Hybrid Cloud with Google Anthos on Hitachi UCP Platform

Reference Architecture Guide

MK-SL-265-00
March 2023

Anthos

Feedback

Hitachi Vantara welcomes your feedback. Please share your thoughts by sending an email message to SolutionLab@HitachiVantara.com. To assist the routing of this message, use the paper number in the subject and the title of this white paper in the text.

Revision history

Changes	Date
Initial release	March 10, 2023

Reference Architecture Guide

Executive overview

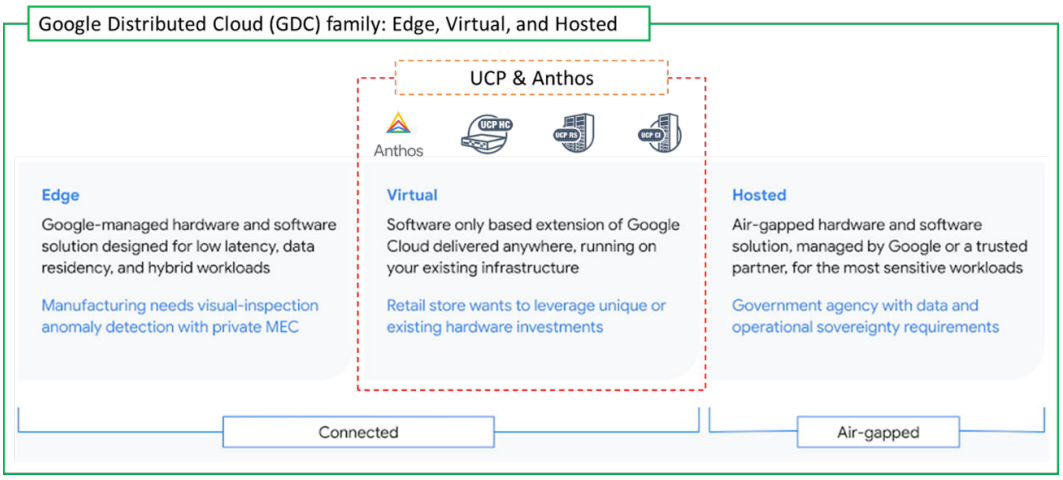
Hitachi Vantara has successfully integrated Anthos, the cloud-native container platform developed by Google Cloud, with the innovative Hitachi Unified Compute Platform (UCP) to achieve Anthos Ready status. Hitachi UCP offers enterprise-class converged, hyper-converged infrastructure, and rack-scale systems that deliver agility, scalability, reliability, resilience, and high-performance to meet the dynamic needs of modern businesses.

Anthos is a robust cloud-native container platform built on open-source technologies, such as Kubernetes, Istio, and Knative, providing secure, scalable, and consistent application development and deployment across on-premises and cloud environments. Anthos enables businesses to accelerate application development across hybrid edge-core-cloud environments, promoting flexibility and agility to stay ahead in a competitive market.

Anthos on-premises is delivered as part of Google's Distributed Cloud (GDC) family, a portfolio of hardware, software, and services that brings Google infrastructure services to the edge and into your data centers. This provides a uniform set of experiences for development, security, and management across any IT environment backed by a common Anthos API. With this Anthos Ready status, customers can now select Hitachi UCP when choosing to use GDC Virtual approach to deploy and manage Anthos on VMware vSphere and Anthos on bare metal services.

By integrating Anthos on UCP, Hitachi Vantara provides customers with a powerful portfolio of solutions that combines the benefits of the Anthos cloud-native container platform with the uncompromised performance and reliability of UCP. Anthos on UCP can leverage either hyperconverged or converged configurations with external storage, complete with our rich set of data services for container storage. This integration has been successfully tested, validated, and now officially listed as an Anthos Ready platform, helping businesses to modernize their IT infrastructure, develop new applications, and meet the evolving demands of the market with ease.

This document details how to deploy and manage a hybrid multi-cloud environment using the Hitachi UCP platform with Google Anthos on-prem.



Ref: https://cloud.google.com/blog/topics/anthos/anthos-on-prem-and-bare-metal-are-now-gdc-virtual

Overview

This reference architecture serves as a proof point that Hitachi Unified Compute platform has been tested and validated as an Anthos Ready platform that supports the latest version. It also provides the foundation for Anthos clusters on VMware and Anthos clusters on bare metal, components of Google Distributed Cloud Virtual (GDC Virtual). This document describes how to deploy and manage a hybrid multi-cloud environment using Google's Anthos and Hitachi UCP platform.

This paper covers the functional aspects of Anthos core components and provides an architecture overview and implementation of Anthos on top of the Hitachi UCP environment. In addition, it provides an example deployment of a stateful application with persistent volumes on Hitachi Virtual Storage Platform (VSP) and VMware vSAN, as well as the integration with third-party Kubernetes clusters deployed on top of UCP, all managed with Google Cloud console.

A key element in the successful deployment of a container platform is having a robust and flexible infrastructure that can meet a wide variety of requirements in a highly dynamic environment. Hitachi UCP provides converged solutions with VSP and hyper-converged solutions certified as VMware vSAN Ready Nodes with a variety of configurable options that can meet any application workload and business needs. Hitachi infrastructure together with Anthos capabilities provides a highly available, high-performance infrastructure, scalable, centralized management, hybrid, and multi-cluster management for containerized workloads.

The intended audience of this document is IT administrators, system architects, consultants, and sales engineers to assist in planning, designing, and implementing the UCP product portfolio with Google Anthos solutions.

Solution components

This section outlines the components used in this reference architecture.

Hitachi Unified Compute Platform deployment options

The following Hitachi Unified Compute Platform deployment options are used in this solution.

Hitachi Unified Compute Platform CI

Hitachi Unified Compute Platform CI (UCP CI) is an optimized, preconfigured, and pretested converged infrastructure appliance for VMware vSphere. It offers a broad range of compute and storage components that can be scaled and configured independently to eliminate overprovisioning. You have a choice of operating environments to maximize your flexibility.

With Unified Compute Platform CI, you can choose between single-rack configurations and multi-rack configurations. More details about UCP CI can be found at https://www.hitachivantara.com/en-us/products/integrated-systems/converged-infrastructure.html.

Hitachi Unified Compute Platform RS

To simplify your hybrid cloud journey, Hitachi Unified Compute Platform RS (UCP RS) provides a turnkey solution that reduces total cost of ownership (TCO) and improves security. The software-defined data center solution accelerates the time to market with a natively integrated cloud infrastructure stack. It comes prepackaged with management software, to provide automated, policy-based IT operations.

Unified Compute Platform RS has automation that enables the deployment of an entire cloud infrastructure in hours, not weeks or months. There is rapid and repeatable application deployment.

Move your workload across data centers to meet changing business needs. Manage your applications across private and public cloud from a common toolset. Scale your data enter without increasing IT headcount. Automate your data center with policies.

More details about UCP RS can be found at https://www.hitachivantara.com/en-us/products/integrated-systems/cloud-foundation.html.

Hitachi Unified Compute Platform HC

Hitachi Unified Compute Platform HC (UCP HC) is an integrated turnkey appliance that combines compute, storage, and virtualization to deliver certainty for edge to core to cloud operations.

This market-proven Hitachi solution provides a scalable, seamless, and simplified cloud foundation for enterprise and mid-market customers. Advanced automation and intelligence for day 0-2 operations accelerate innovation and improve productivity while lowering the TCO.

More details about UCP HC can be found at https://www.hitachivantara.com/en-us/products/integrated-systems/hyper-converged-infrastructure.html.

Hitachi UCP hardware components

The following tables list the versions of hardware and software tested in this reference architecture.

For more information, see Hitachi Vantara Support UCP Product Compatibility at https://compatibility.hitachivantara.com/ and https://compatibility.hitachivantara.com/assets/vmware.

Hardware	Description	Version	Quantity
Hitachi Advanced Server HA810 G2 (for VMware compute cluster)	▪ 2 × Intel Xeon Gold 6338 CPU @ 2.00GHz processors ▪ 8 × 32 GB DIMM, 256 GB memory ▪ NS204i-r NVMe OS Boot Device (Two 480 GB M.2) ▪ Emulex LPe36000 Fibre Channel Adapter ▪ 1 × Intel(R) Ethernet Controller E810-XXV for SFP NIC dual-port ▪ For vSAN configuration: • 1 × SAS SSD 1.92TB (cache) • 3 × SAS SSD 1.92TB (capacity)	iLO 5: 2.65 System ROM: U46 v1.58	3
Hitachi Virtual Storage Platform E1090	▪ 1 TB cache ▪ 8 × 15 TB NVMe drives ▪ 4 × 32 Gbps Fibre Channel ports	93-06-42-80/00	1
Cisco Nexus 9332C switch (spine)	▪ 32-port 40/100 GbE ▪ 2-port 1/10 GbE	NXOS 9.3.5	2
Cisco Nexus 93180YC-FX switch (leaf)	▪ 48-port 10/25 GbE ▪ 6-port 40/100 GbE	NXOS 9.3.5	2
Cisco Nexus 92348	▪ 48-port 1 GbE ▪ 4-port 1/10/25 GbE ▪ 2-port 40/100 GbE	NXOS 9.3.5	1
Brocade G620	▪ 48-port 16/32 Gbps Fibre Channel switch	9.0.0b	2

Software components

The following table lists the key software components.

Software	Version
Hitachi Storage Virtualization Operating System RF	90-05-02-00/01 83-05-33-40/00
VMware vSphere	7.0 Update 3 or newer
VMware vSAN	7.0 Update 3 or newer
Anthos on VMware	1.14.1-gke.39
Kubernetes	1.25.5-gke.100
F5 Big-IP Virtual Edition	17.0.0.1

Google Cloud Anthos

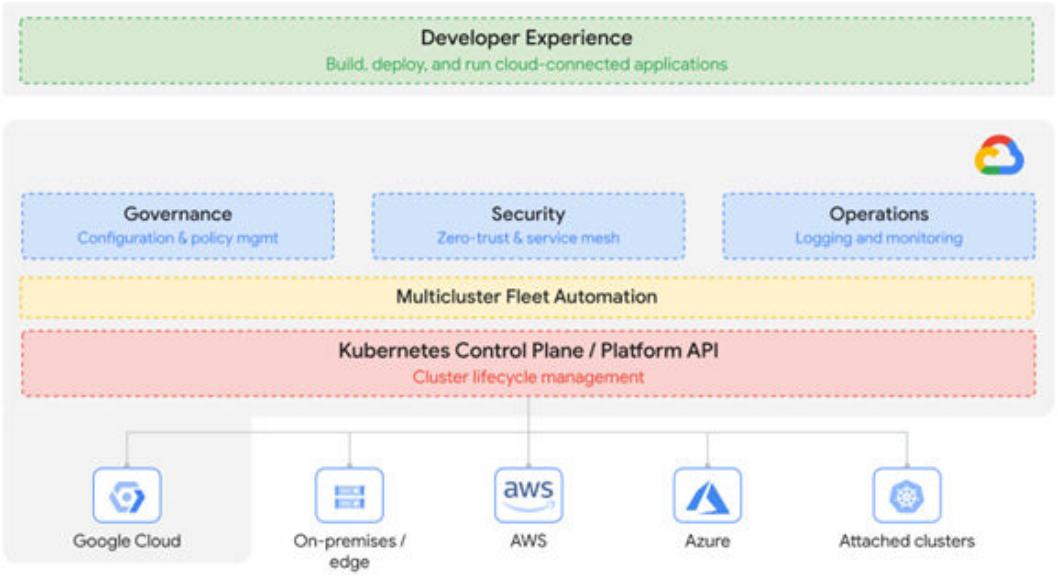
Google Anthos is a cloud-centric container platform that provides you with a consistent platform to construct and manage modern hybrid and multi-cloud environments through a single pane of glass with Google Cloud console. Anthos runs on-premises in a VMware vSphere-based or bare metal environment.

Anthos clusters on VMware and Anthos clusters on bare metal, components of Google Distributed Cloud Virtual (GDC Virtual), are software that bring Google Kubernetes Engine (GKE) to on-premises data centers such as Hitachi UCP platform, which has been qualified as an Anthos Ready platform.

The following figure shows an Anthos solution and its capabilities to manage your fleet clusters and the applications that run on them. A fleet can be made up of GKE clusters on Google Cloud or include clusters outside Google Cloud running on-premises or other public clouds such as Amazon AWS and Microsoft Azure. Anthos helps simplify working across multiple clusters and infrastructure providers, and provides the following features:

- Configuration and policy management
- Fleet-wide networking features
- Identity management features
- Observability features

Anthos Service Mesh provides powerful tools for application security, networking, and observability.



Anthos deployment options

Google Cloud and Anthos features can be used on the following Anthos environments:

- Google Kubernetes Engine (GKE) on Google Cloud
- Google Distributed Cloud Virtual (Anthos on-premises):
  - Anthos clusters on VMware
  - Anthos clusters on bare metal
- Google Distributed Cloud Edge
- Anthos multi-cloud:
  - Anthos clusters on Amazon AWS
  - Anthos clusters on Microsoft Azure
- Attached clusters, these are third-party Kubernetes clusters (EKS, AKS, and other Kubernetes clusters) registered to your fleet.

This paper focuses on the deployment of Anthos clusters on VMware on top of Hitachi Unified Compute Platform.

Anthos on-prem cluster components

The following components make up an Anthos cluster on VMware installation:

- Admin cluster

  The admin cluster is where the Kubernetes control planes for the admin cluster and its associated user clusters run, as well as any add-ons. A single admin cluster can manage multiple user clusters.

  The following nodes are in the admin cluster:

  - Admin cluster control plane — runs the control plane for the admin cluster. The machine that runs the admin control plane is called the admin master.
  - User cluster control plane — runs the control plane for a user cluster. The machine that runs the user cluster control plane is called the user master; there will be a VM for each deployed user cluster.
  - Add-ons — run Kubernetes add-ons such as Prometheus or Grafana. Two VMs, separate from the admin master, run the control plane.

- User clusters

  A user cluster is where you deploy and run your organization‘s workloads and services. Each node in a user cluster is called a worker node. The number (default 3 nodes) and resources for these nodes in a user cluster depend on the workloads your organization plans to run.

- Admin workstation

  The admin workstation is a separate VM with the tools cluster creators and developers need to manage Anthos clusters on VMware. The following tools are used from the admin workstation:

  - `Kubectl` — used to interact with your admin and user clusters, including deploying and managing workloads.
  - `Gkectl` — used to create and update clusters and perform other administrative tasks.

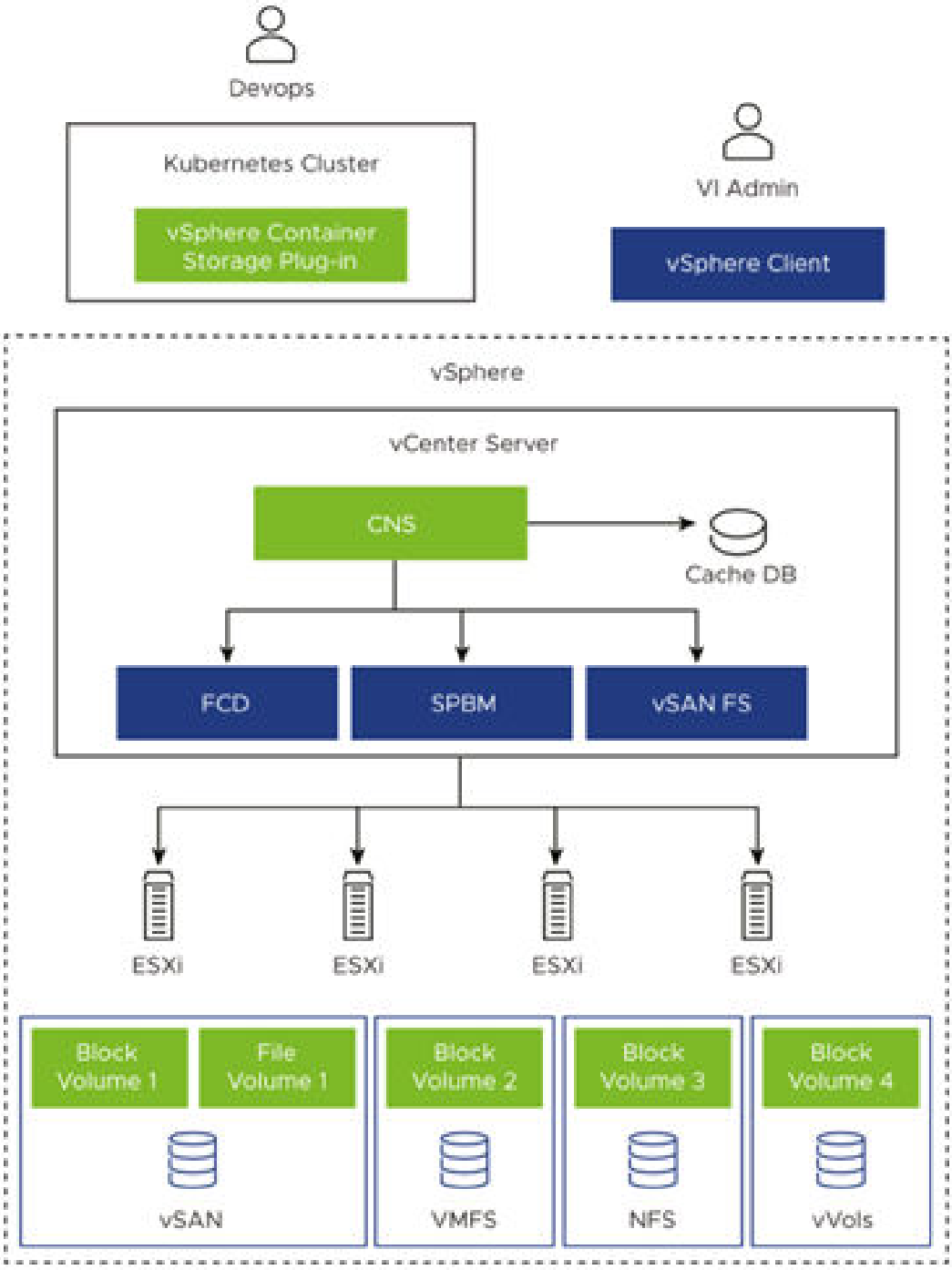
As an alternative to logging into the admin workstation, Google Cloud console provides a web interface where you can perform a subset of Anthos clusters on VMware administrative tasks, including creating new user clusters.

vSphere Cloud Native Storage (CNS)

Cloud Native Storage (CNS) integrates vSphere and Kubernetes and offers capabilities to create and manage container volumes deployed in a vSphere environment. CNS consists of two components, a CNS component in vCenter Server and a vSphere volume driver (also called the vSphere CSI driver) in Kubernetes, called vSphere Container Storage Plug-in.

- CNS enables vSphere and vSphere storage (VMFS, vVols, NFS), including vSAN, as a platform to run stateful applications. CNS enables access of this data path to Kubernetes and brings an understanding of Kubernetes volume and pod abstractions to vSphere. CNS uses several components to work with vSphere storage; this includes VMFS or vVols provided by the Hitachi Storage Provider for VMware vCenter. After you create PVs, you can review them and the virtual disks that back them in the vSphere Client and monitor their storage policy compliance.
- The vSphere Container Storage Plug-in has different components that provide an interface used by the Container Orchestrators such as GKE to manage the lifecycle of vSphere volumes. It also allows you to create, expand and delete volumes, attach, and detach volumes to the cluster worker node VMs and use bind mounts for the volumes inside the pods.

The following figure illustrates how CNS components, CNS in vCenter Server, and vSphere Container Storage Plug-in interact with other components in a vSphere environment (credit to VMware).



Anthos clusters on VMware (GKE on-prem) require installation of the vSphere CSI driver. This CSI driver is installed automatically in Anthos clusters when the clusters are provisioned.

Solution design

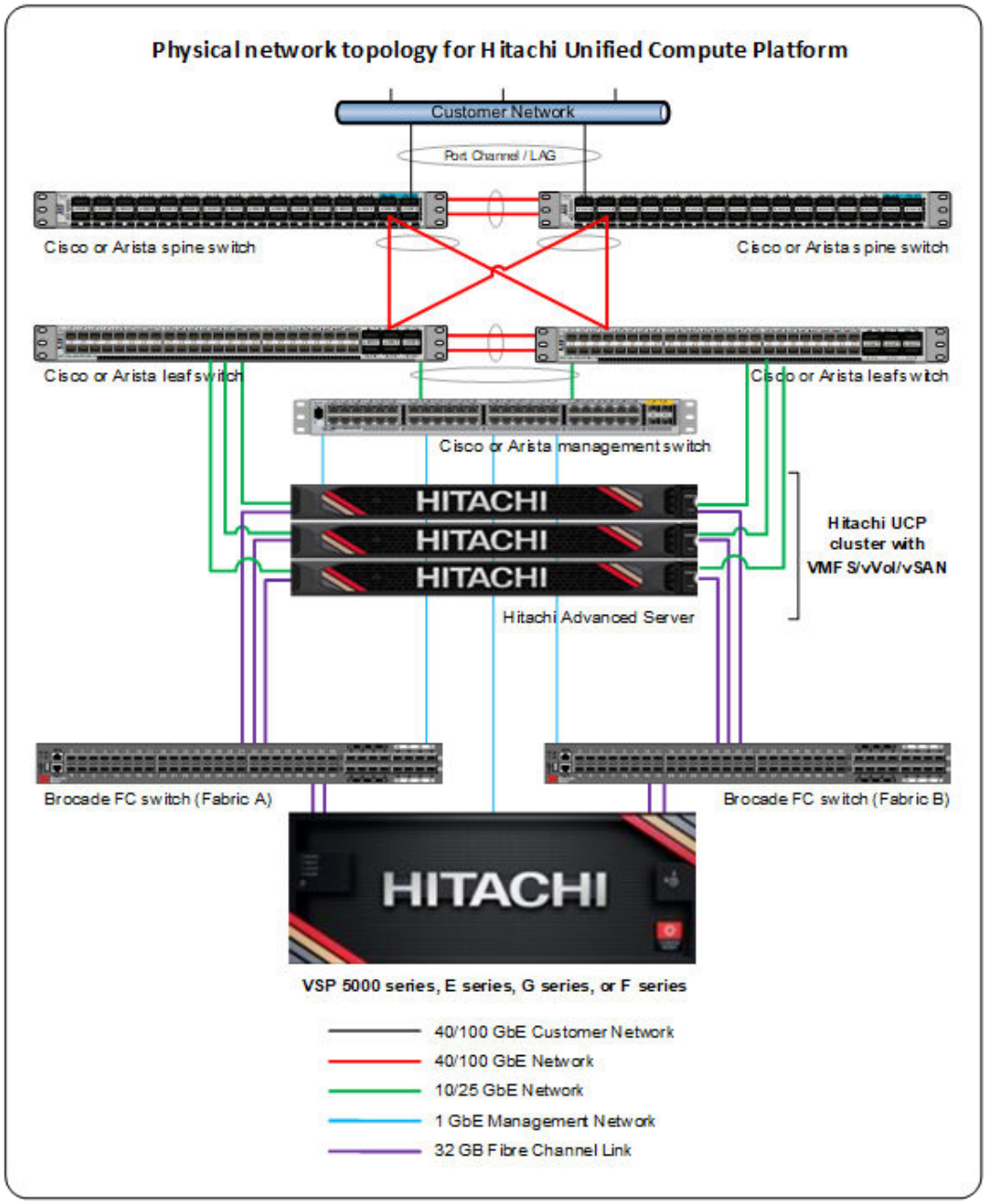
This section describes the detailed solution example for the Hitachi Unified Compute Platform and Google Anthos.

UCP infrastructure components

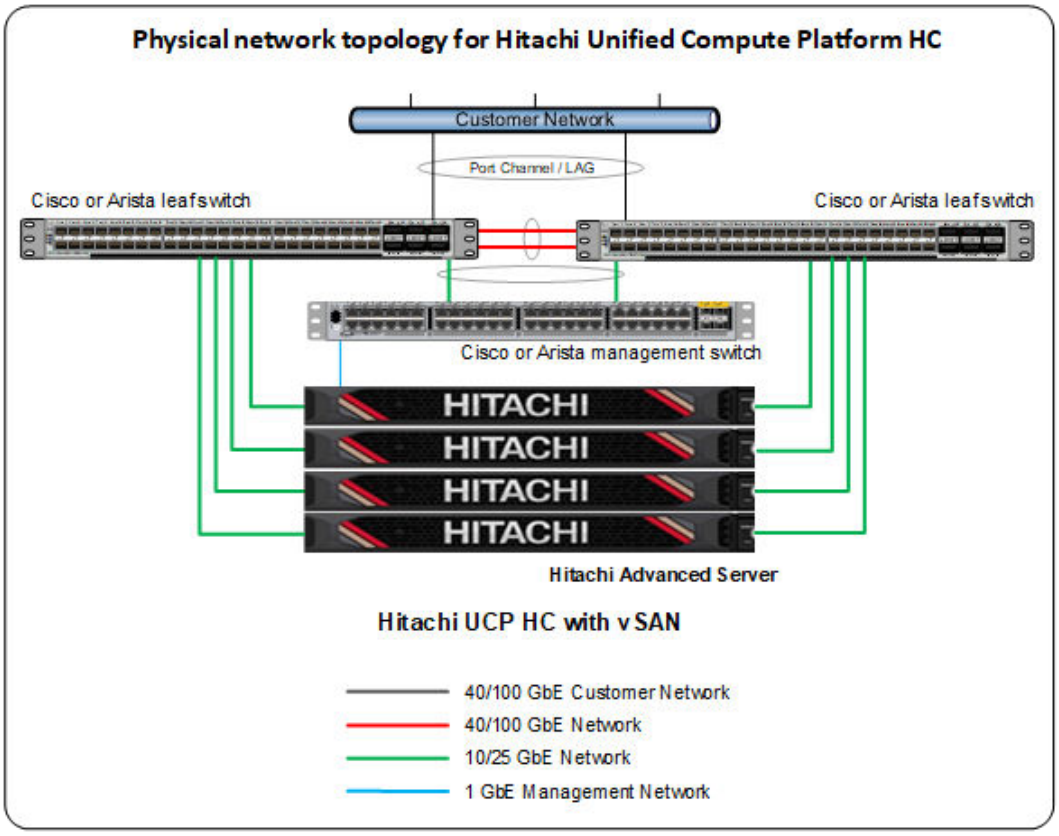
The following figure shows a high availability configuration of Hitachi Unified Compute Platform used to validate the Google Anthos on-prem solution. It includes the following components:

- Two Cisco 9332C or Arista 7050CX3 spine Ethernet switches.
- Two Cisco 93180YC-FX or Arista 7050SX3 leaf Ethernet switches.
- One Cisco 92348 or Arista 7010T management switch.
- Three or four Hitachi Advanced Server models for the vSAN cluster.
  - For vSAN compute nodes, leverage supported internal drives. These compute nodes are vSAN Ready Node Certified as UCP HC.
  - For vVols or VMFS compute nodes, leverage the HBA PCIe card, which is optionally configured together with the UCP HC vSAN Ready Nodes, or when configuring UCP Fibre Channel-only nodes in UCP RS.
- One Hitachi Virtual Storage Platform storage system for the UCP CI option.

The following diagram represents a standard architecture for the Hitachi Unified Compute Platform product portfolio.



The following diagram represents a standard architecture for the Hitachi Unified Compute Platform HC (with VMware vSAN).



The configuration with Hitachi Virtual Storage Platform is described in Unified Compute Platform product portfolio documentation. See the Hitachi Unified Compute Platform CI for VMware vSphere Reference Architecture Guide at https://knowledge.hitachivantara.com/Documents/Application_Optimized_Solutions/VMWare/Unified_Compute_Platform_CI_for_VMware_vSphere_Reference_Architecture_Guide for more information regarding Unified Compute Platform CI configurations.

Hitachi UCP Advisor (optional)

Hitachi Unified Compute Platform Advisor (UCP Advisor) brings simplified IT administration to virtualized, converged, and hyperconverged systems from Hitachi. UCP Advisor supports guided life-cycle management to the server, network, and storage elements within supported Unified Compute Platform systems.

VMware vVols and storage policy-based management (SPBM) (optional)

Storage Provider for VMware vCenter (VASA Provider) enables organizations to deploy Hitachi Storage infrastructure with VMware vSphere virtual volumes (vVols) to bring customers on a reliable enterprise journey to a software-defined, policy-controlled data center.

Hitachi storage policy-based management allows automated provisioning of virtual machines (VMs) and quicker adjustment to business changes. Virtual infrastructure (VI) administrators can make changes to policies to reflect changes in their business environment, dynamically matching storage policy requirements for VMs to available storage pools and services. The vVols solution reduces the operational burden between VI administrators and storage administrators with an efficient collaboration framework leading to faster and better VM and application services provisioning.

To use VMware vVols with Hitachi storage, install Hitachi Storage Provider for VMware vCenter. See *VMware vSphere Virtual Volumes (vVols) with Hitachi Virtual Storage Platform Quick Start and Reference Guide* at https://knowledge.hitachivantara.com/Documents/Application_Optimized_Solutions/VMWare/VMware_vSphere_Virtual_Volumes_(vVols)_with_Hitachi_Virtual_Storage_Platform_Quick_Start_and_Reference_Guide for details.

See *Storage Provider for VMware vCenter (VASA)* https://knowledge.hitachivantara.com/Documents/Adapters_and_Drivers/Storage_Adapters_and_Drivers/VMware/Storage_Provider_for_VMware_vCenter_(VASA) to deploy this environment.

Anthos on-prem configuration

As indicated in previous sections, the Anthos on-prem deployment consists of three types of virtual machines:

- Admin workstation VM — used to configure and manage the Anthos clusters.
- Admin cluster VMs — used to run the admin control plane, user cluster's control plane, and add-ons.
- User Cluster VMs — used to run user workloads and services.

The following tables list the minimum requirements for these different types of virtual machines, basically using the default values.

Node	Requirements	Purpose
Admin workstation	▪ 4 vCPUs ▪ 8 GiB RAM ▪ 100 GiB	This is a standalone VM with the tools and resources needed to create Anthos clusters in your vSphere environment.

Node	Requirements	Purpose
Admin cluster control-plane	▪ 4 vCPUs ▪ 16 GiB RAM	One VM, runs the control plane for the admin cluster

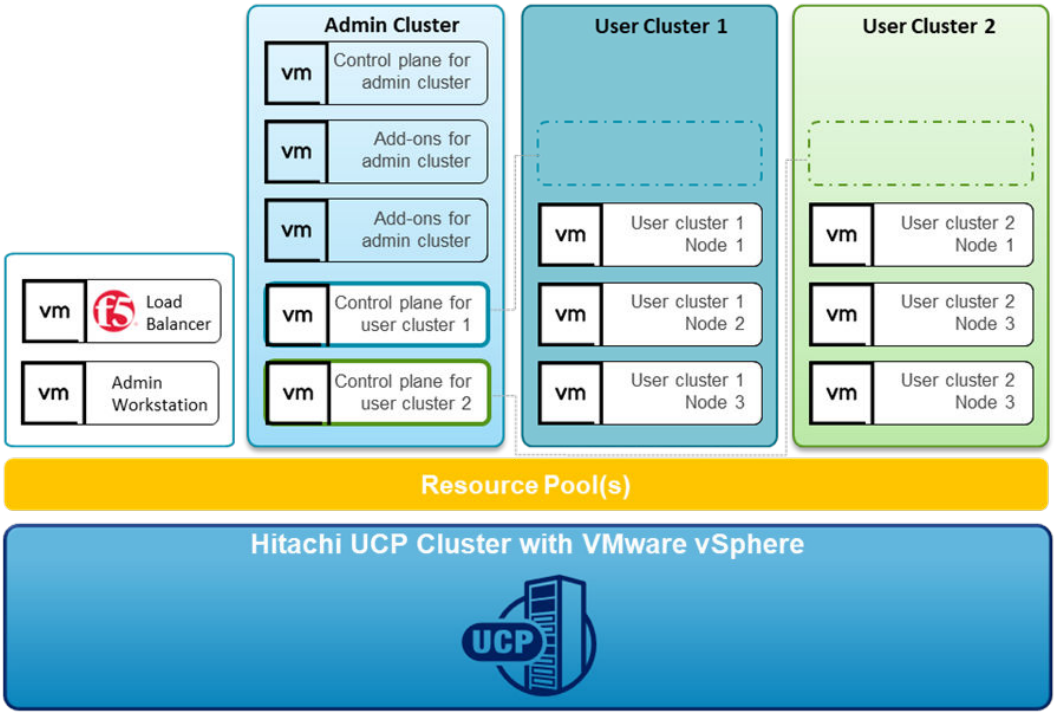
Node	Requirements	Purpose
	▪ 40 GiB disk ▪ 100 GiB disk	
Add-ons	▪ 4 vCPUs ▪ 16 GiB RAM ▪ 40 GiB disk	Two VMs that run the admin cluster's add-ons.
User cluster control-plane	▪ 4 vCPUs ▪ 8 GiB RAM ▪ 40 GiB disk	For each user cluster, one or three VMs. Runs the control plane for user clusters.

Node	Requirements	Purpose
User cluster worker node	▪ 4 vCPU(s) ▪ 8 GiB ▪ 40 GiB	A user cluster node is where your workloads run. These values are the default. The number of nodes and resources required will depend on the workloads you plan to run.

See https://cloud.google.com/anthos/clusters/docs/on-prem/latest/how-to/cpu-ram-storage for additional details about hardware requirements for Anthos.

Anthos on-prem deployment example

In this solution, an admin cluster and two user clusters have been deployed on-premises on top of Hitachi UCP with VMware vSphere. The following figure shows that for this deployment, the admin cluster consists of five virtual machines, one control plane for the admin cluster, one control plane for user cluster-1, one control plane for user cluster-2, and two add-ons for the admin cluster. Both user clusters consist of three virtual machines or worker nodes for user workloads. The environment includes the admin workstation virtual machine and F5 Big-IP Load Balancer.



Deploy Anthos on-prem clusters

On-premises Anthos clusters can be installed on VMware or on bare metal using Hitachi Unified Compute Platform (UCP), depending on your application and business needs. For complete guides to Anthos on-premises options with Google Distributed Cloud Virtual, including cluster setup and administration, see the following resources:

- Anthos clusters on VMware: https://cloud.google.com/anthos/gke/docs/on-prem
- Anthos clusters on bare metal: https://cloud.google.com/anthos/clusters/docs/bare-metal

This guide covers the setup of Anthos clusters on VMware using the Hitachi Unified Compute Platform portfolio.

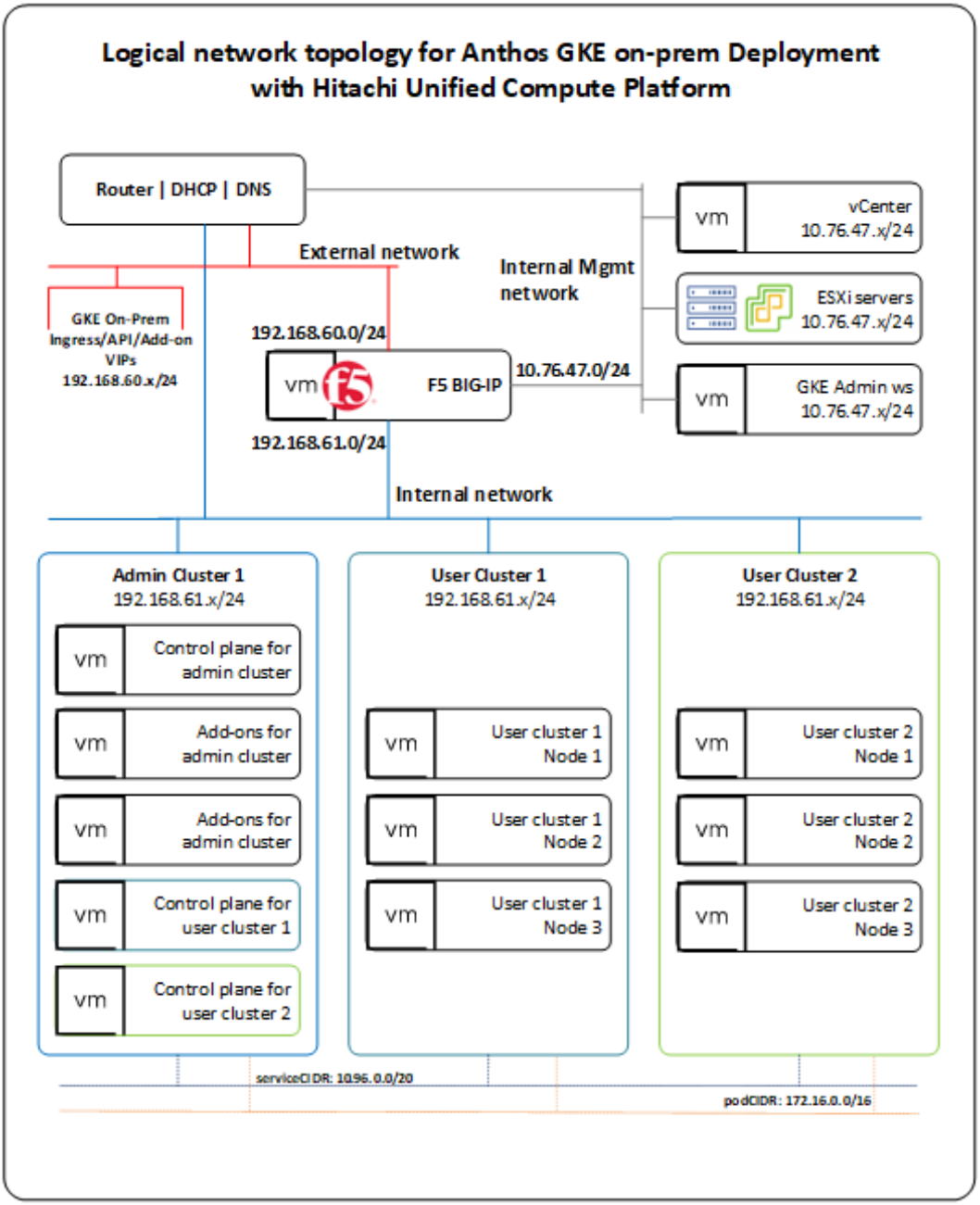
Set up Hitachi UCP CI or UCP HC clusters

To validate this reference architecture, a UCP CI/UCP HC cluster type was configured as described in UCP infrastructure components (on page 13). The VMware cluster was configured following best practices as described in the Hitachi UCP documentation.

The deployment environment consists of the following components:

- VMware vCenter cluster configured to support block storage (with Hitachi Virtual Storage Platform) and VMware vSAN storage:
  - UCP CI deployment for block storage — ESXi cluster connected to Hitachi Virtual Storage Platform (using Fibre channel)
  - UCP HC deployment (vSAN) — ESXi/vSAN cluster (vSAN Ready Nodes)
- Load balancer
- Anthos admin workstation
- Red Hat client workstation (or jump server)
- DNS server
- DHCP server

The following illustration shows a high-level logical network topology for the deployment of Anthos on-prem on top of Hitachi Unified Compute Platform.



Deploy and configure a load balancer

There are several load balancing options supported by Anthos clusters on VMware; choose the option most suited according to your needs.

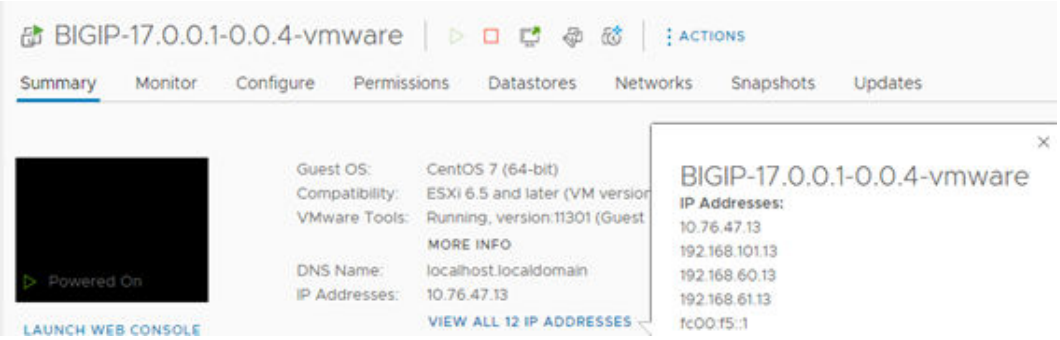
In this validation, Anthos cluster on VMware was configured to be integrated with F5 Big-IP. When choosing this option, Anthos cluster automatically configured the required VIPs on the load balancer.

The following summarizes the steps to deploy and configure F5 Big-IP Virtual Edition appliance.

Procedure

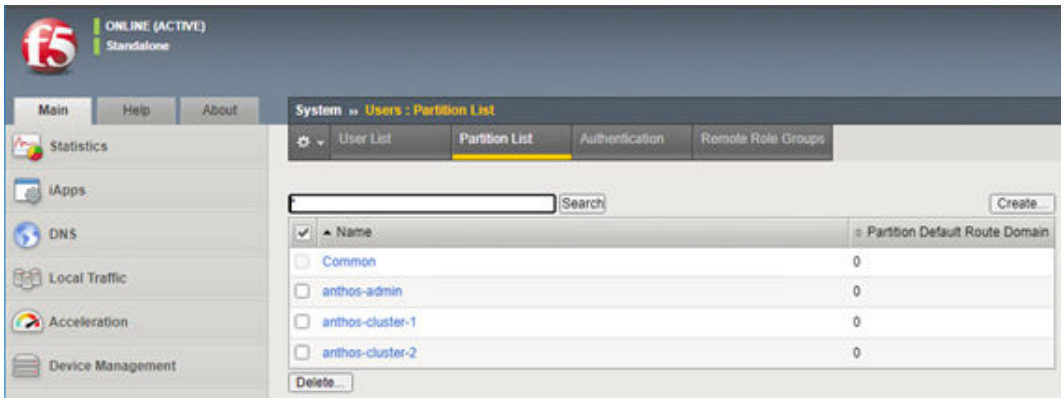
1. Download the F5 Big-IP Virtual Edition (OVA) from F5.

   This requires registration and login to the official F5 site.
2. Deploy the OVA into the UCP cluster using either DHCP or static for the management interface.



3. Log in using the management IP and activate the license.
4. Configure the internal network, external network, and HA network if deploying multiple virtual appliances for the solution.
5. Create individual partitions for the Anthos admin cluster and for each user cluster to be deployed.

   The partition list should look like the following.



   See *Installing F5 BIG-IP ADC for Anthos clusters on VMware* at https://cloud.google.com/architecture/partners/installing-f5-big-ip-adc-for-gke-on-prem for specific details about the setup of F5 Big-IP for Anthos.

> **Note:** The F5 Big-IP load balancer is not bundled with Anthos, so you must get a license and set up the load balancer separately from installing Anthos clusters on VMware. The load balancer must be configured before configuring Anthos clusters.

Set up Google Cloud resources

After the UCP on-prem infrastructure has been configured, you can start the Anthos deployment process. This requires certain tools that are in the Google CLI and several other prerequisites are needed to deploy and access the solution. See https://cloud.google.com/anthos/clusters/docs/on-prem/latest for specific details.

To prepare the environment for Anthos on-prem on VMware, follow these steps.

Procedure

1. Create a Google Cloud project, following the steps from *Creating and managing projects* at https://cloud.google.com/resource-manager/docs/creating-managing-projects.

   Note: Request that your cloud administration team create a project configured for access to Anthos on VMware. All projects intended for use with Anthos must be whitelisted by Google.

2. Deploy a client workstation to manage the installation of Anthos.

   This client workstation can be Linux, MacOS, or Windows. The validation for this paper was done on a client workstation using Red Hat Enterprise Linux 8.4. This workstation must be able to communicate with the VMware vCenter server and the Internet.

3. Install Google Cloud CLI and related tooling on the client workstation.
   a. Install Google Cloud CLI, but skip the `gcloud init` command, and follow instructions at https://cloud.google.com/sdk/docs or see https://cloud.google.com/sdk/docs/downloads-interactive#linux-mac to use the Google Cloud CLI installer in an interactive mode.
   b. After the Cloud CLI has been installed, verify the installed components with the following command:

   ```
   gcloud components list
   ```

   c. If needed, update the gcloud CLI using the following command:

   ```
   gcloud components update
   ```

   d. Install anthos-auth and kubectl using the following commands:

   ```
   gcloud components install kubectl
   gcloud components install anthos-auth
   ```

4. After the workstation has been configured with Google Cloud CLI and related tooling, log in to Google Cloud using the credentials from your organization. Enter the login command and it will display a URL that can be copied into a browser to allow sign-in to Google services. After login, it will present an authorization code that you can copy and paste back into the client workstation and then press **Enter**, as follows:

```
[root@jp-gke-adminws ~]# gcloud auth login
Go to the following link in your browser:
    https://accounts.google.com/o/oauth2/auth?
response_type=code&client_id=32555940559.apps.googleusercontent.com&redirect_uri=
https%3A%2F%2Fsdk.cloud.google.com%2Fauthcode.html&scope=openid+https%3A%2F
%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+https%3A%2F%2Fwww.googleapis.com
%2Fauth%2Fcloud-platform+https%3A%2F%2Fwww.googleapis.com%2Fauth
%2Fappengine.admin+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fsqlservice.login
+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcompute+https%3A%2F
%2Fwww.googleapis.com%2Fauth
%2Faccounts.reauth&state=JpRTK02S4iYbGcRk0xD8AhIeTkVeZo&prompt=consent&access_typ
```

```
e=offline&code_challenge=PuLG0MKEYxyDpWi1H22C7nZsyDNk4Xoci98v95PT2mA&code_challen
ge_method=S256
Enter authorization code:
4/0AWtgzh7cbWJCGroizNyu6Pl0tP0hh6ByEbmIttvcrhpJVjHYxNozMaODOvU_AtaspOdrtw



                                                                          You
are now logged in as
[cccc.ccccc2@hitachivantara.com].
                                                                 Your current
project is [hv-ucp-anthos].  You can change this setting by
running:
                    $ gcloud config set project PROJECT_ID
```

5. Enable Google APIs in your Cloud project so that your on-prem environment can communicate with Google Cloud.

   For this validation, we used the project hv-ucp-anthos. The following example shows how to enable Google APIs in your Cloud project:

```
gcloud services enable --project hv-ucp-anthos \
anthos.googleapis.com \
anthosgke.googleapis.com \
anthosaudit.googleapis.com \
cloudresourcemanager.googleapis.com \
container.googleapis.com \
gkeconnect.googleapis.com \
gkehub.googleapis.com \
serviceusage.googleapis.com \
stackdriver.googleapis.com \
opsconfigmonitoring.googleapis.com \
monitoring.googleapis.com \
logging.googleapis.com \
iam.googleapis.com \
storage.googleapis.com \
connectgateway.googleapis.com
```

6. Create service accounts and grant required roles.

   Before you create your admin and user clusters, you must create these service accounts:

Service Account Name	Purpose
Component access service account	This service account is used to download cluster components on your behalf, from the Container Registry.
Connect-register service account	This service account is used to register your clusters with Google Cloud.
Logging-monitoring service account	This service account is used to export logs and metrics from clusters to Cloud Logging and Cloud Monitoring.

Depending on the features you want to enable, you might also need to have some optional service accounts. See https://cloud.google.com/anthos/clusters/docs/on-prem/latest/how-to/service-accounts#optional_service_accounts for details.

The following steps provide examples of how to manually create these service accounts and grant the required roles to these service accounts. For each service account, first create the service account, then create a JSON key, and then grant the required roles.

7. Create a component access service account.

```
gcloud iam service-accounts create component-access-sa \
--display-name "Component Access Service Account" \
--project hv-ucp-anthos

gcloud iam service-accounts keys create component-access-key.json \
--iam-account component-access-sa@hv-ucp-anthos.iam.gserviceaccount.com \
--project hv-ucp-anthos
```

Note: Depending on your organization policies, service account key creation might be disabled; check with your cloud administrator to create the JSON keys if necessary.

8. After the accounts have been created, grant Identity and Access Management (IAM) roles to your component access service account.

   The following roles are required so Anthos clusters on VMware can do preflight checks:

   - `serviceusage.serviceUsageViewer`
   - `iam.roleViewer`
   - `iam.serviceAccountViewer`
   - `compute.viewer`

```
gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:component-access-sa@hv-ucp-
anthos.iam.gserviceaccount.com" \
--role "roles/serviceusage.serviceUsageViewer"

gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:component-access-sa@hv-ucp-
anthos.iam.gserviceaccount.com" \
--role "roles/iam.roleViewer"

gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:component-access-sa@hv-ucp-
anthos.iam.gserviceaccount.com" \
--role "roles/iam.serviceAccountViewer"

gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:component-access-sa@hv-ucp-
anthos.iam.gserviceaccount.com" \
--role "roles/compute.viewer"
```

9. Create a connect-register service account.

```
gcloud iam service-accounts create connect-register-sa \
--display-name "Connect-register Service Account" \
--project hv-ucp-anthos

gcloud iam service-accounts keys create connect-register-key.json \
--iam-account connect-register-sa@hv-ucp-anthos.iam.gserviceaccount.com \
--project hv-ucp-anthos
```

10. The connect-register service account must be granted the gkehub.admin role on your fleet host project. This is the Cloud project where you view and manage your clusters.

```
gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:connect-register-sa@hv-ucp-
anthos.iam.gserviceaccount.com" \
--role "roles/gkehub.admin"
```

11. Create a logging-monitoring service account.

```
gcloud iam service-accounts create logging-monitoring-sa \
--display-name "Logging-monitoring Service Account" \
--project=hv-ucp-anthos

gcloud iam service-accounts keys create logging-monitoring-key.json \
--iam-account logging-monitoring-sa@hv-ucp-anthos.iam.gserviceaccount.com \
--project hv-ucp-anthos
```

The logging-monitoring service account must be granted the following roles on your logging-monitoring project. This is the Cloud project where you view logs for your clusters.

- `stackdriver.resourceMetadata.writer`
- `opsconfigmonitoring.resourceMetadata.writer`
- `logging.logWriter`
- `monitoring.metricWriter`
- `monitoring.dashboardEditor`

```
gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:logging-monitoring-sa@hv-ucp-
anthos.iam.gserviceaccount.com" \
--role "roles/stackdriver.resourceMetadata.writer"

gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:logging-monitoring-sa@hv-ucp-
anthos.iam.gserviceaccount.com" \
--role "roles/opsconfigmonitoring.resourceMetadata.writer"

gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:logging-monitoring-sa@hv-ucp-
```

```
anthos.iam.gserviceaccount.com" \
--role "roles/logging.logWriter"

gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:logging-monitoring-sa@hv-ucp-
anthos.iam.gserviceaccount.com" \
--role "roles/monitoring.metricWriter"

gcloud projects add-iam-policy-binding hv-ucp-anthos \
--member "serviceAccount:logging-monitoring-sa@hv-ucp-
anthos.iam.gserviceaccount.com" \
--role "roles/monitoring.dashboardEditor"
If needed, use the following command to list the created service accounts:

[root@jp-gke-adminws ~]# gcloud iam service-accounts list

DISPLAY NAME
EMAIL                                                    DISABLED
Connect-register Service Account        connect-register-sa@hv-ucp-
anthos.iam.gserviceaccount.com    False
Component Access Service Account        component-access-sa@hv-ucp-
anthos.iam.gserviceaccount.com    False
Logging-monitoring Service Account      logging-monitoring-sa@hv-ucp-
anthos.iam.gserviceaccount.com  False
```

Note: In these examples, make sure to substitute your Project ID and service account name.

Deploy Anthos on-prem admin workstation on Hitachi UCP

An admin workstation is required to create Anthos clusters on VMware (GKE on-prem). The admin workstation is a standalone VM that is deployed within your Hitachi UCP cluster and is preinstalled with all the tools and resources required to create Anthos clusters on the VMware solution.

In this validation we used the `gkeadm` command-line tool, which is available for Linux, Windows, or MacOS. To deploy the admin workstation, follow these steps.

Procedure

1. Download the `gkeadm` tool from https://cloud.google.com/anthos/clusters/docs/on-prem/latest/how-to/download-gkeadm.

   Version 1.14.1 was the latest available at the time of this validation.

```
[root@jp-gke-adminws ~]# gsutil cp gs://gke-on-prem-release/gkeadm/1.14.1-gke.39/
linux/gkeadm ./
Copying gs://gke-on-prem-release/gkeadm/1.14.1-gke.39/linux/gkeadm...
\ [1 files][ 84.4 MiB/ 84.4 MiB]
Operation completed over 1 objects/84.4 MiB.
```

```
[root@jp-gke-adminws ~]# chmod +x gkeadm
```

2. Get the vCenter CA root certificate, which is used by gkeadm and GKE on-prem to authenticate to the vCenter.

```
true | openssl s_client -connect vcsoleng.sce.lab:443 -showcerts 2>/dev/null |
sed -ne '/-BEGIN/,/-END/p' > vcsoleng-sce-lab.pem
```

3. Copy the vCenter certificate file to the location of your choice. The path will be used on the configuration file when creating the admin workstation.
4. To view the decoded certificate, use the following command:

```
openssl x509 -in vcsoleng-sce-lab.pem -text -noout
```

Another way to get the certificate is described in *Getting your vCenter CA root certificate* at https://cloud.google.com/anthos/clusters/docs/on-prem/latest/how-to/vcenter-ca-cert-path.

5. Use the `gkeadm` tool to generate the following template configuration files: `credential.yaml` and `admin-ws-config.yaml`.

```
[root@jp-gke-adminws ~]# ./gkeadm create config
Created credential template at "credential.yaml".
Created config template at "admin-ws-config.yaml".
```

a. Update the `credential.yaml` file with the vCenter server's username and password:

```
kind: CredentialFile
# list of credentials
items:
# reference name for this credential entry
- name: vCenter
  username: "administrator@vsphere.local"
  password: "vCenterAdminPassword"
```

b. Update the `admin-ws-config.yaml` configuration file with the values specific to your environment:

- `Path to the JSON key file for your component access service account`
- `vCenter IP address or hostname, datacenter, datastore, cluster, resource pool, folder, and network`
- `Path to the root CA certificate for your vCenter server`
- `IP allocation mode: static or dhcp, in this case we used "static"`

- IP address, netmask, gateway, and DNS for the admin workstation
- NTP server address

The following is an example of the `admin workstation` file edited for this validation:

```
gcp:
  # Path of the component access service account's JSON key file
  componentAccessServiceAccountKeyPath: "/root/gke-files/component-access-
key.json"
# Specify which vCenter resources to use
vCenter:
  # The credentials and address GKE On-Prem should use to connect to vCenter
  credentials:
    address: "vcsoleng.sce.lab"
    # reference to vCenter credentials file
    fileRef:
      # read credentials from this file
      path: credential.yaml
      # entry in the credential file
      entry: vCenter
  datacenter: "scdc1"
  datastore: "vsp-1090-mgmt"
  cluster: "HA810G2-GKE-CL1"
  network: "DPortGroup-ha810g2-mgmt"
  # vSphere vm folder to deploy vms into. defaults to datacenter top level
folder
  folder: "ucp-gke"
  resourcePool: "Anthos-Resource-Pool"
  # Provide the path to vCenter CA certificate pub key for SSL verification
  caCertPath: "/root/gke-files/certs/lin/vcsoleng-sce-lab.pem"
# The URL of the proxy for the jump host
proxyUrl: ""
adminWorkstation:
  name: gke-admin-ws-221202-142644
  cpus: 4
  memoryMB: 8192
  # The boot disk size of the admin workstation in GB. It is recommended to
use a
  # disk with at least 100 GB to host images decompressed from the bundle.
  diskGB: 100
  # Name for the persistent disk to be mounted to the home directory
(ending in .vmdk).
  # Any directory in the supplied path must be created before deployment.
  dataDiskName: gke-on-prem-admin-workstation-data-disk/gke-admin-ws-221202-
142644-data-disk.vmdk
  # The size of the data disk in MB.
  dataDiskMB: 512
  network:
    # The IP allocation mode: 'dhcp' or 'static'
```

```
    ipAllocationMode: "static"
    # # The host config in static IP mode. Do not include if using DHCP
    hostConfig:
    #   # The IPv4 static IP address for the admin workstation
      ip: "10.76.47.16"
    #   # The IP address of the default gateway of the subnet in which the
admin workstation
    #   # is to be created
      gateway: "10.76.47.12"
    #   # The subnet mask of the network where you want to create your
admin workstation
    #   # (e.g. 255.255.255.0)
      netmask: "255.255.255.0"
    #   # The list of DNS nameservers to be used by the admin workstation
      dns:
      - "10.76.46.10"
  # The URL of the proxy for the admin workstation
  proxyUrl: ""
  ntpServer: "10.76.47.1"
```

6. Create the admin workstation using the following command:

```
[root@jp-gke-adminws ~]# ./gkeadm create admin-workstation
Using config file "admin-ws-config.yaml"...
Running preflight validations...
- Validation Category: Tools
    - [SUCCESS] gcloud
    - [SUCCESS] ssh
    - [SUCCESS] ssh-keygen
    - [SUCCESS] scp

- Validation Category: Config Check
...

- Validation Category: vCenter
    - [SUCCESS] Credentials
    - [SUCCESS] vCenter Version
    - [SUCCESS] ESXi Version
    - [SUCCESS] Datacenter
    - [SUCCESS] Datastore
    - [SUCCESS] Resource Pool
    - [SUCCESS] Folder
    - [SUCCESS] Network
    - [SUCCESS] Datadisk

All validation results were SUCCESS.

Downloading OS image ...
Creating admin workstation VM ...
...
****************************************
```

```
Admin workstation VM successfully created:
...
- SSH Key: /root/.ssh/gke-admin-workstation
****************************************
...

Preparing "admin-cluster.yaml" for gkectl...
Preparing "user-cluster.yaml" for gkectl...

********************************************************************
Admin workstation is ready to use.

Admin workstation information saved to /root/gke-admin-ws-221202-142644
This file is required for future upgrades
SSH into the admin workstation with the following command:
ssh -i /root/.ssh/gke-admin-workstation ubuntu@10.76.47.16
********************************************************************
```

7. Connect to the admin workstation.

   Use the command displayed in the previous output to SSH to your admin workstation. For example:

```
ssh -i /root/.ssh/gke-admin-workstation ubuntu@10.76.47.16
```

8. After you are connected to the admin workstation, verify that the following generated files are in the home directory:

   - `admin-cluster.yaml` — a template config file for creating your admin cluster.
   - `user-cluster.yaml` — a template config file for creating your user cluster.
   - The JSON key for the component service account. If you let `gkeadm` create the service accounts (when using the `--auto-create-service-accounts` flag), the folder should have all the JSON key files. Otherwise you must manually copy the remaining JSON key files from the client workstation to the Anthos admin workstation. Make note of the name and path because you will need them later to create the clusters.
   - `credential.yaml` — a template config file with vCenter credentials. This file needs to be updated with the load balancer (for example F5 BigIP) and private registry credentials.
   - vCenter cert file

Deploy Anthos on-prem admin cluster on Hitachi UCP

An admin cluster must be created before creating any user cluster to run your workloads. The admin cluster runs the Kubernetes control plane for the admin cluster itself and for the user clusters.

Follow these steps to create Anthos user clusters.

Procedure

1. On the admin workstation, make a copy of the admin-cluster.yaml template with a new name (for example admin-cluster-ucp.yaml) and start editing with the corresponding IP addresses and load balancing information.

   Most of the fields are already filled in with the values used when you created the admin workstation.

   See Appendix A: Example User Cluster Configuration File (on page 51) for an example of the `admin-cluster-ucp.yaml` file used for this validation.

2. When the edits are complete, run the following command to validate the configuration file:

   ```
   gkectl check-config --config admin-cluster-ucp.yaml
   ```

3. After the configuration checks have passed, run the following command to initialize your vSphere environment.

   This will import the OS images to vSphere and mark them as templates. If an issue is identified during the configuration check, and if the issue has already been remediated, you can skip the validation using the `--skip-validation-all` flag.

   ```
   gkectl prepare --config admin-cluster-ucp.yaml --skip-validation-all
   ```

4. If you have chosen to use Seesaw load balancer, create and configure the VMs for your Seesaw load balancer with the following command; otherwise skip this command:

   ```
   gkectl create loadbalancer --config admin-cluster-ucp.yaml
   ```

5. Create the Anthos admin cluster using the following command:

   ```
   gkectl create admin --config admin-cluster-ucp.yaml --skip-validation-all
   ```

   The `gkectl` command creates a kubeconfig file named `kubeconfig` in the current directory. This is the kubeconfig file that must be used to interact with the admin cluster using kubectl or run a diagnosis with `gkectl`. For example, you can list the cluster or list the nodes in the admin cluster using `kubectl`.

   The following is the output for these commands:

   ```
   ubuntu@gke-admin-ws-221202-142644:~$ kubectl --kubeconfig kubeconfig get clusters
   NAME            AGE
   gke-admin-ucp   64d

   ubuntu@gke-admin-ws-221202-142644:~$ kubectl --kubeconfig kubeconfig get nodes
   NAME                                          STATUS
   ROLES                  AGE   VERSION
   gke-admin-master-75tlg                        Ready    control-plane,
   master   17h   v1.25.5-gke.100
   gke-admin-node-6cf77f44f4-hzkhf               Ready
   <none>                 16h   v1.25.5-gke.100
   gke-admin-node-6cf77f44f4-t929t               Ready
   <none>                 16h   v1.25.5-gke.100
   ```

Deploy Anthos on-prem user clusters on Hitachi UCP

User clusters can be created using Anthos on-prem API clients, `gkectl`, and Control plane V2. For this validation we created the clusters using the `gkectl` methods described in *Create a user cluster* at https://cloud.google.com/anthos/clusters/docs/on-prem/latest/how-to/create-user-cluster.

See Appendix A: Example User Cluster Configuration File (on page 51) for an example of the `admin-cluster-ucp.yaml` file used for this validation.

Follow these steps to create Anthos user clusters.

Procedure

1. On the admin workstation, make a copy of the user-cluster.yaml template with a new name (for example `user-cluster-1.yaml`) and start editing with the corresponding IP addresses, load balancing information, cluster name, and service accounts.

   Most of the fields are already filled in with the values used when you created the admin workstation.

   See Appendix A: Example User Cluster Configuration File (on page 51) for an example of the `admin-cluster-ucp.yaml` file used for this validation.

2. When the edits are complete, run the following command to validate the configuration file:

```
gkectl check-config --kubeconfig ADMIN_CLUSTER_KUBECONFIG --config
USER_CLUSTER_CONFIG
Replace the ADMIN_CLUSTER_KUBECONFIG  with the path of the kubeconfig file for
your admin cluster, and the USER_CLUSTER_CONFIG with the file name of your user
cluster configuration file as shown in the following example.
gkectl check-config --kubeconfig kubeconfig --config user-cluster-1.yaml
```

3. If you have chosen to use Seesaw load balancer, create and configure the VMs for your Seesaw load balancer with the following command, otherwise skip this command:

```
gkectl create loadbalancer --kubeconfig kubeconfig --config user-cluster-1.yaml
```

4. Create the first Anthos user cluster using the following command:

```
gkectl create cluster --kubeconfig kubeconfig --config user-cluster-1.yaml
```

   The `gkectl` tool creates a kubeconfig file named `USER_CLUSTER_NAME-kubeconfig` in the current directory. This is the kubeconfig file that must be used to interact with the user cluster using `kubectl` or run a diagnosis with `gkectl`. For example, you can list the cluster or list the nodes in the user cluster using `kubectl`.

   The following is the output for these commands:

```
ubuntu@gke-admin-ws-221202-142644:~$ kubectl --kubeconfig anthos-user-ucpcluster-
1-kubeconfig get clusters
NAME                        AGE
anthos-user-ucpcluster-1    64d
```

```
ubuntu@gke-admin-ws-221202-142644:~$ kubectl --kubeconfig anthos-user-ucpcluster-
1-kubeconfig get nodes
NAME                                              STATUS   ROLES    AGE   VERSION
anthos-user-ucpcluster-1-pool-1-787b9d7d4f-rgvpx   Ready    <none>   21h
v1.25.5-gke.100
anthos-user-ucpcluster-1-pool-1-787b9d7d4f-sjvrd   Ready    <none>   21h
v1.25.5-gke.100
anthos-user-ucpcluster-1-pool-1-787b9d7d4f-zfsdz   Ready    <none>   21h
v1.25.5-gke.100
```

Also, the `gkectl` tool can be used to diagnose the cluster:

```
ubuntu@gke-admin-ws-221202-142644:~$ gkectl diagnose cluster --kubeconfig
kubeconfig --cluster-name anthos-user-ucpcluster-1
Preparing for the diagnose tool...
Diagnosing the cluster......  DONE
Diagnose result is saved successfully in /home/ubuntu/diagnose-user-anthos-user-
ucpcluster-1-20230209235440.json

- Validation Category: User Cluster F5 BIG-IP
Checking f5 (credentials, partition)...SUCCESS

- Validation Category: OS Images
Checking User cluster OS images exist...SUCCESS

- Validation Category: VCenter
Checking Credentials...SUCCESS
Checking VSphere CSI Driver...SUCCESS
Checking vCenter Version...SUCCESS
Checking ESXi Version...SUCCESS
Checking Datacenter...SUCCESS
Checking Resource pool...SUCCESS
Checking Folder...SUCCESS
Checking Network...SUCCESS

- Validation Category: Datastore
Checking Datastore...SUCCESS

- Validation Category: Cluster Healthiness
Checking user cluster and node pools...SUCCESS
Checking user cluster certificates...SUCCESS
...
Checking anthos-identity-service pods...SUCCESS
Checking gke-managed-metrics-server pods...SUCCESS
Checking cert-manager pods...SUCCESS
Checking kube-public pods...SUCCESS
Checking GKE Hub Membership...SUCCESS
Checking all poddisruptionbudgets...SUCCESS
Checking storage...SUCCESS
Checking resource...SUCCESS
```

```
Checking virtual machine resource contention...SUCCESS
Checking host resource contention...SUCCESS

- Validation Category: Connectivity
Checking VMs TOD (availability)...SUCCESS
Some validations were SKIPPED. Check the report above.
Cluster is healthy!
ubuntu@gke-admin-ws-221202-142644:~$
```

5. To create additional user clusters in your solution, follow these steps:
   a. Copy the original `user-cluster.yaml` template or the configuration file used for user-cluster-1 to a new file (for example, `user-cluster-2.yaml`) and start editing with the corresponding IP addresses, load balancing information, and new user cluster name.
   b. When the edits are complete, run the following command to validate the configuration file:

```
gkectl check-config --kubeconfig kubeconfig --config user-cluster-2.yaml
```

   c. Create an additional Anthos user cluster with the following command:

```
gkectl create cluster --kubeconfig kubeconfig --config user-cluster-1.yaml
```

   d. Verify the new cluster and its nodes. Make sure to use the kubeconfig file corresponding to the newly created user cluster:

```
ubuntu@gke-admin-ws-221202-142644:~$ kubectl --kubeconfig anthos-user-
ucpcluster-2-kubeconfig get clusters
NAME                        AGE
anthos-user-ucpcluster-2   8d

ubuntu@gke-admin-ws-221202-142644:~$ kubectl --kubeconfig anthos-user-
ucpcluster-2-kubeconfig get nodes
NAME                                                STATUS   ROLES    AGE
VERSION
anthos-user-ucpcluster-2-pool-1-5cb9b895dc-4czj8   Ready    <none>   17h
v1.25.5-gke.100
anthos-user-ucpcluster-2-pool-1-5cb9b895dc-nr7jz   Ready    <none>   17h
v1.25.5-gke.100
anthos-user-ucpcluster-2-pool-1-5cb9b895dc-sbdjb   Ready    <none>   17h
v1.25.5-gke.100
```

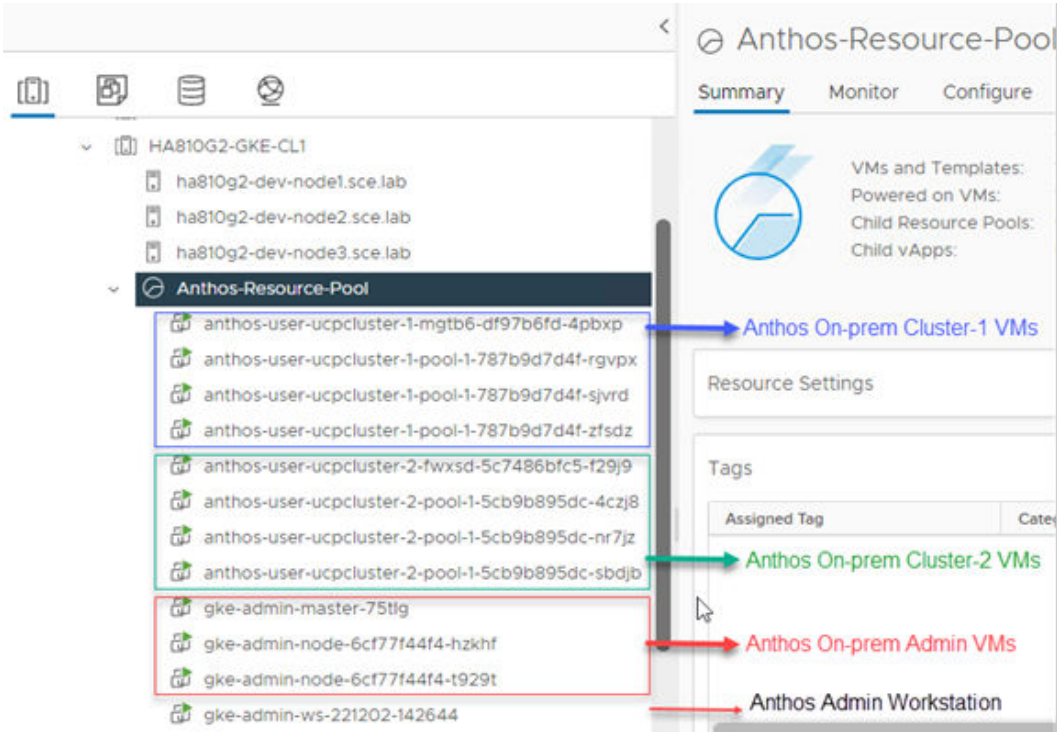
   In addition to the kubectl, you can use the `gkectl` tool to list additional details about the clusters:

```
ubuntu@gke-admin-ws-221202-142644:~$ gkectl list admin --kubeconfig
kubeconfig
NAME            AGE   VERSION
gke-admin-ucp   66d   1.14.1-gke.39
```

```
ubuntu@gke-admin-ws-221202-142644:~$ gkectl list clusters --kubeconfig
kubeconfig
NAMESPACE                                    NAME
READY   STATE     AGE   VERSION
anthos-user-ucpcluster-1-gke-onprem-mgmt   anthos-user-ucpcluster-1
True    RUNNING   66d   1.14.1-gke.39
anthos-user-ucpcluster-2-gke-onprem-mgmt   anthos-user-ucpcluster-2
True    RUNNING   9d    1.14.1-gke.39
```

You can see all the deployed Anthos VMs on the vCenter that is managing the Hitachi UCP/Anthos environment. These include the Anthos admin workstation, the admin cluster VMs, and the user cluster VMs under the resource pool defined for the Anthos on-prem environment.

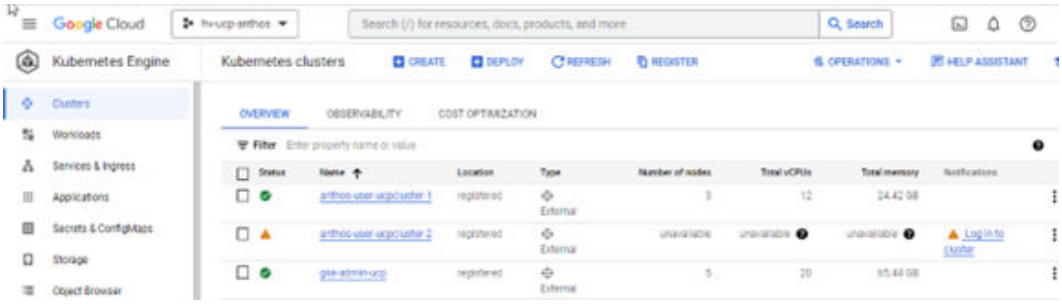
For this validation, the clusters were created using the default number of nodes. The following illustration shows the default number of nodes for each of the clusters. Also, a master VM is deployed and added to the admin cluster for each user cluster.



Manage clusters from the Google Cloud console

Anthos clusters on VMware are registered to a fleet with Google Cloud at creation time, and they are displayed in the console in your fleet host project, along with other fleet clusters such as GKE on Google Cloud. All your clusters are displayed on a single dashboard on the Anthos Clusters and the GKE Clusters pages in the console.

The following illustration shows a view of the Anthos on-prem clusters on the Google Cloud console.



To manage the Anthos clusters from Google Cloud console, you must set up authentication and grant some specific roles so you can log in to these clusters directly from the Google Cloud console. There are different authentication methods as described in *Manage clusters from the Google Cloud console* at https://cloud.google.com/anthos/clusters/docs/on-prem/latest/how-to/connect-to-cluster-console. For this validation we used the bearer token authentication method.

Complete the following steps to enable access to the Anthos on-prem clusters.

Procedure

1. Grant IAM roles for access through the Google Cloud console.

   The recommended roles are:

   - roles/container.viewer
   - roles/gkehub.viewer
   - roles/gkeonprem.admin

   In the following example, change the project ID and user's email based on your organization:

```
gcloud projects add-iam-policy-binding hv-ucp-anthos \
    --member="user:jose.perez2@hitachivantara.com" \
    --role=roles/container.viewer
gcloud projects add-iam-policy-binding hv-ucp-anthos \
    --member="user:jose.perez2@hitachivantara.com" \
    --role=roles/gkehub.viewer
gcloud projects add-iam-policy-binding hv-ucp-anthos \
      --member="user:jose.perez2@hitachivantara.com" \
      --role=roles/gkeonprem.admin
```

2. Configure role-based access control (RBAC).

   a. Create a `cloud-console-reader.yaml` file and apply it to the cluster:

```
cat <<EOF > cloud-console-reader.yaml
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: cloud-console-reader
rules:
```

```
- apiGroups: [""]
  resources: ["nodes", "persistentvolumes", "pods"]
  verbs: ["get", "list", "watch"]
- apiGroups: ["storage.k8s.io"]
  resources: ["storageclasses"]
  verbs: ["get", "list", "watch"]
EOF
```

b. Apply this clusterRole to the cluster. Make sure the kubeconfig file corresponds to the cluster to which you want to log in. The following is an example for cluster-2:

```
kubectl apply -f cloud-console-reader.yaml --kubeconfig anthos-user-
ucpcluster-2-kubeconfig
```

c. Create and authorize a Kubernetes service account (KSA):

```
KSA_NAME=KSA_NAME
kubectl create serviceaccount ${KSA_NAME}
kubectl create clusterrolebinding VIEW_BINDING_NAME \
   --clusterrole view --serviceaccount default:${KSA_NAME}
kubectl create clusterrolebinding CLOUD_CONSOLE_READER_BINDING_NAME \
   --clusterrole cloud-console-reader --serviceaccount default:${KSA_NAME}
```

The following is an example for cluster-2 created in the previous steps:

```
KSA_NAME=ucp-gke-user01
kubectl create serviceaccount ${KSA_NAME} --kubeconfig anthos-user-
ucpcluster-2-kubeconfig

kubectl create clusterrolebinding ucp-gke-user-view \
   --clusterrole view --serviceaccount default:${KSA_NAME} --kubeconfig
anthos-user-ucpcluster-2-kubeconfig

kubectl create clusterrolebinding ucp-gke-user-cloudconsole-reader \
   --clusterrole cloud-console-reader --serviceaccount default:${KSA_NAME} -
-kubeconfig anthos-user-ucpcluster-2-kubeconfig
```

If admin permissions are needed, such as deploying a Kubernetes application from Cloud Marketplace, bind the cluster-admin role to the KSA:

```
kubectl create clusterrolebinding ucp-gke-user-admin \
   --clusterrole cluster-admin --serviceaccount default:${KSA_NAME} --
kubeconfig anthos-user-ucpcluster-2-kubeconfig
```

3. After the service accounts and role bindings have been created, retrieve the KSA's bearer token with the following commands:

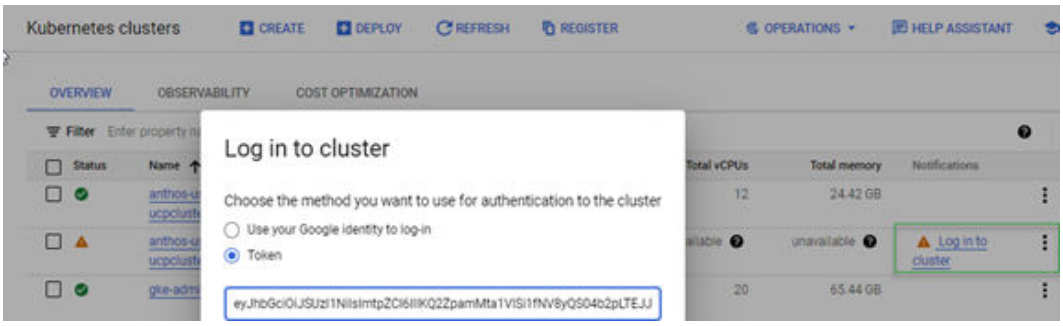
```
SECRET_NAME=${KSA_NAME}-token

kubectl apply -f - << __EOF__
apiVersion: v1
```

```
kind: Secret
metadata:
  name: "${SECRET_NAME}"
  annotations:
    kubernetes.io/service-account.name: "${KSA_NAME}"
type: kubernetes.io/service-account-token
__EOF__

until [[ $(kubectl get -o=jsonpath="{.data.token}" "secret/${SECRET_NAME}") ]];
do
  echo "waiting for token..." >&2;
  sleep 1;
done

kubectl get secret ${SECRET_NAME} -o jsonpath='{$.data.token}' | base64 --decode
```

The following example shows how to get the bearer token for ucpcluster-1:

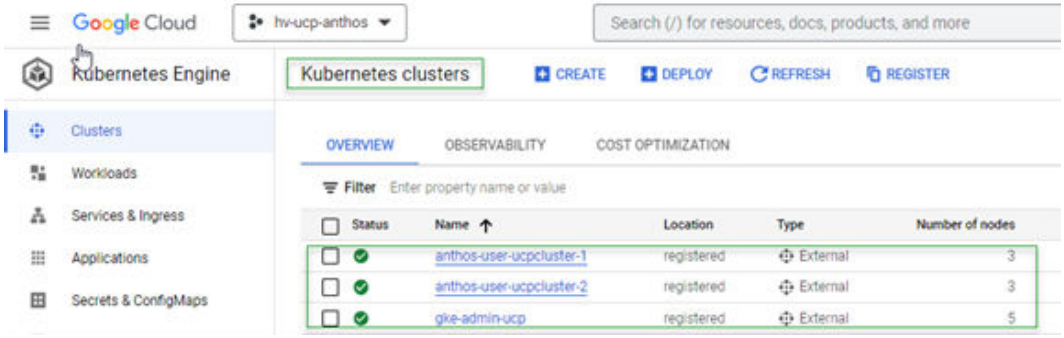
```
ubuntu@gke-admin-ws-221202-142644:~$ SECRET_NAME=ucp-gke-user01-token
ubuntu@gke-admin-ws-221202-142644:~$ kubectl get secret ${SECRET_NAME} --
kubeconfig anthos-user-ucpcluster-2-kubeconfig -o jsonpath='{$.data.token}' |
base64 --decode
eyJhbGciOiJSUzI1NiIsImtpZCI6IkU2UzhSc1lYUUR1dnplQmkxZnlYWlhPRjNLVjRMRnI2R2M0VjdOY
zBJRkEifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ
2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJkZWZhdWx0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3Vu
dC9zZWNyZXQubmFtZSI6InVjcC1na2UtdXNlcjAxLXRva2VuIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlY
WNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6InVjcC1na2UtdXNlcjAxIiwia3ViZXJuZXRlcy5pby
9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQudWlkIjoiMDc1M2Q2NTItMDg0OC00YmQ3LTgzMjA
tYTJkOWNhMjk5YWFjIiwic3ViIjoic3lzdGVtOnNlcnZpY2VhY2NvdW50OmRlZmF1bHQ6dWNwLWdrZS11
c2VyMDEifQ.e7ePsPTLh2hPQrqUKis06VQOT9Vi6wUTUeYYucv0rObyJgqUbQrrLYs68C8Bvb9pUbYGCs
e0RsJwO9xEAqyesUcuxtBH41TaJreWjgAB-25M7ZCXA0GM-igUcKvGf7JROcvq5QTz1Hbl9-
4h6G7uvLLnDU2lDlrVyNcOX6rbi3sH6duVGS0Di-
PX3MMFeXMz3NtJfCoddl5ZCetHzZV1TJVqKDjJ2U0qhsT003x6vlBuzqZDVlAqcDfnp_Tz2auwh0od4uQ
IbgJ_8jQp0FvgPcwiC--qB7etNUiAMYfmh9V6AwPpFvZiAHD7h5UuivBvW--AGGubVY6dWW_ROhU2aQ
```

4. Copy the token and save so it can be used to log in to the Google Cloud console.
   a. In the Google Cloud console, on the GKE Clusters page, click the 3 dots next to the registered cluster and click **Log in**, select **Token**, enter the token obtained in the previous step, and then click **Login**.
   b. In the Google Cloud console, on the Anthos Clusters page, select the cluster, click **Log in**, select **Token**, enter the token obtained in the previous step, and then click **Login**.

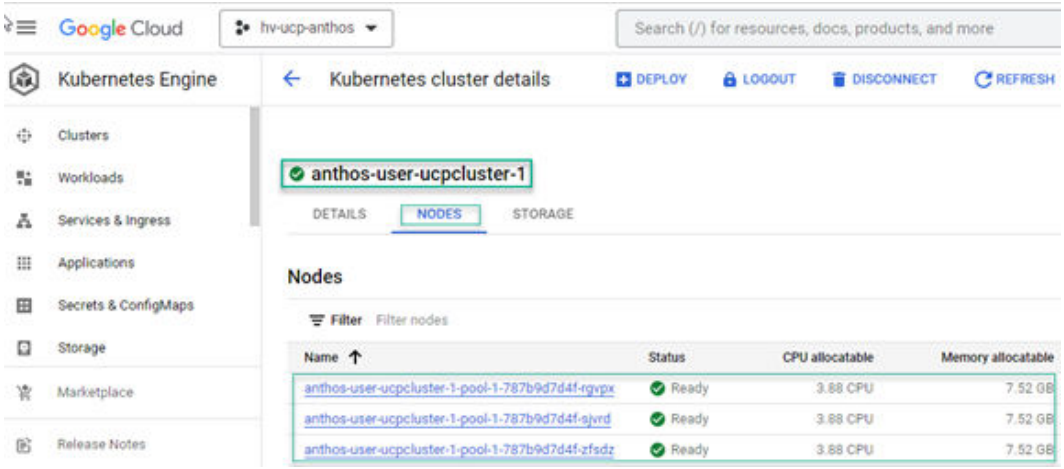


5. Repeat this process for each of the Anthos clusters you want to manage from the Google Cloud console.

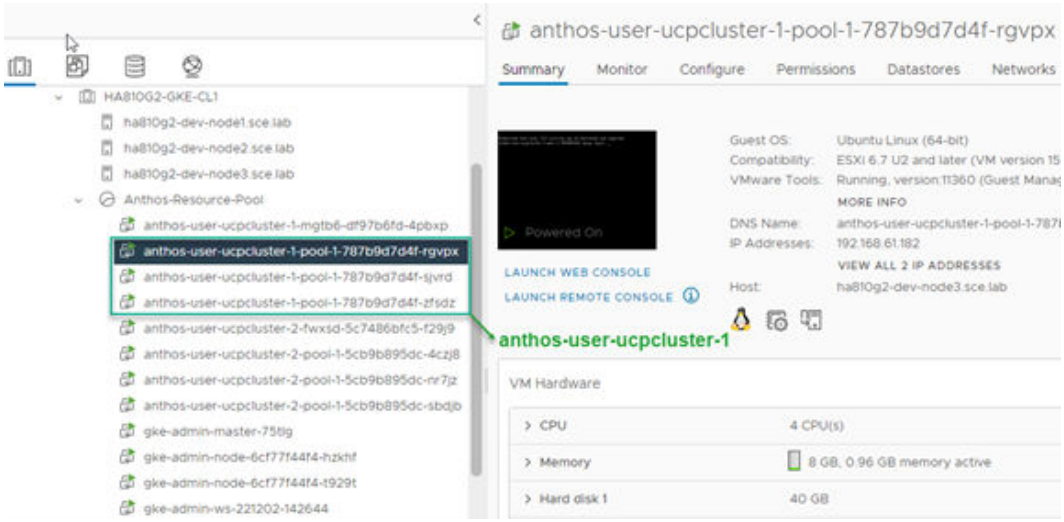
   The following is a view of the Anthos on-prem clusters running as vSphere VMs on Hitachi UCP platform, all with green check marks.



6. Click a specific cluster to display additional information about the cluster such as cluster nodes, Kubernetes version, storage classes, and persistent volumes.



   The same cluster worker nodes can be seen with the same name on the vCenter/UCP environment.



Solution validation

If you have followed the guidance in the Solution Design section, your infrastructure is prepared, and you can try these example deployments. This reference architecture was validated by the following:

- Deploying stateful applications on Anthos on-prem clusters, deployed on a Hitachi UCP platform, using the Google Cloud console and Google Cloud Marketplace.
- Connecting and registering an existing Red Hat OpenShift cluster deployed on Hitachi UCP platform, leveraging the Anthos attached cluster features to demonstrate how easy it is to connect, register any Kubernetes cluster running anywhere, and manage from a single-pane-of-glass using the Google Cloud console.

Deploy applications on Anthos on-prem on UCP

After the Anthos on-prem clusters have been deployed and registered, you can start deploying workloads using the Google Cloud console or the command line.

Google Cloud Marketplace is a catalog of curated container applications that you can use for easy deployment to your Anthos clusters running anywhere.

Deploy a multi-instance of MariaDB with Persistent Volumes on Hitachi Virtual Storage Platform

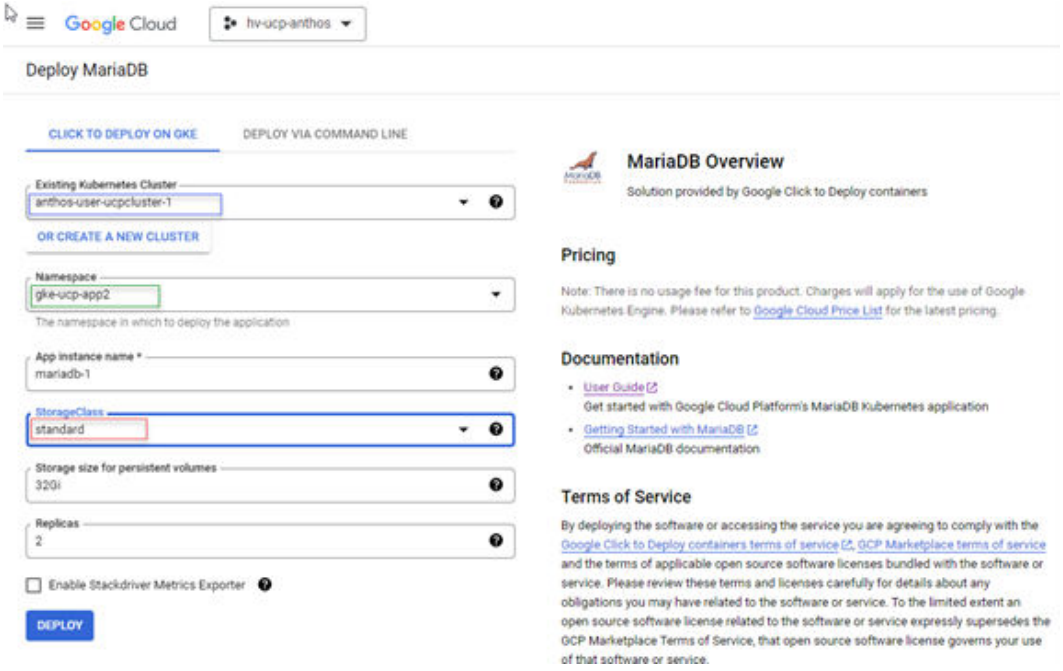
This example deploys a stateful multi-instance MariaDB with replication. The deployment includes two StatefulSets, a primary (read/write access), and a secondary (read-only access).

As indicated previously, we can deploy an application with a few clicks from the Google Cloud console using Google Cloud Marketplace.

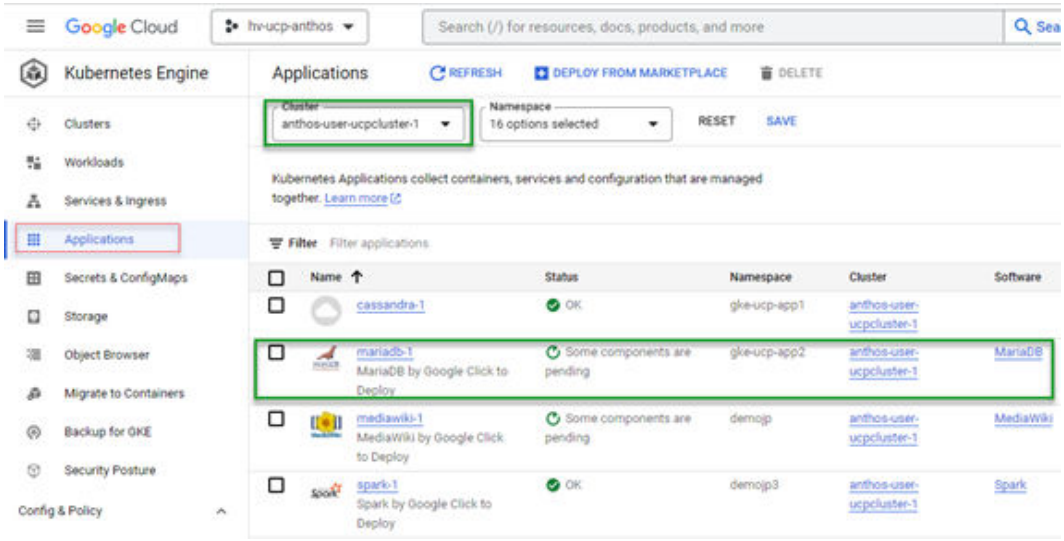
The following example shows how to quickly deploy MariaDB using Google Cloud console.

Procedure

1. On the Google Cloud console, click **Marketplace** and then select **Kubernetes Apps**.
   You will be presented with a list of available applications that are ready to deploy.
2. Click the MariaDB app.
3. Click **Configure** and do the following:
   a. Select the cluster.
   b. Select or create a namespace.
   c. Enter the app instance name.
   d. Select the storage class.
   e. Enter the capacity for the persistent volumes and number of replicas.

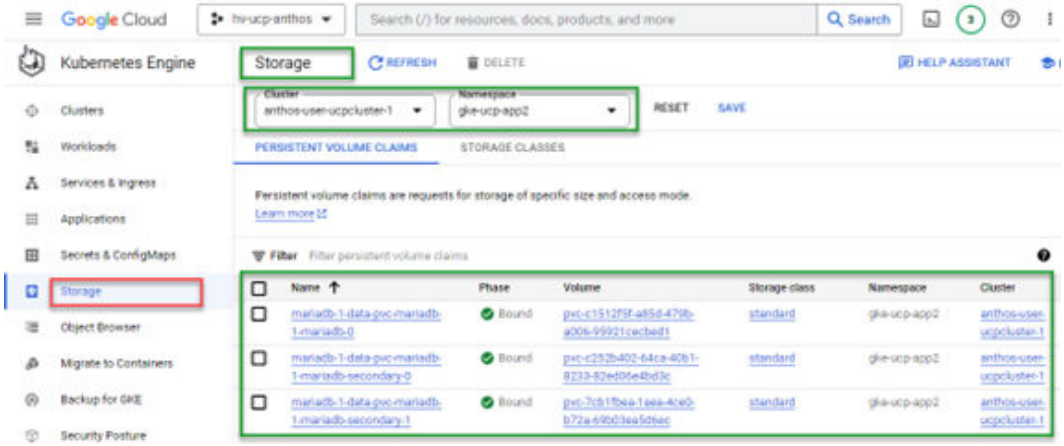


4. Click **Deploy** to start the deployment process.
5. After the deployment is complete, click Applications to verify, and filter by cluster if needed.
   The following illustration shows the new MariaDB app deployed into the `Anthos-user-ucpcluster-1 cluster`.

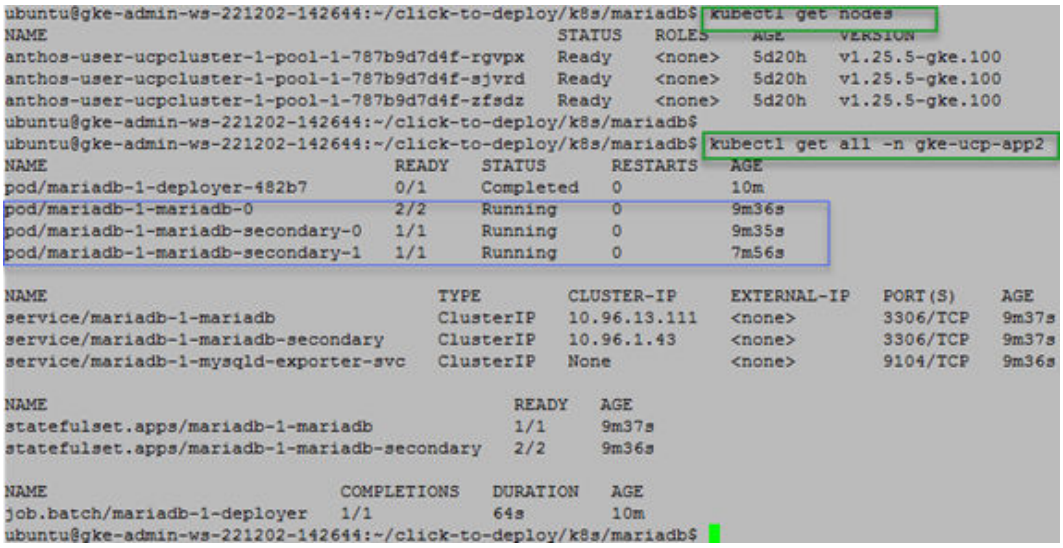


6. To see the persistent volumes claims you can select Storage and then filter by cluster and even namespace.

   In the following illustration you can see the PVCs/PVs using the StorageClass standard, which is backed by a VMFS datastore on the Hitachi UCP cluster using Hitachi Virtual Storage Platform.



You can also see these details from the command line using the `kubectl` tool.



Connect and manage an on-prem OCP cluster with Anthos and Google Cloud

Anthos clusters on VMware, Anthos clusters on bare metal, and multi-cloud Anthos clusters (AWS and Azure) are automatically registered to your project fleet on Google Cloud when they are created. However, GKE clusters on Google Cloud, EKS clusters (AWS), AKS clusters (Azure), and other third-party Kubernetes clusters (also called attached clusters) must be manually registered to join your project fleet on Google Cloud.

Use the Anthos attached cluster feature to manage any standard, Cloud Native Computing Foundation (CNCF) compliant Kubernetes cluster from the Google Cloud console, across multiple cloud providers, along with your Anthos clusters, and enable Anthos features such as centralized configuration control with Anthos Config Management and Microservices with Anthos Service Mesh.

The following is a summary of the steps required to register third party Kubernetes clusters into your project fleet on Google Cloud. For specific details see *Anthos attached clusters* at https://cloud.google.com/anthos/clusters/docs/multi-cloud/attached.

Procedure

1. Download and install Google Cloud CLI, and then use `gcloud` for registration.
2. Install `kubectl`.

   The recommendation is to install `kubectl` with Google Cloud CLI.
3. Enable APIs.

   The following APIs are required to be enabled in your fleet host project:

   - `container.googleapis.com`
   - `gkeconnect.googleapis.com`
   - `gkehub.googleapis.com`, also known as the Fleet API. This is the Google Cloud service that manages cluster registration and fleet membership.
   - `cloudresourcemanager.googleapis.com`

4. Grant access permissions.

   Cluster registration requires both permission to register the cluster, and admin permissions on the cluster itself.
5. Create a Google Cloud service account and create a JSON key file that contains the service account credentials. Make sure to bind the corresponding roles.
6. For Red Hat OpenShift, create a custom Security Context Constraints (SCCs) before registering the cluster to allow installing Connect Agent in your OCP cluster.
7. To register the third-party cluster, run the following command:

```
gcloud container hub memberships register [MEMEBERSHIP_NAME] \
                --context=[CLUSTER_CONTEXT] \
                --service-account-key-file=[LOCAL_KEY_PATH] \
                --kubeconfig=[KUBECONFIG_PATH] \
                --project=[PROJECT_ID}
```

   Replace the following:

   - MEMBERSHIP_NAME: the name that you choose for your cluster being registered to the fleet.
   - SERVICE_ACCOUNT_KEY_PATH: the local file path to the service account's downloaded private key JSON file.
   - KUBECONFIG_CONTEXT: the cluster context of the cluster being registered as it appears in the kubeconfig file.
   - KUBECONFIG_PATH: the local file path where your kubeconfig containing an entry for the cluster being registered is stored.

   The following is an example of the registration of an on-prem OCP cluster deployed on top of Hitachi UCP:

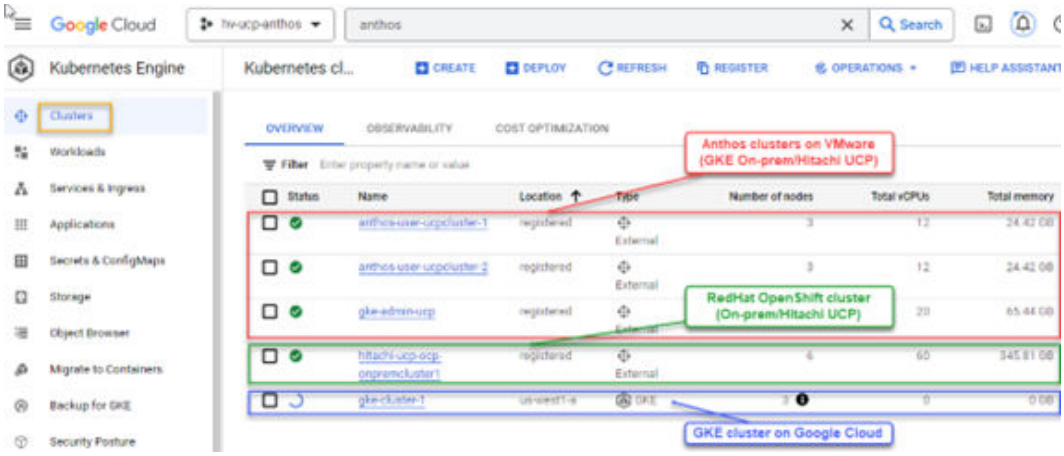
```
[ocpinstall@jpc3-ocp-admin-ws gke-files]$ gcloud container hub memberships
register hitachi-ucp-ocp-onpremcluster1 \
> --context=default/api-jpc3-ocp-hvlab-local:6443/cluster_admin \
> --service-account-key-file=/home/ocpinstall/gke-files/connect-register-
key.json \
> --kubeconfig=/home/ocpinstall/ocp-upi-install/auth/kubeconfig \
> --project=hv-ucp-anthos
Waiting for membership to be created...done.
Created a new membership [projects/hv-ucp-anthos/locations/global/memberships/
hitachi-ucp-ocp-onpremcluster1] for the cluster [hitachi-ucp-ocp-onpremcluster1]
Generating the Connect Agent manifest...
Deploying the Connect Agent on cluster [hitachi-ucp-ocp-onpremcluster1] in
namespace [gke-connect]...
Deleting namespace [gke-connect] in the cluster...done.
Deployed the Connect Agent on cluster [hitachi-ucp-ocp-onpremcluster1] in
namespace [gke-connect].
Finished registering the cluster [hitachi-ucp-ocp-onpremcluster1] with the fleet.
[ocpinstall@jpc3-ocp-admin-ws gke-files]$
```

8. After registration, verify that the Connect Agent is running on the namespace `gke-connect`:

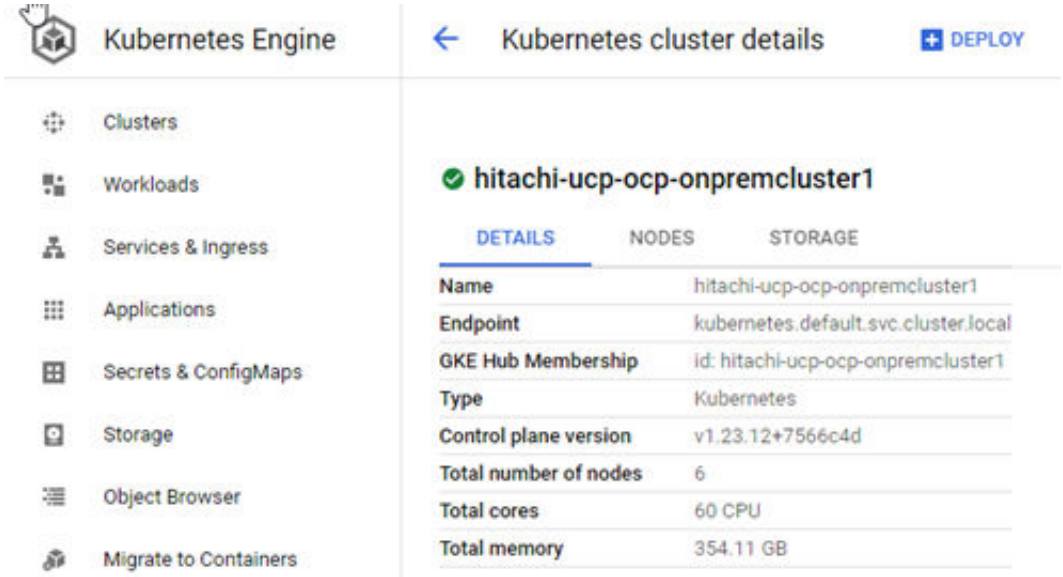
```
oc get all -n gke-connect
```

```
[ocpinstall@jpc3-ocp-admin-ws gke-files]$ oc get all -n gke-connect
NAME                                                  READY   STATUS    RESTARTS   AGE
pod/gke-connect-agent-20230127-00-00-8b44b8975-8htdz  1/1     Running   0          5d3h
pod/gke-connect-agent-20230127-00-00-8b44b8975-mhgsx  1/1     Running   0          5d3h

NAME                                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
service/gke-connect-monitoring       ClusterIP   172.30.167.214   <none>        8080/TCP   5d3h

NAME                                                 READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/gke-connect-agent-20230127-00-00     2/2     2            2           5d3h

NAME                                                        DESIRED   CURRENT   READY   AGE
replicaset.apps/gke-connect-agent-20230127-00-00-8b44b8975  2         2         2       5d3h
[ocpinstall@jpc3-ocp-admin-ws gke-files]$
```

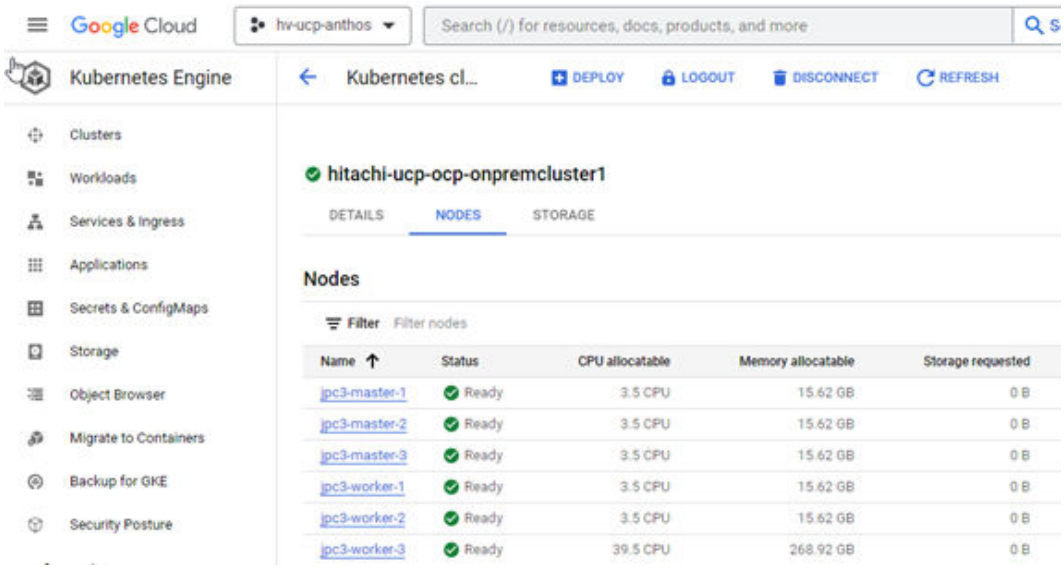
After the registration is complete, your cluster will appear in the GKE and Anthos cluster pages in the Google Cloud console, with the other Anthos clusters, as shown.



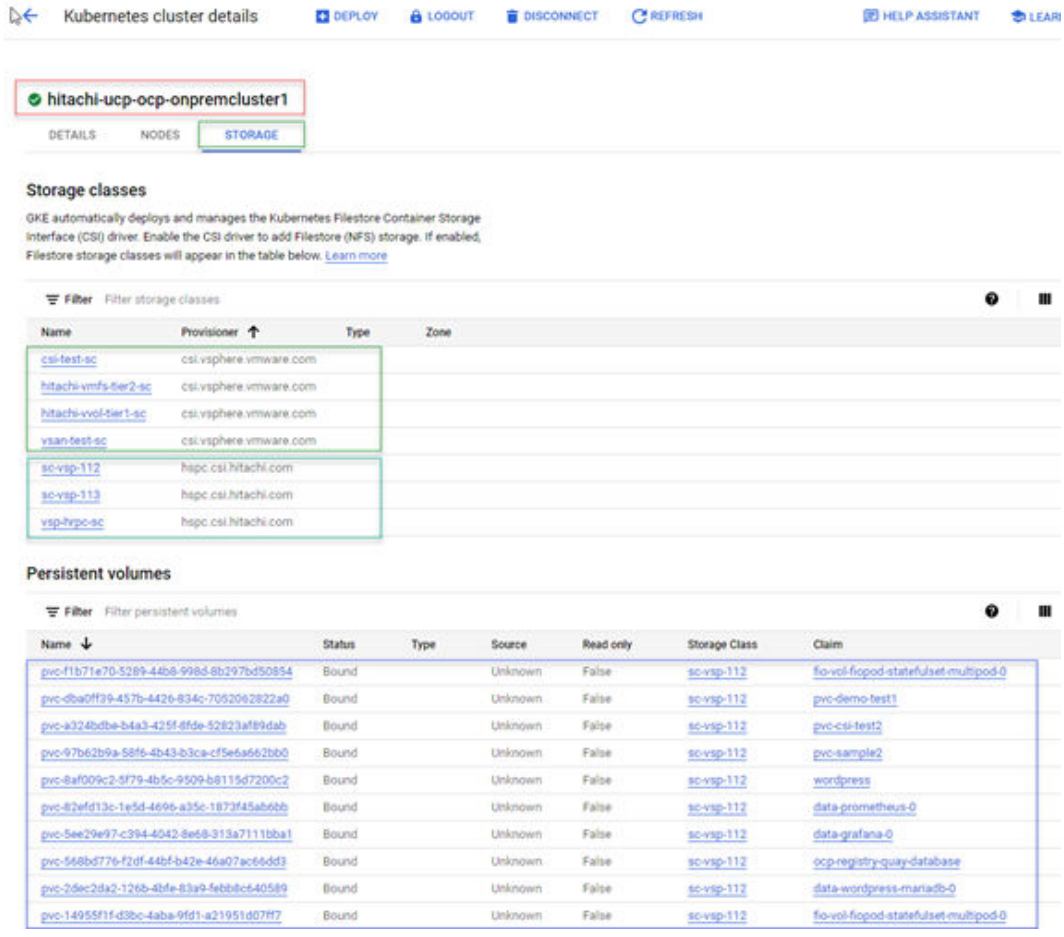
From here we can see more details of the cluster:



The following shows the cluster nodes:



Additional details can be seen from the cluster such as Storage Classes and Persistent Volumes.



Result

At this point you can enable any of the supported Anthos features such as Anthos Service Mesh and Anthos Config Management.

Conclusion

This reference architecture validates how Hitachi Unified Compute Platform, Hitachi Virtual Storage Platform, VMware, and Google Cloud Anthos combine to deliver a powerful and flexible Kubernetes platform for a secure and enterprise-ready hybrid multi-cloud solution.

For customers looking to implement an enterprise class hybrid multi-cloud solution, Hitachi UCP with Google Anthos provides the best platform that can integrate with other cloud providers, leverage existing hardware investments in their own data centers, and manage with a modern hybrid multi-cloud framework through a single pane of glass.

Product descriptions

This section includes information about the hardware and software components used in this solution.

Unified Compute Platform CI

Hitachi Unified Compute Platform CI (UCP CI) is an optimized and preconfigured converged infrastructure platform. It offers a broad range of compute and storage components that can be scaled and configured independently to eliminate overprovisioning. With Unified Compute Platform CI, you can optimize your data center to run any container application workload, at any scale.

Unified Compute Platform HC

Unified Compute Platform HC (UCP HC) is an integrated turnkey appliance that combines compute, storage, and virtualization to deliver certainty for edge to core to cloud operations. This market-proven Hitachi solution provides a scalable, seamless, and simplified cloud foundation for enterprise and mid-market customers. Advanced automation and intelligence for day 0-2 operations accelerate innovation and improve productivity while lowering the TCO.

Hitachi Unified Compute Platform RS

To simplify your hybrid cloud journey, Hitachi Unified Compute Platform RS (UCP RS) provides a turnkey solution that reduces total cost of ownership (TCO) and improves security. The software-defined data center solution accelerates the time to market with a natively integrated cloud infrastructure stack. It comes prepackaged with management software, to provide automated, policy-based IT operations.

UCP RS has automation that enables the deployment of an entire cloud infrastructure in hours, not weeks or months. There is rapid and repeatable application deployment.

Move your workload across data centers to meet changing business needs. Manage your applications across private and public cloud from a common toolset. Scale your data center without increasing IT headcount. Automate your data center with policies.

There is a hypervisor-enabled firewall with Unified Compute Platform CI for enhanced security. Granular security prevents east-west breach. Security policies align with workload, regardless of physical location.

Hitachi Virtual Storage Platform E1090

The Hitachi Virtual Storage Platform E1090 (VSP E1090) storage system is a high-performance, large-capacity data storage system. The VSP E1090 all-flash arrays (AFAs) support NVMe and SAS solid-state drives (SSDs). The VSP E1090H hybrid models can be configured with both SSDs and hard disk drives (HDDs).

- The NVMe flash architecture delivers consistent, low-microsecond latency, which reduces the transaction costs of latency-critical applications and delivers predictable performance to optimize storage resources.
- The hybrid architecture allows for greater scalability and provides data-in-place migration support.

Hitachi Advanced Server

Designed to unlock the full benefits of the hybrid cloud, Hitachi Advanced Server models deliver high performance and enhanced security while reducing operational costs. Global enterprises, cloud service providers, and governments trust Hitachi servers to run bare metal, virtualized, or containerized applications. Powered by industry-leading Intel Xeon Scalable Processors, Hitachi servers are ideal to deliver edge, core, and cloud IT services.

Hitachi servers are designed and optimized to maximize performance for VMware, Oracle, Virtual Desktop Infrastructure (VDI), SAP, analytics, high-performance computing (HPC), and other enterprise workloads.

Hitachi compute systems introduce the NVIDIA graphics acceleration for artificial intelligence (AI), machine learning (ML), and other modern workloads. Purpose-built systems provide scalable compute and storage resources to meet the varying needs of a wide range of applications.

Test drive the power of Hitachi servers by contacting Hitachi or a partner representative today.

Hitachi Advanced Server HA820 G2

Hitachi Advanced Server HA820 G2 is a high-performance two-socket rackmount server designed for optimal performance and power efficiency. This allows owners to upgrade computing performance without overextending power consumption and offers non-latency support to virtualization environments that require maximum memory capacity. Hitachi Advanced Server HA820 G2 provides flexible I/O scalability for today’s diverse data center application requirements.

Cisco Nexus switches

The Cisco Nexus switch product line provides a series of solutions that make it easier to connect and manage disparate data center resources with software-defined networking (SDN). Leveraging the Cisco Unified Fabric, which unifies storage, data and networking (Ethernet/IP) services, the Nexus switches create an open, programmable network foundation built to support a virtualized data center environment.

Brocade switches from Broadcom

Brocade and Hitachi Vantara have partnered to deliver storage networking and data center solutions. These solutions reduce complexity and cost, as well as enable virtualization and cloud computing to increase business agility.

Brocade Fibre Channel switches deliver industry-leading performance, simplifying scale-out network architectures. Get the high-performance, availability, and ease of management you need for a solid foundation to grow the storage network you want.

Hitachi Storage Virtualization Operating System RF

Hitachi Storage Virtualization Operating System RF powers the Hitachi Virtual Storage Platform (VSP) family. It integrates storage system software to provide system element management and advanced storage system functions. Used across multiple platforms, Storage Virtualization Operating System includes storage virtualization, thin provisioning, storage service level controls, dynamic provisioning, and performance instrumentation.

Flash performance is optimized with a patented flash-aware I/O stack, which accelerates data access. Adaptive inline data reduction increases storage efficiency while enabling a balance of data efficiency and application performance. Industry-leading storage virtualization allows SVOS RF to use third-party all-flash and hybrid arrays as storage capacity, consolidating resources for a higher ROI and providing a high-speed front end to slower, less-predictable arrays.

Hitachi Unified Compute Platform Advisor

Hitachi Unified Compute Platform Advisor (UCP Advisor) is a comprehensive cloud infrastructure management and automation software that enables IT agility and simplifies day 0-N operations for edge, core, and cloud environments. The fourth-generation UCP Advisor accelerates application deployment and drastically simplifies converged and hyperconverged infrastructure deployment, configuration, life cycle management, and ongoing operations with advanced policy-based automation and orchestration for private and hybrid cloud environments.

The centralized management plane enables remote, federated management for the entire portfolio of converged, hyperconverged, and storage data center infrastructure solutions to improve operational efficiency and reduce management complexity. Its intelligent automation services accelerate infrastructure deployment and configuration, significantly minimizing deployment risk and reducing provisioning time and complexity, automating hundreds of mandatory tasks.

VMware vCenter Server Appliance

The VMware vCenter Server Appliance is a preconfigured Linux virtual machine, which is optimized for running VMware vCenter Server and the associated services on Linux.

vCenter Server Appliance is an Open Virtualization Format (OVF) template. The appliance is imported to an ESXi host and configured through the web-based interface. It comes pre-installed with all the components needed to run a vCenter Server. These include vCenter SSO (Single Sign-on), Inventory Service, vSphere Web Client, and the vCenter Server itself.

VMware vSAN

Seamlessly extending virtualization to storage with an integrated hyper-converged solution that works with your overall VMware environment, VMware vSAN reduces the risk in digital transformation by using existing tools, skillsets, and solutions.

Built by VMware, enjoy the best integration with VMware vSphere features with vSAN. Discover the flexibility to expand with other VMware SDDC and multi-cloud offerings as your needs grow. Protect current storage infrastructure investments with the only hyperconverged infrastructure solution built on policy-based management that extends per-virtual machine policies and automated provisioning to modern SAN and NAS storage systems.

With this NVMe storage design, you can host virtual SAP HANA virtual machines of 128 GB to 4 TB in production or create up to 4 virtual machines in non-production environments on the 2-socket Hitachi Advanced Server DS220 G2 V224N.

Appendix A: Example User Cluster Configuration File

The following is an example of the user cluster configuration file `user-cluster-1.yaml` that was used in this reference architecture.

```
apiVersion: v1
kind: UserCluster
# (Required) A unique name for this cluster
name: "anthos-user-ucpcluster-1"
# (Required) GKE on-prem version (example: 1.3.0-gke.16)
gkeOnPremVersion: 1.14.1-gke.39
# # (Optional) Specify the prepared secret configuration which can be added or edited
# # only during cluster creation
# preparedSecrets:
#   # reference to the secret namespace for a group of secrets; it should be prepared
#   # beforehand by 'gkectl prepare secrets' command; it is immutable.
#   namespace: ""
# # (Optional/Preview) Specify whether or not to use kubeception for managing this
cluster.
# # Default is true
# kubeception: true
# # (Optional) vCenter configuration (default: inherit from the admin cluster)
# vCenter:
#   # # (Optional/Preview) vCenter server to use. kubeception needs to be false when
the
#   # # address is different from that in the admin cluster configuration
#   # address: ""
#   datacenter: ""
#   cluster: ""
#   # Resource pool to use. Specify [VSPHERE_CLUSTER_NAME]/Resources to use the
default
#   # resource pool
#   resourcePool: ""
#   datastore: ""
#   # Provide the path to vCenter CA certificate pub key for SSL verification
#   caCertPath: ""
#   # The credentials to connect to vCenter
#   credentials:
#     # reference to external credentials file
#     fileRef:
#       # read credentials from this file
#       path: ""
#       # entry in the credential file
#       entry: ""
#     # # (Optional) reference to the credential secret; it should be prepared
beforehand
#     # # by 'gkectl prepare secrets' command
#     # secretRef:
#     #   # The version for this prepared secret; it can be specified as 'latest' or
integer
```

```
#     #   # string; it will be defaulted to latest version if it is not specified
when creating
#     #   # a cluster; it is allowed to be empty when creating a cluster; it is not
allowed
#     #   # to be empty when rotating credentials
#     #   version: ""
#   # (Optional) vSphere folder where cluster VMs will be located. Defaults to the the
#   # datacenter wide folder if unspecified.
#   folder: ""
# # (Optional) The absolute or relative path to the GCP service account key for
pulling
# # GKE images (default: inherit from the admin cluster)
# componentAccessServiceAccountKeyPath: ""
# # (Optional) The prepared credentials for component access service account key
# componentAccessServiceAccountKey:
#   # reference to the credential secret; it should be prepared beforehand by 'gkectl
#   # prepare secrets' command
#   secretRef:
#     # The version for this prepared secret; it can be specified as 'latest' or
integer
#     # string; it will be defaulted to latest version if it is not specified when
creating
#     # a cluster; it is allowed to be empty when creating a cluster; it is not
allowed
#     # to be empty when rotating credentials
#     version: ""
# (Required) Network configuration; vCenter section is optional and inherits from
# the admin cluster if not specified
network:
  # # (Required when using "static" ipMode.type; "Seesaw" loadBalancer.kind; or
setting
  # # kubeception to "false") This section overrides ipMode.ipBlockFilePath values
when
  # # ipMode.type=static. It's also used for seesaw nodes and control plane nodes of a
  # # non-kubeception user cluster
  # hostConfig:
  #   # List of DNS servers
  #   dnsServers:
  #   - ""
  #   # List of NTP servers
  #   ntpServers:
  #   - ""
  #   # # List of DNS search domains
  #   # searchDomainsForDNS:
  #   # - ""
  ipMode:
    # (Required) Define what IP mode to use ("dhcp" or "static")
    type: dhcp
    # # (Required when using "static" mode) The absolute or relative path to the yaml
file
    # # to use for static IP allocation. Hostconfig part will be overwritten by
```

```
network.hostconfig
    # # if specified
    # ipBlockFilePath: ""
  # (Required) The Kubernetes service CIDR range for the cluster. Must not overlap
  # with the pod CIDR range
  serviceCIDR: 10.96.0.0/20
  # (Required) The Kubernetes pod CIDR range for the cluster. Must not overlap with
  # the service CIDR range
  podCIDR: 172.16.0.0/16
  vCenter:
    # vSphere network name
    networkName: DPortGroup-ha810g2-GKE-Clusters
  # # (Optional) List of additional node network interfaces feature enabled by
multipleNetworkInterfaces
  # additionalNodeInterfaces:
  # # vSphere network name
  # - networkName: ""
  #   # (Required) Define what IP mode to use ("dhcp" "static" or "none")
  #   type: dhcp
  #   # # (Required when using "static" mode) The absolute or relative path to the
yaml file
  #   # # to use for static IP allocation. Hostconfig part will be overwritten by
network.hostconfig
  #   # # if specified
  #   # ipBlockFilePath: ""
  # # (Optional/Preview) Specify the IPs to use for control plane machines of a non-
kubeception
  # # cluster. 1 IP is needed for non-HA cluster and 3 for HA cluster. Non-empty
controlPlaneIPBlock
  # # is not allowed for a kubeception cluster
  # controlPlaneIPBlock:
  #   netmask: ""
  #   gateway: ""
  #   ips:
  #   - ip: ""
  #     hostname: ""
# (Required) Load balancer configuration
loadBalancer:
  # (Required) The VIPs to use for load balancing
  vips:
    # Used to connect to the Kubernetes API
    controlPlaneVIP: "192.168.60.27"
    # Shared by all services for ingress traffic
    ingressVIP: "192.168.60.28"
  # (Required) Which load balancer to use "F5BigIP" "Seesaw" "ManualLB" or "MetalLB".
  # Uncomment the corresponding field below to provide the detailed spec
  #kind: Seesaw
  kind: F5BigIP
  # # (Required when using "ManualLB" kind) Specify pre-defined nodeports
  # manualLB:
  #   # NodePort for ingress service's http (only needed for user cluster)
```

```
  #   ingressHTTPNodePort: 30243
  #   # NodePort for ingress service's https (only needed for user cluster)
  #   ingressHTTPSNodePort: 30879
  #   # NodePort for konnectivity server service (only needed for user cluster)
  #   konnectivityServerNodePort: 30563
  #   # NodePort for control plane service
  #   controlPlaneNodePort: 30562
  #   # NodePort for addon service (only needed for admin cluster)
  #   addonsNodePort: 0
  # # (Required when using "F5BigIP" kind) Specify the already-existing partition and
  # # credentials
  f5BigIP:
    address: "10.76.47.13"
    credentials:
  #     # reference to external credentials file
      fileRef:
  #       # read credentials from this file
        path: credential.yaml
  #       # entry in the credential file
        entry: f5BigIP
  #     # # (Optional) reference to the credential secret; it should be prepared
beforehand
  #     # # by 'gkectl prepare secrets' command
  #     # secretRef:
  #     #   # The version for this prepared secret; it can be specified as 'latest'
or integer
  #     #   # string; it will be defaulted to latest version if it is not specified
when creating
  #     #   # a cluster; it is allowed to be empty when creating a cluster; it is not
allowed
  #     #   # to be empty when rotating credentials
  #     #   version: ""
    partition: "anthos-cluster-1"
  #   # # (Optional) Specify a pool name if using SNAT
  #   # snatPoolName: ""
  # (Required when using "Seesaw" kind) Specify the Seesaw configs
  #seesaw:
    # (Required) The absolute or relative path to the yaml file to use for IP
allocation
    # for LB VMs. Must contain one or two IPs. Hostconfig part will be overwritten
    # by network.hostconfig if specified.
    #ipBlockFilePath: ""
    # (Required) The Virtual Router IDentifier of VRRP for the Seesaw group. Must
    # be between 1-255 and unique in a VLAN.
    #vrid: 0
    # (Required) The IP announced by the master of Seesaw group
    #masterIP: ""
    # (Required) The number CPUs per machine
    #cpus: 4
    # (Required) Memory size in MB per machine
    #memoryMB: 3072
```

```
    # (Optional) Network that the LB interface of Seesaw runs in (default: cluster
    # network)
    #vCenter:
      # vSphere network name
      #networkName: DPortGroup-ha810g2-mgmt
    # (Optional) Run two LB VMs to achieve high availability (default: false)
    #enableHA: false
    # (Optional) Avoid using VRRP MAC and rely on gratuitous ARP to do failover. In
    # this mode MAC learning is not needed but the gateway must refresh arp table
    # based on gratuitous ARP. It's recommended to turn this on to avoid MAC learning
    # configuration. In vsphere 7+ it must be true to enable HA. It is supported in
    # GKE on-prem version 1.7+. (default: false)
    #disableVRRPMAC: true
  # # (Required when using "MetalLB" kind in user clusters) Specify the MetalLB
configs
  # metalLB:
  #   # (Required) A list of non-overlapping IP pools used by load balancer typed
services.
  #   # Must include ingressVIP of the cluster.
  #   addressPools:
  #   # (Required) Name of the address pool
  #   - name: ""
  #     # (Required) The addresses that are part of this pool. Each address must be
either
  #     # in the CIDR form (1.2.3.0/24) or range form (1.2.3.1-1.2.3.5).
  #     addresses:
  #     - ""
  #     # # (Optional) Avoid using IPs ending in .0 or .255. This avoids buggy
consumer devices
  #     # # mistakenly dropping IPv4 traffic for those special IP addresses (default:
false)
  #     # avoidBuggyIPs: false
  #     # # (Optional) Prevent IP addresses to be automatically assigned from this
pool (default:
  #     # # false)
  #     # manualAssign: false
# # (Optional) Enable dataplane v2
# enableDataplaneV2: false
# # (Optional) Enable support for multiple networking interfaces
# multipleNetworkInterfaces: false
# # (Optional) Enable advanced dataplane v2 networking features such as Egress NAT
Gateway
# # and it requires enableDataplaneV2 to be set
# advancedNetworking: false
# # (Optional) Enable dataplane v2 for Windows
# enableWindowsDataplaneV2: false
# # (Optional) Storage specification for the cluster
# storage:
#   # Whether to disable vSphere CSI components deployment. The feature is enabled by
#   # default.
#   vSphereCSIDisabled: false
```

```
# (Optional) User cluster master nodes must have either 1 or 3 replicas (default:
# 4 CPUs; 8192 MB memory; 1 replica)
masterNode:
  cpus: 4
  memoryMB: 8192
  # How many machines of this type to deploy
  replicas: 1
  # # Enable auto resizing on master
  # autoResize:
  #   # Whether to enable auto resize for master. Defaults to false.
  #   enabled: false
  # vsphere:
  #   # (Optional) vSphere datastore the master nodes will be created on (default:
vCenter.datastore)
  #   datastore: ""
# (Required) List of node pools. The total un-tainted replicas across all node pools
# must be greater than or equal to 3
nodePools:
- name: anthos-user-ucpcluster-1-pool-1
  cpus: 4
  memoryMB: 8192
  # How many machines of this type to deploy
  replicas: 3
  # # (Optional) boot disk size; must be at least 40 (default: 40)
  # bootDiskSizeGB: 40
  # (Optional) Specify the type of OS image; available options can be set to "ubuntu"
  # "ubuntu_containerd" "cos" or "windows". Default is "ubuntu_containerd".
  osImageType: ubuntu_containerd
  # # (Required when using "windows" osImageType) Specify the OS image template in
vCenter
  # osImage: ""
  # # Labels to apply to Kubernetes Node objects
  # labels: {}
  # # Taints to apply to Kubernetes Node objects
  # taints:
  # - key: ""
  #   value: ""
  #   effect: ""
  # vsphere:
  #   # (Optional) vSphere datastore the node pool will be created on (default:
vCenter.datastore)
  #   datastore: ""
  #   # (Optional) vSphere tags to be attached to the virtual machines in the node
pool.
  #   # It is supported in GKE on-prem version 1.7+
  #   tags:
  #   - category: ""
  #     name: ""
  # # (Optional) Horizontal autoscaling for the nodepool; replicas should not be
edited
  # # while updating the nodepool if this is turned on
```

```
  # autoscaling:
  #   # min number of replicas in the NodePool
  #   minReplicas: 0
  #   # max number of replicas in the NodePool
  #   maxReplicas: 0
  # # (Optional) Allow traffic of LoadBalancer typed services flow through nodes of
this
  # # pool. Can only be true in MetalLB mode. Default is false.
  # enableLoadBalancer: false
# Spread nodes across at least three physical hosts (requires at least three hosts)
antiAffinityGroups:
  # Set to false to disable DRS rule creation
  enabled: false
# # (Optional/Preview) Track user cluster VMs with vSphere tags
# enableVMTracking: false
# # Configure node pool update policy for the cluster
# nodePoolUpdatePolicy:
#   # (Optional/Preview) Number of node pools to update at a time. 0 means no limit.
#   # 1 means updating one by one.
#   maximumConcurrentNodePoolUpdate: 0
# # (Optional) Configure additional authentication.
# authentication:
#   # (Optional) Provide an additional serving certificate for the API server
#   sni:
#     certPath: ""
#     keyPath: ""
# (Required) Specify which GCP project to connect your GKE clusters to
gkeConnect:
  projectID: "hv-ucp-anthos"
  # The absolute or relative path to the key file for a GCP service account used to
  # register the cluster
  registerServiceAccountKeyPath: "/home/ubuntu/connect-register-key.json"
  # # (Optional) The prepared credentials for register service account key
  # registerServiceAccountKey:
  #   # reference to the credential secret; it should be prepared beforehand by
'gkectl
  #   # prepare secrets' command
  #   secretRef:
  #     # The version for this prepared secret; it can be specified as 'latest' or
integer
  #     # string; it will be defaulted to latest version if it is not specified when
creating
  #     # a cluster; it is allowed to be empty when creating a cluster; it is not
allowed
  #     # to be empty when rotating credentials
  #     version: ""
# (Required) Specify which GCP project to connect your logs and metrics to
stackdriver:
  # The project ID for logs and metrics. It should be the same with
gkeconnect.projectID.
  projectID: "hv-ucp-anthos"
```

```
  # A GCP region where you would like to store logs and metrics for this cluster.
  clusterLocation: "us-west1"
  enableVPC: false
  # The absolute or relative path to the key file for a GCP service account used to
  # send logs and metrics from the cluster
  serviceAccountKeyPath: "/home/ubuntu/logging-monitoring-key.json"
  # # (Optional) The prepared credentials for stackdriver service account key
  # serviceAccountKey:
  #   # reference to the credential secret; it should be prepared beforehand by
'gkectl
  #   # prepare secrets' command
  #   secretRef:
  #     # The version for this prepared secret; it can be specified as 'latest' or
integer
  #     # string; it will be defaulted to latest version if it is not specified when
creating
  #     # a cluster; it is allowed to be empty when creating a cluster; it is not
allowed
  #     # to be empty when rotating credentials
  #     version: ""
  # (Optional) Disable vsphere resource metrics collection from vcenter.  False by
  # default
  disableVsphereResourceMetrics: false
# # (Optional/Preview) Configure the GKE usage metering feature
# usageMetering:
#   bigQueryProjectID: ""
#   # The ID of the BigQuery Dataset in which the usage metering data will be stored
#   bigQueryDatasetID: ""
#   # The absolute or relative path to the key file for a GCP service account used by
#   # gke-usage-metering to report to BigQuery
#   bigQueryServiceAccountKeyPath: ""
#   # # (Optional) The prepared credentials for big query service account key
#   # bigQueryServiceAccountKey:
#   #   # reference to the credential secret; it should be prepared beforehand by
'gkectl
#   #   # prepare secrets' command
#   #   secretRef:
#   #     # The version for this prepared secret; it can be specified as 'latest' or
integer
#   #     # string; it will be defaulted to latest version if it is not specified
when creating
#   #     # a cluster; it is allowed to be empty when creating a cluster; it is not
allowed
#   #     # to be empty when rotating credentials
#   #     version: ""
#   # Whether or not to enable consumption-based metering
#   enableConsumptionMetering: false
# # (Optional) Configure kubernetes apiserver audit logging
# cloudAuditLogging:
#   # The project ID for logs and metrics. It should be the same with
gkeconnect.projectID.
```

```
#   projectID: ""
#   # A GCP region where you would like to store audit logs for this cluster.
#   clusterLocation: ""
#   # The absolute or relative path to the key file for a GCP service account used to
#   # send audit logs from the cluster
#   serviceAccountKeyPath: ""
#   # # (Optional) The prepared credentials for cloud audit logging service account
key
#   # serviceAccountKey:
#   #   # reference to the credential secret; it should be prepared beforehand by
'gkectl
#   #   # prepare secrets' command
#   #   secretRef:
#   #     # The version for this prepared secret; it can be specified as 'latest' or
integer
#   #     # string; it will be defaulted to latest version if it is not specified
when creating
#   #     # a cluster; it is allowed to be empty when creating a cluster; it is not
allowed
#   #     # to be empty when rotating credentials
#   #     version: ""
# Enable auto repair for the cluster
autoRepair:
  # Whether to enable auto repair feature. Set false to disable.
  enabled: true
# # Encrypt Kubernetes secrets at rest
# secretsEncryption:
#   # Secrets Encryption Mode. Possible values are: GeneratedKey
#   mode: GeneratedKey
#   # GeneratedKey Secrets Encryption config
#   generatedKey:
#     # # key version
#     # keyVersion: 1
#     # # disable secrets encryption
#     # disabled: false
```

Hitachi Vantara

Corporate Headquarters
2535 Augustine Drive
Santa Clara, CA 95054 USA

HitachiVantara.com/contact