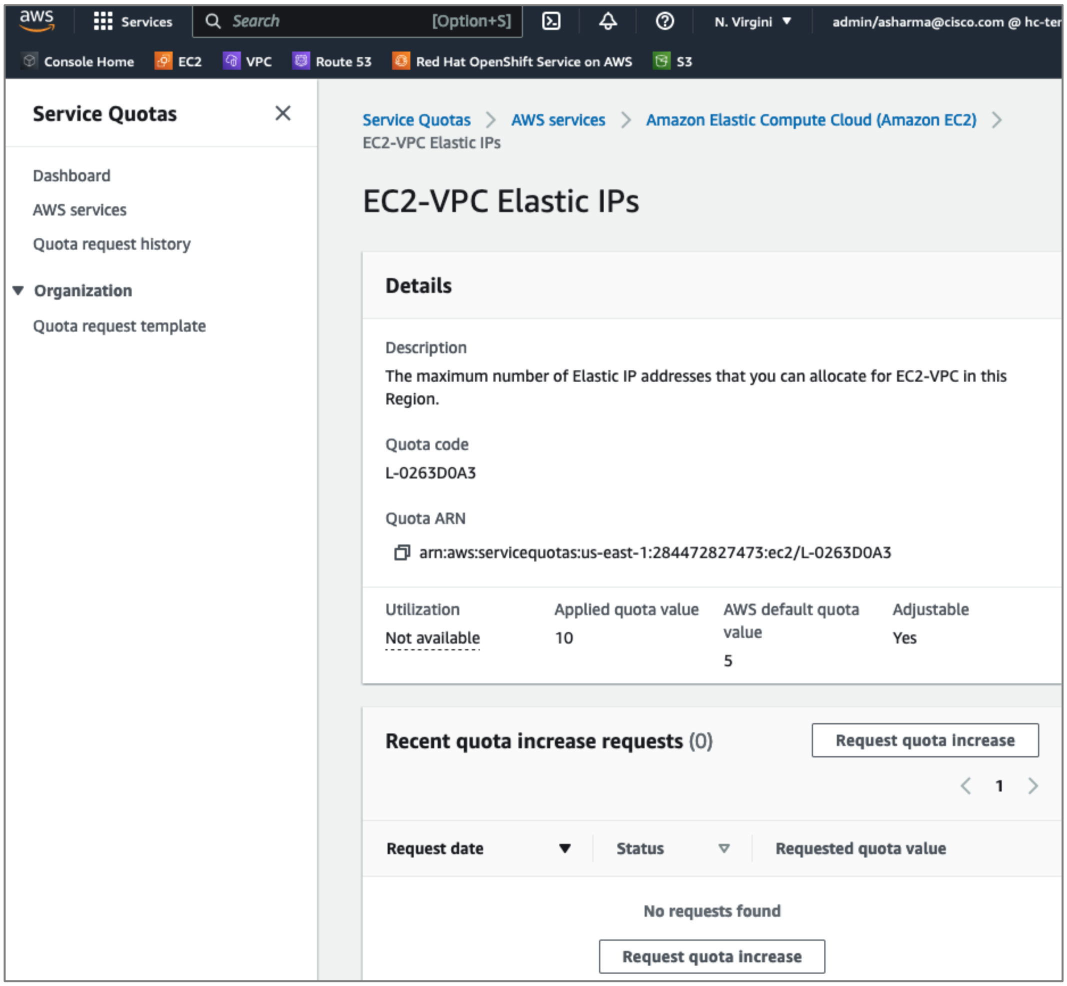
Step 2. Right-click on your account name in the top right-hand corner of the window and select **Service Quotas** from the drop-down list.



Step 3. Click on **Service Quotas** and from the Dashboard in the left navigation pane, select the service category (for example, **Amazon EC2**).

Step 4. Identify the and click on the specific service item (for example, **EC2-VPC Elastic IPs**)

Step 5. Click on the **Request Quota Increase** to provide the necessary information to generate the request.



EC2-VPC Elastic IPs

Details

Description

The maximum number of Elastic IP addresses that you can allocate for EC2-VPC in this Region.

Quota code

L-0263D0A3

Quota ARN

arn:aws:servicequotas:us-east-1:284472827473:ec2/L-0263D0A3

Utilization	Applied quota value	AWS default quota value	Adjustable
Not available	10	5	Yes

Recent quota increase requests (0)

Request date	Status	Requested quota value

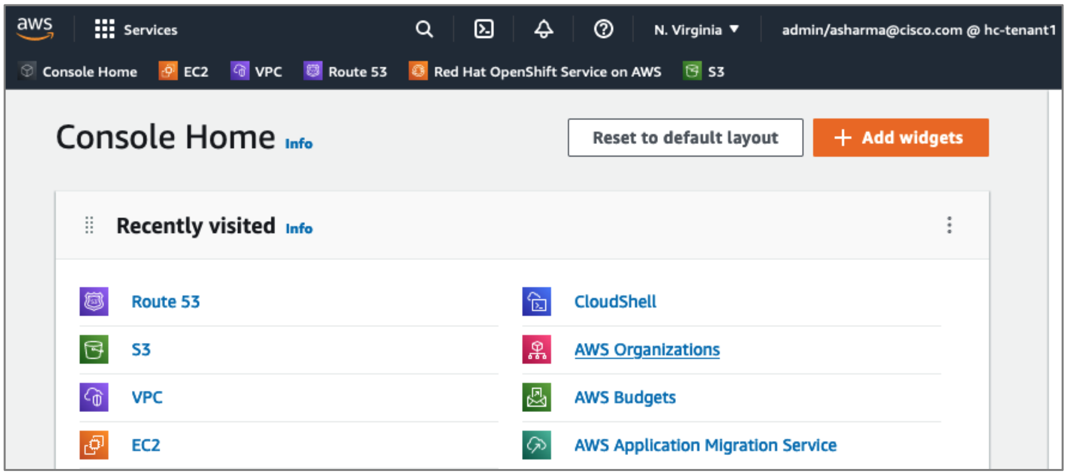
No requests found

Note: You can also request service limits for Elastic IPs by navigating to **EC2 > Limits** from the console home.

Procedure 3. Provision a base DNS domain in AWS Route 53

Red Hat OCP in AWS requires a dedicated public hosted zone defined in Route 53 for use by the cluster and for name resolution when accessing the cluster from external locations.

Step 1. Use a web browser to navigate to **console.aws.amazon.com**. Login using your AWS account. Click on **Route 53**.

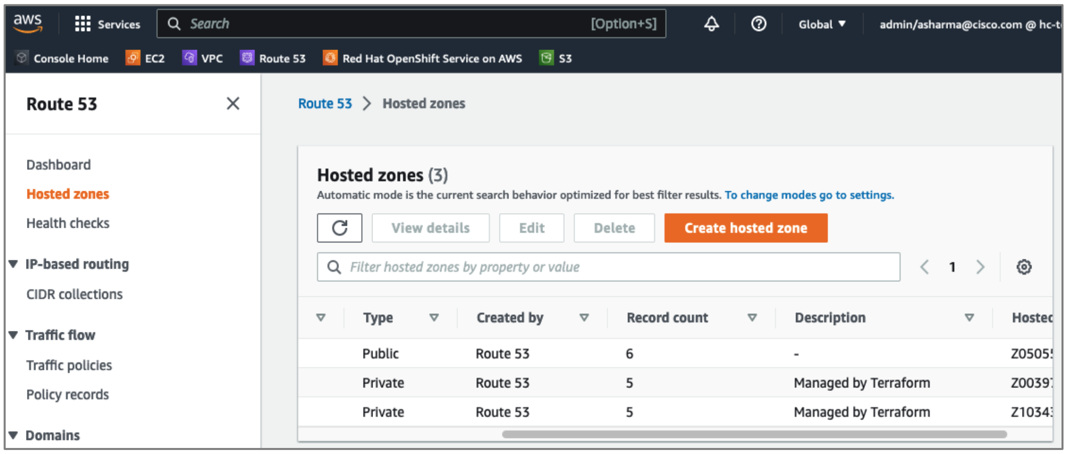


Console Home Info

Recently visited Info

Route 53	CloudShell
S3	AWS Organizations
VPC	AWS Budgets
EC2	AWS Application Migration Service

Step 2. From the navigation pane, select **Hosted Zones > Create hosted zone**.

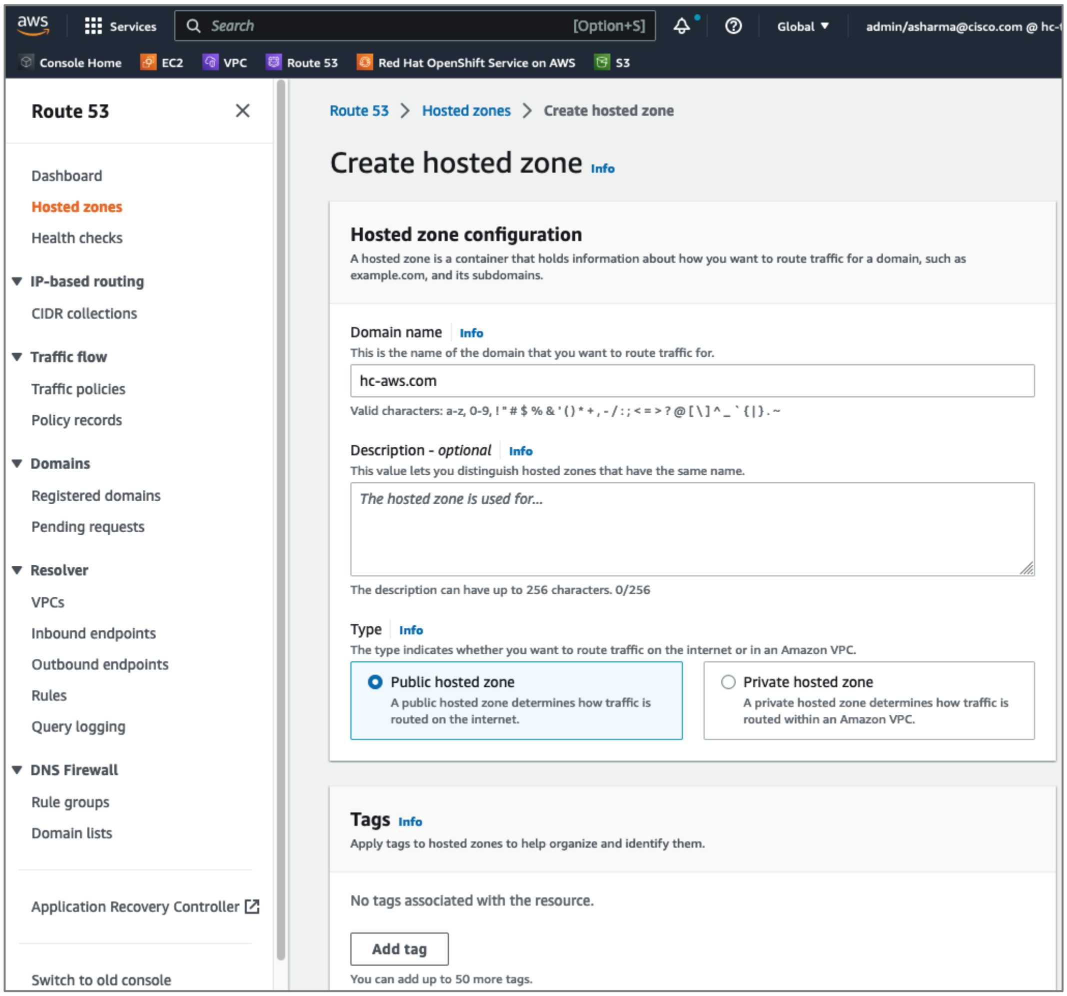


Hosted zones (3)

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

Type	Created by	Record count	Description	Hostec
Public	Route 53	6	-	Z0505!
Private	Route 53	5	Managed by Terraform	Z0039
Private	Route 53	5	Managed by Terraform	Z1034

Step 3. In the **Create Hosted Zone** window, enter a domain name (for example, **hc-aws.com**) that you want to use. For **Type**, use **Public Hosted Zone**. OCP does use a private zone for the cluster, but it is deployed and configured by the installer.



Create hosted zone

Hosted zone configuration

A hosted zone is a container that holds information about how you want to route traffic for a domain, such as example.com, and its subdomains.

Domain name

This is the name of the domain that you want to route traffic for.

hc-aws.com

Valid characters: a-z, 0-9, ! " # $ % & ' ( ) * + , - / : ; < = > ? @ [\ ] ^ _ ` { | } . ~

Description - *optional*

This value lets you distinguish hosted zones that have the same name.

The hosted zone is used for...

The description can have up to 256 characters. 0/256

Type

The type indicates whether you want to route traffic on the internet or in an Amazon VPC.

● Public hosted zone
A public hosted zone determines how traffic is routed on the internet.

○ Private hosted zone
A private hosted zone determines how traffic is routed within an Amazon VPC.

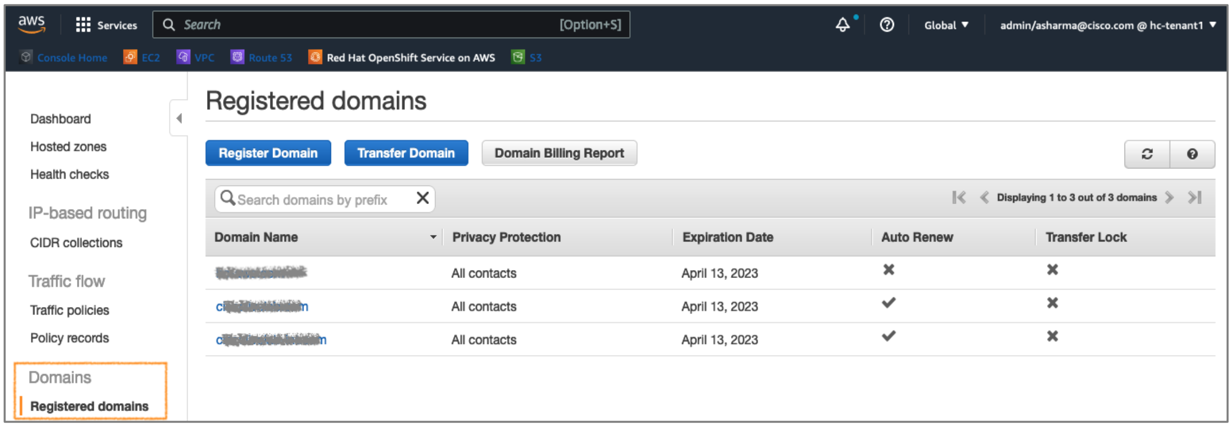
Tags

Apply tags to hosted zones to help organize and identify them.

No tags associated with the resource.

You can add up to 50 more tags.

Step 4. Scroll down to the bottom and click on **Create hosted zone**. A new hosted zone will now be created to enable traffic to be routed to your domain.



Registered domains

Displaying 1 to 3 out of 3 domains

Domain Name	Privacy Protection	Expiration Date	Auto Renew	Transfer Lock
[redacted]	All contacts	April 13, 2023	✗	✗
[redacted]	All contacts	April 13, 2023	✓	✗
[redacted]	All contacts	April 13, 2023	✓	✗

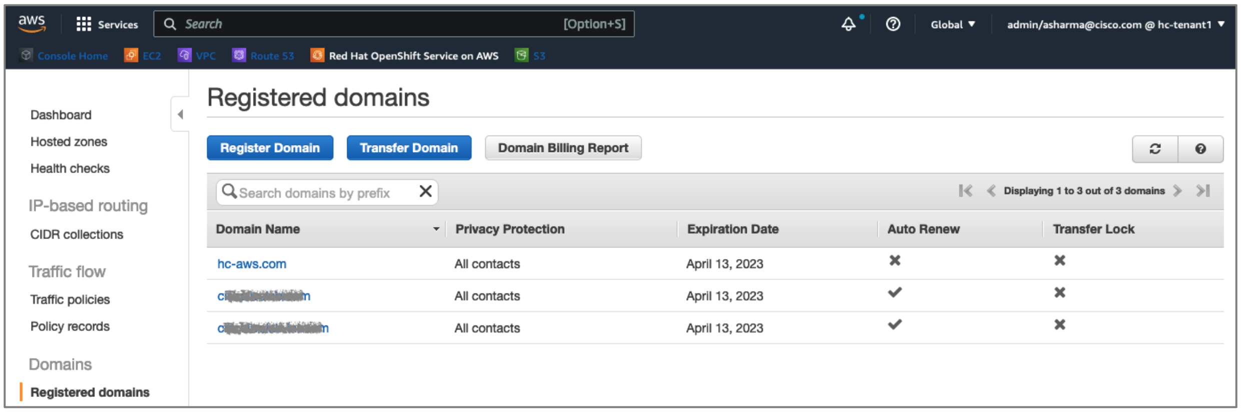
Step 5. From the navigation pane, under **Domains**, select **Registered domains** and click on **Register Domain**.

Step 6. In the **Domain Search** window, choose a domain name (for example, **hc-aws**) and extension (**.com**) from the drop-down list. Click **Check** to verify it is available.

Step 7. If it is available, you can click **Add to cart** and click **Continue** to proceed further.

Step 8. In the **Contact Details** window, provide contact information for your domain. Click **Continue**.

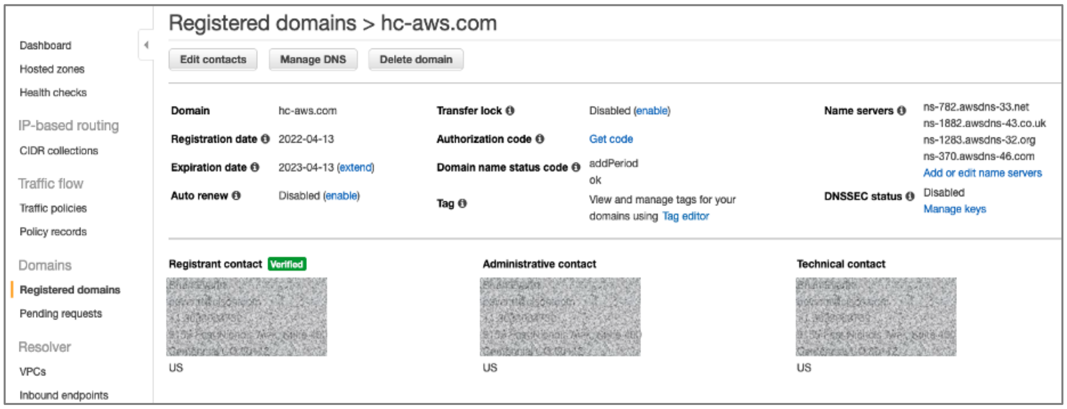
Step 9. In the **Verify and Purchase** window, verify the information provided, agree to the terms and conditions, and click on **Complete Order** to register the domain. You will receive notification from AWS when the domain is available – it may take a few minutes to complete.



Registered domains

Domain Name	Privacy Protection	Expiration Date	Auto Renew	Transfer Lock
hc-aws.com	All contacts	April 13, 2023	✗	✗
[redacted]	All contacts	April 13, 2023	✓	✗
[redacted]	All contacts	April 13, 2023	✓	✗

Step 10. You can view details of your domain by clicking on the newly created domain from the list of **Registered domains**.



Registered domains > hc-aws.com

Domain: hc-aws.com
Registration date: 2022-04-13
Expiration date: 2023-04-13 (extend)
Auto renew: Disabled (enable)

Transfer lock: Disabled (enable)
Authorization code: Get code
Domain name status code: addPeriod
ok
Tag: View and manage tags for your domains using Tag editor

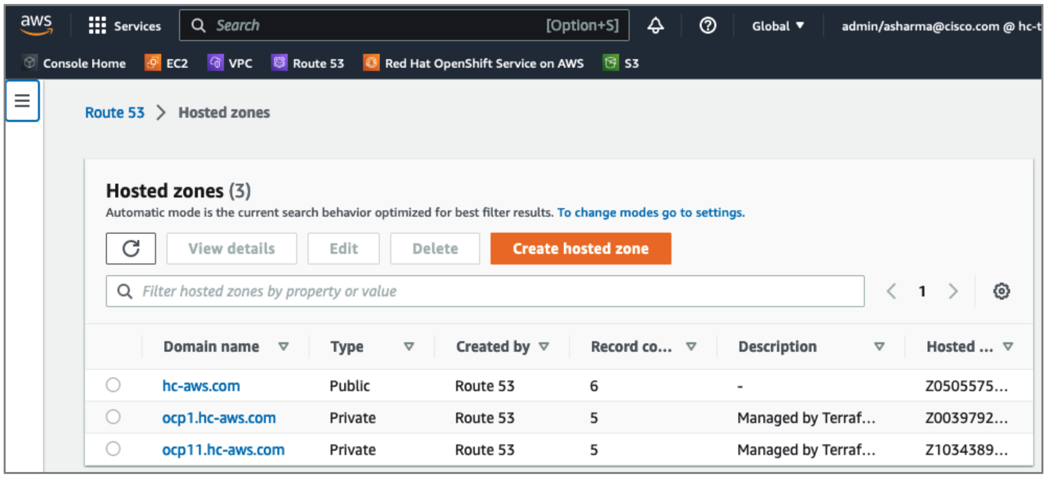
Name servers: ns-782.awsdns-33.net
ns-1882.awsdns-43.co.uk
ns-1283.awsdns-32.org
ns-370.awsdns-46.com
Add or edit name servers
DNSSEC status: Disabled
Manage keys

Registrant contact Verified
[redacted]
US

Administrative contact
[redacted]
US

Technical contact
[redacted]
US

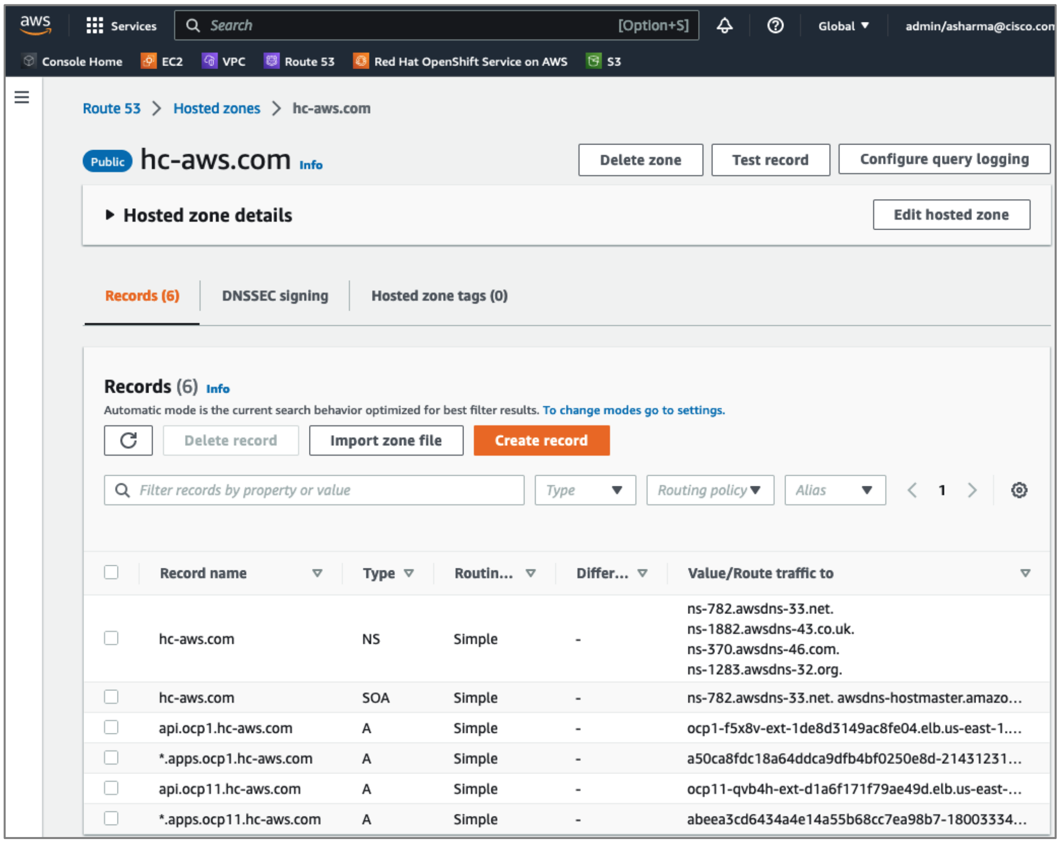
Step 11. Once the cluster is deployed, you can go back to **Route 53 > Hosted zones** to see the Red Hat OCP installer provisioned records in the public and private zones, and routing info for the DNS records.



Hosted zones (3)

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

	Domain name	Type	Created by	Record co...	Description	Hosted ...
○	hc-aws.com	Public	Route 53	6	-	Z0505575...
○	ocp1.hc-aws.com	Private	Route 53	5	Managed by Terraf...	Z0039792...
○	ocp11.hc-aws.com	Private	Route 53	5	Managed by Terraf...	Z1034389...



Route 53 > Hosted zones > hc-aws.com

Public hc-aws.com Info

▸ Hosted zone details

Records (6)

Records (6) Info

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

□	Record name	Type	Routin...	Differ...	Value/Route traffic to
□	hc-aws.com	NS	Simple	-	ns-782.awsdns-33.net. ns-1882.awsdns-43.co.uk. ns-370.awsdns-46.com. ns-1283.awsdns-32.org.
□	hc-aws.com	SOA	Simple	-	ns-782.awsdns-33.net. awsdns-hostmaster.amazo...
□	api.ocp1.hc-aws.com	A	Simple	-	ocp1-f5x8v-ext-1de8d3149ac8fe04.elb.us-east-1....
□	*.apps.ocp1.hc-aws.com	A	Simple	-	a50ca8fdc18a64ddca9dfb4bf0250e8d-21431231...
□	api.ocp11.hc-aws.com	A	Simple	-	ocp11-qvb4h-ext-d1a6f171f79ae49d.elb.us-east-...
□	*.apps.ocp11.hc-aws.com	A	Simple	-	abeea3cd6434a4e14a55b68cc7ea98b7-18003334...

Procedure 4. Generate a key pair for SSH access to Red Hat OCP cluster

The commands you will need for this are:

Commands
`ssh-keygen -t rsa -N '' -f <path>/<file_name>`
`eval "$(ssh-agent -s)"`
`ssh-add <path>/<file_name>`

Step 1. On the installer workstation running a Linux operating system, use the following command. You can generate the key using **rsa** or **edcsa** algorithm.

```
ssh-keygen -t rsa -N '' -f ~/.ssh/id_rsa
```

Step 2. Add the SSH private key identity to the SSH agent for your local user. If the **`ssh-agent`** process is not already running for your local user, start it as a background task:

```
eval "$(ssh-agent -s)"
```

Step 3. Add your SSH private key to the **`ssh-agent`** using the command:

```
ssh-add ~/.ssh/id_rsa
```

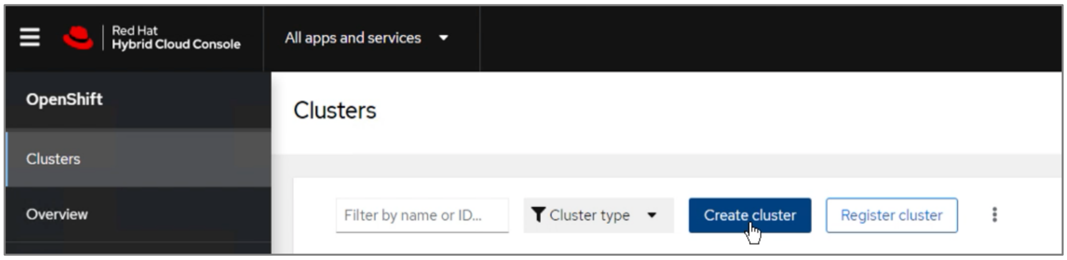
The above key is provided as input to the installer. Installer will add it to the ignition files that are used to do the initial configuration of the OCP nodes. When the OCP cluster is deployed, you will be able to SSH into the cluster as user '**core**' without the need for a password.

Procedure 5. Download the Red Hat OCP installer and other tools from Red Hat Hybrid Cloud Console

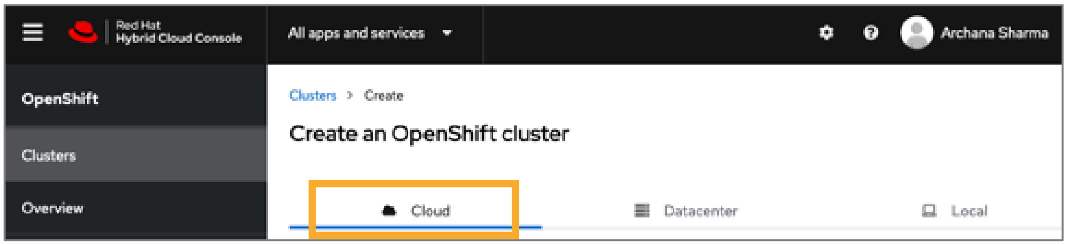
Step 1. Use a web browser to navigate to Red Hat Hybrid Cloud Console at console.redhat.com. Login to your Red Hat account.

Step 2. From the left navigation pane, select and click on **OpenShift**.

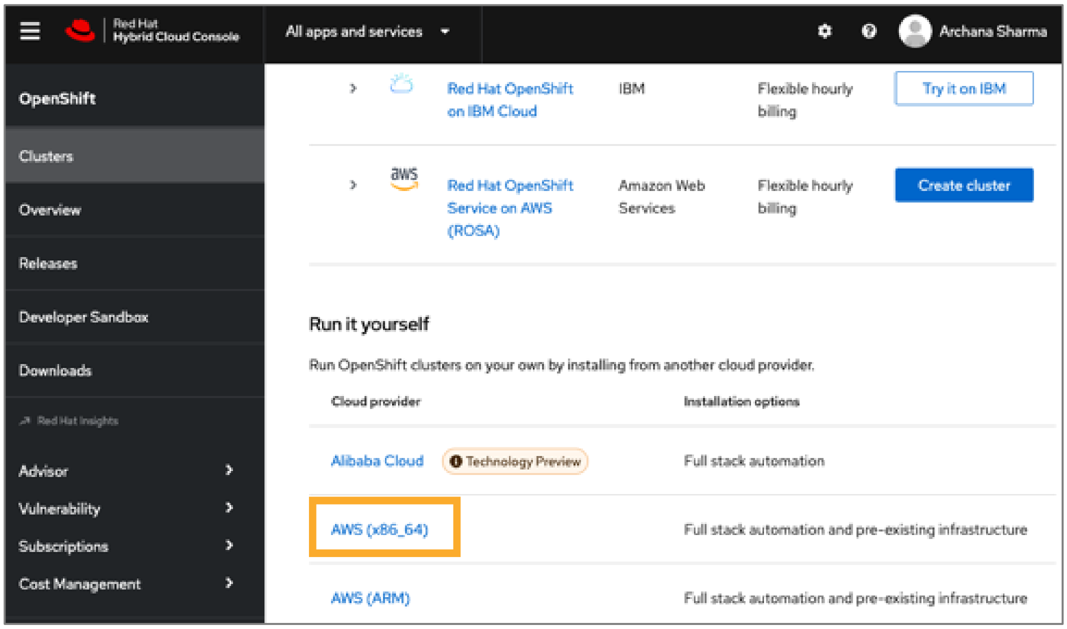
Step 3. Navigate to **Clusters** and click on **Create Cluster**.



Step 4. Navigate to the **Cloud** tab



Step 5. Scroll down and click on AWS for the infrastructure provider.



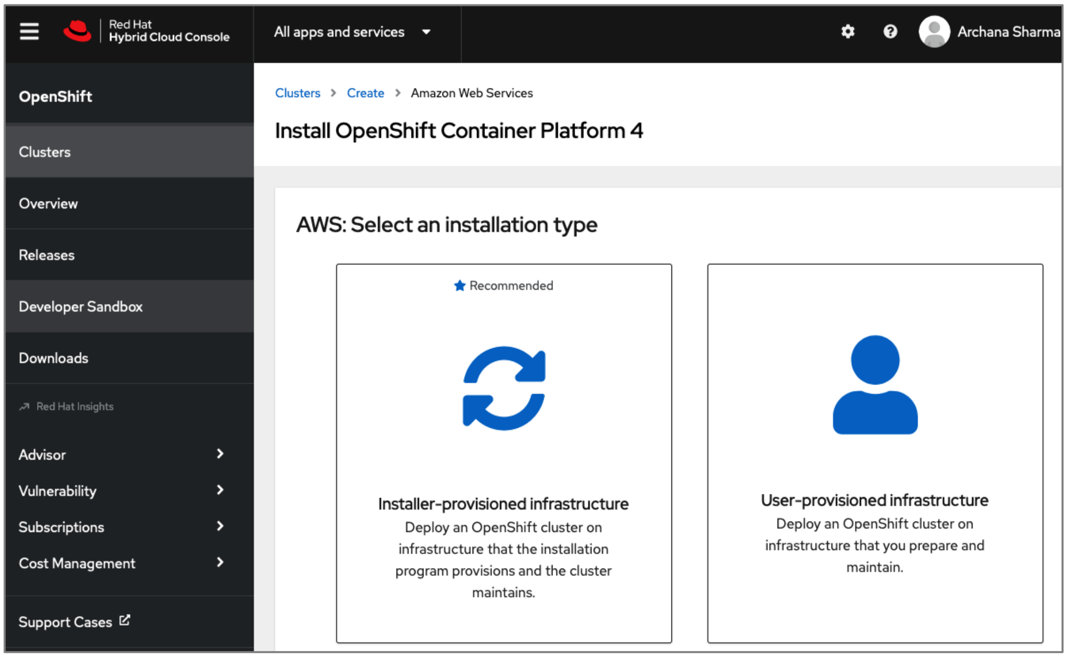
Red Hat OpenShift on IBM Cloud	IBM	Flexible hourly billing	Try it on IBM
Red Hat OpenShift Service on AWS (ROSA)	Amazon Web Services	Flexible hourly billing	Create cluster

Run it yourself

Run OpenShift clusters on your own by installing from another cloud provider.

Cloud provider	Installation options
Alibaba Cloud (Technology Preview)	Full stack automation
AWS (x86_64)	Full stack automation and pre-existing infrastructure
AWS (ARM)	Full stack automation and pre-existing infrastructure

Step 6. Select the Installer-Provisioned Infrastructure (IPI) method.



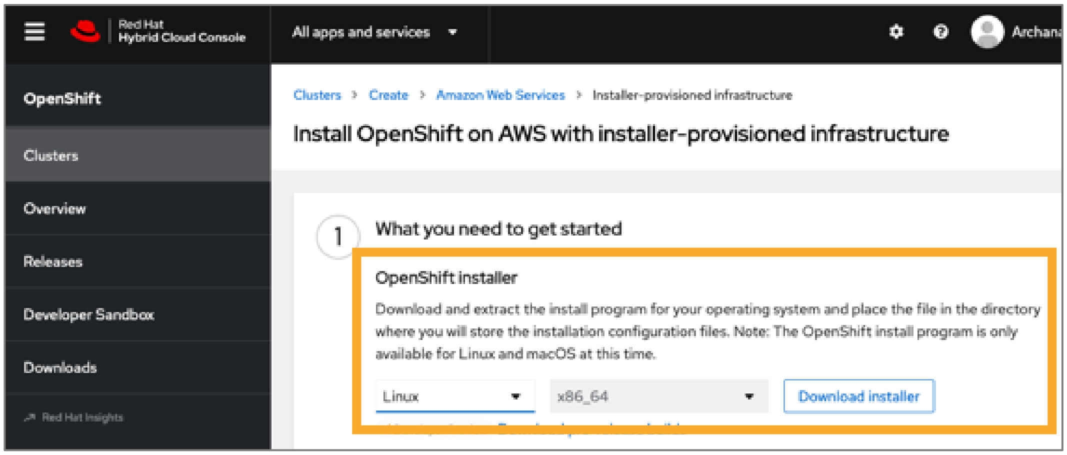
Install OpenShift Container Platform 4

AWS: Select an installation type

Installer-provisioned infrastructure
Deploy an OpenShift cluster on infrastructure that the installation program provisions and the cluster maintains.

User-provisioned infrastructure
Deploy an OpenShift cluster on infrastructure that you prepare and maintain.

Step 7. From the AWS infrastructure page, download the installation program for the operating system running on the EC2 installer workstation.



Install OpenShift on AWS with installer-provisioned infrastructure

What you need to get started

OpenShift installer

Download and extract the install program for your operating system and place the file in the directory where you will store the installation configuration files. Note: The OpenShift install program is only available for Linux and macOS at this time.

Linux

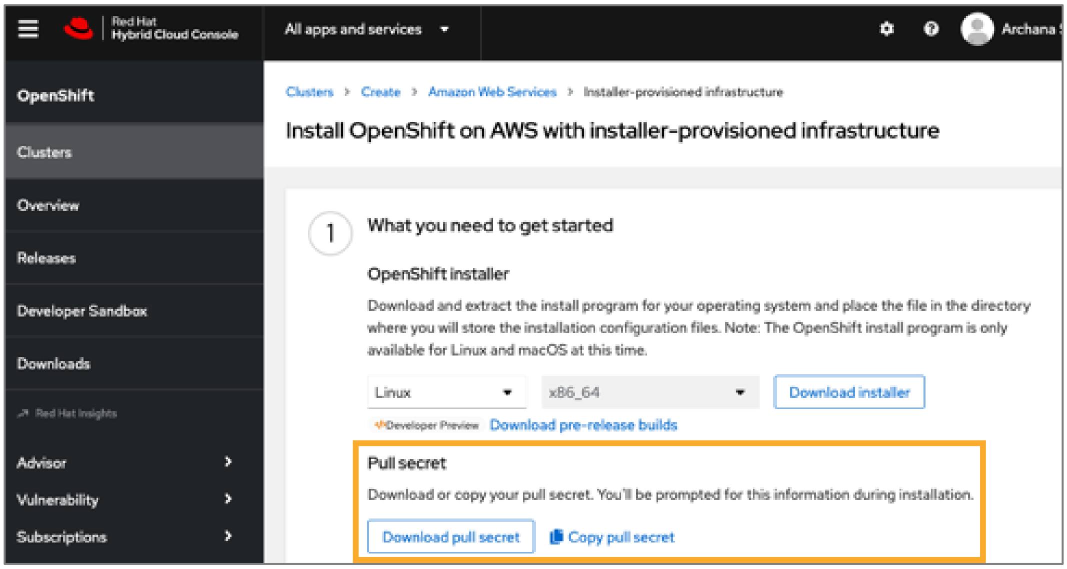
x86_64

Step 8. On the installer workstation, create a directory for the cluster (in this case, **ocp11**) and move the installation package to this directory.

Step 9. Extract the installation package.

```
tar -xvf openshift-install-linux.tar.gz
```

Step 10. Download the pull-secret and save it in a file in the same directory. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components. Also copy the pull secret.



Install OpenShift on AWS with installer-provisioned infrastructure

1 What you need to get started

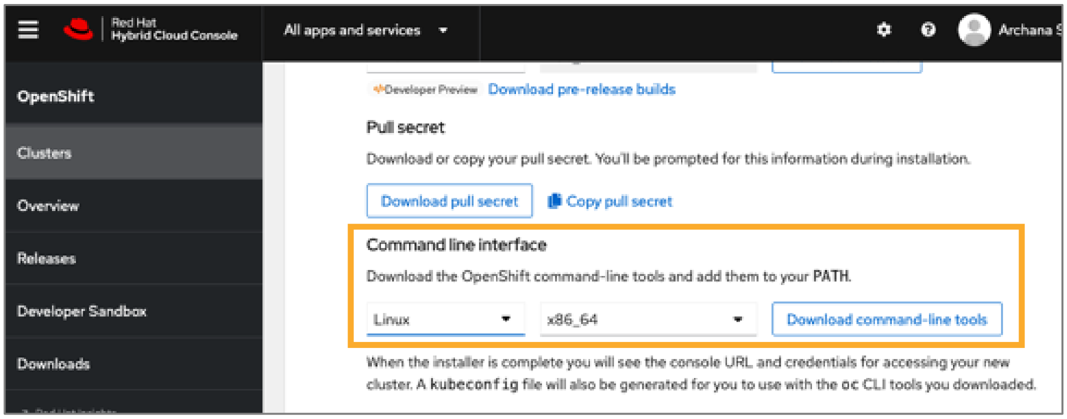
OpenShift installer

Download and extract the install program for your operating system and place the file in the directory where you will store the installation configuration files. Note: The OpenShift install program is only available for Linux and macOS at this time.

Pull secret

Download or copy your pull secret. You'll be prompted for this information during installation.

Step 11. Download the OpenShift CLI tools to the same directory.



Pull secret

Download or copy your pull secret. You'll be prompted for this information during installation.

Command line interface

Download the OpenShift command-line tools and add them to your PATH.

When the installer is complete you will see the console URL and credentials for accessing your new cluster. A kubeconfig file will also be generated for you to use with the oc CLI tools you downloaded.

Procedure 6. Install Red Hat OpenShift Container Platform in AWS

Step 1. Execute the following command to run the installer:

```
./openshift-install create cluster --dir=<installation_directory> --log-level=info
```

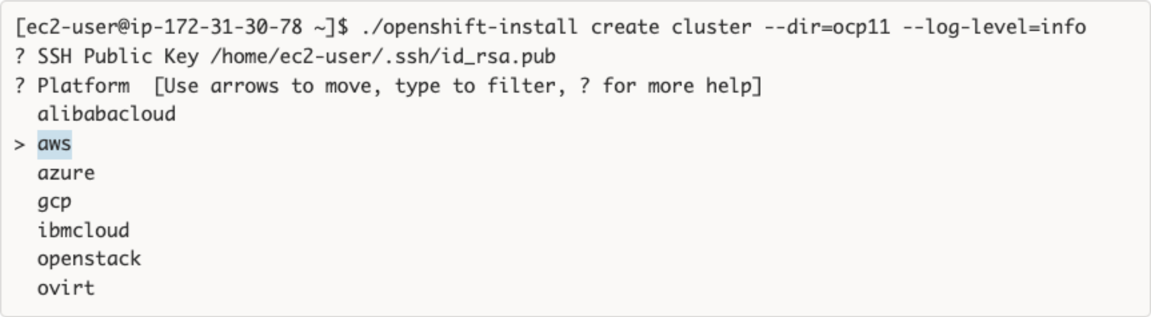
Step 2. From the installer workstation, run the following command to start the installation process

```
./openshift-install create cluster --dir=ocp11 --log-level=info
```

Step 3. Select the SSH public key to pass to the cluster nodes through the installation configuration (ignition) file.

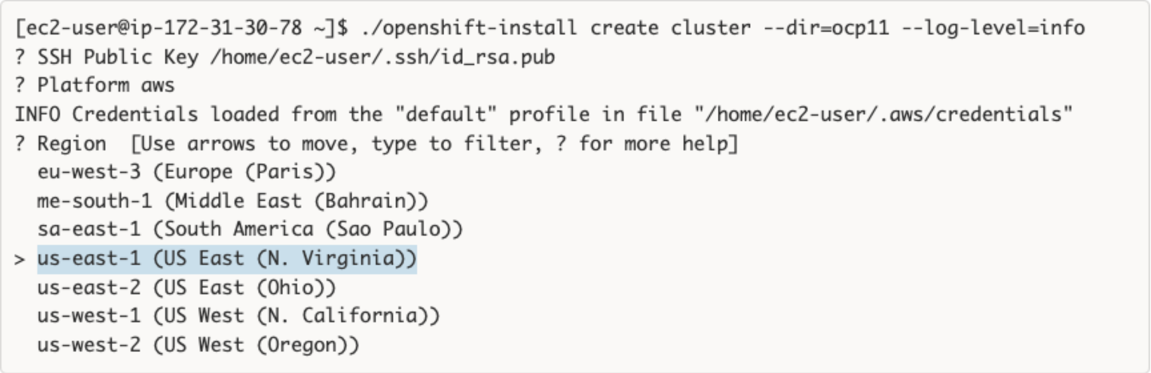
```
[ec2-user@ip-172-31-30-78 ~]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key  [Use arrows to move, type to filter, ? for more help]
> /home/ec2-user/.ssh/id_rsa.pub
  <none>
```

Step 4. Select infrastructure environment.



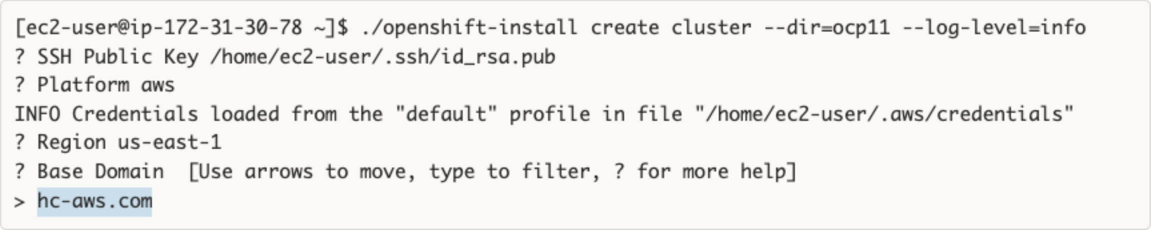
```
[ec2-user@ip-172-31-30-78 ~]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/ec2-user/.ssh/id_rsa.pub
? Platform  [Use arrows to move, type to filter, ? for more help]
  alibabacloud
> aws
  azure
  gcp
  ibmcloud
  openstack
  ovirt
```

Step 5. Select the AWS region to deploy the cluster in.



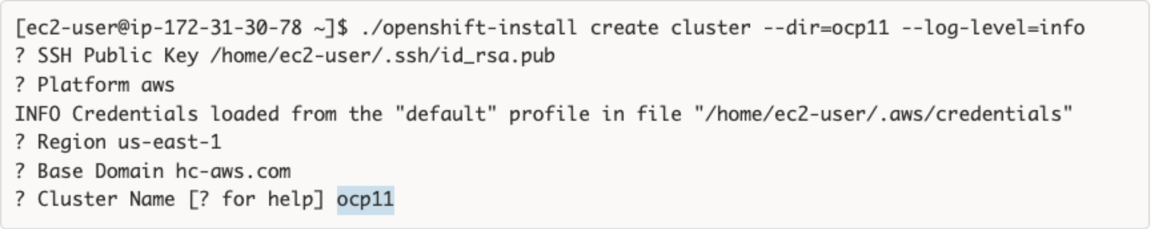
```
[ec2-user@ip-172-31-30-78 ~]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/ec2-user/.ssh/id_rsa.pub
? Platform aws
INFO Credentials loaded from the "default" profile in file "/home/ec2-user/.aws/credentials"
? Region  [Use arrows to move, type to filter, ? for more help]
  eu-west-3 (Europe (Paris))
  me-south-1 (Middle East (Bahrain))
  sa-east-1 (South America (Sao Paulo))
> us-east-1 (US East (N. Virginia))
  us-east-2 (US East (Ohio))
  us-west-1 (US West (N. California))
  us-west-2 (US West (Oregon))
```

Step 6. Select the base domain or the Route 53 public hosted zone that was created for this cluster.



```
[ec2-user@ip-172-31-30-78 ~]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/ec2-user/.ssh/id_rsa.pub
? Platform aws
INFO Credentials loaded from the "default" profile in file "/home/ec2-user/.aws/credentials"
? Region us-east-1
? Base Domain  [Use arrows to move, type to filter, ? for more help]
> hc-aws.com
```

Step 7. Select a name for the OCP cluster.



```
[ec2-user@ip-172-31-30-78 ~]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/ec2-user/.ssh/id_rsa.pub
? Platform aws
INFO Credentials loaded from the "default" profile in file "/home/ec2-user/.aws/credentials"
? Region us-east-1
? Base Domain hc-aws.com
? Cluster Name [? for help] ocp11
```

Step 8. Paste the pull secret copied/downloaded from the Hybrid Cloud Console.

```
[ec2-user@ip-172-31-30-78 ~]$ ./openshift-install create cluster --dir=ocp11 --log-level=info
? SSH Public Key /home/ec2-user/.ssh/id_rsa.pub
? Platform aws
INFO Credentials loaded from the "default" profile in file "/home/ec2-user/.aws/credentials"
? Region us-east-1
? Base Domain hc-aws.com
? Cluster Name ocp11
? Pull Secret [? for help]
*******************************************************************************************************
******************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************************************************************************
*******************************************
```

Step 9. Once the pull-secret is provided, the installer will start the OCP deployment. It will take around 45min. Once the OCP cluster is deployed, there are a few post-install tasks that must be completed.

Note: Do **not** remove the installer or the files created by the installer. These will be necessary to delete the cluster.

Procedure 7. Post-install task – Verify access to new cluster

When the cluster installation finished, complete the following post-install tasks to verify that you can access the new cluster using the command line tools and the web console.

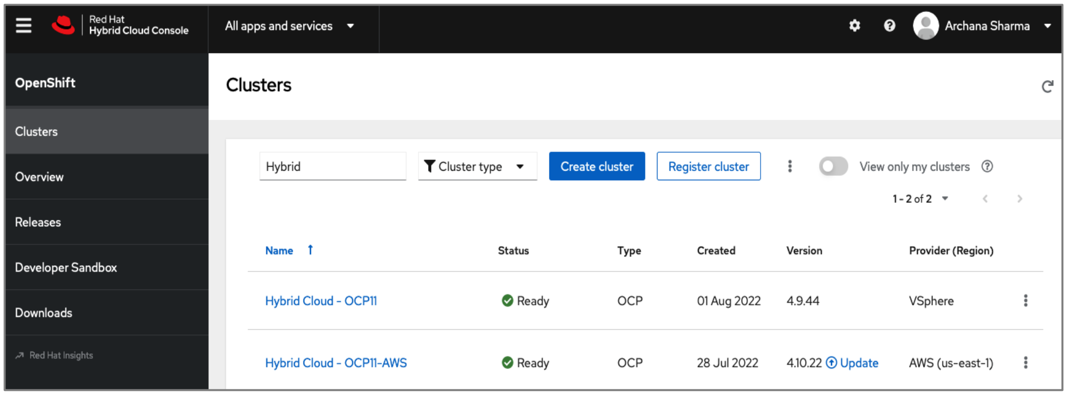
Step 1. To verify access via the command line, execute the following commands from the workstation:

```
export KUBECONFIG=/root/openshift/<ocp_directory>/auth/kubeconfig
oc whoami
oc get nodes -o wide
```

```
...
WARNING failed to find default instance type: no instance type found for the zone constraint
WARNING failed to find default instance type: no instance type found for the zone constraint
INFO Creating infrastructure resources...
INFO Waiting up to 20m0s (until 7:15PM) for the Kubernetes API at https://api.ocp11.hc-
aws.com:6443...
INFO API v1.23.5+3afdacb up
INFO Waiting up to 30m0s (until 7:28PM) for bootstrapping to complete...
INFO Destroying the bootstrap resources...
INFO Waiting up to 40m0s (until 7:48PM) for the cluster at https://api.ocp11.hc-aws.com:6443 to
initialize...
INFO Waiting up to 10m0s (until 7:25PM) for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/ec2-
user/ocp11/auth/kubeconfig' INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.ocp11.hc-aws.com
INFO Login to the console with user: "kubeadmin", and password: "XXXXXXXXXXXXXXXX"
INFO Time elapsed: 28m38s
[ec2-user@ip-172-31-30-78 ~]$ export KUBECONFIG=/root/openshift/ocp11/auth/kubeconfig
[ec2-user@ip-172-31-30-78 ~]$
[ec2-user@ip-172-31-30-78 ~]$ oc whoami
system:admin
[ec2-user@ip-172-31-30-78 ~]$

[ec2-user@ip-172-31-30-78 ~]$ oc get nodes
NAME                          STATUS   ROLES    AGE   VERSION
ip-10-0-129-37.ec2.internal   Ready    worker   31m   v1.23.5+3afdacb
ip-10-0-136-144.ec2.internal  Ready    master   39m   v1.23.5+3afdacb
ip-10-0-148-146.ec2.internal  Ready    worker   31m   v1.23.5+3afdacb
ip-10-0-149-45.ec2.internal   Ready    master   40m   v1.23.5+3afdacb
ip-10-0-161-196.ec2.internal  Ready    master   39m   v1.23.5+3afdacb
ip-10-0-175-170.ec2.internal  Ready    worker   31m   v1.23.5+3afdacb
[ec2-user@ip-172-31-30-78 ~]$
```

Step 2. To verify access via web console, open a web browser and navigate to the OpenShift Hybrid Cloud Console. Login to your Red Hat account. Navigate to **OpenShift > Clusters** and find the newly created OCP cluster. Select and click on the cluster name.

Note: If your cluster has access to the Internet, the Telemetry service will automatically register with the Hybrid Cloud Console, and you will see it in the cluster list.

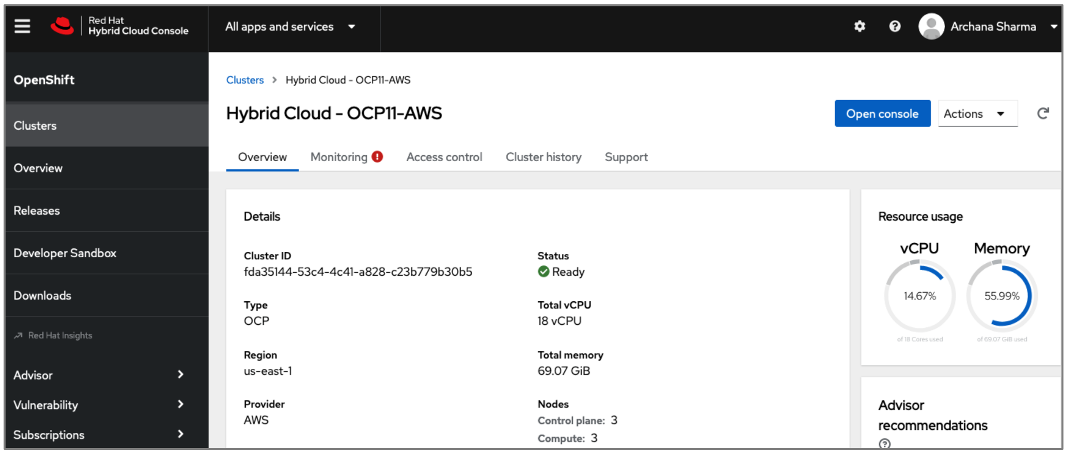


Clusters

Hybrid

Name	Status	Type	Created	Version	Provider (Region)
Hybrid Cloud - OCP11	✓ Ready	OCP	01 Aug 2022	4.9.44	VSphere
Hybrid Cloud - OCP11-AWS	✓ Ready	OCP	28 Jul 2022	4.10.22 Update	AWS (us-east-1)

Step 3. Click on **Open console** from the top-right corner and login to the cluster console as **kubeadm** user.



Hybrid Cloud - OCP11-AWS

Details

Cluster ID
fda35144-53c4-4c41-a828-c23b779b30b5

Status
Ready

Type
OCP

Total vCPU
18 vCPU

Region
us-east-1

Total memory
69.07 GiB

Provider
AWS

Nodes
Control plane: 3
Compute: 3

Step 4. Verify status and health of cluster from OCP cluster console.

Procedure 8. Post-install task: Set up a Bastion Node for SSH Access

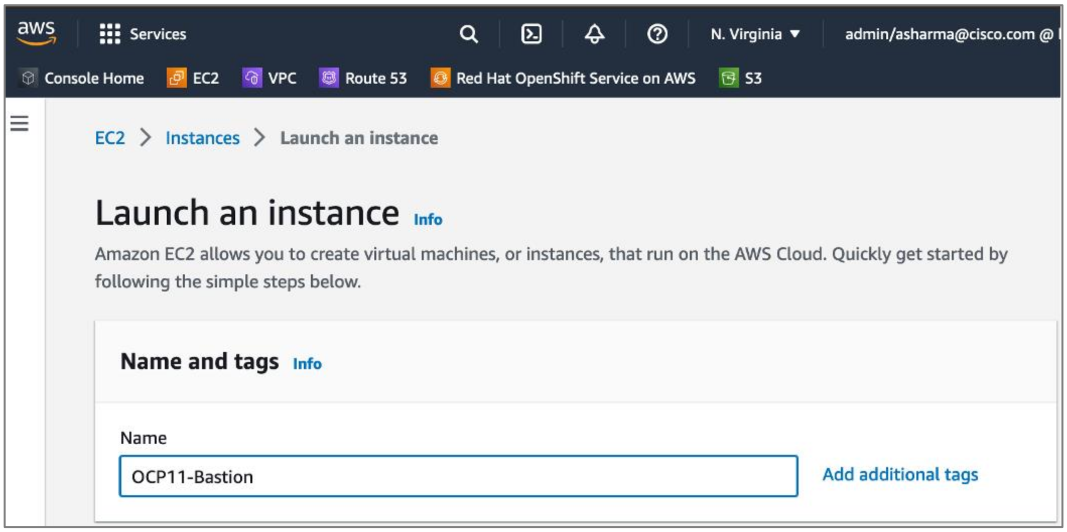
The OCP cluster nodes on EC2 instances in AWS are deployed in a private subnet. To access these nodes using SSH would require a bastion node or jump box in the same VPC as the cluster nodes but with a public IP so that you can access it from outside AWS.

This procedure enables you to deploy an EC2 instance as a bastion node for SSH access.

Step 1. Use a web browser to navigate to **console.aws.amazon.com**. Login to your AWS account.

Step 2. Navigate to **EC2 > Instances**.

Step 3. Click **Launch Instances**. Provide a **Name**.


aws
Services
N. Virginia
admin/asharma@cisco.com @
Console Home
EC2
VPC
Route 53
Red Hat OpenShift Service on AWS
S3
EC2 > Instances > Launch an instance
Info
Info
Add additional tags

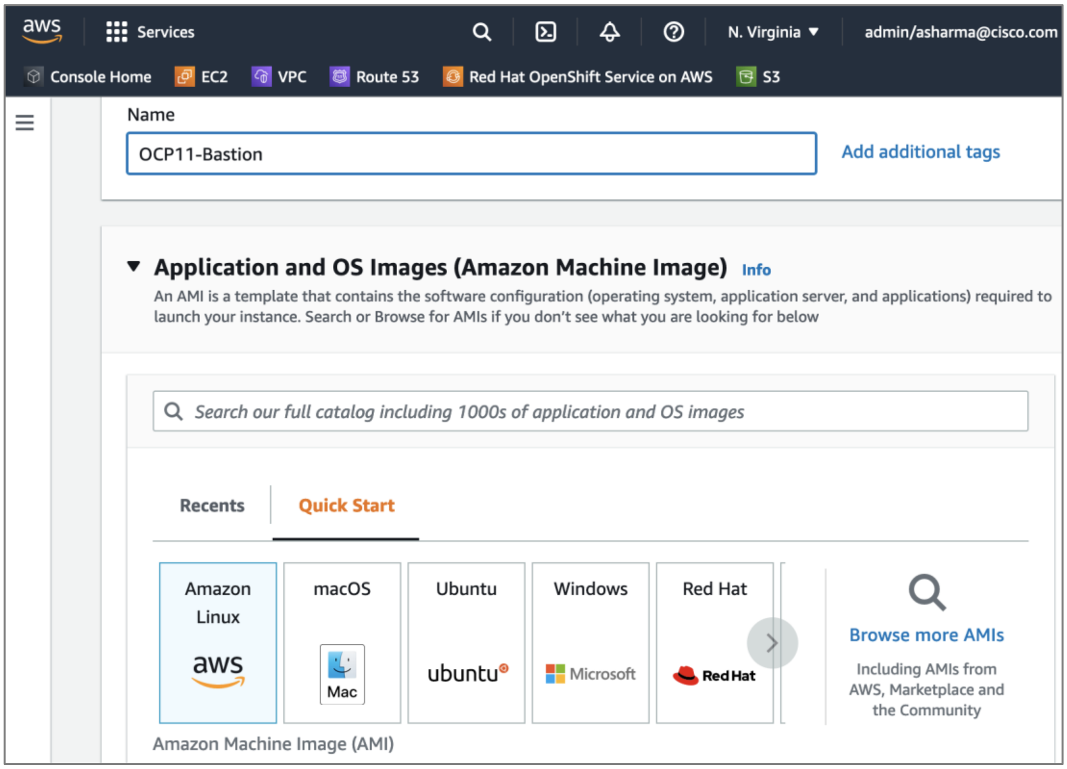
Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name: OCP11-Bastion

Step 4. Scroll down and select an **OS image** from the options.



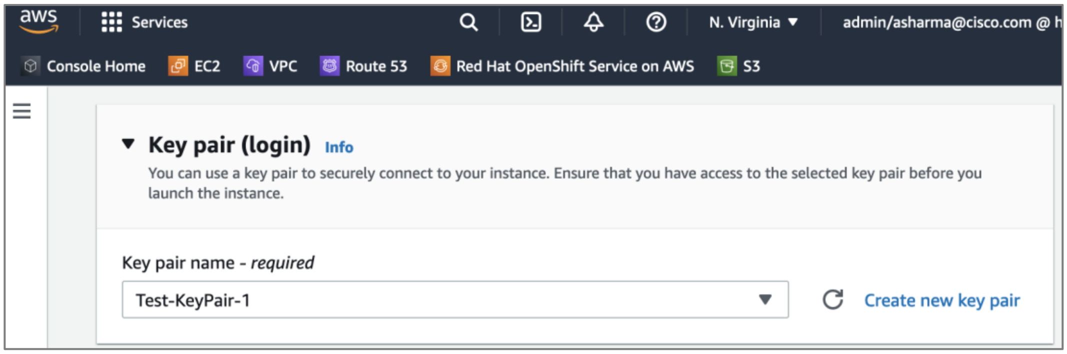
Name: OCP11-Bastion

Application and OS Images (Amazon Machine Image)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Amazon Machine Image (AMI)

Step 5. Scroll down and select a **Key pair** from the drop-down list or create a new pair.



Key pair (login)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

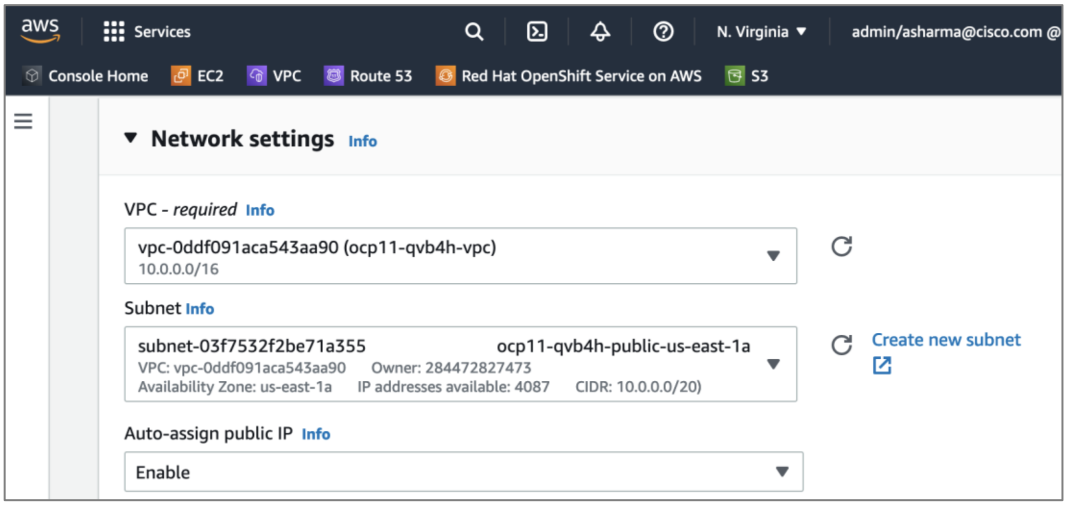
Key pair name - *required*: Test-KeyPair-1

Step 6. Scroll down to **Network settings**. Click **Edit**.

Step 7. For the **VPC**, select the OCP cluster VPC from the drop-down list.

Step 8. For the **Subnet**, select a Public subnet in the deployed by the OCP installer.

Step 9. For the **Auto-Assign public IP**, select **Enable** from the drop-down list.



Network settings Info

VPC - *required* Info

vpc-0ddf091aca543aa90 (ocp11-qvb4h-vpc)
10.0.0.0/16

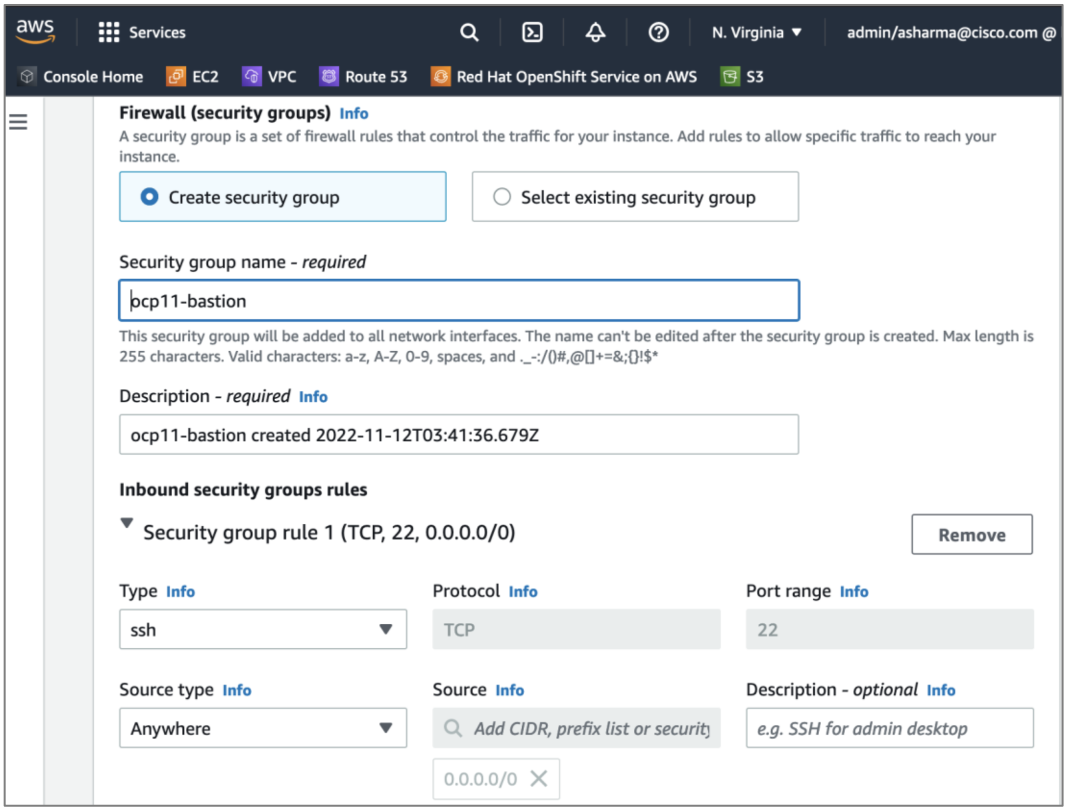
Subnet Info

subnet-03f7532f2be71a355 ocp11-qvb4h-public-us-east-1a
VPC: vpc-0ddf091aca543aa90 Owner: 284472827473
Availability Zone: us-east-1a IP addresses available: 4087 CIDR: 10.0.0.0/20)

Auto-assign public IP Info

Enable

Step 10. Scroll down to **Firewall (security groups)**. Specify a **Security group name** for the new security group. Update the **description** as needed.



Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

● Create security group ○ Select existing security group

Security group name - *required*

ocp11-bastion

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!$*

Description - *required* Info

ocp11-bastion created 2022-11-12T03:41:36.679Z

Inbound security groups rules

Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type Info: ssh

Protocol Info: TCP

Port range Info: 22

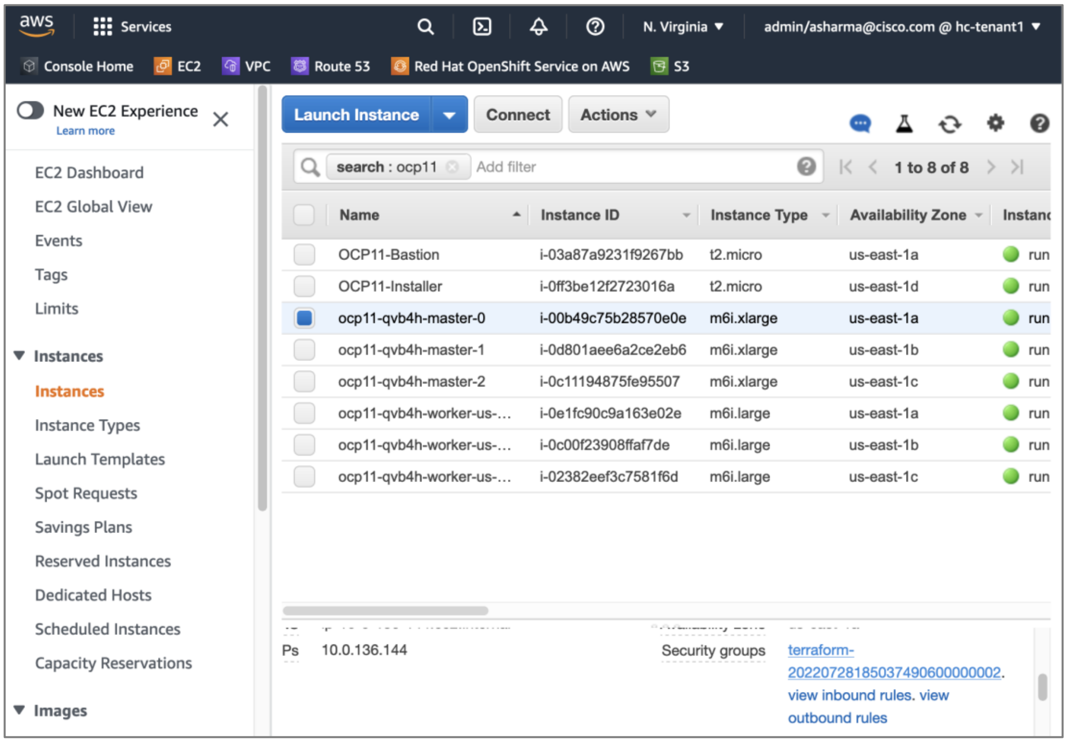
Source type Info: Anywhere

Source Info: *Add CIDR, prefix list or security*
0.0.0.0/0

Description - *optional* Info: *e.g. SSH for admin desktop*

Step 11. Scroll down and click **Launch Instance**.

Step 12. If an inbound rule for SSH does not already exist on the master and worker nodes, add a new inbound rule. The rule would allow SSH access from the new bastion security group. From the AWS console, navigate to **EC2 > Instances** and select one of the master node instances in the cluster. In the bottom window, navigate to **Security grou**ps and click on the name (terraform-xxx) to take you to the security group for master nodes.

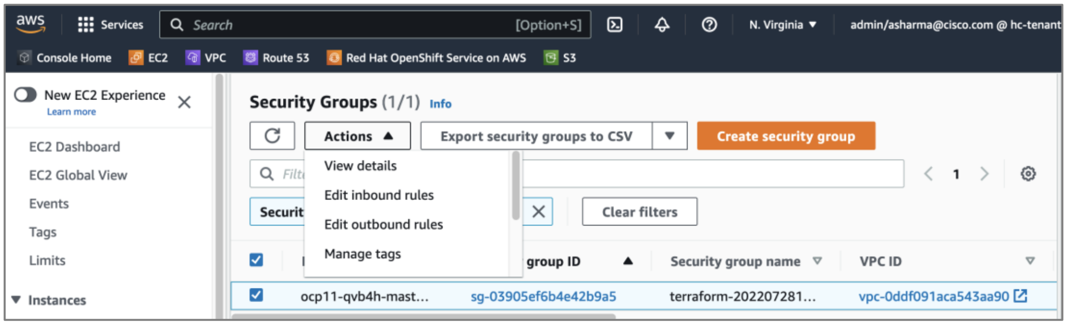


	Name	Instance ID	Instance Type	Availability Zone	Instanc
□	OCP11-Bastion	i-03a87a9231f9267bb	t2.micro	us-east-1a	● run
□	OCP11-Installer	i-0ff3be12f2723016a	t2.micro	us-east-1d	● run
☑	ocp11-qvb4h-master-0	i-00b49c75b28570e0e	m6i.xlarge	us-east-1a	● run
□	ocp11-qvb4h-master-1	i-0d801aee6a2ce2eb6	m6i.xlarge	us-east-1b	● run
□	ocp11-qvb4h-master-2	i-0c11194875fe95507	m6i.xlarge	us-east-1c	● run
□	ocp11-qvb4h-worker-us-…	i-0e1fc90c9a163e02e	m6i.large	us-east-1a	● run
□	ocp11-qvb4h-worker-us-…	i-0c00f23908ffaf7de	m6i.large	us-east-1b	● run
□	ocp11-qvb4h-worker-us-…	i-02382eef3c7581f6d	m6i.large	us-east-1c	● run

Ps 10.0.136.144

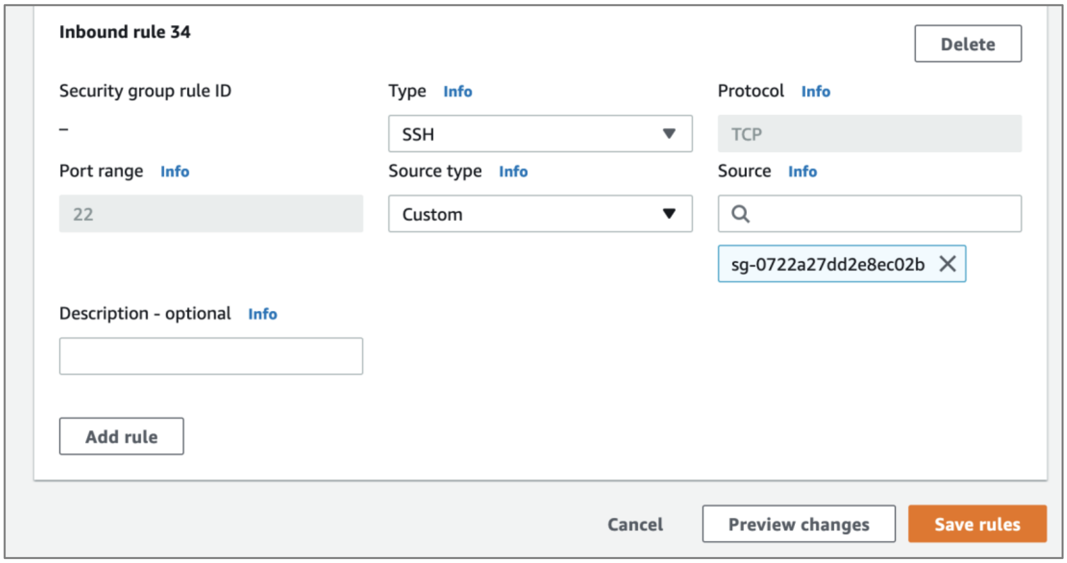
Security groups terraform-20220728185037490600000002. view inbound rules. view outbound rules

Step 13. Click on **Actions** and select **Edit inbound rules** from the drop-down list.



☑	I	group ID	Security group name	VPC ID
☑	ocp11-qvb4h-mast…	sg-03905ef6b4e42b9a5	terraform-202207281…	vpc-0ddf091aca543aa90

Step 14. In the **Edit inbound rules** window, scroll down to the bottom and click on **Add Rule**. In the last row, edit the new rule Type and select **SSH** from the drop-down list. For the **Source,** select the security group rule created earlier for the bastion node.



Inbound rule 34

Security group rule ID: –

Type: SSH

Protocol: TCP

Port range: 22

Source type: Custom

Source: sg-0722a27dd2e8ec02b

Description - optional:

Step 15. Click on **Save Rules**.

Step 16. Repeat steps [12-16] for a worker node.

Step 17. The next few steps are to copy the SSH private key to the bastion node. This is the key pair whose public key was provided to the OCP installer to install the cluster. The private key is in the OCP installer **(~/.ssh/id_rsa** – not **id_rsa.pub**). You can use SCP to copy the file the OCP installer to the public IP of the bastion node as outlined below.

Note: **Secure** the private keys. If your private key is compromised, your OCP cluster and applications running on it could be at risk.

Step 18. From the local workstation used to SSH into OCP installer, copy the PEM file for SSH access to Bastion node, to the OCP Installer.

```
$ scp -i "<PEM_File_To_Access_OCP_Installer>" <PEM_File_To_Access_OCP_Bastion> ec2-user@<Public_IP/DNS_of_OCP_Installer>:
```

Step 19. SSH into the OCP Installer.

```
$ ssh -i "<PEM_File_To_Access_OCP_Installer>" ec2-user@<Public_IP/DNS_of_OCP_Installer
```

Step 20. From OCP Installer, copy private key (not **id_rsa.pub)** to OCP Bastion workstation.

```
$ scp -i "<PEM_File_To_Access_OCP_Bastion>" ~/.ssh/id_rsa ec2-user@<Public_IP/DNS_of_OCP_Bastion>:~/.ssh/id_rsa
```

Step 21. From the OCP installer, note the private DNS hostname of the OCP cluster node. You cannot run the OCP cluster CLI tools from the Bastion node as you haven't installed them on the Bastion node.

```
[ec2-user@ip-172-31-30-78 ~]$ export KUBECONFIG=ocp11/auth/kubeconfig
[ec2-user@ip-172-31-30-78 ~]$ oc get nodes
NAME                         STATUS   ROLES    AGE    VERSION
ip-10-0-129-37.ec2.internal  Ready    worker   106d   v1.23.5+3afdacb
ip-10-0-136-144.ec2.internal Ready    master   106d   v1.23.5+3afdacb
ip-10-0-148-146.ec2.internal Ready    worker   106d   v1.23.5+3afdacb
ip-10-0-149-45.ec2.internal  Ready    master   106d   v1.23.5+3afdacb
ip-10-0-161-196.ec2.internal Ready    master   106d   v1.23.5+3afdacb
ip-10-0-175-170.ec2.internal Ready    worker   106d   v1.23.5+3afdacb
[ec2-user@ip-172-31-30-78 ~]$
```

Step 22. SSH into Bastion workstation directly from OCP Installer.

```
$ ssh -i <PEM_File_To_Access_OCP_Bastion>" ec2-user@<Public_IP/DNS_of_OCP_Bastion>
```

Step 23. SSH from Bastion workstation to OCP cluster node.

```
[ec2-user@ip-10-0-9-213 ~]$ ssh core@ip-10-0-136-144.ec2.internal
```

```
[ec2-user@ip-10-0-9-213 ~]$
[ec2-user@ip-10-0-9-213 ~]$ ssh core@ip-10-0-136-144.ec2.internal
Red Hat Enterprise Linux CoreOS 410.84.202207051718-0
  Part of OpenShift 4.10, RHCOS is a Kubernetes native operating system
  managed by the Machine Config Operator (`clusteroperator/machine-config`).

WARNING: Direct SSH access to machines is not recommended; instead,
make configuration changes via `machineconfig` objects:
  https://docs.openshift.com/container-platform/4.10/architecture/architecture-rhcos.html

---
Last login: Sat Nov 12 13:40:39 2022 from 10.0.9.213
[core@ip-10-0-136-144 ~]$
```

Now you have SSH access to Red Hat OCP cluster in an Amazon VPC.

Procedure 9. (Optional) Remove the Administrator Access policy for the IAM account

The IAM account that was created earlier with Administrator Access policy is only required for installation and can be disabled and changed to Read Only or some other lower-level access policy.

Enable Secure Hybrid Cloud Connectivity

This section describes the deployment of secure hybrid cloud connectivity between an on-prem data center and public cloud. The solution uses IPsec VPN connections established between CSR1000v routers in the Enterprise and Transit Gateway routers in AWS.

Prerequisites

The prerequisites for deploying IPsec VPN between on-prem and public cloud are:

- Enterprise Gateways (in this case, CSR1000v) deployed using stable, recommended version of software with configuration in place for out-of-band management, connectivity to the Enterprise data center network (Cisco ACI in this case), connectivity to the Internet.
- Public IPs allocated for the Enterprise gateways
- Enterprise Firewall provisioned to allows IPsec protocols and traffic to the public IP of the Enterprise gateways from AWS Transit Gateways (you will get this information when TGW is deployed).
- Determine the routing protocol to use (static or dynamic - BGP)
- If using dynamic, then Autonomous System Number (ASN) for BGP on Enterprise side. You can also specify the AWS side or use the default that AWS uses.
- Identify a summary route or routes that should be advertise to AWS VPCs. For static routing, static routes will be added to TGW.
- Identify VPC ID(s) if attaching multiple VPCs to TGW - an attachment per VPC (and VPN) will need to be created for the Transit Gateway.

Setup Information

Table 13 lists the configuration parameters for the site-to-site IPsec VPN.

Table 13. Site-to-Site IPsec VPN - Configuration Parameters

Variable	Variable Name	Value	Additional Info
Transit Gateway Name	Name Tag	HC-TGW-0	
AWS ASN (Optional)	-	65512	You can also AWS default.
Transit Gateway CIDR Block (Optional)	CIDR	99.0.0.0/24	You can also AWS default. Specifying for ease of troubleshooting.
VPC Attachment Name (s)	-	TGW-VPC-Attachment	
Name for CSR1kv-1	Name Tag	customer-gateway-1	
ASN for CSR1kv-1		65251	Not used but specifying it for future use.
Public IP for CSR1kv-1		<specify>	
Name for CSR1kv-2	Name Tag	customer-gateway-2	
ASN for CSR1kv-2		65251	Not used but specifying it for future use.
Public IP for CSR1kv-2		<specify>	
Site-to-Site VPN Name (s)	Name Tag	VPN-181 VPN-182	
VPN Attachment Name (s)	-	TGW-VPN-Attachment1 TGW-VPN-Attachment2	

Deployment Steps

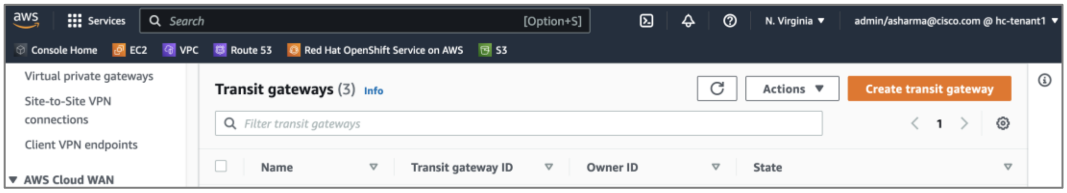
The procedures in this section will enable hybrid cloud connectivity between on-prem and public cloud.

Procedure 1. Deploy Transit Gateway (TGW) in AWS

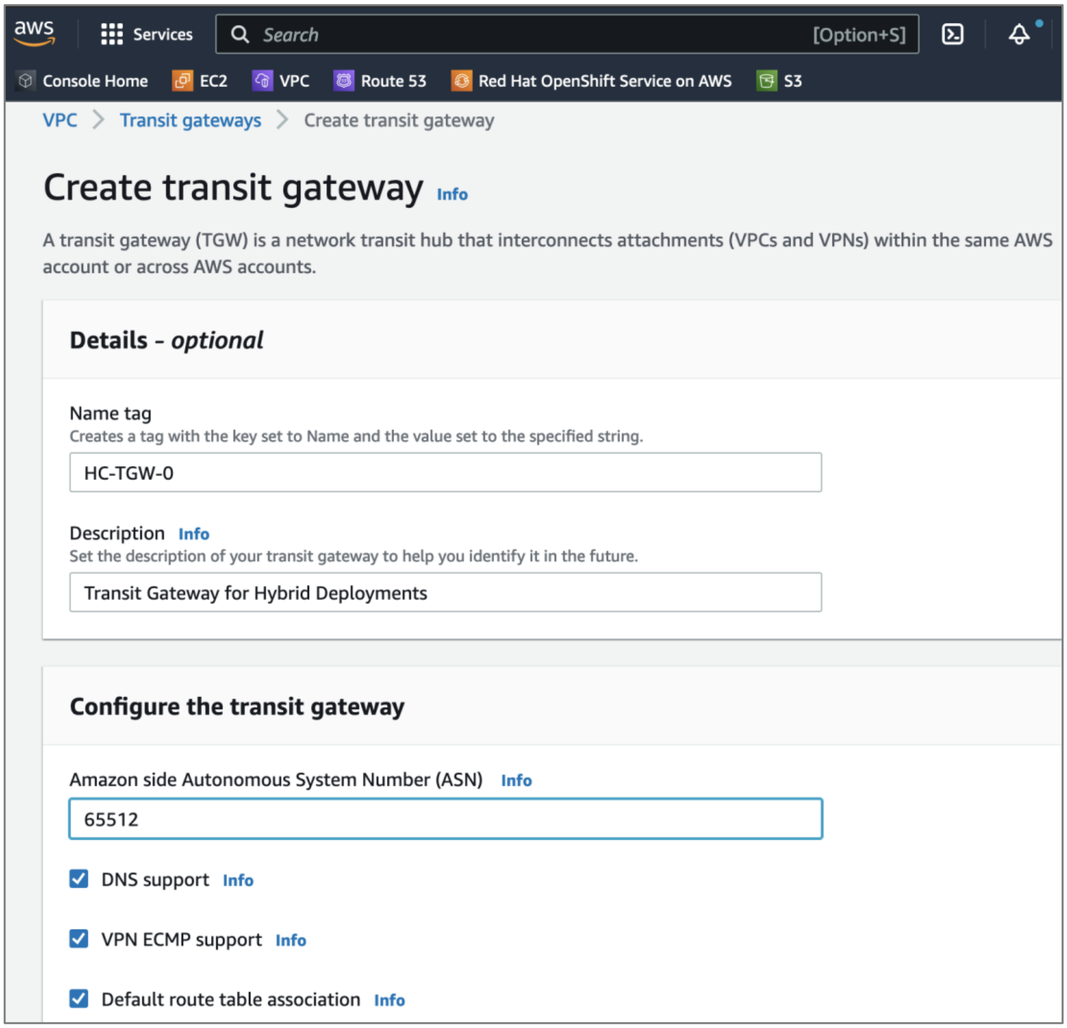
Step 1. Use a web browser to navigate to **console.aws.amazon.com.** Login to your AWS account.

Step 2. Navigate to **VPC > Transit gateways** from the left navigation pane.

Step 3. Click on **Create Transit Gateway.**



Step 4. In the **Create Transit Gateway** window, specify a **Name tag**, **Description**, **ASN** and accept the remaining defaults.



aws
Services
Search
[Option+S]
Console Home
EC2
VPC
Route 53
Red Hat OpenShift Service on AWS
S3
VPC > Transit gateways > Create transit gateway

Create transit gateway Info

A transit gateway (TGW) is a network transit hub that interconnects attachments (VPCs and VPNs) within the same AWS account or across AWS accounts.

Details - *optional*

Name tag

Creates a tag with the key set to Name and the value set to the specified string.

HC-TGW-0

Description Info

Set the description of your transit gateway to help you identify it in the future.

Transit Gateway for Hybrid Deployments

Configure the transit gateway

Amazon side Autonomous System Number (ASN) Info

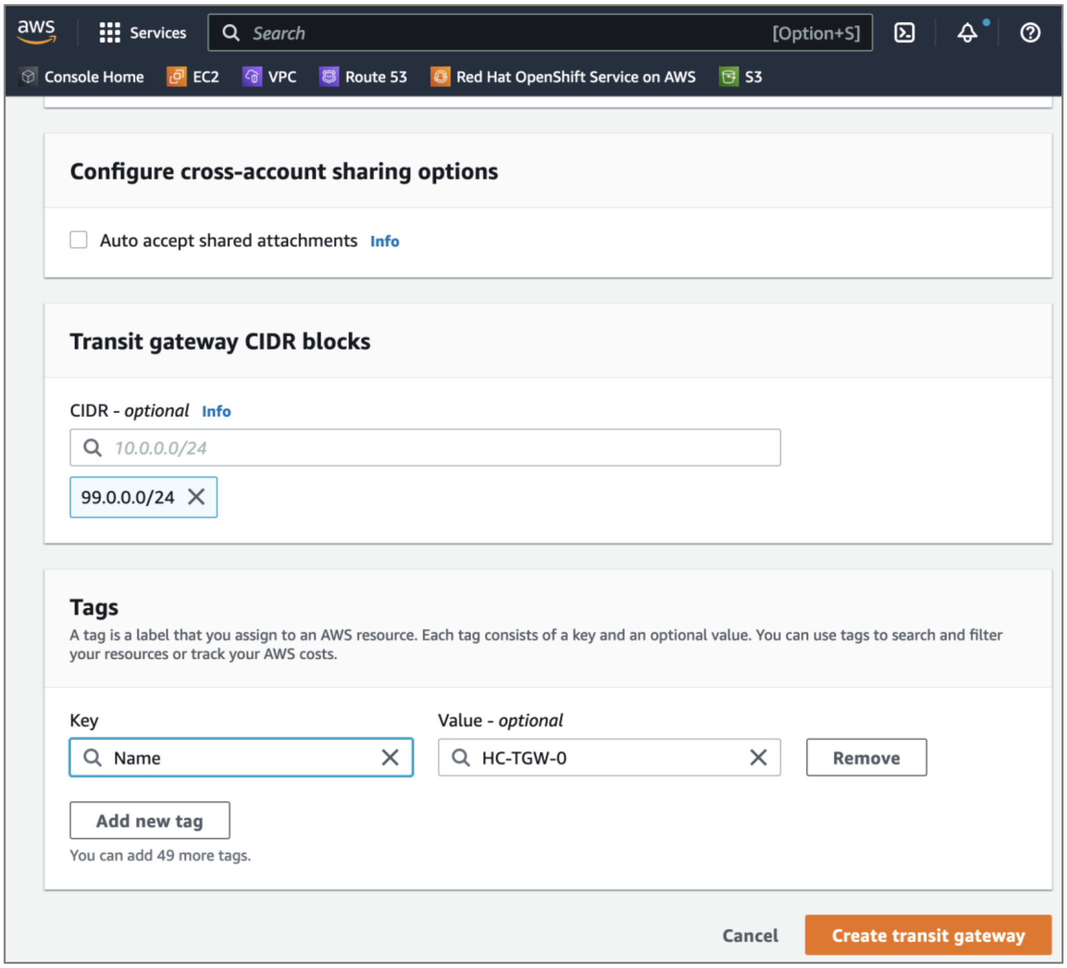
65512

☑ DNS support Info

☑ VPN ECMP support Info

☑ Default route table association Info

Step 5. Provide a **CIDR** subnet for TGW to use.



Configure cross-account sharing options

□ Auto accept shared attachments Info

Transit gateway CIDR blocks

CIDR - *optional* Info

10.0.0.0/24

99.0.0.0/24 ✕

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - *optional*
Name	HC-TGW-0

You can add 49 more tags.

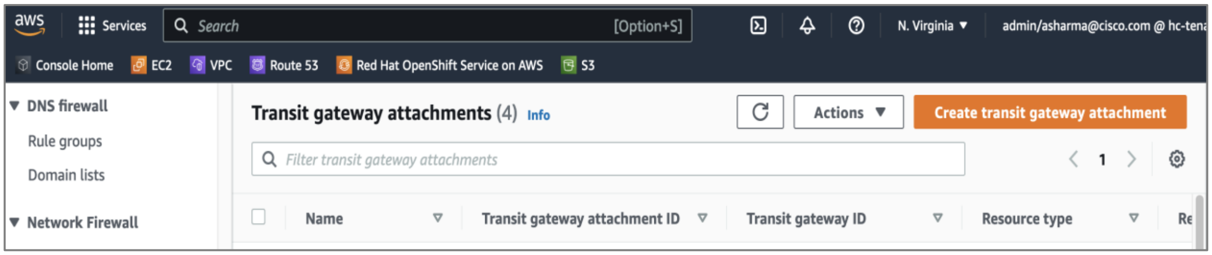
Step 6. Click **Create transit gateway**.

Procedure 2. Create TGW attachment to Red Hat OCP cluster in AWS VPC

Step 1. Use a web browser to navigate to **console.aws.amazon.com.** Login to your AWS account.

Step 2. Navigate to **VPC > Transit gateways attachments** from the left navigation pane.

Step 3. Click on **Create transit gateway attachment.**

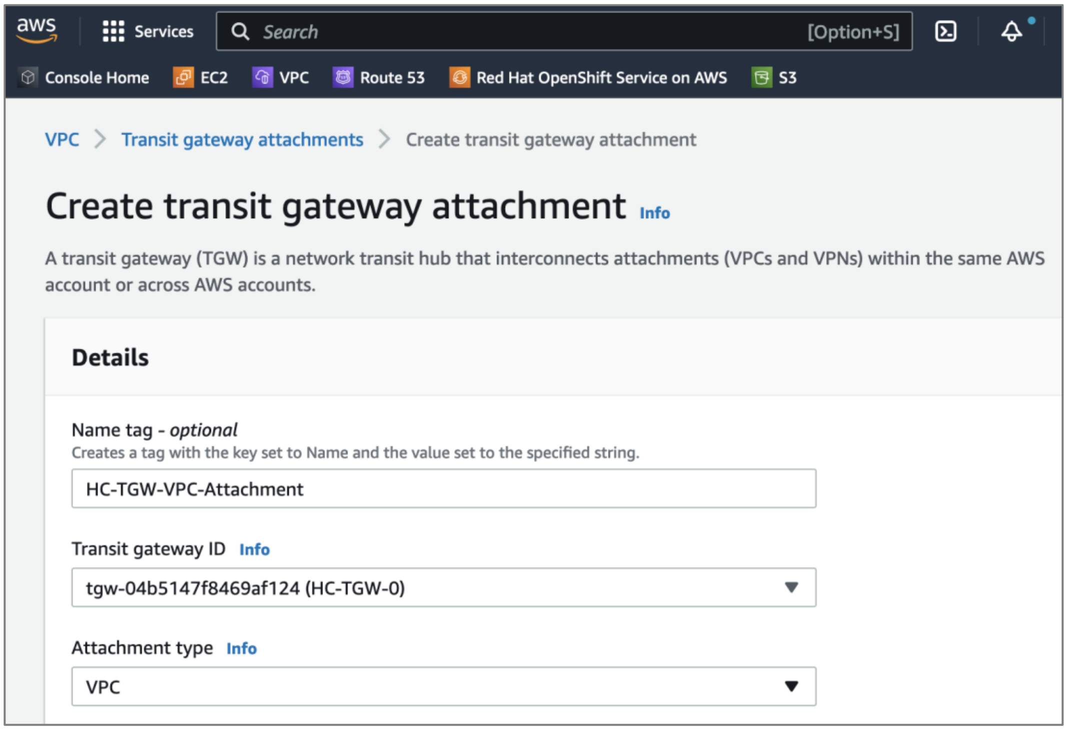


Transit gateway attachments (4) Info

Actions ▼
Create transit gateway attachment
Filter transit gateway attachments
1

□	Name	Transit gateway attachment ID	Transit gateway ID	Resource type	Re

Step 4. In the **Create transit gateway attachment** window, specify a **Name tag**. For the **Transit gateway id**, select the newly created TGW from the drop-down list. For the **Attachment type**, select **VPC**.


aws
Services
Search
[Option+S]
Console Home
EC2
VPC
Route 53
Red Hat OpenShift Service on AWS
S3
VPC > Transit gateway attachments > Create transit gateway attachment

Create transit gateway attachment

A transit gateway (TGW) is a network transit hub that interconnects attachments (VPCs and VPNs) within the same AWS account or across AWS accounts.

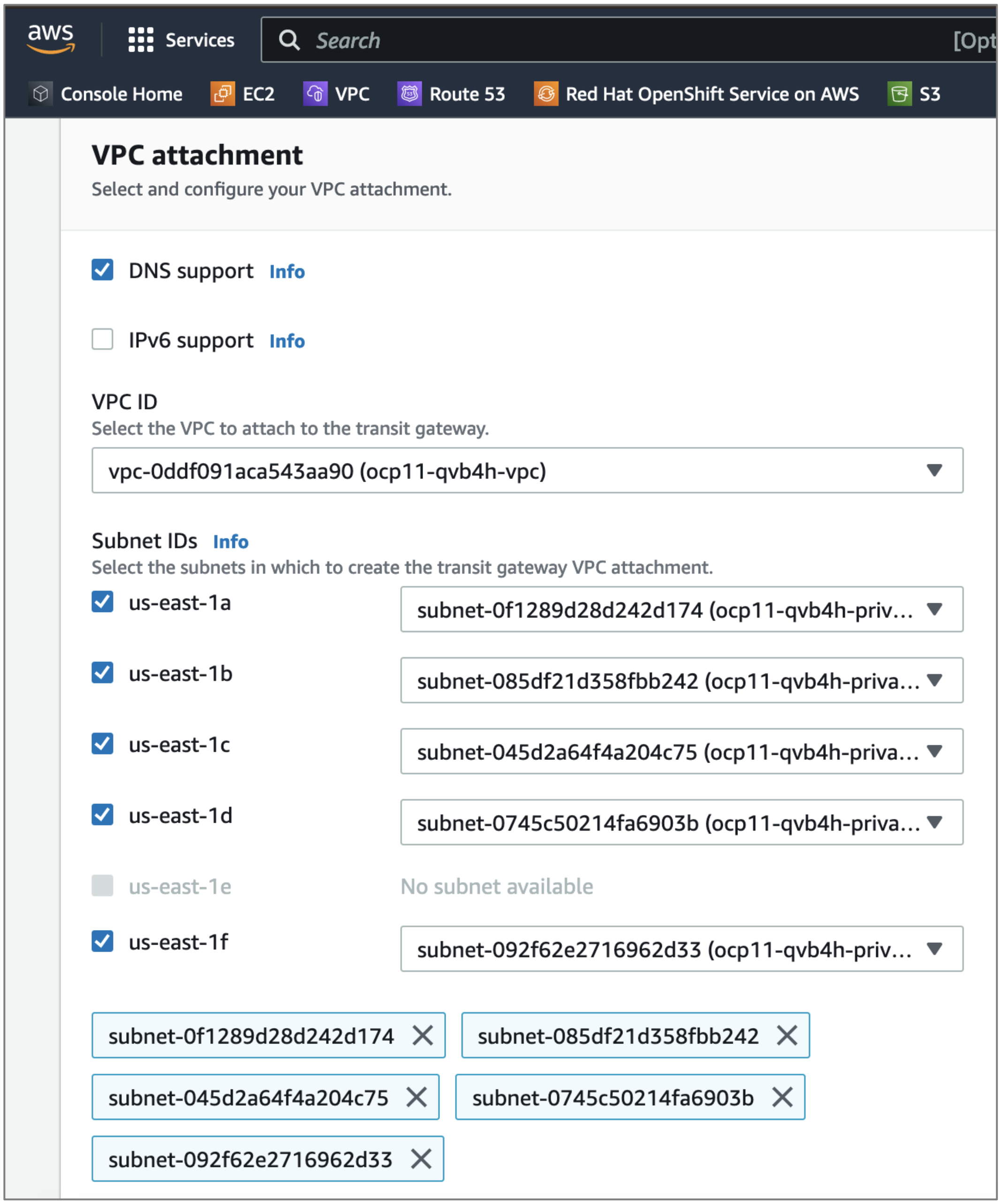
Details

Name tag - *optional*
Creates a tag with the key set to Name and the value set to the specified string.
HC-TGW-VPC-Attachment

Transit gateway ID
tgw-04b5147f8469af124 (HC-TGW-0)

Attachment type
VPC

Step 5. In the VPC Attachment section, for the **VPC ID**, select the OCP cluster VPC from the drop-down list. For **Subnet IDs**, select the subnets to create an attachment for in each availability zone.



VPC attachment

Select and configure your VPC attachment.

☑ DNS support Info

□ IPv6 support Info

VPC ID

Select the VPC to attach to the transit gateway.

vpc-0ddf091aca543aa90 (ocp11-qvb4h-vpc)

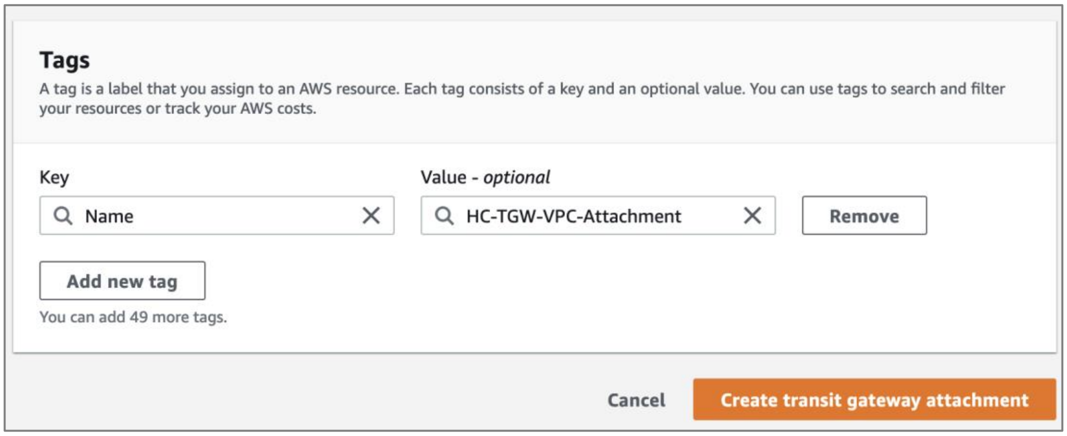
Subnet IDs Info

Select the subnets in which to create the transit gateway VPC attachment.

☑ us-east-1a	subnet-0f1289d28d242d174 (ocp11-qvb4h-priv…
☑ us-east-1b	subnet-085df21d358fbb242 (ocp11-qvb4h-priva…
☑ us-east-1c	subnet-045d2a64f4a204c75 (ocp11-qvb4h-priva…
☑ us-east-1d	subnet-0745c50214fa6903b (ocp11-qvb4h-priva…
□ us-east-1e	No subnet available
☑ us-east-1f	subnet-092f62e2716962d33 (ocp11-qvb4h-priv…

subnet-0f1289d28d242d174 ✕ subnet-085df21d358fbb242 ✕ subnet-045d2a64f4a204c75 ✕ subnet-0745c50214fa6903b ✕ subnet-092f62e2716962d33 ✕

Step 6. Scroll down to the bottom and click on **Create transit gateway attachment**.



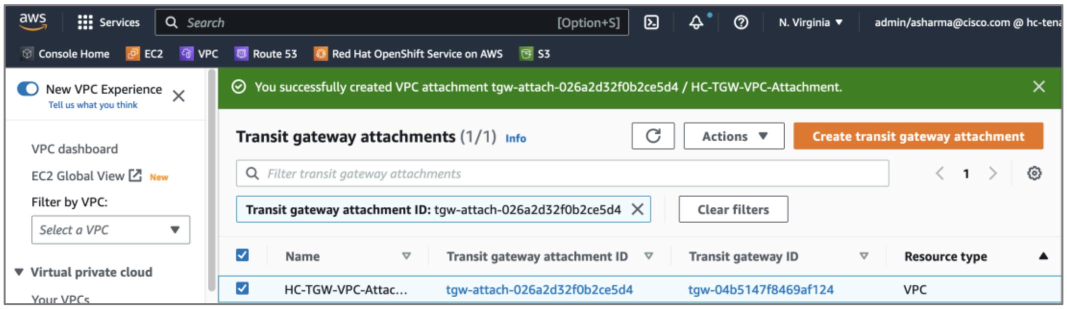
Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - *optional*	
Name	HC-TGW-VPC-Attachment	Remove

You can add 49 more tags.

Step 7. Verify it is successfully created.



You successfully created VPC attachment tgw-attach-026a2d32f0b2ce5d4 / HC-TGW-VPC-Attachment.

Transit gateway attachments (1/1) Info

Transit gateway attachment ID: tgw-attach-026a2d32f0b2ce5d4

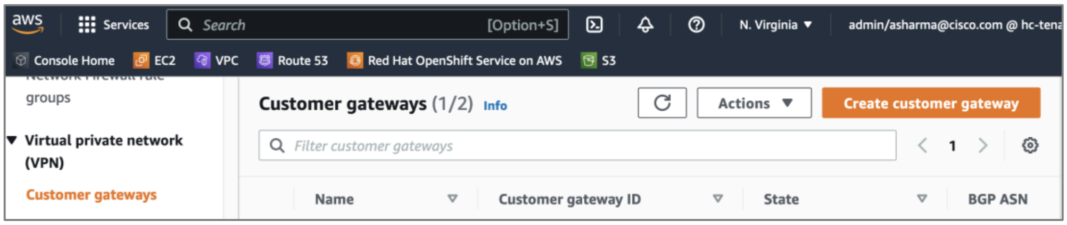
☑	Name	Transit gateway attachment ID	Transit gateway ID	Resource type
☑	HC-TGW-VPC-Attac...	tgw-attach-026a2d32f0b2ce5d4	tgw-04b5147f8469af124	VPC

Procedure 3. Create Customer Gateways in AWS VPC

Customer gateways in AWS represent the on-prem gateways on the other end of the site-to-site VPN.

Step 1. Use a web browser to navigate to **console.aws.amazon.com**. Login to your AWS account.

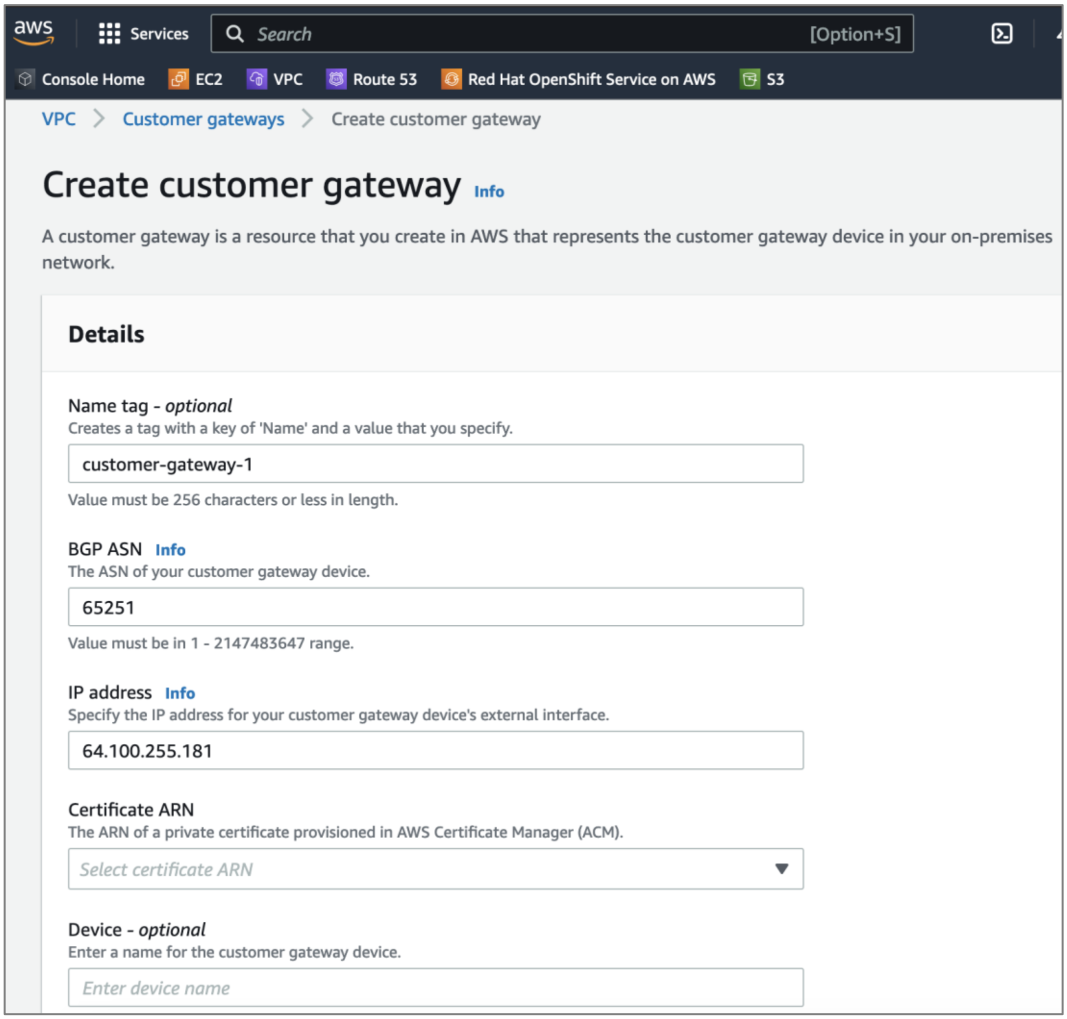
Step 2. Navigate to **VPC > Customer gateways** from the left navigation pane.



Customer gateways (1/2) Info

Name	Customer gateway ID	State	BGP ASN

Step 3. Click on **Create transit gateway attachment**.


aws
Services
Search
[Option+S]
Console Home
EC2
VPC
Route 53
Red Hat OpenShift Service on AWS
S3
VPC > Customer gateways > Create customer gateway

Create customer gateway Info

A customer gateway is a resource that you create in AWS that represents the customer gateway device in your on-premises network.

Details

Name tag - *optional*
Creates a tag with a key of 'Name' and a value that you specify.
customer-gateway-1
Value must be 256 characters or less in length.

BGP ASN Info
The ASN of your customer gateway device.
65251
Value must be in 1 - 2147483647 range.

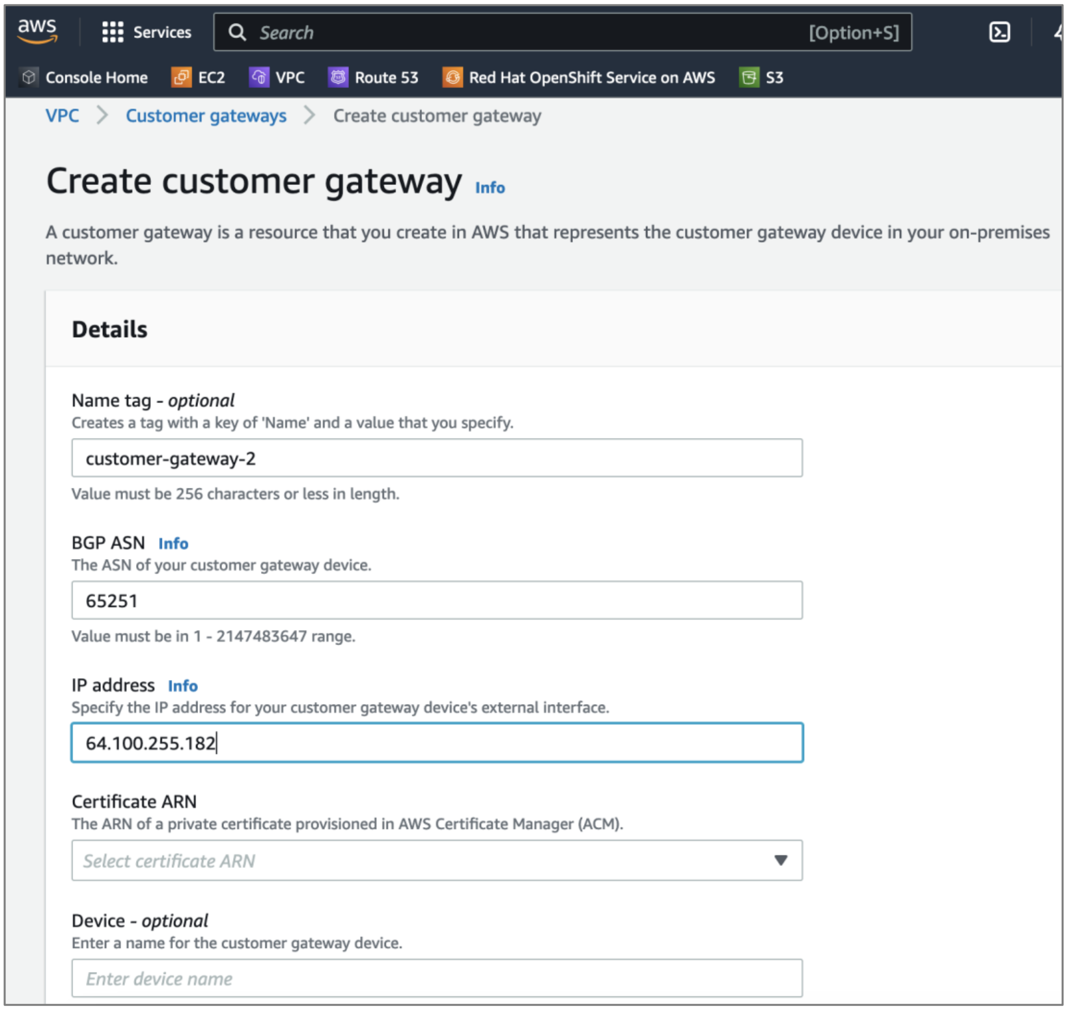
IP address Info
Specify the IP address for your customer gateway device's external interface.
64.100.255.181

Certificate ARN
The ARN of a private certificate provisioned in AWS Certificate Manager (ACM).
Select certificate ARN ▼

Device - *optional*
Enter a name for the customer gateway device.
Enter device name

Step 4. Scroll down to the bottom and click on **Create customer gateway**.

Step 5. Repeat Steps 1-4 for the second on-prem gateway.


aws
Services
Search
[Option+S]
Console Home
EC2
VPC
Route 53
Red Hat OpenShift Service on AWS
S3
VPC > Customer gateways > Create customer gateway

Create customer gateway Info

A customer gateway is a resource that you create in AWS that represents the customer gateway device in your on-premises network.

Details

Name tag - *optional*
Creates a tag with a key of 'Name' and a value that you specify.
customer-gateway-2
Value must be 256 characters or less in length.

BGP ASN Info
The ASN of your customer gateway device.
65251
Value must be in 1 - 2147483647 range.

IP address Info
Specify the IP address for your customer gateway device's external interface.
64.100.255.182

Certificate ARN
The ARN of a private certificate provisioned in AWS Certificate Manager (ACM).
Select certificate ARN

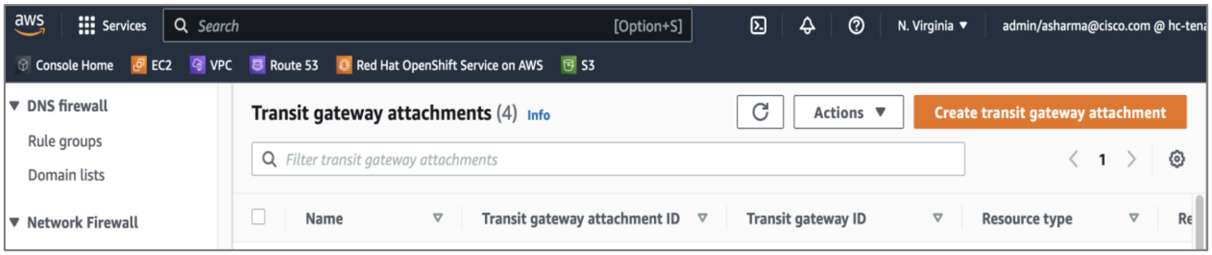
Device - *optional*
Enter a name for the customer gateway device.
Enter device name

Procedure 4. Create TGW attachment to the Site-to-Site VPN

Step 1. Use a web browser to navigate to **console.aws.amazon.com**. Login to your AWS account.

Step 2. Navigate to **VPC > Transit gateways attachments** from the left navigation pane.

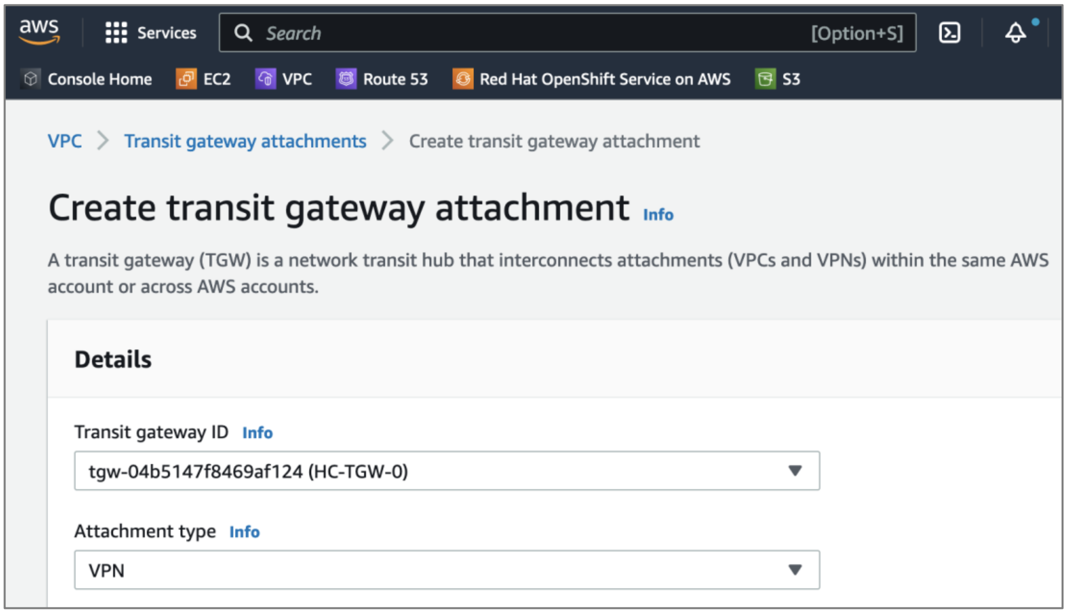
Step 3. Click on **Create transit gateway attachment**.


aws
Services
Search
[Option+S]
N. Virginia
admin/asharma@cisco.com @ hc-tena
Console Home
EC2
VPC
Route 53
Red Hat OpenShift Service on AWS
S3
DNS firewall
Rule groups
Domain lists
Network Firewall
Actions
Create transit gateway attachment
Filter transit gateway attachments
1

Transit gateway attachments (4) Info

□	Name	Transit gateway attachment ID	Transit gateway ID	Resource type	Re

Step 4. In the **Create transit gateway attachment** window, specify a **Name tag (**for example, HC-TGW-VPN-Attachment**)**. For the **Transit gateway id**, select the newly created TGW from the drop-down list. For the **Attachment type**, select **VPN**.


aws
Services
Search [Option+S]
Console Home
EC2
VPC
Route 53
Red Hat OpenShift Service on AWS
S3
VPC > Transit gateway attachments > Create transit gateway attachment

Create transit gateway attachment Info

A transit gateway (TGW) is a network transit hub that interconnects attachments (VPCs and VPNs) within the same AWS account or across AWS accounts.

Details

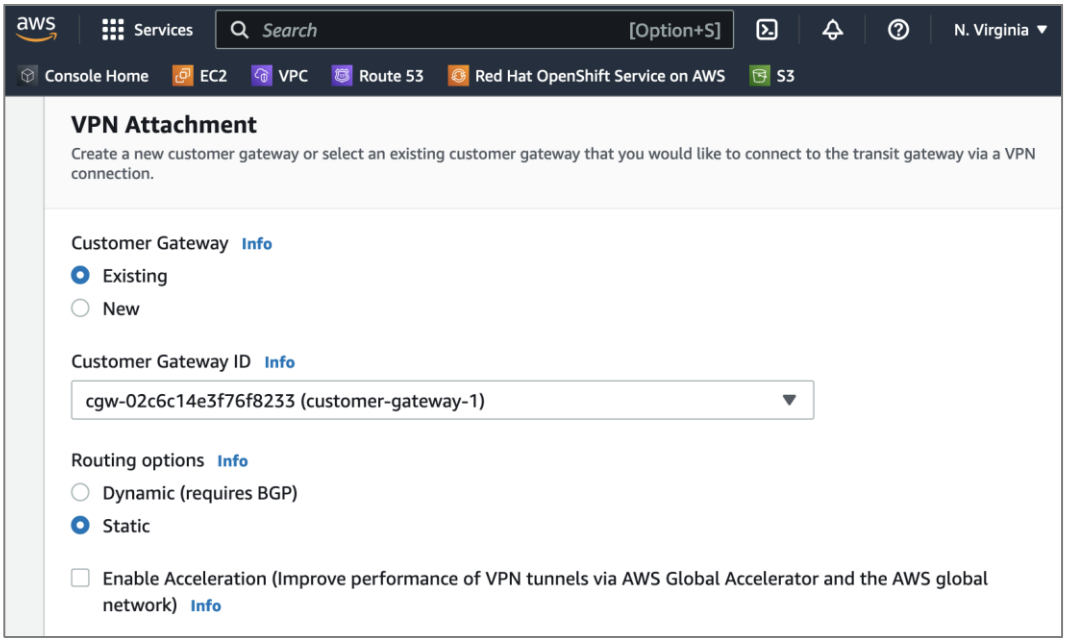
Transit gateway ID Info

tgw-04b5147f8469af124 (HC-TGW-0) ▼

Attachment type Info

VPN ▼

Step 5. In the VPN Attachment section, for **Customer gateway**, select the radio button for **Existing.** For **Customer gateway ID**, select the first customer gateway from the drop-down list. For **Routing Options**, select **Static**.



VPN Attachment

Create a new customer gateway or select an existing customer gateway that you would like to connect to the transit gateway via a VPN connection.

Customer Gateway Info

● Existing

○ New

Customer Gateway ID Info

cgw-02c6c14e3f76f8233 (customer-gateway-1) ▼

Routing options Info

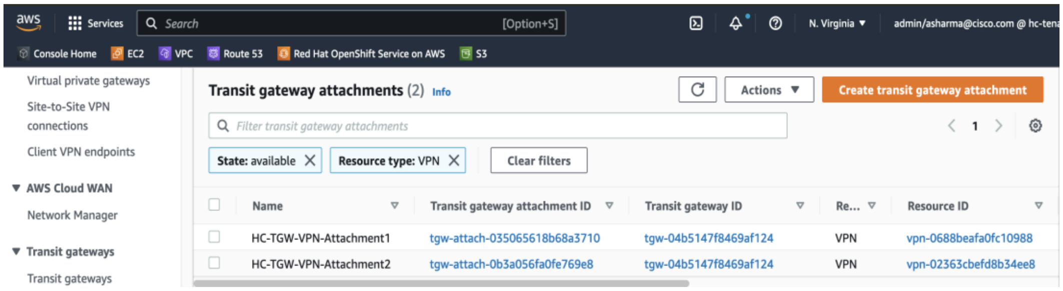
○ Dynamic (requires BGP)

● Static

□ Enable Acceleration (Improve performance of VPN tunnels via AWS Global Accelerator and the AWS global network) Info

Step 6. Scroll down to the bottom and click on **Create transit gateway attachment.**

Step 7. Repeat steps 1-6 to create a second TGW attachment to second customer gateway.



Transit gateway attachments (2) Info

State: available ✕ Resource type: VPN ✕

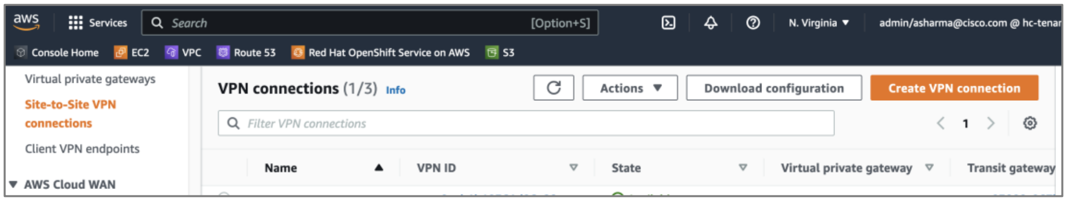
□	Name	Transit gateway attachment ID	Transit gateway ID	Re...	Resource ID
□	HC-TGW-VPN-Attachment1	tgw-attach-035065618b68a3710	tgw-04b5147f8469af124	VPN	vpn-0688beafa0fc10988
□	HC-TGW-VPN-Attachment2	tgw-attach-0b3a056fa0fe769e8	tgw-04b5147f8469af124	VPN	vpn-02363cbefd8b34ee8

Procedure 5. Create Site-to-Site VPN to on-prem data center

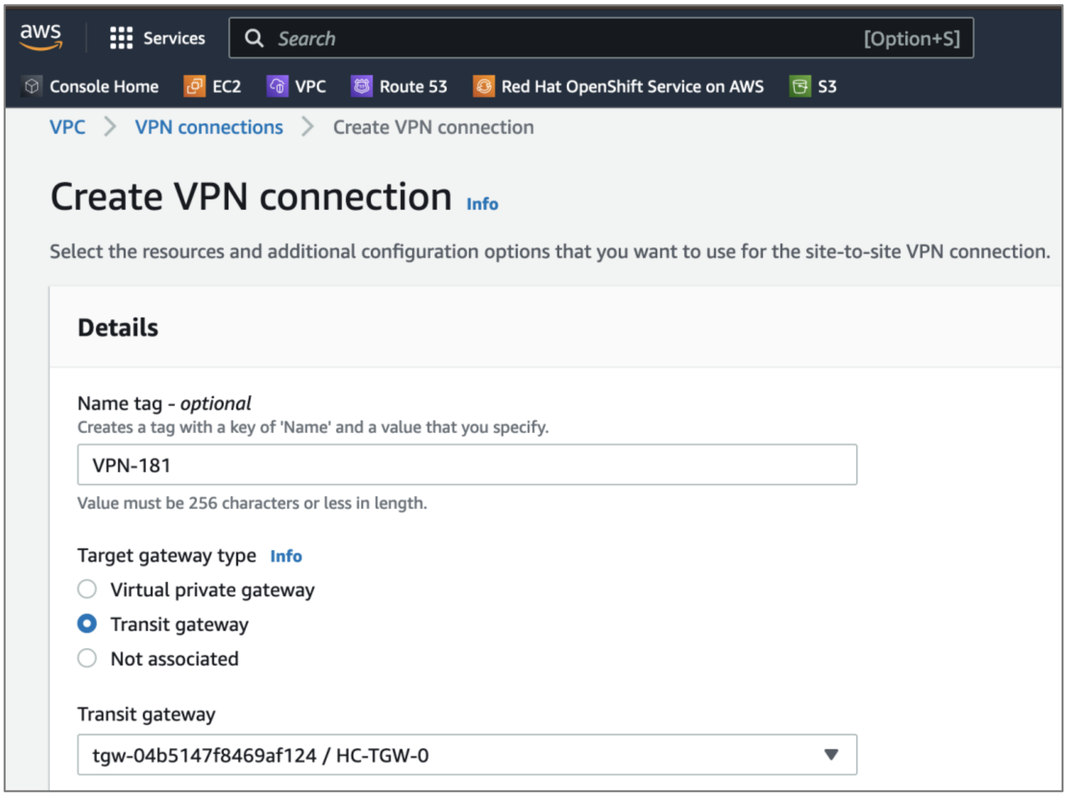
Step 1. Use a web browser to navigate to **console.aws.amazon.com**. Login to your AWS account.

Step 2. Navigate to **VPC > Site-to-Site VPN connections** from the left navigation pane.

Step 3. Click on **Create VPN connection**.



Step 4. In the **Create VPN connection** window, specify a name in **Name tag**, for the **Target gateway type**, select **Transit gateway**, and for the **Transit Gateway**, select the newly created **Transit Gateway** from the drop-down list.


aws
Services
Search
[Option+S]
Console Home
EC2
VPC
Route 53
Red Hat OpenShift Service on AWS
S3
VPC > VPN connections > Create VPN connection
Info
Info

Create VPN connection

Select the resources and additional configuration options that you want to use for the site-to-site VPN connection.

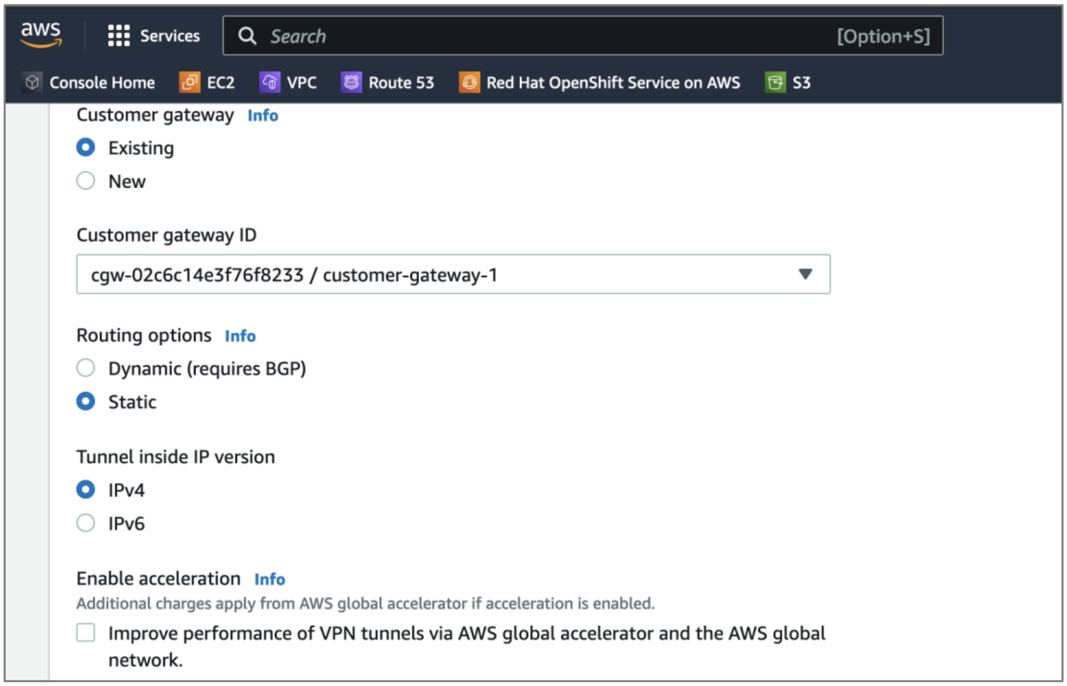
Details

Name tag - *optional*
Creates a tag with a key of 'Name' and a value that you specify.
VPN-181
Value must be 256 characters or less in length.

Target gateway type
○ Virtual private gateway
● Transit gateway
○ Not associated

Transit gateway
tgw-04b5147f8469af124 / HC-TGW-0 ▼

Step 5. For the **Customer gateway**, select **Existing**. For the **Customer gateway ID**, select the first gateway from the drop-down list. For the **Routing options**, select **Static**. For the **Tunnel inside IP version**, select **IPv4**.



Customer gateway Info

● Existing

○ New

Customer gateway ID

cgw-02c6c14e3f76f8233 / customer-gateway-1 ▼

Routing options Info

○ Dynamic (requires BGP)

● Static

Tunnel inside IP version

● IPv4

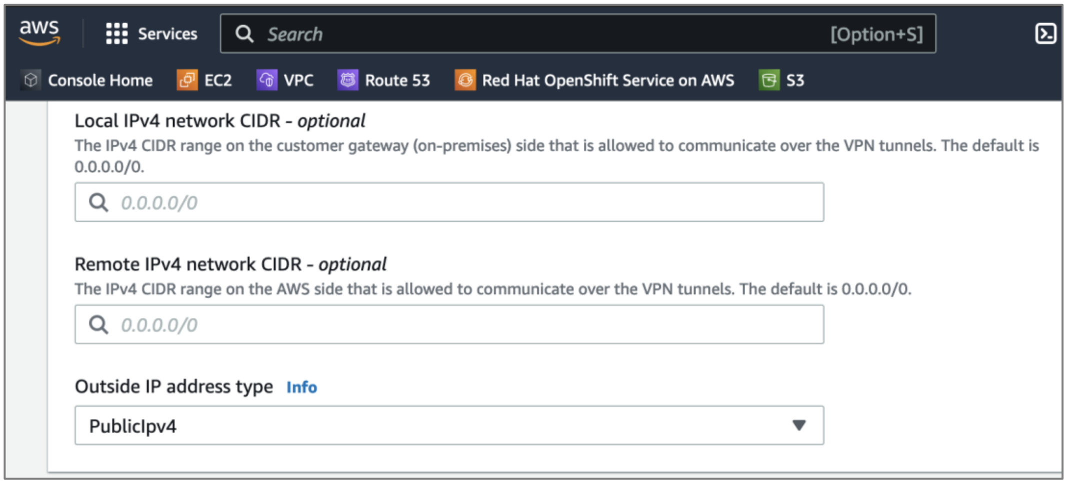
○ IPv6

Enable acceleration Info

Additional charges apply from AWS global accelerator if acceleration is enabled.

□ Improve performance of VPN tunnels via AWS global accelerator and the AWS global network.

Step 6. Select default options for the IPs on either side allowed over the site-to-site VPN. Default allows all traffic. Restrict as needed for your environment.



Local IPv4 network CIDR - *optional*

The IPv4 CIDR range on the customer gateway (on-premises) side that is allowed to communicate over the VPN tunnels. The default is 0.0.0.0/0.

0.0.0.0/0

Remote IPv4 network CIDR - *optional*

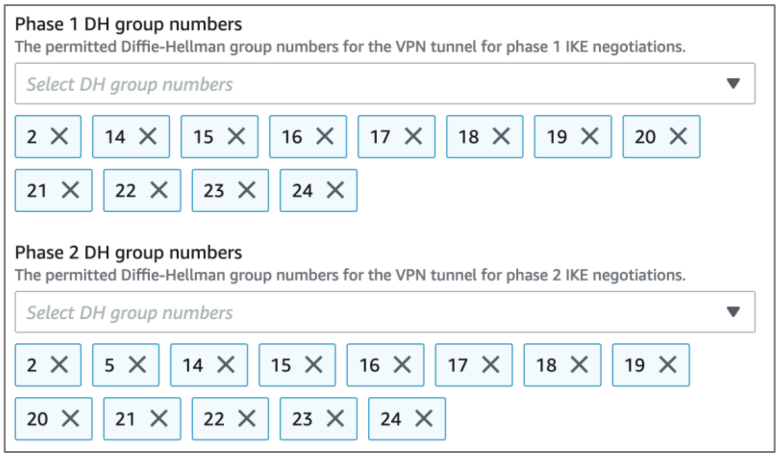
The IPv4 CIDR range on the AWS side that is allowed to communicate over the VPN tunnels. The default is 0.0.0.0/0.

0.0.0.0/0

Outside IP address type Info

PublicIpv4 ▼

Step 7. Expand **Tunnel 1 options** and under **Advanced options for tunnel 1**, select **Edit tunnel 1 options.** This will enable several configuration options. For **Phase 1 DH group numbers**, delete 2 by clicking on the **X** next 2. Similarly, delete 2 from **Phase 2 DH group numbers**.



Phase 1 DH group numbers

The permitted Diffie-Hellman group numbers for the VPN tunnel for phase 1 IKE negotiations.

Select DH group numbers

2 ✕ 14 ✕ 15 ✕ 16 ✕ 17 ✕ 18 ✕ 19 ✕ 20 ✕ 21 ✕ 22 ✕ 23 ✕ 24 ✕

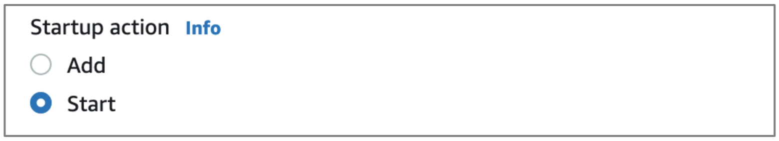
Phase 2 DH group numbers

The permitted Diffie-Hellman group numbers for the VPN tunnel for phase 2 IKE negotiations.

Select DH group numbers

2 ✕ 5 ✕ 14 ✕ 15 ✕ 16 ✕ 17 ✕ 18 ✕ 19 ✕ 20 ✕ 21 ✕ 22 ✕ 23 ✕ 24 ✕

Step 8. For **Startup action**, select **Start** to specify the action to take for establishing a new (or modified) VPN tunnel. This is only supported for customer gateways with IP addresses.

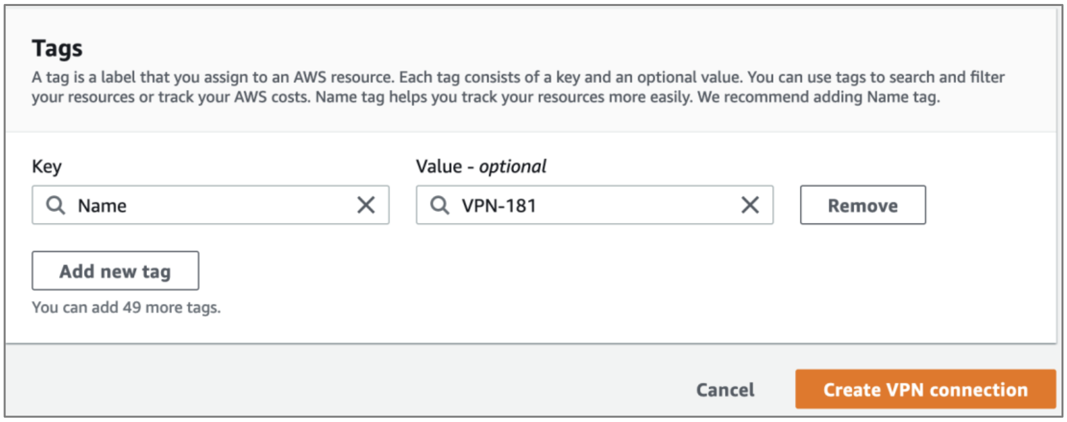


Startup action Info

○ Add

● Start

Step 9. Under **Tags**, specify a name for connection using **Key** (Name) and **Value**.



Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs. Name tag helps you track your resources more easily. We recommend adding Name tag.

Key	Value - *optional*	
Name	VPN-181	Remove

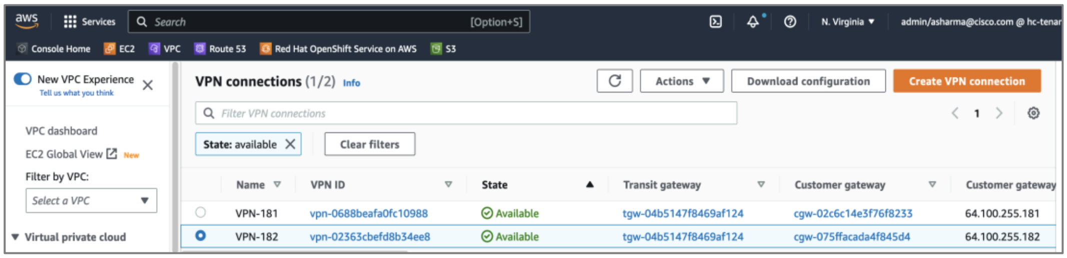
Add new tag

You can add 49 more tags.

Cancel   Create VPN connection

Step 10. Click **Create VPN connection** to create the VPN tunnel on the AWS side.

Step 11. Repeat Steps 1-10 for the second customer gateway.



VPN connections (1/2) Info

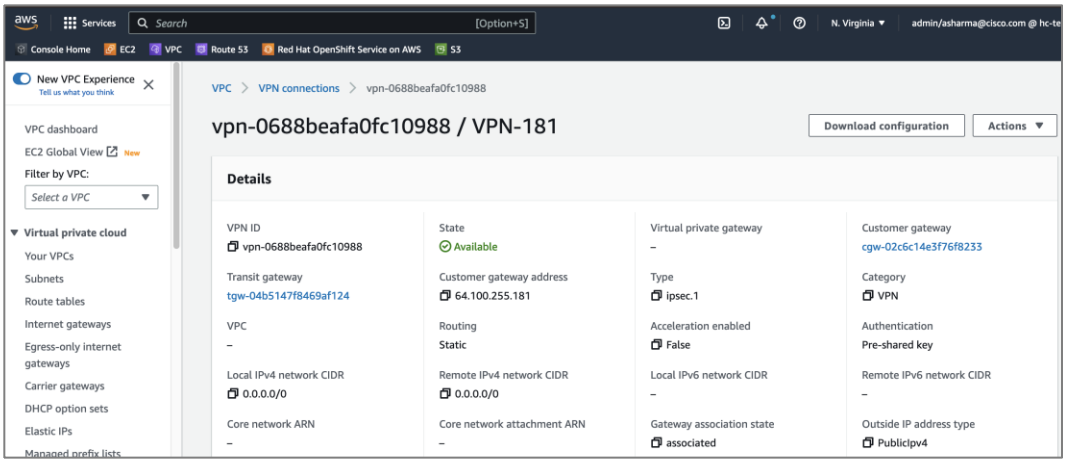
State: available

	Name	VPN ID	State	Transit gateway	Customer gateway	Customer gateway
○	VPN-181	vpn-0688beafa0fc10988	Available	tgw-04b5147f8469af124	cgw-02c6c14e3f76f8233	64.100.255.181
●	VPN-182	vpn-02363cbefd8b34ee8	Available	tgw-04b5147f8469af124	cgw-075ffacada4f845d4	64.100.255.182

Procedure 6. Download site-to-site VPN configuration for on-prem device

Step 1. From **VPC > Site-to-Site VPN**, select ad click the newly deployed VPN configuration for the first customer gateway.

Step 2. From the configuration for the VPN, click on **Download Configuration**.

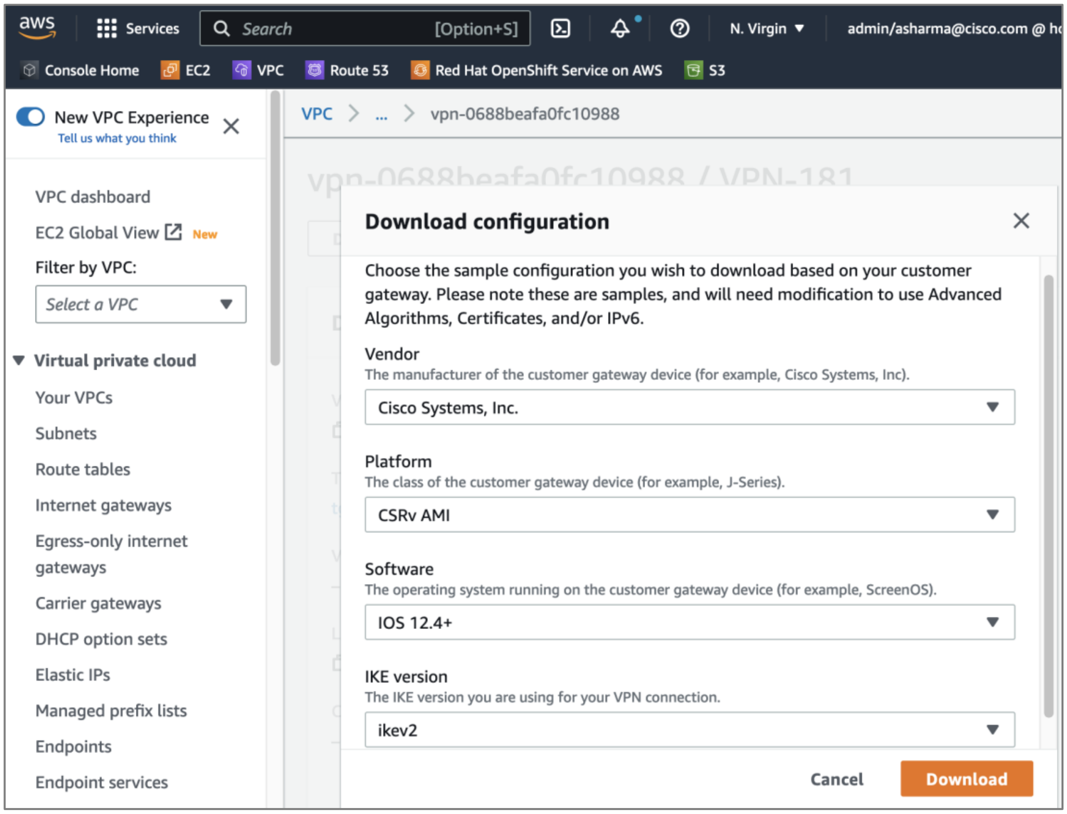


vpn-0688beafa0fc10988 / VPN-181

Details

VPN ID vpn-0688beafa0fc10988	State Available	Virtual private gateway –	Customer gateway cgw-02c6c14e3f76f8233
Transit gateway tgw-04b5147f8469af124	Customer gateway address 64.100.255.181	Type ipsec.1	Category VPN
VPC –	Routing Static	Acceleration enabled False	Authentication Pre-shared key
Local IPv4 network CIDR 0.0.0.0/0	Remote IPv4 network CIDR 0.0.0.0/0	Local IPv6 network CIDR –	Remote IPv6 network CIDR –
Core network ARN –	Core network attachment ARN –	Gateway association state associated	Outside IP address type PublicIpv4

Step 3. In the **Download configuration** pop-up window, for **Vendor**, select **Cisco Systems, Inc**. For **Platform**, select **CSRv AMI**. For **Software**, select **12.4+.** For **IKE** version, select **ikev2.**



Download configuration

Choose the sample configuration you wish to download based on your customer gateway. Please note these are samples, and will need modification to use Advanced Algorithms, Certificates, and/or IPv6.

Vendor
The manufacturer of the customer gateway device (for example, Cisco Systems, Inc).
Cisco Systems, Inc.

Platform
The class of the customer gateway device (for example, J-Series).
CSRv AMI

Software
The operating system running on the customer gateway device (for example, ScreenOS).
IOS 12.4+

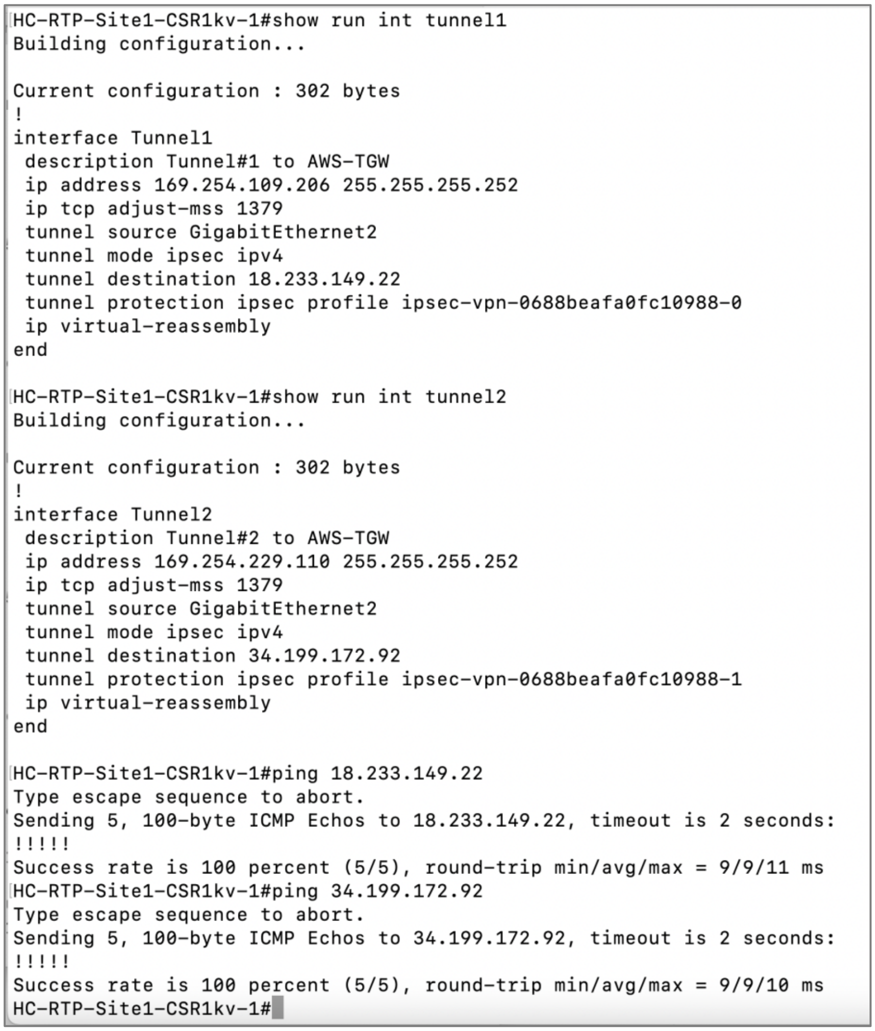
IKE version
The IKE version you are using for your VPN connection.
ikev2

Step 4. Click **Download.**

Step 5. Repeat steps 1-4 for the VPN connection to the second customer gateway.

Step 6. Edit the downloaded configuration files and upload it to the on-prem VPN devices. The VPN devices are CSR1000V in this design. The edited configuration is provided in the Appendix section of this document.

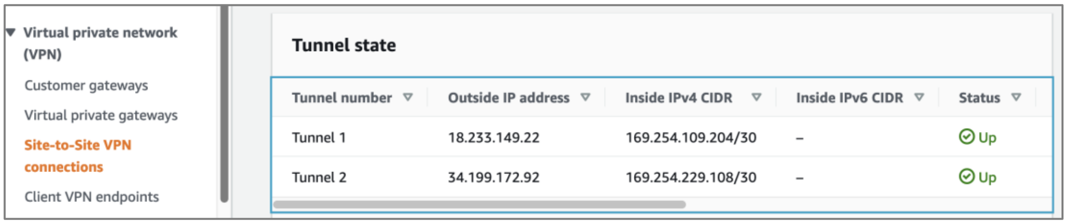
Step 7. Enterprises must allow inbound reachability from the public IP of the AWS VPN tunnel to the public IP (if NAT is used, then to the private IP of the customer gateways. If NAT is used, then to the private IP of the customer gateways. This is the minimum configuration necessary to bring the IPsec VPN tunnel up. Verify that this connectivity is in place by initiating a ping from the customer gateways within the Enterprise to the public IP of the tunnel in AWS. Verify for both customer gateways – configuration and verification for customer gateway-1 is shown below.



```
HC-RTP-Site1-CSR1kv-1#show run int tunnel1
Building configuration...

Current configuration : 302 bytes
!
interface Tunnel1
 description Tunnel#1 to AWS-TGW
 ip address 169.254.109.206 255.255.255.252
 ip tcp adjust-mss 1379
 tunnel source GigabitEthernet2
 tunnel mode ipsec ipv4
 tunnel destination 18.233.149.22
 tunnel protection ipsec profile ipsec-vpn-0688beafa0fc10988-0
 ip virtual-reassembly
end

HC-RTP-Site1-CSR1kv-1#show run int tunnel2
Building configuration...

Current configuration : 302 bytes
!
interface Tunnel2
 description Tunnel#2 to AWS-TGW
 ip address 169.254.229.110 255.255.255.252
 ip tcp adjust-mss 1379
 tunnel source GigabitEthernet2
 tunnel mode ipsec ipv4
 tunnel destination 34.199.172.92
 tunnel protection ipsec profile ipsec-vpn-0688beafa0fc10988-1
 ip virtual-reassembly
end

HC-RTP-Site1-CSR1kv-1#ping 18.233.149.22
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 18.233.149.22, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 9/9/11 ms
HC-RTP-Site1-CSR1kv-1#ping 34.199.172.92
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 34.199.172.92, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 9/9/10 ms
HC-RTP-Site1-CSR1kv-1#
```

Step 8. Verify the site-to-site VPN tunnels are up on both locations as shown below.

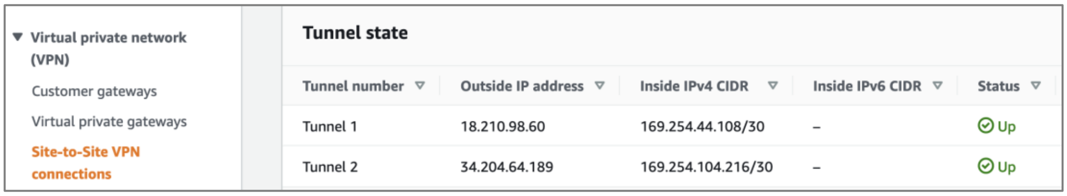
Note: Status of Site-to-Site VPN to Customer Gateway-1



Tunnel state

Tunnel number	Outside IP address	Inside IPv4 CIDR	Inside IPv6 CIDR	Status
Tunnel 1	18.233.149.22	169.254.109.204/30	–	✓ Up
Tunnel 2	34.199.172.92	169.254.229.108/30	–	✓ Up

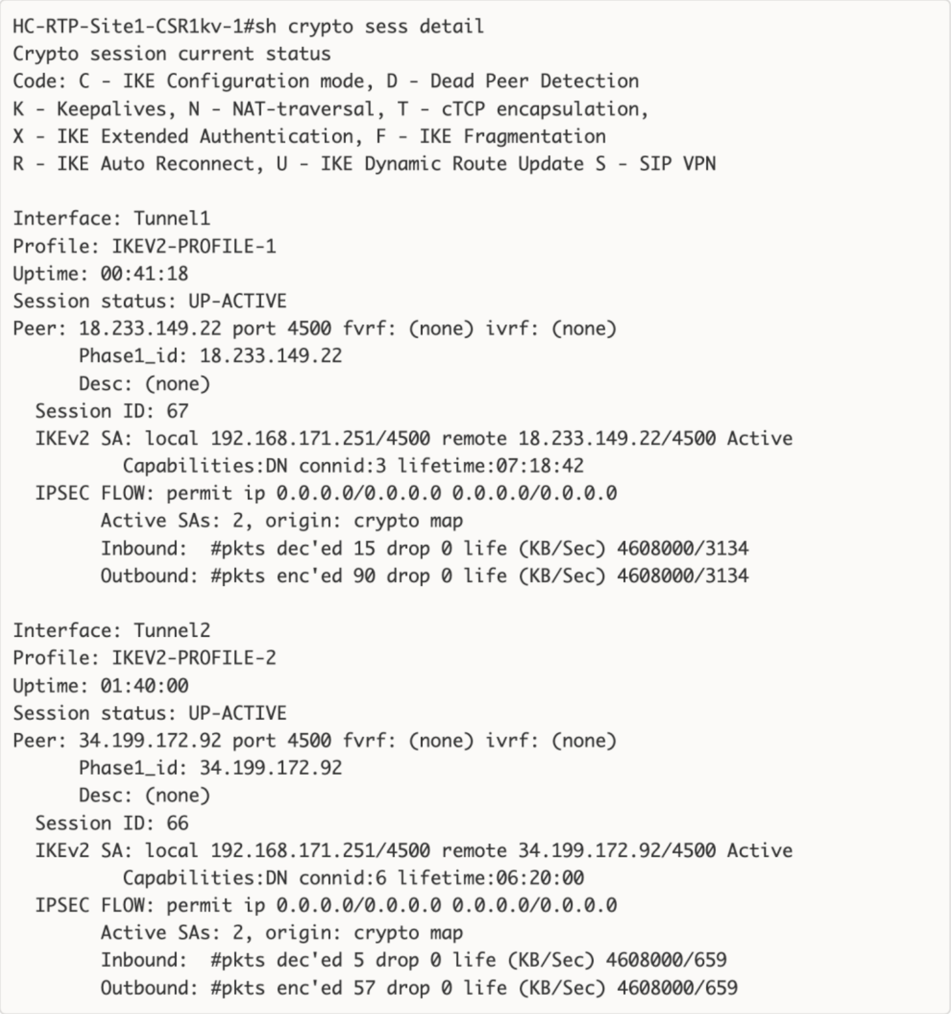
Note: Status of Site-to-Site VPN to Customer Gateway-2



Tunnel state

Tunnel number	Outside IP address	Inside IPv4 CIDR	Inside IPv6 CIDR	Status
Tunnel 1	18.210.98.60	169.254.44.108/30	–	✓ Up
Tunnel 2	34.204.64.189	169.254.104.216/30	–	✓ Up

Note: Enterprise side Status of Site-to-Site VPN on VPN Gateway-1



```
HC-RTP-Site1-CSR1kv-1#sh crypto sess detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation,
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect, U - IKE Dynamic Route Update S - SIP VPN

Interface: Tunnel1
Profile: IKEV2-PROFILE-1
Uptime: 00:41:18
Session status: UP-ACTIVE
Peer: 18.233.149.22 port 4500 fvrf: (none) ivrf: (none)
      Phase1_id: 18.233.149.22
      Desc: (none)
  Session ID: 67
  IKEv2 SA: local 192.168.171.251/4500 remote 18.233.149.22/4500 Active
          Capabilities:DN connid:3 lifetime:07:18:42
  IPSEC FLOW: permit ip 0.0.0.0/0.0.0.0 0.0.0.0/0.0.0.0
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 15 drop 0 life (KB/Sec) 4608000/3134
        Outbound: #pkts enc'ed 90 drop 0 life (KB/Sec) 4608000/3134

Interface: Tunnel2
Profile: IKEV2-PROFILE-2
Uptime: 01:40:00
Session status: UP-ACTIVE
Peer: 34.199.172.92 port 4500 fvrf: (none) ivrf: (none)
      Phase1_id: 34.199.172.92
      Desc: (none)
  Session ID: 66
  IKEv2 SA: local 192.168.171.251/4500 remote 34.199.172.92/4500 Active
          Capabilities:DN connid:6 lifetime:06:20:00
  IPSEC FLOW: permit ip 0.0.0.0/0.0.0.0 0.0.0.0/0.0.0.0
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 5 drop 0 life (KB/Sec) 4608000/659
        Outbound: #pkts enc'ed 57 drop 0 life (KB/Sec) 4608000/659
```

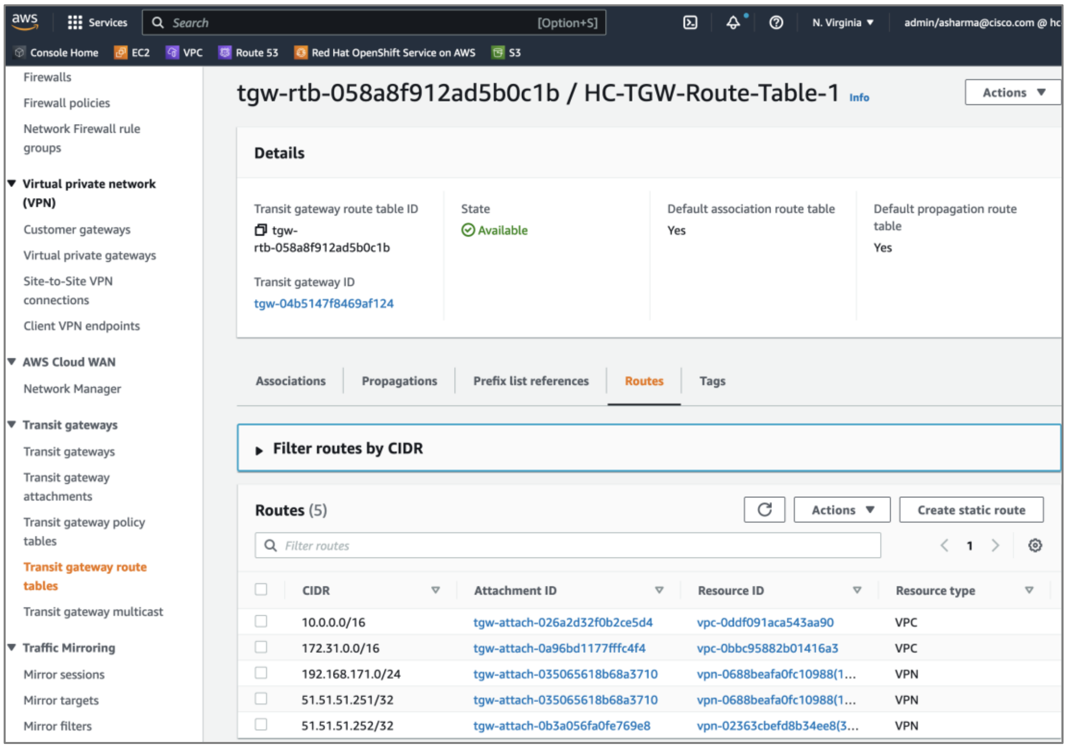
Note: Enterprise side Status of Site-to-Site VPN on VPN Gateway-2

```
HC-RTP-Site1-CSR1kv-2#show crypto sess detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect, U - IKE Dynamic Route Update, S - SIP VPN

Interface: Tunnel1
Profile: IKEV2-PROFILE-1
Uptime: 00:43:31
Session status: UP-ACTIVE
Peer: 18.210.98.60 port 4500 fvrf: (none) ivrf: (none)
      Phase1_id: 18.210.98.60
      Desc: (none)
  Session ID: 28
  IKEv2 SA: local 192.168.171.252/4500 remote 18.210.98.60/4500 Active
          Capabilities:DN connid:2 lifetime:07:16:29
  IPSEC FLOW: permit ip 0.0.0.0/0.0.0.0 0.0.0.0/0.0.0.0
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 10 drop 0 life (KB/Sec) 4607998/988
        Outbound: #pkts enc'ed 10 drop 0 life (KB/Sec) 4607999/988

Interface: Tunnel2
Profile: IKEV2-PROFILE-2
Uptime: 00:44:20
Session status: UP-ACTIVE
Peer: 34.204.64.189 port 4500 fvrf: (none) ivrf: (none)
      Phase1_id: 34.204.64.189
      Desc: (none)
  Session ID: 30
  IKEv2 SA: local 192.168.171.252/4500 remote 34.204.64.189/4500 Active
          Capabilities:DN connid:1 lifetime:07:15:40
  IPSEC FLOW: permit ip 0.0.0.0/0.0.0.0 0.0.0.0/0.0.0.0
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 6 drop 0 life (KB/Sec) 4607999/939
        Outbound: #pkts enc'ed 6 drop 0 life (KB/Sec) 4607999/939
```

Step 9. Verify you have IP reachability between on-prem endpoints and AWS VPC endpoint. To verify this quickly, a temporary loopback IP and address is configured on the Enterprise gateways, and then a static route to the /32 IP of these loopbacks are added in the route table of the AWS Transit Gateway. Now you should be able to ping from the Loopback0 IP to any of the cluster nodes in AWS.

```
HC-RTP-Site1-CSR1kv-1#show run int loop0
Building configuration...

Current configuration : 98 bytes
!
interface Loopback0
 description TEST-AWS-INTERFACE
 ip address 51.51.51.251 255.255.255.0
end

HC-RTP-Site1-CSR1kv-1#
```

```
HC-RTP-Site1-CSR1kv-2#show run int loop0
Building configuration...

Current configuration : 98 bytes
!
interface Loopback0
 description TEST-AWS-INTERFACE
 ip address 51.51.51.252 255.255.255.0
end

HC-RTP-Site1-CSR1kv-2#
```



tgw-rtb-058a8f912ad5b0c1b / HC-TGW-Route-Table-1

Details

Transit gateway route table ID: tgw-rtb-058a8f912ad5b0c1b

State: Available

Default association route table: Yes

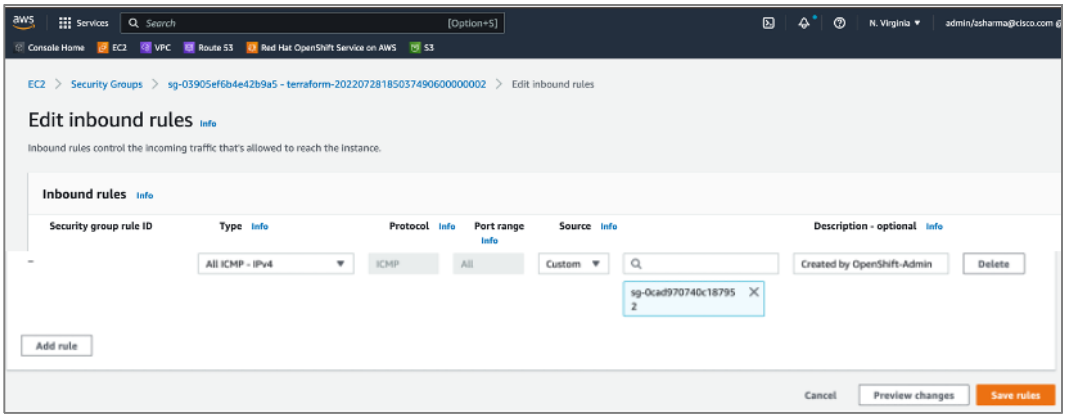
Default propagation route table: Yes

Transit gateway ID: tgw-04b5147f8469af124

Routes (5)

CIDR	Attachment ID	Resource ID	Resource type
10.0.0.0/16	tgw-attach-026a2d32f0b2ce5d4	vpc-0ddf091aca543aa90	VPC
172.31.0.0/16	tgw-attach-0a96bd1177fffc4f4	vpc-0bbc95882b01416a3	VPC
192.168.171.0/24	tgw-attach-035065618b68a3710	vpn-0688beafa0fc10988(1...	VPN
51.51.51.251/32	tgw-attach-035065618b68a3710	vpn-0688beafa0fc10988(1...	VPN
51.51.51.252/32	tgw-attach-0b3a056fa0fe769e8	vpn-02363cbefd8b34ee8(3...	VPN

You also need to allow ICMP traffic from the Loopback IPs by adding a security group rule in the security group for master nodes/worker nodes or other endpoints that you want to allow ping access to as shown below.



EC2 > Security Groups > sg-03905ef6b4e42b9a5 - terraform-20220728185037490600000002 > Edit inbound rules

Edit inbound rules

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
-	All ICMP - IPv4	ICMP	All	Custom	sg-0cad970740c18795 2	Created by OpenShift-Admin

Step 10. You can now add additional routes as needed for reachability across this site-to-site VPN for applications, services, and other networks across this hybrid deployment.

Enable Cisco Intersight Workload Optimizer (On-Prem)

This section describes the deployment of a resource optimization management tool for a cloud-native environment in an on-prem Enterprise data center. The solution uses Cisco Intersight Workload Optimizer (IWO), a service on Cisco Intersight, to ensure application performance and optimize resource usage.

The procedures outlined in this section will deploy Cisco IWO to monitor and optimize resource usage on a Red Hat OCP cluster running on an Application HyperFlex VSI cluster.

Prerequisites

- A valid cisco.com account to download IWO collector.
- OCP Installer or some other workstation to deploy the Cisco IWO collector. The workstation should be able to execute CLI commands on the Red Hat OCP cluster.
- Helm v2 (requires Tiller) or Helm v3 to install the collector in the OCP cluster.
- IWO collector Pods (two identical pods are deployed for high availability) require Kubelet access to every node in the cluster. Default: HTTPs + port=10250
- TCP port 80 and TCP/UDP port 443 must be open for collector to securely connect with Cisco Intersight

Setup Information

Table 14 lists the installation parameters for the IWO Collector.

Table 14. IWO Collector - Installation Parameters

Variable	Variable Name	Value	Additional Info
OCP Project or Namespace	namespace	iwo-collector	
IWO Collector - name	name	iwo-collector-pod	
IWO Collector Version	iwoServerVersion	8.5	
IWO Collector Image Tag	collectoryImage.tag	8.5.6	
IWO Name of Target cluster	targetName	ocp11-OnPrem.hc.com	Doesn't need to be same as OCP cluster name but it should be unique within IWO

Deployment Steps

The procedures in this section will deploy Cisco IWO to monitor and optimize resource usage on a Red Hat OCP cluster running on an Application HyperFlex VSI cluster.

Procedure 1. Deploy Cisco IWO collector on the Red Hat OCP cluster

Step 1. Navigate using a web browser to Cisco Software Downloads page on cisco.com. Login using your Cisco account.

Step 2. In the **Select a Product** search area, enter **Intersight**.

Step 3. Select and click **Intersight** under **Cloud and Systems Management**.

Step 4. Select and click **Intersight Kubernetes Collector**.

Step 5. Download the latest **Kubernetes Collector for Cisco Intersight Workload Optimizer** to the on-prem OCP Installer (or any workstation that can execute CLI commands on the OCP cluster).

Step 6. From the OCP installer, verify you can access the cluster and create a separate Red Hat OCP project/namespace (**iwo-collector**) for IWO collector. Export kubeconfig if needed, for example: **export KUBECONFIG=ocp11/auth/kubeconfig**

```
[administrator@ocp-installer HC-OpenShift]$ oc create namespace iwo-collector
namespace/iwo-collector created
```

Step 7. Extract the collector image downloaded from cisco.com. For example: **tar -xvf iwo-k8s-collector-v1.2.0.tgz**

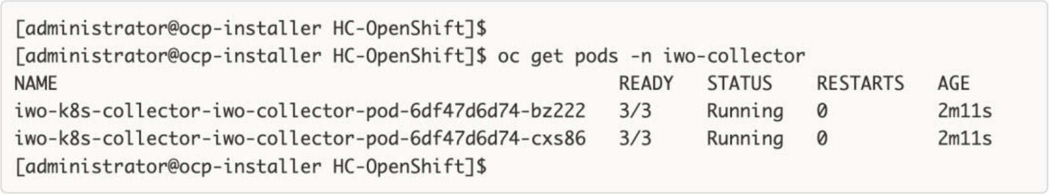
Step 8. Do a dry-run of the collector install before deploying it. For Helm v3, execute the following command for a dry run:

```
helm install --dry-run --debug iwo-collector-pod iwo-k8s-collector  --namespace
iwo-collector --set iwoServerVersion=8.5 --set collectorImage.tag=8.5.6 --set
targetName=ocp11-OnPrem.hc.com
```

Step 9. Install the collector if the dry-run is successful. For Helm v3, execute the following command to install:

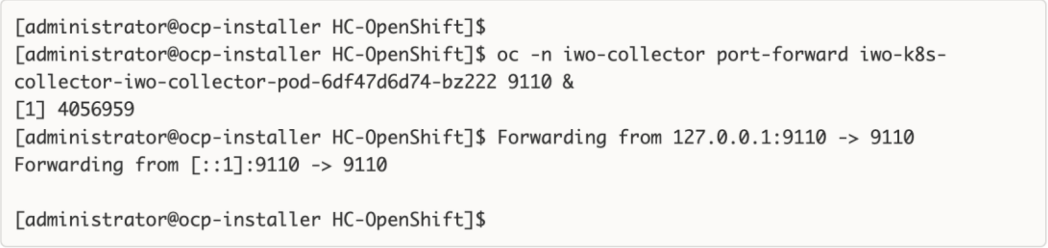
```
helm install --debug iwo-collector-pod iwo-k8s-collector  --namespace iwo-
collector --set iwoServerVersion=8.5 --set collectorImage.tag=8.5.6 --set
targetName=ocp11-Onprem.hc.com
```

Step 10. Verify IWO collector Pods are up and running. Note one of the Pod names for use in next step.



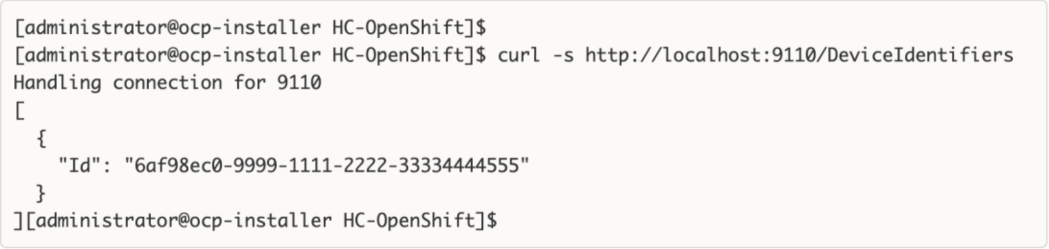
```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ oc get pods -n iwo-collector
NAME                                              READY   STATUS    RESTARTS   AGE
iwo-k8s-collector-iwo-collector-pod-6df47d6d74-bz222   3/3     Running   0          2m11s
iwo-k8s-collector-iwo-collector-pod-6df47d6d74-cxs86   3/3     Running   0          2m11s
[administrator@ocp-installer HC-OpenShift]$
```

Step 11. Configure port forwarding as follows using one of the pods from previous step.



```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ oc -n iwo-collector port-forward iwo-k8s-
collector-iwo-collector-pod-6df47d6d74-bz222 9110 &
[1] 4056959
[administrator@ocp-installer HC-OpenShift]$ Forwarding from 127.0.0.1:9110 -> 9110
Forwarding from [::1]:9110 -> 9110

[administrator@ocp-installer HC-OpenShift]$
```

Step 12. Collect the **Device ID** using the following command: **curl -s http://localhost:9110/DeviceIdentifiers**



```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ curl -s http://localhost:9110/DeviceIdentifiers
Handling connection for 9110
[
  {
    "Id": "6af98ec0-9999-1111-2222-33334444555"
  }
][administrator@ocp-installer HC-OpenShift]$
```

Step 13. Collect the **Claim Code** using the following command: **curl -s http://localhost:9110/SecurityTokens**

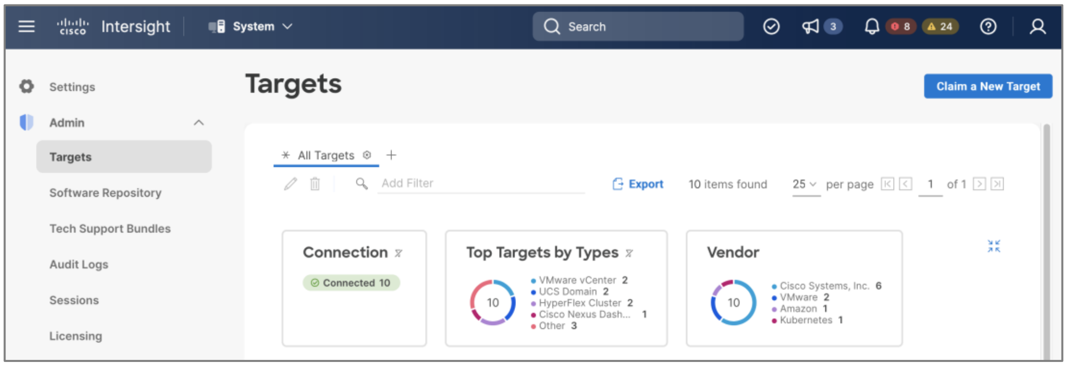
```
[administrator@ocp-installer HC-OpenShift]$
[administrator@ocp-installer HC-OpenShift]$ curl -s http://localhost:9110/SecurityTokens
Handling connection for 9110
[
  {
    "Token": "5D051A7BCCFA",
    "Duration": 599
  }
][administrator@ocp-installer HC-OpenShift]$
```

Procedure 2. Claim the on-prem IWO collector as a target in Intersight

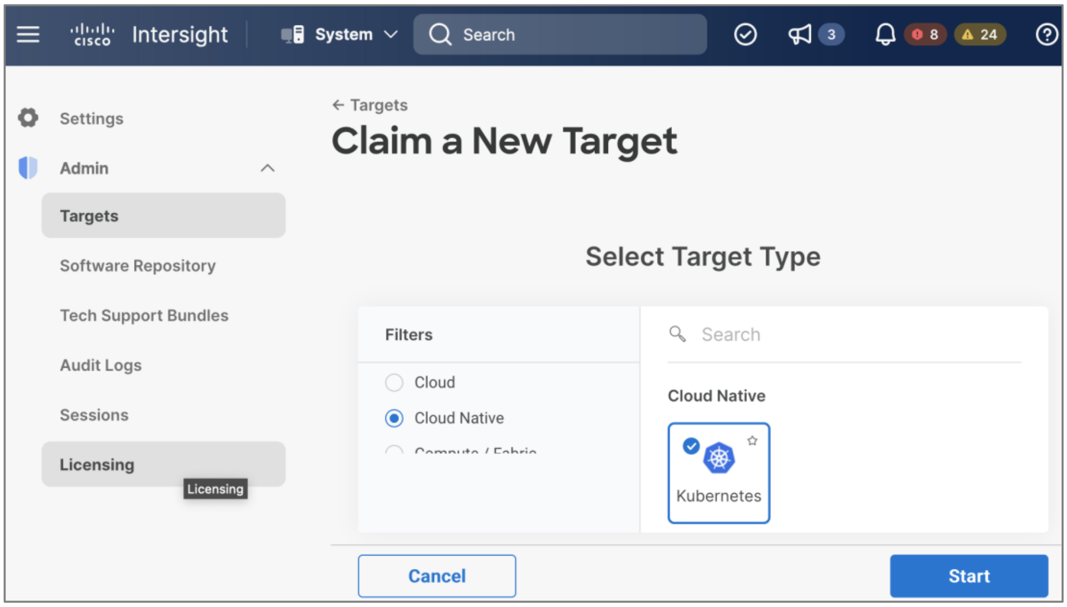
Step 1. Navigate to **intersight.com** and login using your account.

Step 2. Select **System** from the drop-down list in the top left side of the window.

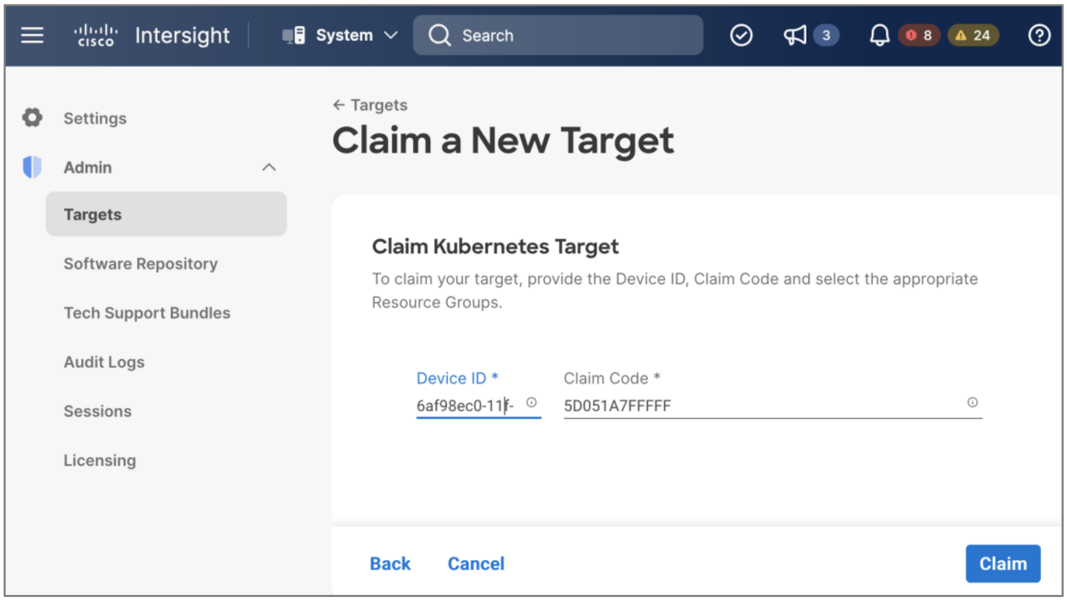
Step 3. Navigate to **Targets** and click on **Claim a New Target**.



Step 4. In the **Claim a New Target** window, navigate to **Filters** and select **Cloud Native**. Select **Kubernetes** and click **Start**.



Step 5. Provide the **Device ID** and **Claim Code** collected from the IWO collector earlier.



Claim a New Target

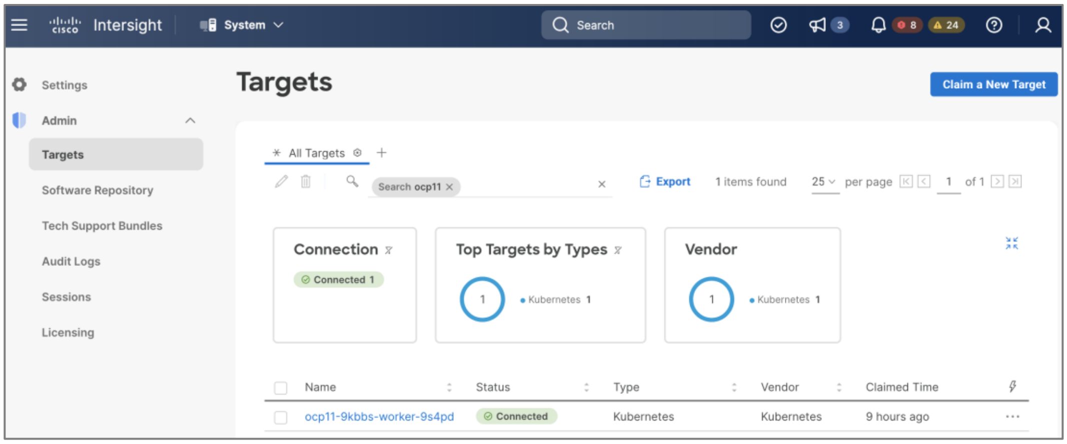
Claim Kubernetes Target

To claim your target, provide the Device ID, Claim Code and select the appropriate Resource Groups.

Device ID * 6af98ec0-11f-

Claim Code * 5D051A7FFFFF

Step 6. Click **Claim.** Verify the on-prem collector for the OCP cluster is claimed in Cisco Intersight.



Targets

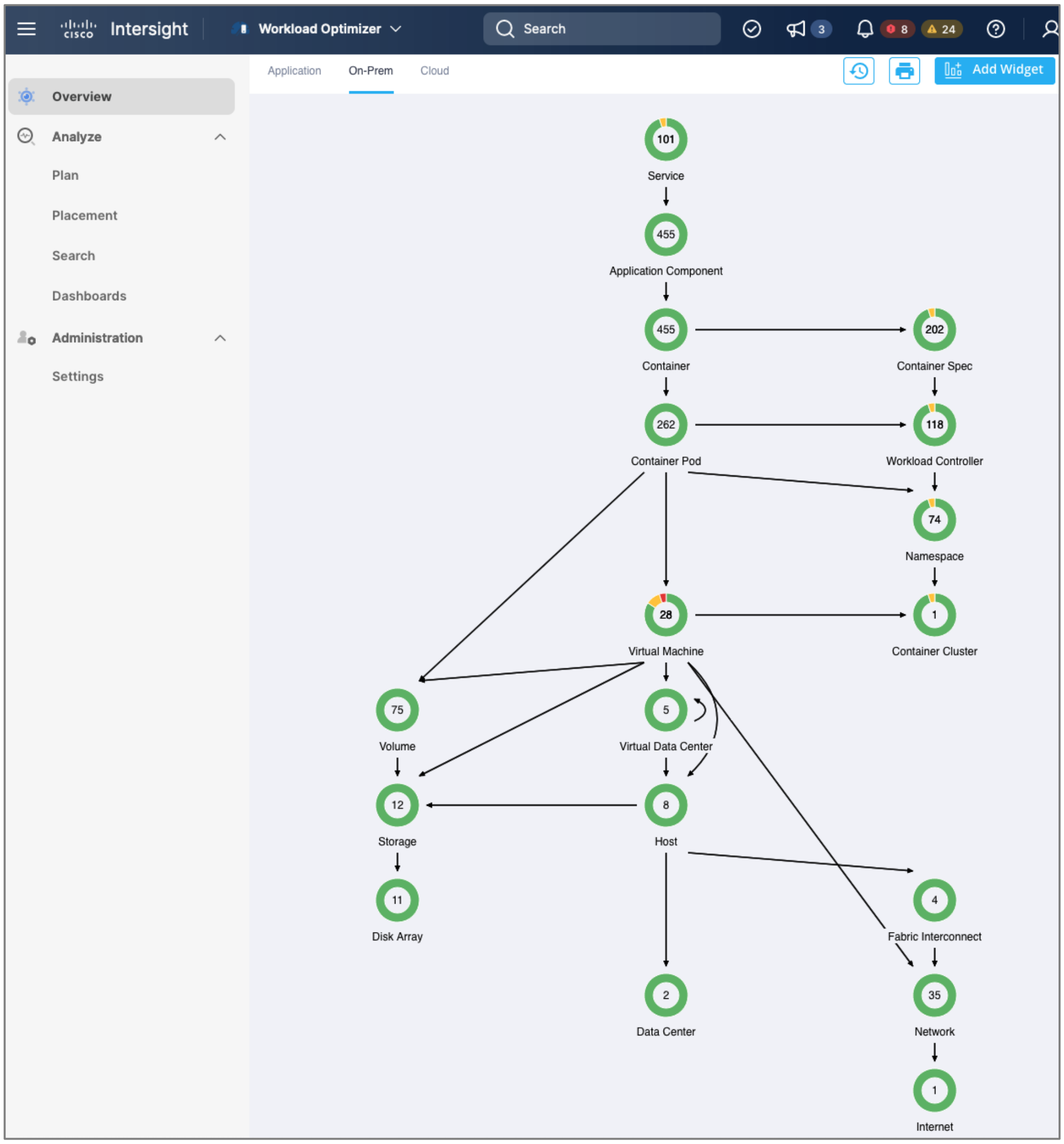
Name	Status	Type	Vendor	Claimed Time
ocp11-9kbbs-worker-9s4pd	Connected	Kubernetes	Kubernetes	9 hours ago

Procedure 3. Verify Cisco IWO is monitoring resources on the on-prem infrastructure

Step 1. From **intersight.com**, select **Workload Optimizer** from the drop-down list at the top left side of the window.

Step 2. Navigate to **Overview** and select the **On-Prem tab** from the right-side window.

Step 3. Verify that you are seeing everything from Container, Application and Service Components to underlying host, storage and other infrastructure in the dependency mapping created by IWO.



Step 4. The Cisco IWO environment is now ready for use by the administrator to monitor and implement (manually or automated) policies to ensure application performance on the on-prem Red Hat OCP cluster.

Enable Cisco Intersight Workload Optimizer (Public Cloud)

This section describes the deployment of a resource optimization tool for managing cloud-native resources in AWS. The solution uses Cisco Intersight Workload Optimizer, a Cisco Intersight service, to ensure application performance and manage cloud costs.

The procedures outlined in this section will deploy Cisco Intersight Workload Optimizer on AWS EC2 instances hosting a Red Hat OCP cluster.

Prerequisites

- A valid cisco.com account to download IWO collector.
- OCP Installer in AWS or some other workstation to deploy the Cisco IWO collector. The workstation should be able to execute CLI commands on the Red Hat OCP cluster.
- Helm v2 (requires Tiller) or Helm v3 to install the collector in the OCP cluster.

- IWO collector Pods (two identical pods are deployed for high availability) require Kubelet access to every node in the OCP cluster. Default: HTTPs + port=10250
- Collector should be able to connect to Cisco Intersight through the Internet.

Setup Information

Table 15 lists the installation parameters for the IWO collector.

Table 15. IWO Collector - Installation Parameters

Variable	Variable Name	Value	Additional Info
OCP Project or Namespace	namespace	iwo-collector	
IWO Collector - name	name	iwo-collector-pod	
IWO Collector Version	iwoServerVersion	8.5	
IWO Collector Image Tag	collectoryImage.tag	8.5.6	
IWO Name of Target cluster	targetName	ocp11-AWS.hc.com	Doesn't need to be same as OCP cluster name but it should be unique within IWO

Deployment Steps

The procedures in this section will deploy Cisco Intersight Workload Optimizer on AWS EC2 instances hosting a Red Hat OCP cluster.

Procedure 1. Deploy Cisco IWO collector on Red Hat OCP cluster

Step 1. Follow Procedure 1 steps in the on-prem Cisco IWO deployment section to deploy the collector in AWS OCP cluster using the AWS setup information provided earlier.

Procedure 2. Claim Cisco IWO collector as a target in Intersight

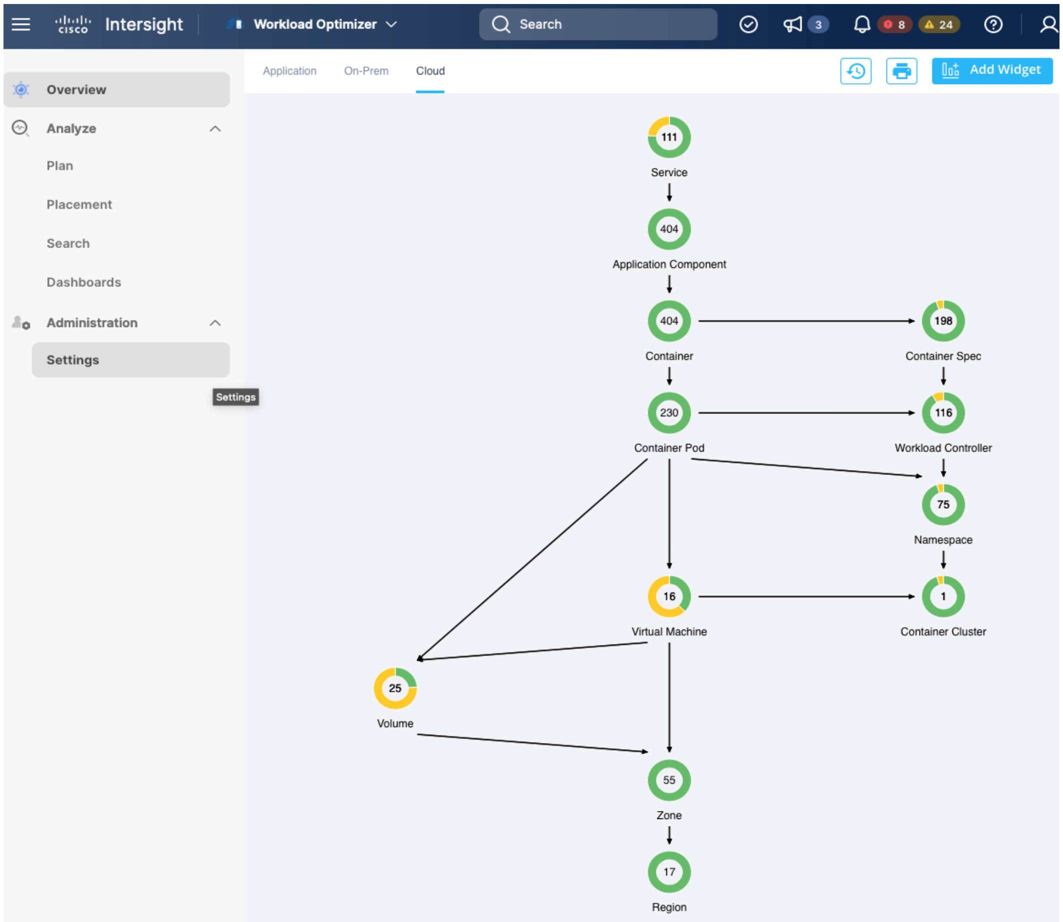
Step 1. Follow Procedure 2 steps in the on-prem Cisco IWO deployment section to claim the collector as a target in Intersight.

Procedure 3. Verify Cisco IWO is monitoring resources in AWS

Step 2. From **intersight.com**, select **Workload Optimizer** from the drop-down list at the top left side of the window.

Step 3. Navigate to **Overview** and select the **Cloud tab** from the right-side window.

Step 4. Verify that you are seeing the Container, Application and Service Components and the underlying AWS resources in the dependency mapping created by IWO.



Step 5. The Cisco IWO environment is now ready for use by the administrator to monitor and implement (manually or automated) policies to ensure application performance on the Red Hat OCP cluster in AWS.

Conclusion

Hybrid cloud is the de facto operating model in most Enterprises today. Cisco HyperFlex with Red Hat OCP solution provides a flexible foundational hybrid cloud architecture that Enterprises can adopt and standardize on from enterprise edge to core data centers. The solution offers an enterprise-grade Kubernetes environment for an Enterprise’s cloud native efforts platform with Enterprise-level support from Cisco and Red Hat. Cisco HyperFlex is an essential building block that provides the fastest path to hybrid cloud with enterprise-grade, software-defined compute, and storage infrastructure. Coupled with Cisco Intersight, HyperFlex can be deployed and managed with simplicity and ease across all Enterprise locations, around the globe. Cisco Intersight can deliver a production-ready Virtual Server Infrastructure (VSI) in less than an hour using a fully automated deployment process. The SaaS model enables you to install, deploy, monitor, and maintain all of your clusters wherever they reside from a central portal. Intersight offers a comprehensive set of day-2 management capabilities that greatly simplifies and accelerates operations. It includes features such as cluster expansion, full-stack upgrades, day-2 storage management with performance monitoring, connected TAC support, hardware compatibility checks and tools for capacity planning and cluster health checks. Red Hat OpenShift Container Platform provides an enterprise-grade Kubernetes platform with consistent management and development experience across a hybrid environment for both operations and development teams. Cisco Intersight Workload Optimizer, a service in Cisco Intersight, can continuously monitor and optimizes resources across the stack to lower cloud costs and ensure application performance across hybrid cloud environment.

About the Authors

Archana Sharma, Technical Leader, Cisco UCS Data Center Solutions, Cisco Systems Inc.

Archana Sharma is Technical Marketing Engineer with over 20 years of experience at Cisco on a range of technologies that span Data Center, Desktop Virtualization, Collaboration, and other Layer2 and Layer3 technologies. Archana is focused on systems and solutions for Enterprise and Provider deployments, including delivery of Cisco Validated designs for over 10 years. Archana is currently working on designing and integrating Cisco UCS-based Converged Infrastructure solutions. Archana holds a CCIE (#3080) in Routing and Switching and a bachelor's degree in Electrical Engineering from North Carolina State University.

Appendices

This appendix is organized into the following sections:

- Appendix A – On-Prem Site-to-Site VPN Configuration
- Appendix B – References Used in Guide
- Appendix C – Glossary
- Appendix D – Acronyms
- Appendix E – Recommended for You

Appendix A – On-Prem Site-to-Site VPN Configuration

Customer Gateway – 1 (On-Prem)

```
HC-RTP-Site1-CSR1kv-1#show run
Building configuration...

Current configuration : 12630 bytes
!
! Last configuration change at 21:06:20 UTC Thu Jul 28 2022 by admin
!
version 17.3
service timestamps debug datetime msec
service timestamps log datetime msec
service call-home
platform qfp utilization monitor load 80
platform punt-keepalive disable-kernel-core
platform console virtual
platform hardware throughput level MB 5000
!
hostname HC-RTP-Site1-CSR1kv-1
!
boot-start-marker
boot-end-marker
!
enable secret 9 $9$85NtD8GXl77uIk$BNe35miWi7qFO72Nb0mknPjZBMJVng/h.jQlIt4ug6g
!
no aaa new-model
!
ip domain lookup source-interface GigabitEthernet1
ip domain name cspg.local
!
login on-success log
!
subscriber templating
!
multilink bundle-name authenticated
!
```

```
crypto pki trustpoint TP-self-signed-2015064199
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-2015064199
 revocation-check none
 rsakeypair TP-self-signed-2015064199
!
crypto pki trustpoint SLA-TrustPoint
 enrollment terminal
 revocation-check crl
!
crypto pki certificate chain TP-self-signed-2015064199
 certificate self-signed 01
  <!! REMOVED !!>
        quit
crypto pki certificate chain SLA-TrustPoint
 certificate ca 01
   <!! REMOVED !!>
        quit
!
crypto pki certificate pool
 cabundle nvram:ios_core.p7b
!
license udi pid CSR1000V sn 9CCCQFFFGV
license boot level ax
diagnostic bootup level minimal
memory free low-watermark processor 71489
!
spanning-tree extend system-id
!
username admin privilege 15 secret 9
$9$g8936Xyf7Fy64E$QdjuY/IIN6RXBRvc40NMLc6Bd5uXP36LhdalczF4Y4g
!
redundancy
!
crypto ikev2 proposal PROPOSAL1
 encryption aes-cbc-128 aes-cbc-192 aes-cbc-256
 integrity sha1 sha512 sha384 sha256
 group 24 21 20 19 16 15 14 2
!
crypto ikev2 policy POLICY1
 match address local 192.168.171.251
 proposal PROPOSAL1
!
crypto ikev2 keyring KEYRING1
 peer 18.233.149.22
  address 18.233.149.22
  pre-shared-key DPd4oIk8bflTRRqZXh6xjShL8.AyLfwr
 !
!
crypto ikev2 keyring KEYRING2
 peer 34.199.172.92
  address 34.199.172.92
  pre-shared-key T3XV4of7JGljGRmiCZMLdMyrq.uwNm8L
 !
crypto ikev2 profile IKEV2-PROFILE-1
 match address local 192.168.171.251
 match identity remote address 18.233.149.22 255.255.255.255
 identity local address 64.100.255.181
 authentication remote pre-share
 authentication local pre-share
 keyring local KEYRING1
 lifetime 28800
 dpd 10 10 on-demand
!
crypto ikev2 profile IKEV2-PROFILE-2
 match address local 192.168.171.251
 match identity remote address 34.199.172.92 255.255.255.255
 identity local address 64.100.255.181
 authentication remote pre-share
 authentication local pre-share
 keyring local KEYRING2
 lifetime 28800
 dpd 10 10 on-demand
!
lldp run
cdp run
!
crypto isakmp keepalive 10 10
!
crypto ipsec security-association replay window-size 128
!
crypto ipsec transform-set ipsec-prop-vpn-0688beafa0fc10988-0 esp-aes esp-sha256-hmac
 mode tunnel
crypto ipsec transform-set ipsec-prop-vpn-0688beafa0fc10988-1 esp-aes esp-sha256-hmac
 mode tunnel
crypto ipsec df-bit clear
!
crypto ipsec profile ipsec-vpn-0688beafa0fc10988-0
 set transform-set ipsec-prop-vpn-0688beafa0fc10988-0
 set pfs group14
 set ikev2-profile IKEV2-PROFILE-1
!
crypto ipsec profile ipsec-vpn-0688beafa0fc10988-1
 set transform-set ipsec-prop-vpn-0688beafa0fc10988-1
 set pfs group14
 set ikev2-profile IKEV2-PROFILE-2
!
interface Loopback0
 description TEST-AWS-INTERFACE
 ip address 51.51.51.251 255.255.255.0
!
interface Tunnel1
 description Tunnel#1 to AWS-TGW
 ip address 169.254.109.206 255.255.255.252
 ip tcp adjust-mss 1379
 tunnel source GigabitEthernet2
 tunnel mode ipsec ipv4
 tunnel destination 18.233.149.22
 tunnel protection ipsec profile ipsec-vpn-0688beafa0fc10988-0
 ip virtual-reassembly
!
interface Tunnel2
 description Tunnel#2 to AWS-TGW
 ip address 169.254.229.110 255.255.255.252
 ip tcp adjust-mss 1379
 tunnel source GigabitEthernet2
 tunnel mode ipsec ipv4
 tunnel destination 34.199.172.92
 tunnel protection ipsec profile ipsec-vpn-0688beafa0fc10988-1
 ip virtual-reassembly
!
interface GigabitEthernet1
 description OOB Mgmt
 ip address 172.26.163.251 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
 description To On-Prem Networks and source IPsec interface to AWS
 ip address 192.168.171.251 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
ip forward-protocol nd
ip http server
ip http authentication local
ip http secure-server
ip http client source-interface GigabitEthernet1
!
```

```
ip route 0.0.0.0 0.0.0.0 172.26.163.254
ip route 10.0.0.0 255.255.0.0 Tunnel1
ip route 10.0.0.0 255.255.0.0 Tunnel2
ip route 18.233.149.22 255.255.255.255 GigabitEthernet2 192.168.171.254
ip route 34.199.172.92 255.255.255.255 GigabitEthernet2 192.168.171.254
ip route 64.100.255.189 255.255.255.255 GigabitEthernet2 192.168.171.254
ip route 172.26.163.0 255.255.255.0 GigabitEthernet1 172.26.163.254
ip route 172.31.0.0 255.255.0.0 Tunnel1
ip route 172.31.0.0 255.255.0.0 Tunnel2
ip ssh rsa keypair-name ssh-key
ip ssh version 2
ip scp server enable
!
control-plane
!
line con 0
 exec-timeout 90 0
 stopbits 1
line vty 0 4
 exec-timeout 90 0
 login local
 transport input ssh
!
call-home
 ! If contact email address in call-home is configured as sch-smart-licensing@cisco.com
 ! the email address configured in Cisco Smart License Portal will be used as contact
 ! email address to send SCH notifications.
 contact-email-addr sch-smart-licensing@cisco.com
 profile "CiscoTAC-1"
  active
  destination transport-method http
!
end
HC-RTP-Site1-CSR1kv-1#
```

Customer Gateway - 2 (On-Prem)

```
HC-RTP-Site1-CSR1kv-2#show run
Building configuration...

Current configuration : 11938 bytes
!
! Last configuration change at 15:11:46 UTC Mon Jul 28 2022 by admin
!
version 17.3
service timestamps debug datetime msec
service timestamps log datetime msec
service call-home
```

```
platform qfp utilization monitor load 80
platform punt-keepalive disable-kernel-core
platform console virtual
platform hardware throughput level MB 5000
!
hostname HC-RTP-Site1-CSR1kv-2
!
boot-start-marker
boot-end-marker
!
enable secret 9 $9$Y8hVg5t/EgKMK.$NhS1soDgDKK4u5.ExrGUYI3hcPlceUYVBCoL7MYhCgM
!
no aaa new-model
!
ip domain lookup source-interface GigabitEthernet1
ip domain name cspg.local
!
login on-success log
!
subscriber templating
!
multilink bundle-name authenticated
!
crypto pki trustpoint TP-self-signed-1201348278
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-1201348278
 revocation-check none
 rsakeypair TP-self-signed-1201348278
!
crypto pki trustpoint SLA-TrustPoint
 enrollment terminal
 revocation-check crl
!
crypto pki certificate chain TP-self-signed-1201348278
 certificate self-signed 01
    <!! REMOVED !!>
        quit
crypto pki certificate chain SLA-TrustPoint
 certificate ca 01
  <!! REMOVED !!>
        quit
!
crypto pki certificate pool
 cabundle nvram:ios_core.p7b
!
license udi pid CSR1000V sn 9Q28ZTVBYYH
license boot level ax
```

```
diagnostic bootup level minimal
memory free low-watermark processor 71489
!
spanning-tree extend system-id
!
username admin privilege 15 secret 9
$9$5gt0bisfQriQVk$VDkvYrDTRJE2QkIJ0ci5vo5dfV2P561XjSI1wf4eX2A
!
redundancy
!
crypto ikev2 proposal PROPOSAL1
 encryption aes-cbc-128 aes-cbc-192 aes-cbc-256
 integrity sha1 sha512 sha384 sha256
 group 24 21 20 19 16 15 14 2
crypto ikev2 proposal ikev2-proposal-default
 encryption aes-cbc-256 aes-cbc-192 aes-cbc-128
 integrity sha512 sha384 sha256 sha1
 group 24 21 20 19 16 15 14 2
!
crypto ikev2 policy POLICY1
 match address local 192.168.171.252
 proposal PROPOSAL1
crypto ikev2 policy ikev2-policy-default
 proposal ikev2-proposal-default
!
crypto ikev2 keyring KEYRING1
 peer 18.210.98.60
  address 18.210.98.60
  pre-shared-key x_X0D_MOOrTSHQK.p84TNKmgQdv2PkxP
!
crypto ikev2 keyring KEYRING2
 peer 34.204.64.189
  address 34.204.64.189
  pre-shared-key sCpDlVSrqWHqSnZHUENYL5JWapQS0sJd
!
crypto ikev2 profile IKEV2-PROFILE-1
 match address local 192.168.171.252
 match identity remote address 18.210.98.60 255.255.255.255
 identity local address 64.100.255.182
 authentication remote pre-share
 authentication local pre-share
 keyring local KEYRING1
 lifetime 28800
 dpd 10 10 on-demand
!
crypto ikev2 profile IKEV2-PROFILE-2
 match address local 192.168.171.252
 match identity remote address 34.204.64.189 255.255.255.255
 identity local address 64.100.255.182
 authentication remote pre-share
 authentication local pre-share
 keyring local KEYRING2
 lifetime 28800
 dpd 10 10 on-demand
!
lldp run
cdp run
!
crypto isakmp keepalive 10 10
!
crypto ipsec security-association replay window-size 128
!
crypto ipsec transform-set ipsec-prop-vpn-02363cbefd8b34ee8-0 esp-aes esp-sha256-hmac
 mode tunnel
crypto ipsec transform-set ipsec-prop-vpn-02363cbefd8b34ee8-1 esp-aes esp-sha256-hmac
 mode tunnel
crypto ipsec df-bit clear
!
crypto ipsec profile ipsec-vpn-02363cbefd8b34ee8-0
 set transform-set ipsec-prop-vpn-02363cbefd8b34ee8-0
 set pfs group14
 set ikev2-profile IKEV2-PROFILE-1
!
crypto ipsec profile ipsec-vpn-02363cbefd8b34ee8-1
 set transform-set ipsec-prop-vpn-02363cbefd8b34ee8-1
 set pfs group14
 set ikev2-profile IKEV2-PROFILE-2
!
interface Loopback0
 description TEST-AWS-INTERFACE
 ip address 51.51.51.251 255.255.255.0
!
interface Tunnel1
 ip address 169.254.44.110 255.255.255.252
 ip tcp adjust-mss 1379
 tunnel source GigabitEthernet2
 tunnel mode ipsec ipv4
 tunnel destination 18.210.98.60
 tunnel protection ipsec profile ipsec-vpn-02363cbefd8b34ee8-0
 ip virtual-reassembly
!
interface Tunnel2
 ip address 169.254.104.218 255.255.255.252
 ip tcp adjust-mss 1379
 tunnel source GigabitEthernet2
 tunnel mode ipsec ipv4
 tunnel destination 34.204.64.189
 tunnel protection ipsec profile ipsec-vpn-02363cbefd8b34ee8-1
 ip virtual-reassembly
!
interface GigabitEthernet1
 description OOB Mgmt
 ip address 172.26.163.252 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
 description To On-Prem Networks and source IPsec interface to AWS
 ip address 192.168.171.252 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
ip forward-protocol nd
ip http server
ip http authentication local
ip http secure-server
ip http client source-interface GigabitEthernet1
!
ip route 0.0.0.0 0.0.0.0 172.26.163.254
ip route 10.0.0.0 255.255.255.0 Tunnel1
ip route 10.0.0.0 255.255.255.0 Tunnel2
ip route 18.210.98.60 255.255.255.255 GigabitEthernet2 192.168.171.254
ip route 34.204.64.189 255.255.255.255 GigabitEthernet2 192.168.171.254
ip route 64.100.255.189 255.255.255.255 GigabitEthernet2 192.168.171.254
ip route 172.26.163.0 255.255.255.0 GigabitEthernet1 172.26.163.254
ip ssh rsa keypair-name ssh-key
ip ssh version 2
ip scp server enable
!
control-plane
!
line con 0
 exec-timeout 90 0
 stopbits 1
line vty 0 4
 exec-timeout 90 0
 login local
 transport input ssh
!
```

```
call-home
 ! If contact email address in call-home is configured as sch-smart-licensing@cisco.com
 ! the email address configured in Cisco Smart License Portal will be used as contact
email address to send SCH notifications.
 contact-email-addr sch-smart-licensing@cisco.com
 profile "CiscoTAC-1"
  active
  destination transport-method http
!
end

HC-RTP-Site1-CSR1kv-2#
```

Appendix B – References Used in Guide

26 Kubernetes Statistics to Reference:
https://www.containiq.com/post/kubernetes-statistics

Red Hat OpenShift vs. Kubernetes:
https://www.redhat.com/en/topics/containers/red-hat-openshift-kubernetes

Cisco HyperFlex Data Sheets:
https://www.cisco.com/c/en/us/products/hyperconverged-infrastructure/hyperflex-hx-series/datasheet-listing.html

Cisco Intersight:
https://www.cisco.com/c/en/us/products/servers-unified-computing/intersight/index.html

Cisco Unified Computing System:
http://www.cisco.com/en/US/products/ps10265/index.html

Cisco UCS Manager:
http://www.cisco.com/en/US/products/ps10281/index.html

Red Hat OpenShift:
https://www.openshift.com/

Red Hat Hybrid Cloud Console:
https://access.redhat.com/products/red-hat-hybrid-cloud-console

Red Hat Ansible:
https://www.ansible.com/resources/get-started

Cisco HyperFlex Container Storage Interface (CSI):
https://catalog.redhat.com/software/operators/detail/615212f8b6d5b845070b7da0

Cisco IWO User Guide:
https://www.cisco.com/c/dam/en/us/td/docs/unified_computing/ucs/Intersight/Intersight_Workload_Optimizer/Cisco_Intersight_Workload_Optimizer_User_Guide.pdf

Cisco IWO Data Sheet:
https://www.cisco.com/c/en/us/products/collateral/cloud-systems-management/intersight-workload-optimizer/datasheet-c78-744509.html

Cisco IWO Documentation:
https://intersight.com/help/resources#cisco_intersight_workload_optimizer

Appendix C – Glossary

This glossary addresses some terms used in this document, for the purposes of aiding understanding. This is not a complete list of all multicloud terminology. Some Cisco product links are supplied here also, where considered useful for the purposes of clarity, but this is by no means intended to be a complete list of all applicable Cisco products.

aaS/XaaS **(IT capability provided as a Service)**	Some IT capability, X, provided as a service (XaaS). Some benefits are: • The provider manages the design, implementation, deployment, upgrades, resiliency, scalability, and overall delivery of the service and the infrastructure that supports it. • There are very low barriers to entry, so that services can be quickly adopted and dropped in response to business demand, without the penalty of inefficiently utilized CapEx. • The service charge is an IT OpEx cost (pay-as-you-go), whereas the CapEx and the service infrastructure is the responsibility of the provider. • Costs are commensurate to usage and hence more easily controlled with respect to business demand and outcomes. Such services are typically implemented as “microservices,” which are accessed via REST APIs. This architectural style supports composition of service components into systems. Access to and management of aaS assets is via a web GUI and/or APIs, such that Infrastructure-as-code (IaC) techniques can be used for automation, for example, Ansible and Terraform. The provider can be any entity capable of implementing an aaS “cloud-native” architecture. The cloud-native architecture concept is well-documented and supported by open-source software and a rich ecosystem of services such as training and consultancy. The provider can be an internal IT department or any of many third-party companies using and supporting the same open-source platforms. Service access control, integrated with corporate IAM, can be mapped to specific users and business activities, enabling consistent policy controls across services, wherever they are delivered from.
Ansible	An infrastructure automation tool, used to implement processes for instantiating and configuring IT service components, such as VMs on an IaaS platform. Supports the consistent execution of processes defined in YAML “playbooks” at scale, across multiple targets. Because the Ansible artefacts (playbooks) are text-based, they can be stored in a Source Code Management (SCM) system, such as GitHub. This allows for software development like processes to be applied to infrastructure automation, such as, Infrastructure-as-code (see IaC below). https://www.ansible.com
AWS **(Amazon Web Services)**	Provider of IaaS and PaaS. https://aws.amazon.com
Azure	Microsoft IaaS and PaaS. https://azure.microsoft.com/en-gb/
Co-located data center	“A colocation center (CoLo)...is a type of data center where equipment, space, and bandwidth are available for rental to retail customers. Colocation facilities provide space, power, cooling, and physical security for the server, storage, and networking equipment of other firms and also connect them to a variety of telecommunications and network service providers with a minimum of cost and complexity.”

	https://en.wikipedia.org/wiki/Colocation_centre

Containers **(Docker)**	A (Docker) container is a means to create a package of code for an application and its dependencies, such that the application can run on different platforms which support the Docker environment. In the context of aaS, microservices are typically packaged within Linux containers orchestrated by Kubernetes (K8s). https://www.docker.com https://www.cisco.com/c/en/us/products/cloud-systems-management/containerplatform/index.html
DevOps	The underlying principle of DevOps is that the application development and operations teams should work closely together, ideally within the context of a toolchain that automates the stages of development, test, deployment, monitoring, and issue handling. DevOps is closely aligned with IaC, continuous integration and deployment (CI/CD), and Agile software development practices. https://en.wikipedia.org/wiki/DevOps https://en.wikipedia.org/wiki/CI/CD
Edge compute	Edge compute is the idea that it can be more efficient to process data at the edge of a network, close to the endpoints that originate that data, or to provide virtualized access services, such as at the network edge. This could be for reasons related to low latency response, reduction of the amount of unprocessed data being transported, efficiency of resource utilization, and so on. The generic label for this is Multi-access Edge Computing (MEC), or Mobile Edge Computing for mobile networks specifically. From an application experience perspective, it is important to be able to utilize, at the edge, the same operations model, processes, and tools used for any other compute node in the system. https://en.wikipedia.org/wiki/Mobile_edge_computing
IaaS **(Infrastructure as-a-Service)**	Infrastructure components provided aaS, located in data centers operated by a provider, typically accessed over the public Internet. IaaS provides a base platform for the deployment of workloads, typically with containers and Kubernetes (K8s).
IaC **(Infrastructure as-Code)**	Given the ability to automate aaS via APIs, the implementation of the automation is typically via Python code, Ansible playbooks, and similar. These automation artefacts are programming code that define how the services are consumed. As such, they can be subject to the same code management and software development regimes as any other body of code. This means that infrastructure automation can be subject to all of the quality and consistency benefits, CI/CD, traceability, automated testing, compliance checking, and so on, that could be applied to any coding project. https://en.wikipedia.org/wiki/Infrastructure_as_code
IAM **(Identity and Access Management)**	IAM is the means to control access to IT resources so that only those explicitly authorized to access given resources can do so. IAM is an essential foundation to a secure multicloud environment. https://en.wikipedia.org/wiki/Identity_management
IBM **(Cloud)**	IBM IaaS and PaaS. https://www.ibm.com/cloud
Intersight	Cisco Intersight™ is a Software-as-a-Service (SaaS) infrastructure lifecycle management platform that delivers simplified configuration, deployment, maintenance, and support. https://www.cisco.com/c/en/us/products/servers-unified-computing/intersight/index.html

GCP **(Google Cloud Platform)**	Google IaaS and PaaS. https://cloud.google.com/gcp
Kubernetes **(K8s)**	Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications. https://kubernetes.io
Microservices	A microservices architecture is characterized by processes implementing fine-grained services, typically exposed via REST APIs and which can be composed into systems. The processes are often container-based, and the instantiation of the services often managed with Kubernetes. Microservices managed in this way are intrinsically well suited for deployment into IaaS environments, and as such, are the basis of a cloud native architecture. https://en.wikipedia.org/wiki/Microservices
PaaS **(Platform-as-a-Service)**	PaaS is a layer of value-add services, typically for application development, deployment, monitoring, and general lifecycle management. The use of IaC with IaaS and PaaS is very closely associated with DevOps practices.
Private on-premises data center	A data center infrastructure housed within an environment owned by a given enterprise is distinguished from other forms of data center, with the implication that the private data center is more secure, given that access is restricted to those authorized by the enterprise. Thus, circumstances can arise where very sensitive IT assets are only deployed in a private data center, in contrast to using public IaaS. For many intents and purposes, the underlying technology can be identical, allowing for hybrid deployments where some IT assets are privately deployed but also accessible to other assets in public IaaS. IAM, VPNs, firewalls, and similar are key technologies needed to underpin the security of such an arrangement.
REST API	Representational State Transfer (REST) APIs is a generic term for APIs accessed over HTTP(S), typically transporting data encoded in JSON or XML. REST APIs have the advantage that they support distributed systems, communicating over HTTP, which is a well-understood protocol from a security management perspective. REST APIs are another element of a cloud-native applications architecture, alongside microservices. https://en.wikipedia.org/wiki/Representational_state_transfer
SaaS **(Software-as-a-Service)**	End-user applications provided “aaS” over the public Internet, with the underlying software systems and infrastructure owned and managed by the provider.
SAML **(Security Assertion Markup Language)**	Used in the context of Single-Sign-On (SSO) for exchanging authentication and authorization data between an identity provider, typically an IAM system, and a service provider (some form of SaaS). The SAML protocol exchanges XML documents that contain security assertions used by the aaS for access control decisions. https://en.wikipedia.org/wiki/Security_Assertion_Markup_Language
Terraform	An open-source IaC software tool for cloud services, based on declarative configuration files. https://www.terraform.io

Appendix D - Acronym Glossary

AAA–Authentication, Authorization, and Accounting

ACP–Access-Control Policy

ACI–Cisco Application Centric Infrastructure

ACK–Acknowledge or Acknowledgement

ACL–Access-Control List

AD–Microsoft Active Directory

AFI–Address Family Identifier

AMP–Cisco Advanced Malware Protection

AP–Access Point

API–Application Programming Interface

APIC– Cisco Application Policy Infrastructure Controller (ACI)

ASA–Cisco Adaptative Security Appliance

ASM–Any-Source Multicast (PIM)

ASR–Aggregation Services Router

Auto-RP–Cisco Automatic Rendezvous Point protocol (multicast)

AVC–Application Visibility and Control

BFD–Bidirectional Forwarding Detection

BGP–Border Gateway Protocol

BMS–Building Management System

BSR–Bootstrap Router (multicast)

BYOD–Bring Your Own Device

CAPWAP–Control and Provisioning of Wireless Access Points Protocol

CDP–Cisco Discovery Protocol

CEF–Cisco Express Forwarding

CMD–Cisco Meta Data

CPU–Central Processing Unit

CSR–Cloud Services Routers

CTA–Cognitive Threat Analytics

CUWN–Cisco Unified Wireless Network

CVD–Cisco Validated Design

CYOD–Choose Your Own Device

DC–Data Center

DHCP–Dynamic Host Configuration Protocol

DM–Dense-Mode (multicast)

DMVPN–Dynamic Multipoint Virtual Private Network

DMZ–Demilitarized Zone (firewall/networking construct)

DNA–Cisco Digital Network Architecture

DNS–Domain Name System

DORA–Discover, Offer, Request, ACK (DHCP Process)

DWDM–Dense Wavelength Division Multiplexing

ECMP–Equal Cost Multi Path

EID–Endpoint Identifier

EIGRP–Enhanced Interior Gateway Routing Protocol

EMI–Electromagnetic Interference

ETR–Egress Tunnel Router (LISP)

EVPN–Ethernet Virtual Private Network (BGP EVPN with VXLAN data plane)

FHR–First-Hop Router (multicast)

FHRP–First-Hop Redundancy Protocol

FMC–Cisco Firepower Management Center

FTD–Cisco Firepower Threat Defense

GBAC–Group-Based Access Control

GbE–Gigabit Ethernet

Gbit/s–Gigabits Per Second (interface/port speed reference)

GRE–Generic Routing Encapsulation

GRT–Global Routing Table

HA–High-Availability

HQ–Headquarters

HSRP–Cisco Hot-Standby Routing Protocol

HTDB–Host-tracking Database (SD-Access control plane node construct)

IBNS–Identity-Based Networking Services (IBNS 2.0 is the current version)

ICMP– Internet Control Message Protocol

IDF–Intermediate Distribution Frame; essentially a wiring closet.

IEEE–Institute of Electrical and Electronics Engineers

IETF–Internet Engineering Task Force

IGP–Interior Gateway Protocol

IID–Instance-ID (LISP)

IOE–Internet of Everything

IoT–Internet of Things

IP–Internet Protocol

IPAM–IP Address Management

IPS–Intrusion Prevention System

IPSec–Internet Protocol Security

ISE–Cisco Identity Services Engine

ISR–Integrated Services Router

IS-IS–Intermediate System to Intermediate System routing protocol

ITR–Ingress Tunnel Router (LISP)

LACP–Link Aggregation Control Protocol

LAG–Link Aggregation Group

LAN–Local Area Network

L2 VNI–Layer 2 Virtual Network Identifier; as used in SD-Access Fabric, a VLAN.

L3 VNI– Layer 3 Virtual Network Identifier; as used in SD-Access Fabric, a VRF.

LHR–Last-Hop Router (multicast)

LISP–Location Identifier Separation Protocol

MAC–Media Access Control Address (OSI Layer 2 Address)

MAN–Metro Area Network

MEC–Multichassis EtherChannel, sometimes referenced as ***MCEC***

MDF–Main Distribution Frame; essentially the central wiring point of the network.

MnT–Monitoring and Troubleshooting Node (Cisco ISE persona)

MOH–Music on Hold

MPLS–Multiprotocol Label Switching

MR–Map-resolver (LISP)

MS–Map-server (LISP)

MSDP–Multicast Source Discovery Protocol (multicast)

MTU–Maximum Transmission Unit

NAC–Network Access Control

NAD–Network Access Device

NAT–Network Address Translation

NBAR–Cisco Network-Based Application Recognition (NBAR2 is the current version).

NFV–Network Functions Virtualization

NSF–Non-Stop Forwarding

OSI–Open Systems Interconnection model

OSPF–Open Shortest Path First routing protocol

OT–Operational Technology

PAgP–Port Aggregation Protocol

PAN–Primary Administration Node (Cisco ISE persona)

PCI DSS–Payment Card Industry Data Security Standard

PD–Powered Devices (PoE)

PETR–Proxy-Egress Tunnel Router (LISP)

PIM–Protocol-Independent Multicast

PITR–Proxy-Ingress Tunnel Router (LISP)

PnP–Plug-n-Play

PoE–Power over Ethernet (Generic term, may also refer to IEEE 802.3af, 15.4W at PSE)

PoE+–Power over Ethernet Plus (IEEE 802.3at, 30W at PSE)

PSE–Power Sourcing Equipment (PoE)

PSN–Policy Service Node (Cisco ISE persona)

pxGrid–Platform Exchange Grid (Cisco ISE persona and publisher/subscriber service)

PxTR–Proxy-Tunnel Router (LISP - device operating as both a PETR and PITR)

QoS–Quality of Service

RADIUS–Remote Authentication Dial-In User Service

REST–Representational State Transfer

RFC–Request for Comments Document (IETF)

RIB–Routing Information Base

RLOC–Routing Locator (LISP)

RP–Rendezvous Point (multicast)

RP–Redundancy Port (WLC)

RP–Route Processer

RPF–Reverse Path Forwarding

RR–Route Reflector (BGP)

RTT–Round-Trip Time

SA–Source Active (multicast)

SAFI–Subsequent Address Family Identifiers (BGP)

SD–Software-Defined

SDA–Cisco Software Defined-Access

SDN–Software-Defined Networking

SFP–Small Form-Factor Pluggable (1 GbE transceiver)

SFP+– Small Form-Factor Pluggable (10 GbE transceiver)

SGACL–Security-Group ACL

SGT–Scalable Group Tag, sometimes reference as Security Group Tag

SM–Spare-mode (multicast)

SNMP–Simple Network Management Protocol

SSID–Service Set Identifier (wireless)

SSM–Source-Specific Multicast (PIM)

SSO–Stateful Switchover

STP–Spanning-tree protocol

SVI–Switched Virtual Interface

SVL–Cisco StackWise Virtual

SWIM–Software Image Management

SXP–Scalable Group Tag Exchange Protocol

Syslog–System Logging Protocol

TACACS+–Terminal Access Controller Access-Control System Plus

TCP–Transmission Control Protocol (OSI Layer 4)

UCS– Cisco Unified Computing System

UDP–User Datagram Protocol (OSI Layer 4)

UPoE–Cisco Universal Power Over Ethernet (60W at PSE)

UPoE+– Cisco Universal Power Over Ethernet Plus (90W at PSE)

URL–Uniform Resource Locator

VLAN–Virtual Local Area Network

VM—Virtual Machine

VN–Virtual Network, analogous to a VRF in SD-Access

VNI–Virtual Network Identifier (VXLAN)

vPC–virtual Port Channel (Cisco Nexus)

VPLS–Virtual Private LAN Service

VPN–Virtual Private Network

VPNv4–BGP address family that consists of a Route-Distinguisher (RD) prepended to an IPv4 prefix

VPWS–Virtual Private Wire Service

VRF–Virtual Routing and Forwarding

VSL–Virtual Switch Link (Cisco VSS component)

VSS–Cisco Virtual Switching System

VXLAN–Virtual Extensible LAN

WAN–Wide-Area Network

WLAN–Wireless Local Area Network (generally synonymous with IEEE 802.11-based networks)

WoL–Wake-on-LAN

xTR–Tunnel Router (LISP – device operating as both an ETR and ITR)

Appendix E – Recommended for You

Cisco UCS Solutions GitHub Repo: https://github.com/ucs-compute-solutions

Feedback

For comments and suggestions about this guide and related guides, join the discussion on Cisco Community at https://cs.co/en-cvds.

CVD Program

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV Amsterdam,
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **https://www.cisco.com/go/offices**.