Anthos with NetApp

NetApp Solutions

NetApp
January 12, 2023

Table of Contents

NVA-1165: Anthos with NetApp

Alan Cowles and Nikhil Kulkarni, NetApp

This reference document provides deployment validation of the Anthos with NetApp solution by NetApp and our engineering partners when it is deployed in multiple data-center environments. It also details storage integration with NetApp storage systems by using the Astra Trident storage orchestrator for the management of persistent storage. Lastly, we explore and document a number of solution validations and real-world use cases.

Use cases

The Anthos with NetApp solution is architected to deliver exceptional value for customers with the following use cases:

- Easy to deploy and manage Anthos environment deployed using the provided `bmctl` tool on bare metal or the `gkectl` tool on VMware vSphere.
- Combined power of enterprise container and virtualized workloads with Anthos deployed virtually on vSphere or on bare metal with kubevirt.
- Real-world configuration and use cases highlighting Anthos features when used with NetApp storage and Astra Trident, the open-source storage orchestrator for Kubernetes.

Business value

Enterprises are increasingly adopting DevOps practices to create new products, shorten release cycles, and rapidly add new features. Because of their innate agile nature, containers and microservices play a crucial role in supporting DevOps practices. However, practicing DevOps at a production scale in an enterprise environment presents its own challenges and imposes certain requirements on the underlying infrastructure, such as the following:

- High availability at all layers in the stack
- Ease of deployment procedures
- Non-disruptive operations and upgrades
- API-driven and programmable infrastructure to keep up with microservices agility
- Multitenancy with performance guarantees
- The ability to run virtualized and containerized workloads simultaneously
- The ability to scale infrastructure independently based on workload demands

The Anthos with NetApp solution acknowledges these challenges and presents a solution that helps address each concern by implementing the fully automated deployment of Anthos on prem in the customer’s data center environment of choice.

Technology overview

The Anthos with NetApp solution is comprised of the following major components:

Anthos On Prem

Anthos On Prem is a fully supported enterprise Kubernetes platform that can be deployed in the VMware vSphere hypervisor, or on a bare metal infrastructure of your choosing.

For more information about Anthos, see the Anthos website located here.

NetApp storage systems

NetApp has several storage systems perfect for enterprise data centers and hybrid cloud deployments. The NetApp portfolio includes NetApp ONTAP, NetApp Element, and NetApp e-Series storage systems, all of which can provide persistent storage for containerized applications.

For more information visit the NetApp website here.

NetApp storage integrations

Astra Trident is an open-source and fully-supported storage orchestrator for containers and Kubernetes distributions, including Anthos.

For more information, visit the Astra Trident website here.

Advanced configuration options

This section is dedicated to customizations that real world users would likely need to perform when deploying this solution into production, such as creating a dedicated private image registry or deploying custom load balancer instances.

Current support matrix for validated releases

Technology	Purpose	Software version
NetApp ONTAP	Storage	9.8, 9.9.1
NetApp Element	Storage	12.3
NetApp Astra Trident	Storage Orchestration	22.04.0
Anthos Clusters on VMware	Container orchestration	1.11
Anthos on bare metal	Container Orchestration	1.10
VMware vSphere	Data center virtualization	6.7U3, 7.0U3

Next: Anthos Overview.

Anthos Overview

Anthos with NetApp is a verified, best-practice hybrid cloud architecture for the deployment of an on-premises Google Kubernetes Engine (GKE) environment in a reliable and dependable manner. This NetApp Verified Architecture reference document serves as both a design guide and a deployment validation of the Anthos with NetApp solution deployed to bare metal and virtual environments. The architecture described in this document has been validated by subject matter experts at NetApp and Google Cloud to provide the advantages of running Anthos within your enterprise data-center environment.

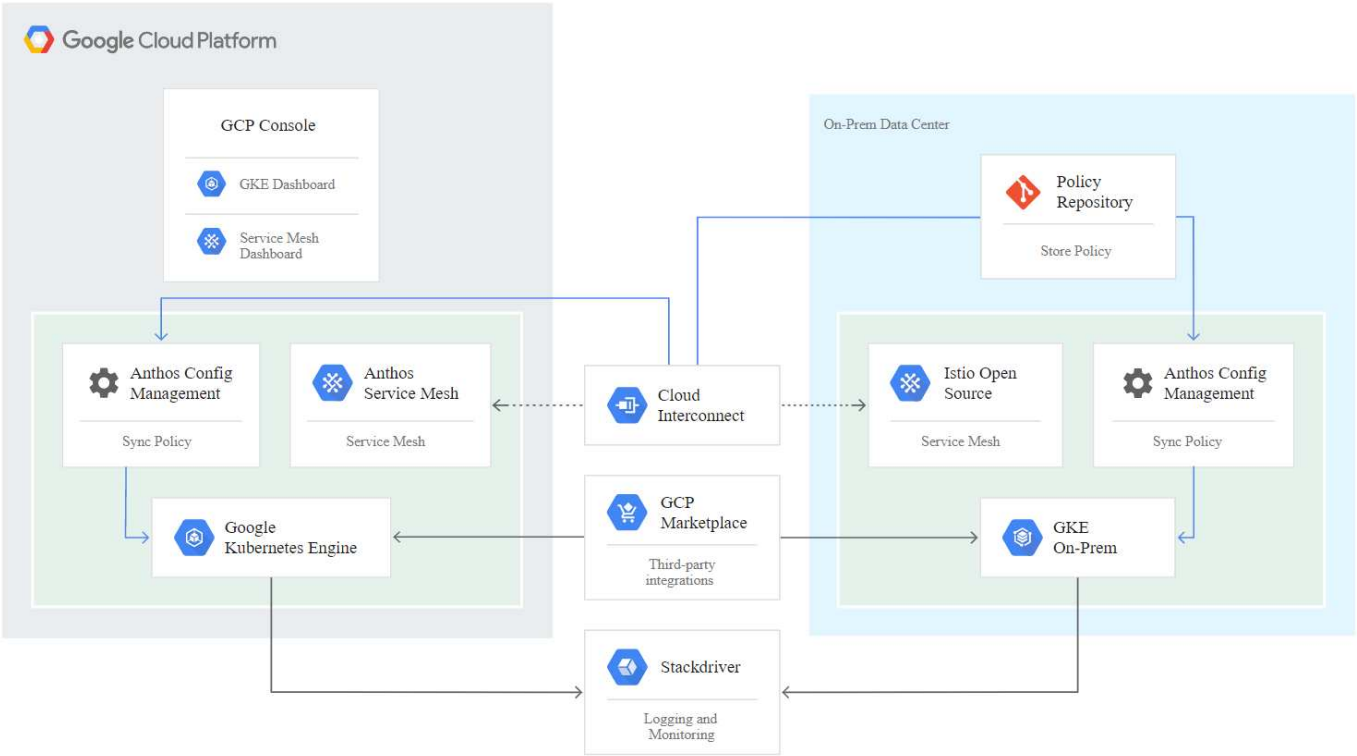
Anthos

Anthos is a hybrid-cloud Kubernetes data center solution that enables organizations to construct and manage modern hybrid-cloud infrastructures while adopting agile workflows focused on application development. Anthos on VMware, a solution built on open-source technologies, runs on-premises in a VMware vSphere-based infrastructure, which can connect and interoperate with Anthos GKE in Google Cloud.
Adopting containers, service mesh, and other transformational technologies enables organizations to experience consistent application development cycles and production-ready workloads in local and cloud-based environments. The following figure depicts the Anthos solution and how a deployment in an on-premises data center interconnects with infrastructure in the cloud.

For more information about Anthos, see the Anthos website located here.

Anthos provides the following features:

- **Anthos configuration management.** Automates the policy and security of hybrid Kubernetes deployments.
- **Anthos Service Mesh.** Enhances application observability, security, and control with an Istio-powered service mesh.
- **Google Cloud Marketplace for Kubernetes Applications.** A catalog of curated container applications available for easy deployment.
- **Migrate for Anthos.** Automatic migration of physical services and VMs from on-premises to the cloud.
- **Stackdriver.** Management service offered by Google for logging and monitoring cloud instances.



Deployment methods for Anthos

Anthos clusters on VMware

Anthos clusters deployed to VMware vSphere environments are easy to deploy, maintain, and scale rapidly for most end-user Kubernetes workloads.

For more information about Anthos clusters on VMware, deployed with NetApp, please visit the page here.

Anthos on bare metal

Anthos clusters deployed on bare metal servers are hardware agnostic and allow you to select a compute platform optimized for your personalized use case.

For more information about Anthos on bare metal clusters deployed with NetApp, visit here.

Next: Anthos Clusters on VMware.

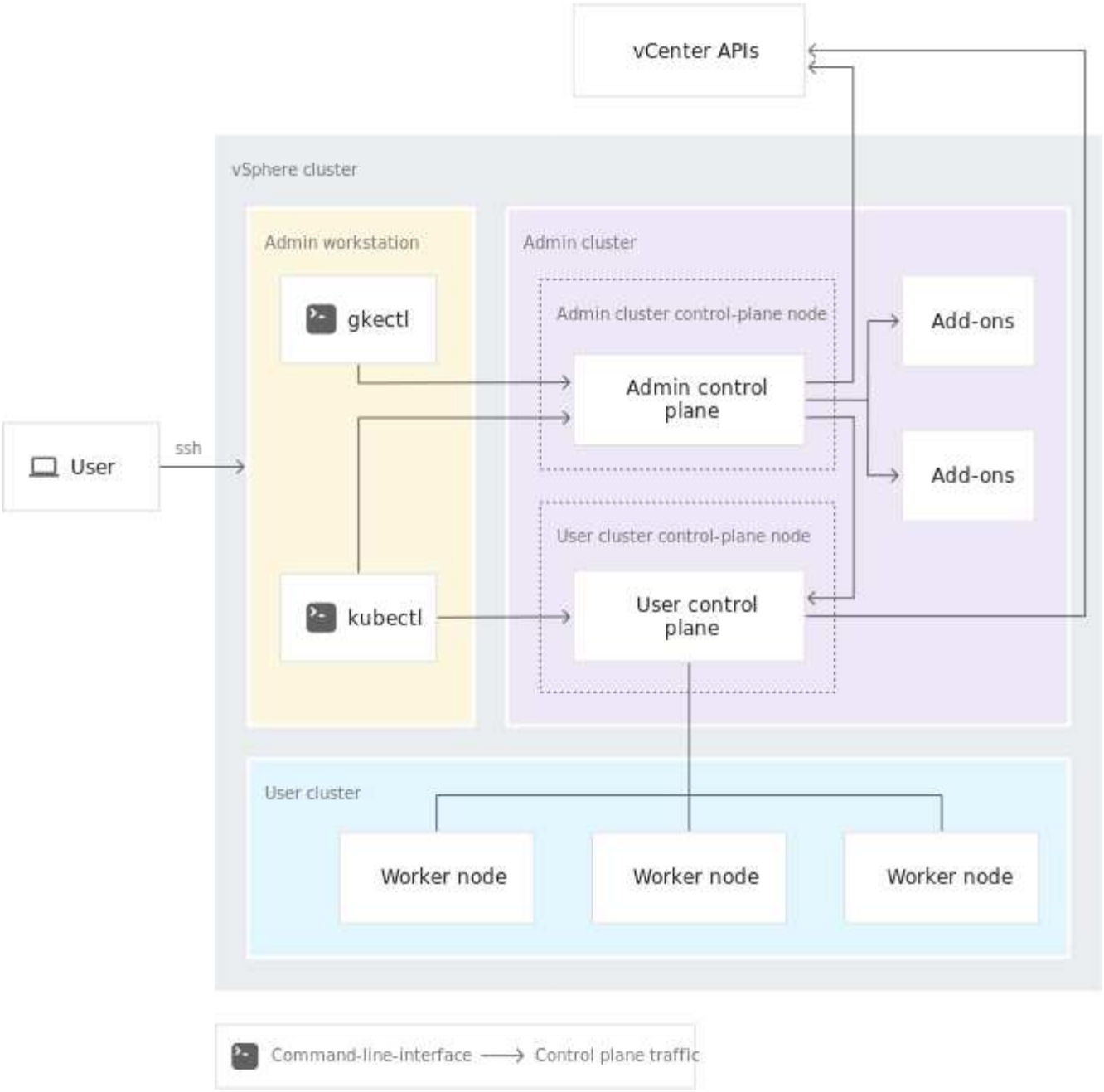
Anthos Clusters on VMware

Anthos clusters on VMware is an extension of Google Kubernetes Engine that is deployed in an end user's private data center. An organization can deploy the same applications designed to run in containers in Google Cloud in Kubernetes clusters on-premises.
Anthos clusters on VMware can be deployed into an existing VMware vSphere environment in your data center, which can save on capital expenses and enable more rapid deployment and scaling operations.

The deployment of Anthos clusters on VMware includes the following components:

- **Anthos admin workstation.** A deployment host from which `gkectl` and `kubectl` commands can be run to deploy and interact with Anthos deployments.
- **Admin cluster.** The initial cluster deployed when setting up Anthos clusters on VMware. This cluster manages all subordinate user cluster actions, including deployment, scaling, and upgrade.
- **User cluster.** Each user cluster is deployed with it's own load balancer instance or partition, allowing it to act as a standalone Kubernetes cluster for individual users or groups, helping to achieve full multitenancy.

The following graphic is a description of an Anthos-clusters-on-VMware deployment.



Benefits

Anthos clusters on VMware offers the following benefits:

- **Advanced multitenancy.** Each end user can be assigned their own user cluster, deployed with the virtual resources necessary for their own development environment.
- **Cost savings.** End users can realize significant cost savings by deploying multiple user clusters to the same physical environment and utilizing their own physical resources for their application deployments instead of provisioning resources in their Google Cloud environment or on large bare-metal clusters.
- **Develop then publish.** On-premises deployments can be used while applications are in development, which allows for testing of applications in the privacy of a local data center before being made publicly available in the cloud.
- **Security requirements.** Customers with increased security concerns or sensitive data sets that cannot be stored in the public cloud are able to run their applications from the security of their own data centers,

thereby meeting organizational requirements.

VMware vSphere

VMware vSphere is a virtualization platform for centrally managing a large number of virtualized servers and networks running on the ESXi hypervisor.

For more information about VMware vSphere, see the VMware vSphere website.

VMware vSphere provides the following features:

- **VMware vCenter Server.** VMware vCenter Server provides unified management of all hosts and VMs from a single console and aggregates performance monitoring of clusters, hosts, and VMs.
- **VMware vSphere vMotion.** VMware vCenter allows you to hot migrate VMs between nodes in the cluster upon request in a non-disruptive manner.
- **vSphere High Availability.** To avoid disruption in the event of host failures, VMware vSphere allows hosts to be clustered and configured for high availability. VMs that are disrupted by host failure are rebooted shortly on other hosts in the cluster, restoring services.
- **Distributed Resource Scheduler (DRS).** A VMware vSphere cluster can be configured to load balance the resource needs of the VMs it is hosting. VMs with resource contentions can be hot migrated to other nodes in the cluster to make sure that enough resources are available.

Hardware requirements

Compute

Google Cloud periodically requests updated validation of partner server platforms with new releases of Anthos through their Anthos Ready platform partner program. A listing of currently validated server platforms and the versions of Anthos supported can be found here.

The following table contains server platforms that have been tested by NetApp and NetApp partner engineers for the validation of Anthos clusters on VMware deployments. These include solutions such as the NetApp FlexPod with Cisco UCS servers and the NetApp HCI hybrid cloud infrastructure platform.

Manufacturer	Make	Model
Cisco	UCS	B200 M5
NetApp	HCI	C410

Operating system

Anthos clusters on VMware can be deployed to both vSphere 6 and 7 environments as chosen by the customer to help match their current datacenter infrastructure.

The following table contains a list vSphere versions that have been used by NetApp and our partners to validate the solution.

Operating System	Release	Anthos Versions
VMware vSphere	6.7U3	1.11
VMware vSphere	7.0U3	1.11

Additional hardware

To complete the deployment of Anthos with NetApp as a fully validated solution, additional data center components for networking and storage have been tested by NetApp and our partner engineers.

The following table includes information about these additional infrastructure components.

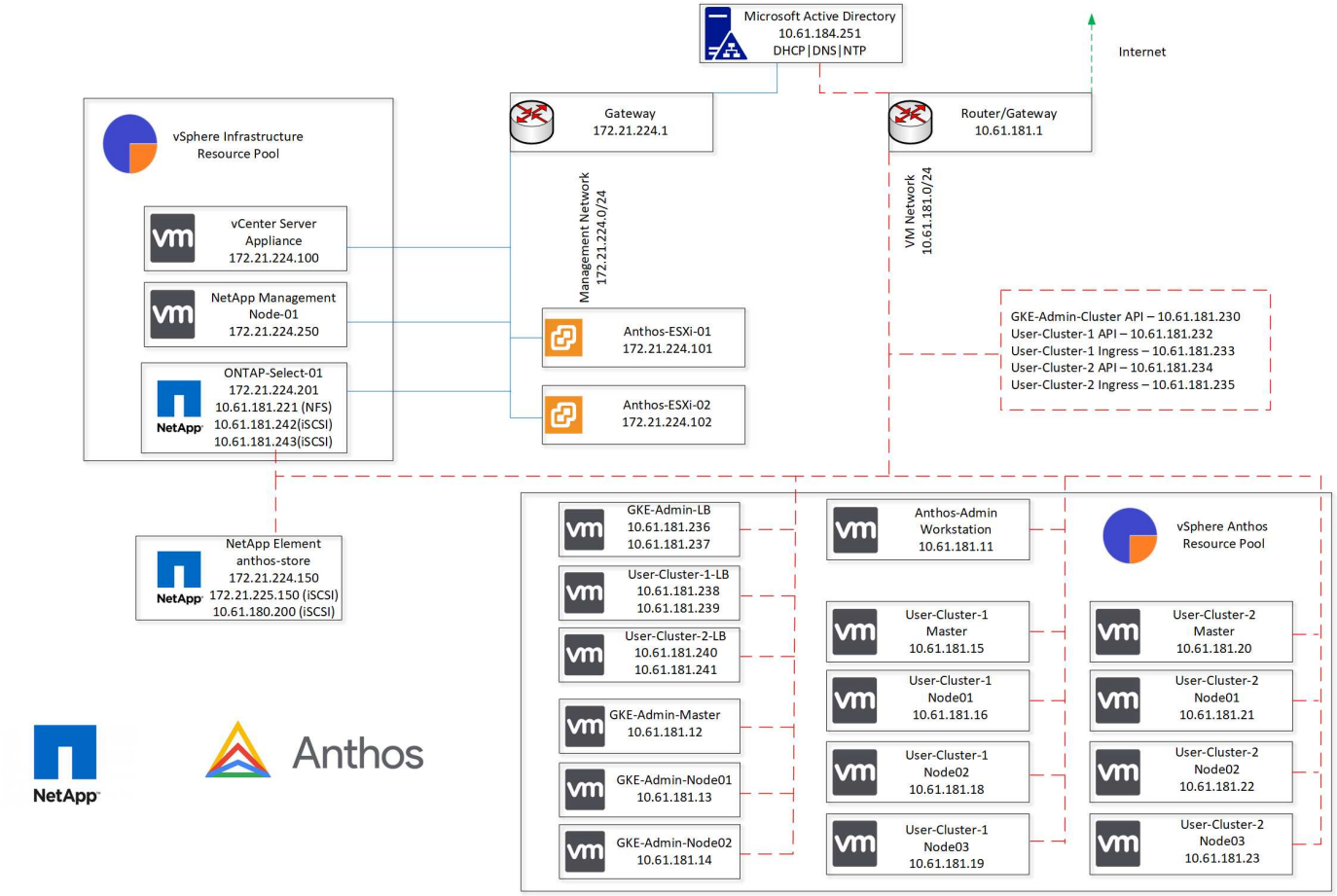
Manufacturer	Hardware Name	Model
Mellanox	SN	2010
NetApp	AFF	A250
NetApp	HCI	S410

Additional software

The following table includes a list of software versions deployed in the validation environment.

Manufacturer	Software Name	Version
Cisco	UCS	4.1(3e)
NetApp	Element	12
NetApp	HCI	1.8
NetApp	ONTAP	9.9.1
NetApp	Astra Trident	22.04

During the Anthos Ready platform validation performed by NetApp, the lab environment was built based on the following diagram, which allowed us to test multiple deployed user clusters alongside multiple NetApp Storage systems and storage backends.



Network infrastructure support resources

The following infrastructure should be in place prior to the deployment of Anthos:

- At least one DNS server providing full host-name resolution that is accessible from the in-band management network and the VM network.
- At least one NTP server that is accessible from the in-band management network and the VM network.
- A DHCP server available to provide network address leases on demand should clusters need to scale dynamically.
- (Optional) Outbound internet connectivity for both the in-band management network and the VM network.

Best practices for production deployments

This section lists several best practices that an organization should take into consideration before deploying this solution into production.

Deploy Anthos to an ESXi cluster of at least three nodes

Although it is possible to install Anthos in a vSphere cluster of less than three nodes for demonstration or evaluation purposes, this is not recommended for production workloads. Although two nodes allow for basic HA and fault tolerance, an Anthos cluster configuration must be modified to disable default host affinity, and this deployment method is not supported by Google Cloud.

Configure virtual machine and host affinity

Distributing Anthos cluster nodes across multiple hypervisor nodes can be achieved by enabling VM and host affinity.

Affinity or anti-affinity is a way to define rules for a set of VMs and/or hosts that determine whether the VMs run together on the same host or hosts in the group or on different hosts. It is applied to VMs by creating affinity groups that consist of VMs and/or hosts with a set of identical parameters and conditions. Depending on whether the VMs in an affinity group run on the same host or hosts in the group or separately on different hosts, the parameters of the affinity group can define either positive affinity or negative affinity.

To configure affinity groups, see the appropriate link below for your version of VMWare vSphere.

vSphere 6.7 Documentation: Using DRS Affinity Rules.
vSphere 7.0 Documentation: Using DRS Affinity Rules.

Anthos has a config option in each individual `cluster.yaml` file to automatically create node affinity rules that can be enabled or disabled based on the number of ESXi hosts in your environment.

Next: Anthos on bare metal.

Anthos on bare metal

Benefits

The hardware-agnostic capabilities of Anthos on bare metal allow you to select a compute platform optimized for your personalized use case and also provide many additional benefits.

Examples include the following:

- **Bring your own server.** You can use servers that match your existing infrastructure to reduce capital expenditure and management costs.
- **Bring your own Linux OS.** By choosing the Linux OS that you wish to deploy your Anthos-on-bare-metal environment to, you can ensure that the Anthos environment fits neatly into your existing infrastructure and management schemes.
- **Improved performance and lowered cost.** Without the requirement of a hypervisor, Anthos-on-bare-metal clusters call for direct access to server hardware resources, including performance-optimized hardware devices like GPUs.
- **Improved network performance and lowered latency.** Because the Anthos-on-bare-metal server nodes are directly connected to your network without a virtualized abstraction layer, they can be optimized for low latency and performance.

Hardware requirements

Compute

Google Cloud periodically requests updated validation of partner server platforms with new releases of Anthos through their Anthos Ready platform partner program. A listing of currently validated server platforms and the versions of Anthos supported can be found here.

The following table contains server platforms that have been tested by NetApp and NetApp partner engineers for the validation of Anthos on bare metal deployments.

Manufacturer	Make	Model
Cisco	UCS	B200 M5
HPE	Proliant	DL360

Operating System

Anthos-on-bare-metal nodes can be configured with several different Linux distributions as chosen by the customer to help match their current datacenter infrastructure.

The following table contains a list of Linux operating systems that have been used by NetApp and our partners to validate the solution.

Operating System	Release	Anthos Versions
CentOS	8.4	1.11
Red Hat Enterprise Linux	8.4	1.11
Ubuntu	18.04 LTS	1.11
Ubuntu	20.04 LTS	1.11

Additional hardware

To complete the deployment of Anthos on bare metal as a fully validated solution, additional data center components for networking and storage have been tested by NetApp and our partner engineers.

The following table includes information about these additional infrastructure components.

Manufacturer	Hardware Name	Model
Cisco	Nexus	C9336C-FX2
NetApp	AFF	A250, A220

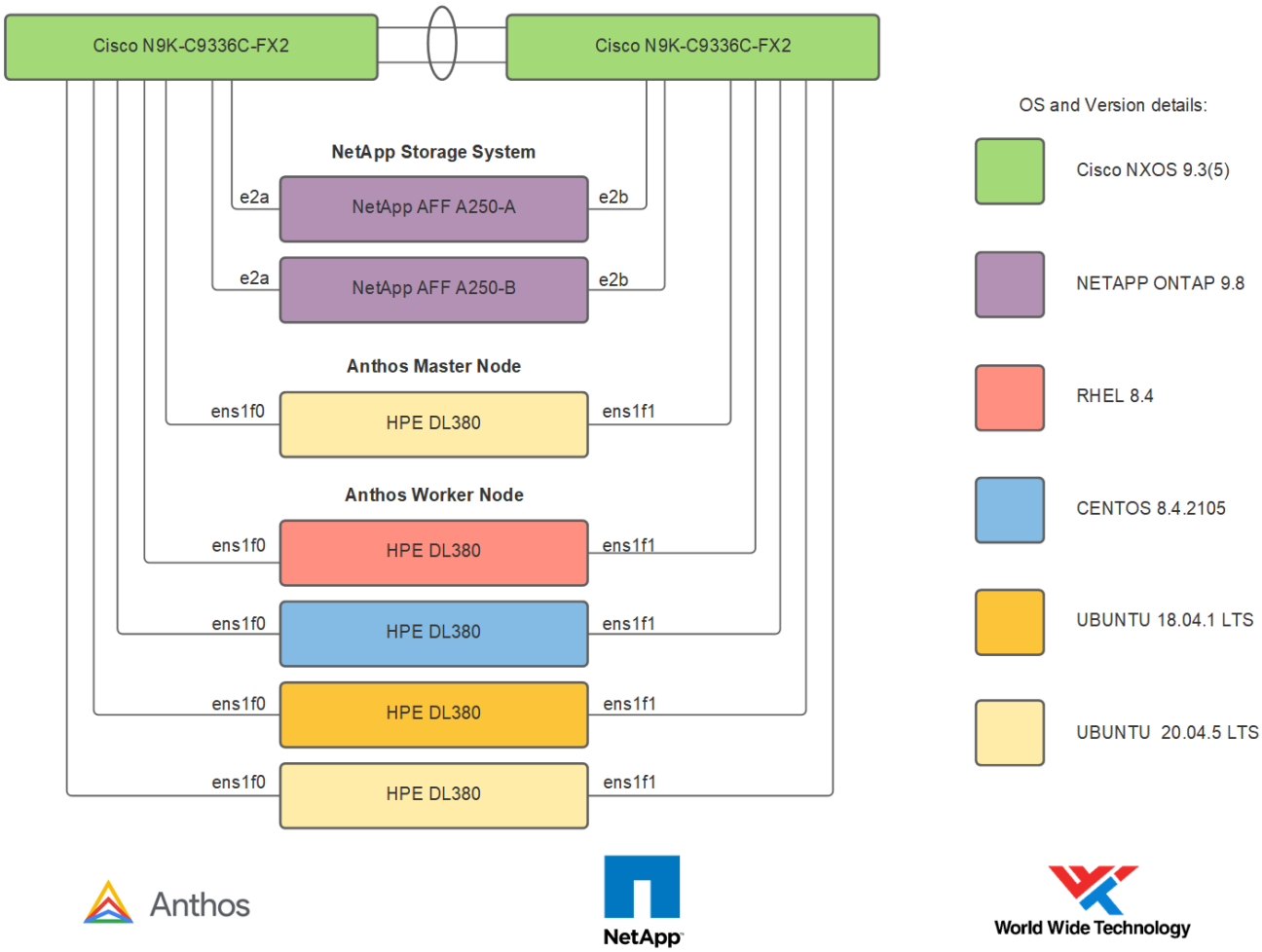
Additional software

The following table includes a list of additional software versions deployed in the validation environment.

Manufacturer	Software name	Version
Cisco	NXOS	9.3(5)
NetApp	ONTAP	9.9.1, 9.10.1
NetApp	Astra Trident	22.04

During the Anthos Ready platform validation performed by NetApp and our partner team at World Wide Technology (WWT), the lab environment was built based on the following diagram, which allowed us to test the functionality of each server type, operating system, the network devices, and storage systems deployed in the solution.

Anthos BareMetal Physical Hardware Diagram



This multi-OS environment shows interoperability with supported OS versions for the Anthos-on-bare-metal solution. We anticipate that customers will standardize on one or a subset of operating systems for their deployment.

Infrastructure support resources

The following infrastructure should be in place prior to the deployment of Anthos on bare metal:

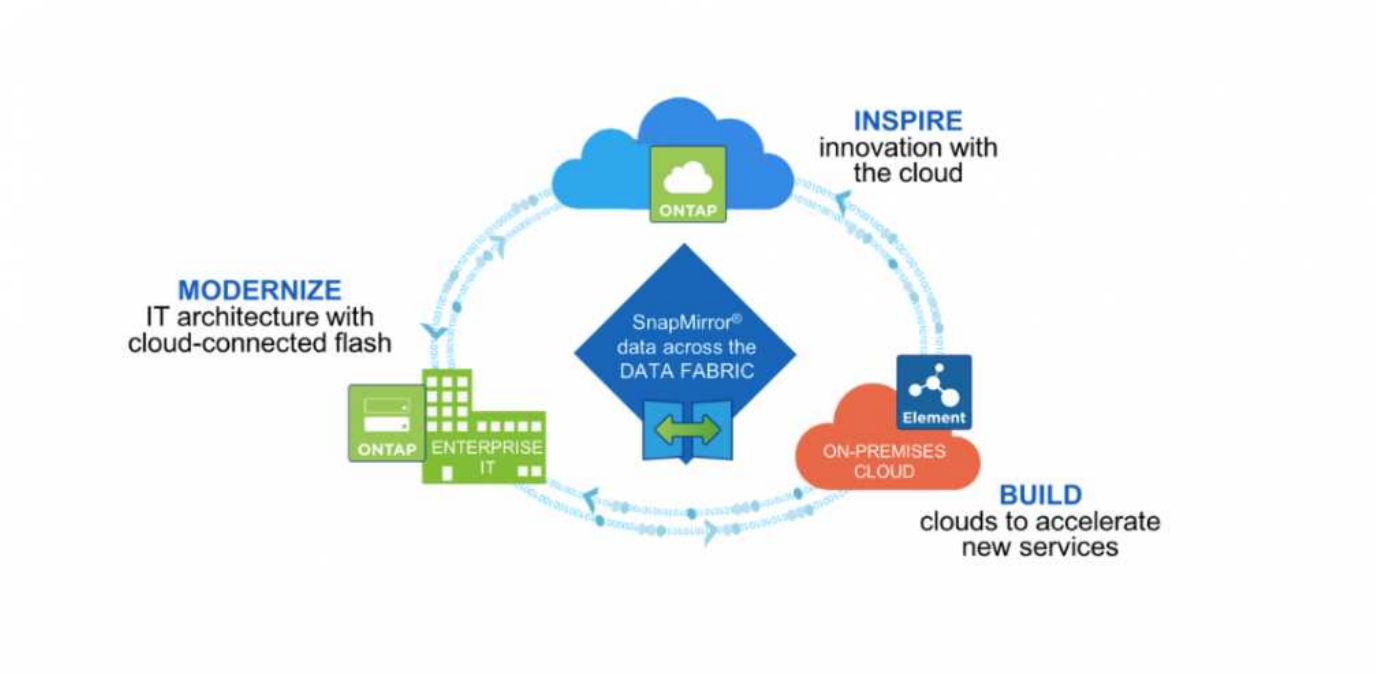
- At least one DNS server that provides a full host-name resolution accessible from the management network.
- At least one NTP server that is accessible from the management network.
- (Optional) Outbound internet connectivity for both the in-band management network.

There is a demo video of an Anthos on bare metal deployment in the Videos and Demos section of this document.

Next: NetApp Storage Systems Overview.

NetApp Storage Overview

NetApp has several storage platforms that are qualified with our Astra Trident Storage Orchestrator to provision storage for applications deployed on Anthos.



- AFF and FAS systems run NetApp ONTAP and provide storage for both file-based (NFS) and block-based (iSCSI) use cases.
- Cloud Volumes ONTAP and ONTAP Select provide the same benefits in the cloud and virtual space respectively.
- NetApp Cloud Volumes Service (AWS/GCP) and Azure NetApp Files provide file-based storage in the cloud.
- NetApp Element storage systems provide for block-based (iSCSI) use cases in a highly scalable environment.

Each storage system in the NetApp portfolio can ease both data management and movement between on-premises sites and the cloud, ensuring that your data is where your applications are.

Next: NetApp ONTAP.

NetApp ONTAP

NetApp ONTAP is a powerful storage-software tool with capabilities such as an intuitive GUI, REST APIs with automation integration, AI-informed predictive analytics and corrective action, nondisruptive hardware upgrades, and cross-storage import.

For more information about the NetApp ONTAP storage system, visit the NetApp ONTAP website.

ONTAP provides the following features:

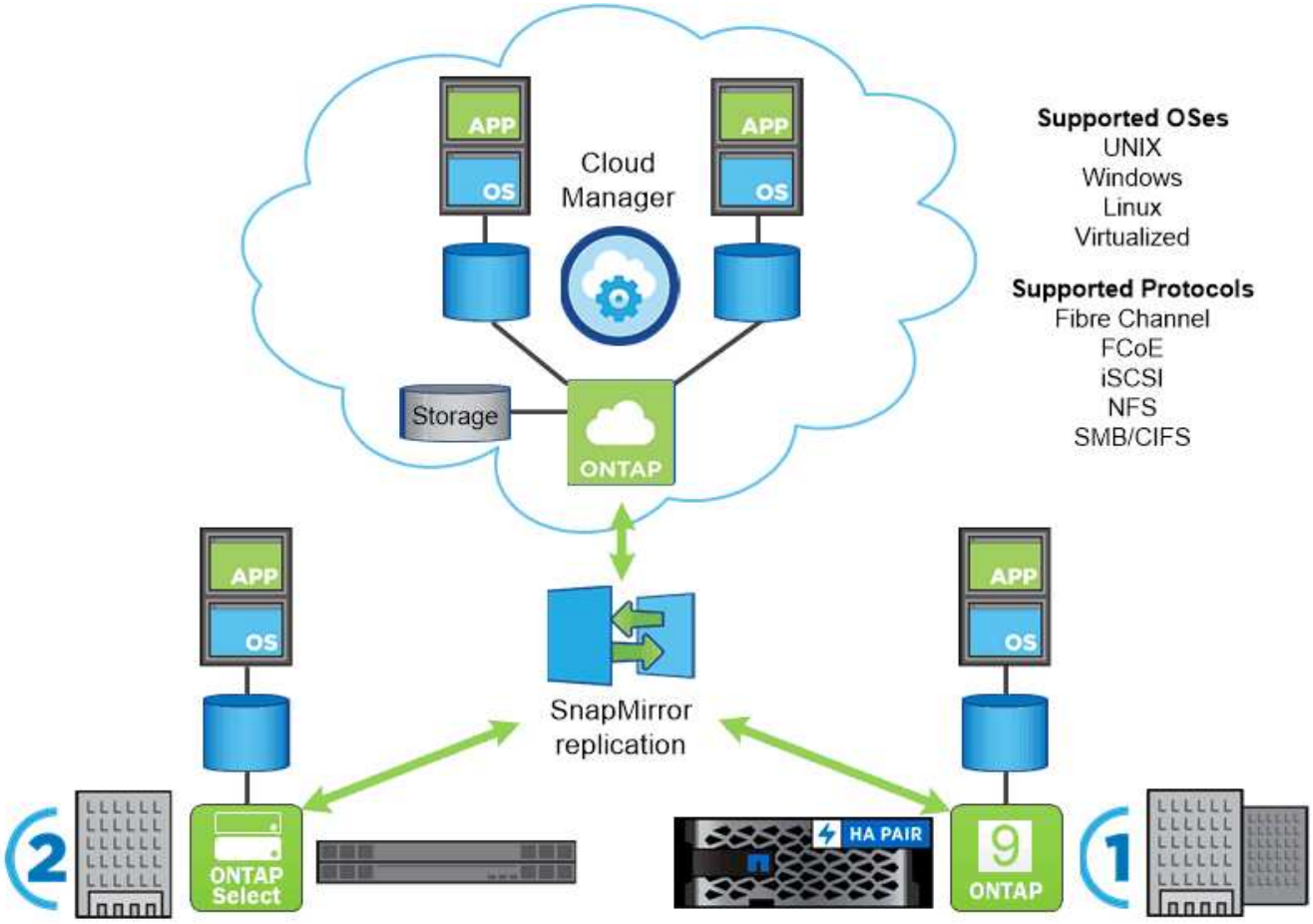
- A unified storage system with simultaneous data access and management of NFS, CIFS, iSCSI, FC,

FCoE, and FC-NVMe protocols.

- Different deployment models include on-premises on all-flash, hybrid, and all-HDD hardware configurations; VM-based storage platforms on a supported hypervisor such as ONTAP Select; and in the cloud as Cloud Volumes ONTAP.
- Increased data storage efficiency on ONTAP systems with support for automatic data tiering, inline data compression, deduplication, and compaction.
- Workload-based, QoS-controlled storage.
- Seamless integration with a public cloud for tiering and protection of data. ONTAP also provides robust data protection capabilities that sets it apart in any environment:
  - **NetApp Snapshot copies.** A fast, point-in-time backup of data using a minimal amount of disk space with no additional performance overhead.
  - **NetApp SnapMirror.** Mirrors the Snapshot copies of data from one storage system to another. ONTAP supports mirroring data to other physical platforms and cloud-native services as well.
  - **NetApp SnapLock.** Efficiently administration of nonrewritable data by writing it to special volumes that cannot be overwritten or erased for a designated period.
  - **NetApp SnapVault.** Backs up data from multiple storage systems to a central Snapshot copy that serves as a backup to all designated systems.
  - **NetApp SyncMirror.** Provides real-time, RAID-level mirroring of data to two different plexes of disks that are connected physically to the same controller.
  - **NetApp SnapRestore.** Provides fast restoration of backed-up data on demand from Snapshot copies.
  - **NetApp FlexClone.** Provides instantaneous provisioning of a fully readable and writeable copy of a NetApp volume based on a Snapshot copy.

For more information about ONTAP, see the ONTAP 9 Documentation Center.

NetApp ONTAP is available on-premises, virtualized, or in the cloud.



NetApp platforms

NetApp AFF/FAS

NetApp provides robust all-flash (AFF) and scale-out hybrid (FAS) storage platforms that are tailor-made with low-latency performance, integrated data protection, and multiprotocol support.

Both systems are powered by NetApp ONTAP data management software, the industry's most advanced data-management software for highly-available, cloud-integrated, simplified storage management to deliver the enterprise-class speed, efficiency, and security your data fabric needs.

For more information about NETAPP AFF and FAS platforms, click here.

ONTAP Select

ONTAP Select is a software-defined deployment of NetApp ONTAP that can be deployed onto a hypervisor in your environment. It can be installed on VMware vSphere or on KVM and provides the full functionality and experience of a hardware-based ONTAP system.

For more information about ONTAP Select, click here.

Cloud Volumes ONTAP

NetApp Cloud Volumes ONTAP is a cloud-deployed version of NetApp ONTAP available to be deployed in a number of public clouds, including: Amazon AWS, Microsoft Azure, and Google Cloud.

For more information about Cloud Volumes ONTAP, click here.

Next: NetApp Element.

NetApp Element

NetApp Element software provides modular, scalable performance, with each storage node delivering guaranteed capacity and throughput to the environment. NetApp Element systems can scale from 4 to 100 nodes in a single cluster and offer a number of advanced storage management features.

For more information about NetApp Element storage systems, visit the NetApp Solidfire website.

iSCSI login redirection and self-healing capabilities

NetApp Element software leverages the iSCSI storage protocol, a standard way to encapsulate SCSI commands on a traditional TCP/IP network. When SCSI standards change or when the performance of Ethernet networks improves, the iSCSI storage protocol benefits without the need for any changes.

Although all storage nodes have a management IP and a storage IP, NetApp Element software advertises a single storage virtual IP address (SVIP address) for all storage traffic in the cluster. As a part of the iSCSI login process, storage can respond that the target volume has been moved to a different address and therefore it cannot proceed with the negotiation process. The host then reissues the login request to the new address in a process that requires no host-side reconfiguration. This process is known as iSCSI login redirection.

iSCSI login redirection is a key part of the NetApp Element software cluster. When a host login request is received, the node decides which member of the cluster should handle the traffic based on the IOPS and the capacity requirements for the volume. Volumes are distributed across the NetApp Element software cluster and are redistributed if a single node is handling too much traffic for its volumes or if a new node is added. Multiple copies of a given volume are allocated across the array.

In this manner, if a node failure is followed by volume redistribution, there is no effect on host connectivity beyond a logout and login with redirection to the new location. With iSCSI login redirection, a NetApp Element software cluster is a self-healing, scale-out architecture that is capable of nondisruptive upgrades and operations.

NetApp Element software cluster QoS

A NetApp Element software cluster allows QoS to be dynamically configured on a per-volume basis. You can use per-volume QoS settings to control storage performance based on SLAs that you define. The following three configurable parameters define the QoS:

- **Minimum IOPS.** The minimum number of sustained IOPS that the NetApp Element software cluster provides to a volume. The minimum IOPS configured for a volume is the guaranteed level of performance for a volume. Per-volume performance does not drop below this level.

- **Maximum IOPS.** The maximum number of sustained IOPS that the NetApp Element software cluster provides to a particular volume.
- **Burst IOPS.** The maximum number of IOPS allowed in a short burst scenario. The burst duration setting is configurable, with a default of 1 minute. If a volume has been running below the maximum IOPS level, burst credits are accumulated. When performance levels become very high and are pushed, short bursts of IOPS beyond the maximum IOPS are allowed on the volume.

Multitenancy

Secure multitenancy is achieved with the following features:

- **Secure authentication.** The Challenge-Handshake Authentication Protocol (CHAP) is used for secure volume access. The Lightweight Directory Access Protocol (LDAP) is used for secure access to the cluster for management and reporting.
- **Volume access groups (VAGs).** Optionally, VAGs can be used in lieu of authentication, mapping any number of iSCSI initiator-specific iSCSI Qualified Names (IQNs) to one or more volumes. To access a volume in a VAG, the initiator's IQN must be in the allowed IQN list for the group of volumes.
- **Tenant virtual LANs (VLANs).** At the network level, end-to-end network security between iSCSI initiators and the NetApp Element software cluster is facilitated by using VLANs. For any VLAN that is created to isolate a workload or a tenant, NetApp Element Software creates a separate iSCSI target SVIP address that is accessible only through the specific VLAN.
- **VRF-enabled VLANs.** To further support security and scalability in the data center, NetApp Element software allows you to enable any tenant VLAN for VRF-like functionality. This feature adds these two key capabilities:
  - **L3 routing to a tenant SVIP address.** This feature allows you to situate iSCSI initiators on a separate network or VLAN from that of the NetApp Element software cluster.
  - **Overlapping or duplicate IP subnets.** This feature enables you to add a template to tenant environments, allowing each respective tenant VLAN to be assigned IP addresses from the same IP subnet. This capability can be useful for in-service provider environments where scale and preservation of IPspace are important.

Enterprise storage efficiencies

The NetApp Element software cluster increases overall storage efficiency and performance. The following features are performed inline, are always on, and require no manual configuration by the user:

- **Deduplication.** The system only stores unique 4K blocks. Any duplicate 4K blocks are automatically associated to an already stored version of the data. Data is on block drives and is mirrored by using the NetApp Element software Helix data protection. This system significantly reduces capacity consumption and write operations within the system.
- **Compression.** Compression is performed inline before data is written to NVRAM. Data is compressed, stored in 4K blocks, and remains compressed in the system. This compression significantly reduces capacity consumption, write operations, and bandwidth consumption across the cluster.
- **Thin-provisioning.** This capability provides the right amount of storage at the time that you need it, eliminating capacity consumption that caused by overprovisioned volumes or underutilized volumes.
- **Helix.** The metadata for an individual volume is stored on a metadata drive and is replicated to a secondary metadata drive for redundancy.

Element was designed for automation. All the storage features are available through APIs. These APIs are the only method that the UI uses to control the system.

Next: NetApp Storage Integrations Overview.

NetApp Storage Integration Overview

Anthos Ready storage partner program.

Google Cloud periodically requests updated validation of partner storage integrations with new releases of Anthos through their Anthos Ready storage partner program. A list of currently validated storage solutions, CSI drivers, available features, and the versions of Anthos supported can be found here.

NetApp has maintained regular compliance on a quarterly basis with requests to validate our Astra Trident CSI-compliant storage orchestrator and our ONTAP and Element storage systems with versions of Anthos.

The following table contains the Anthos versions tested by NetApp and NetApp partner engineers for validation of NetApp Astra Trident CSI drivers and feature sets as a part of the Anthos Ready storage partner program:

Deployment Type	Version	Storage System	Astra Trident Version	Protocol	Features
VMware	1.11	ONTAP	22.04	NAS	Multiwriter, Volume Expansion, SnapShots
VMware	1.11	ONTAP	22.04	SAN	Raw Block, Volume Expansion, SnapShots
VMware	1.11	Element	22.04	SAN	Raw Block, Volume Expansion, SnapShots
bare metal	1.10	ONTAP	22.01	NAS	Multiwriter, Volume Expansion, SnapShots
bare metal	1.10	ONTAP	22.01	SAN	Raw Block, Volume Expansion, SnapShots

NetApp storage integrations

NetApp provides a number of products to help you with orchestrating and managing persistent data in container-based environments such as Anthos.

NetApp Astra Trident is an open-source, fully-supported storage orchestrator for containers and Kubernetes distributions, including Anthos. For more information, visit the Astra Trident website here.

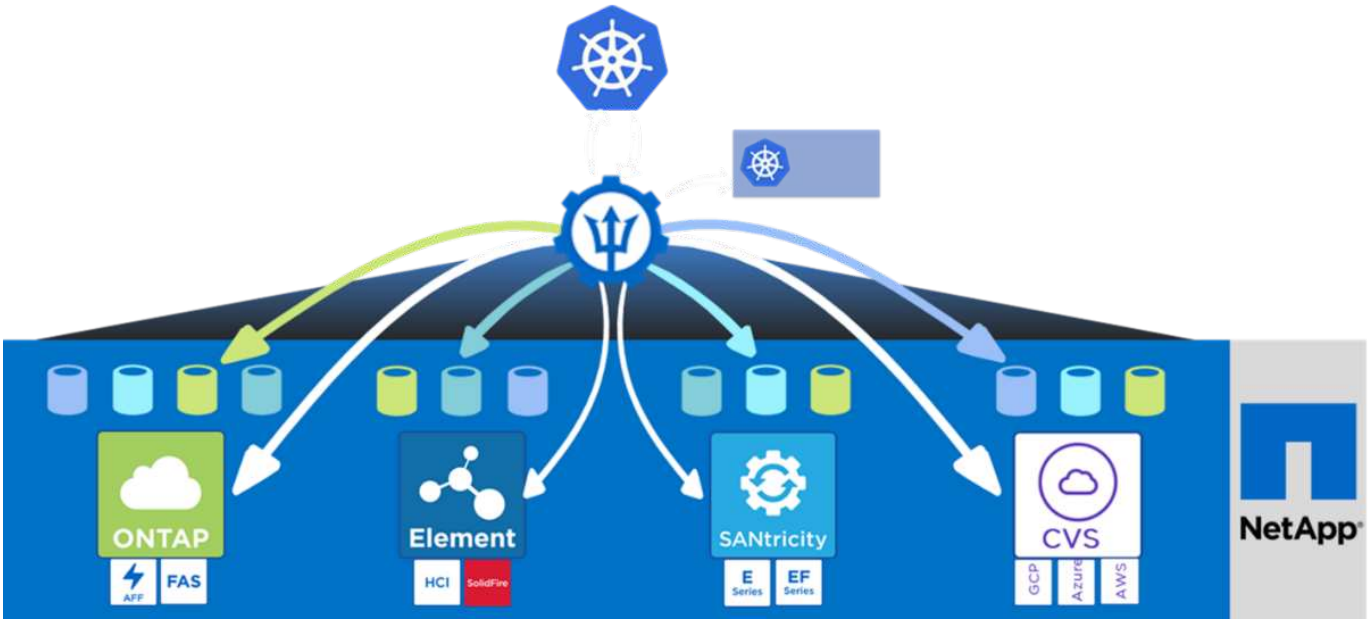
The following pages have additional information about the NetApp products that have been validated for application and persistent-storage management in the Anthos with NetApp solution.

Next: NetApp Astra Trident Overview.

Astra Trident Overview

Astra Trident is an fully supported, open-source storage orchestrator for containers and Kubernetes distributions, including Anthos. Trident works with the entire NetApp storage portfolio, including NetApp ONTAP and Element storage systems, and it also supports NFS and iSCSI connections. Trident accelerates the DevOps workflow by allowing end users to provision and manage storage from their NetApp storage systems without requiring intervention from a storage administrator.

An administrator can configure a number of storage backends based on project needs and storage system models that enable advanced storage features, including compression, specific disk types, and QoS levels that guarantee a certain level of performance. After they are defined, these backends can be used by developers in their projects to create persistent volume claims (PVCs) and to attach persistent storage to their containers on demand.



Astra Trident has a rapid development cycle and, like Kubernetes, is released four times a year.

The latest version of Astra Trident, 22.04, was released in April 2022. A support matrix for what version of Trident has been tested with which Kubernetes distribution can be found here.

Starting with the 20.04 release, Trident setup is performed by the Trident operator. The operator makes large scale deployments easier and provides additional support including self healing for pods that are deployed as a part of the Trident install.

With the 22.04 release, a Helm chart was made available to ease the installation of the Trident Operator.

Install the Trident Operator with Helm

To use Helm to automate installation of Trident on the deployed user cluster and provision a persistent volume, complete the following steps:

Helm is not installed by default on the GKE-Admin workstation. It can be downloaded in a binary format that works with Ubuntu from the Helm Install Page.

1. First, set the location of the user cluster's `kubeconfig` file as an environment variable so that you don't have to reference it, because Trident has no option to pass this file.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ export
KUBECONFIG=~/user-cluster-1/user-cluster-1-kubeconfig
```

2. Add the Trident Helm repository:

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer] helm repo add netapp-
trident https://netapp.github.io/trident-helm-chart
```

3. Run the Helm command to install the Trident operator from the tarball in the helm directory while creating the trident namespace in your user cluster.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ helm install trident
netapp-trident/trident-operator --version 22.4.0 --create-namespace
--namespace trident
NAME: trident
LAST DEPLOYED: Fri May  6 12:54:25 2022
NAMESPACE: trident
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
Thank you for installing trident-operator, which will deploy and manage
NetApp's Trident CSI
storage provisioner for Kubernetes.

Your release is named 'trident' and is installed into the 'trident'
namespace.
Please note that there must be only one instance of Trident (and
trident-operator) in a Kubernetes cluster.

To configure Trident to manage storage resources, you will need a copy
of tridentctl, which is
available in pre-packaged Trident releases.  You may find all Trident
releases and source code
online at https://github.com/NetApp/trident.

To learn more about the release, try:

  $ helm status trident
  $ helm get all trident
```

4. You can verify that Trident is successfully installed by checking the pods that are running in the namespace or by using the tridentctl binary to check the installed version.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl get pods -n
trident
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-5z45l                   1/2     Running   2          30s
trident-csi-696b685cf8-htdb2        6/6     Running   0          30s
trident-csi-b74p2                   2/2     Running   0          30s
trident-csi-lrw4n                   2/2     Running   0          30s
trident-operator-7c748d957-gr2gw    1/1     Running   0          36s

[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ ./tridentctl -n
trident version
+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 22.04.0          | 22.04.0          |
+----------------+----------------+
```

In some cases, customer environments might require the customization of the Trident deployment. In these cases, it is also possible to manually install the Trident operator and update the included manifests to customize the deployment.

Manually install the Trident Operator

To manually install Trident on the deployed user cluster and provision a persistent volume, complete the following steps:

1. Download the installation archive to the admin workstation and extract the contents. The current version of Trident is 22.04, which can be downloaded here.

```
[ubuntu@gke-admin-ws-2022-05-03 ~]$ wget
https://github.com/NetApp/trident/releases/download/v22.04.0/trident-
installer-22.04.0.tar.gz
--2022-05-06 15:17:30--
https://github.com/NetApp/trident/releases/download/v22.04.0/trident-
installer-22.04.0.tar.gz
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://github-
releases.githubusercontent.com/77179634/a4fa9f00-a9f2-11eb-9053-
98e8e573d4ae?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIWNJYAX4CSVEH53A%2F20210506%2Fus-east-
1%2Fs3%2Faws4_request&X-Amz-Date=20210506T191643Z&X-Amz-Expires=300&X-
```

```
Amz-
Signature=8a49a2a1e08c147d1ddd8149ce45a5714f9853fee19bb1c507989b9543eb36
30&X-Amz-
SignedHeaders=host&actor_id=0&key_id=0&repo_id=77179634&response-
content-disposition=attachment%3B%20filename%3Dtrident-installer-
22.04.0.tar.gz&response-content-type=application%2Foctet-stream
[following]
--2022-05-06 15:17:30--  https://github-
releases.githubusercontent.com/77179634/a4fa9f00-a9f2-11eb-9053-
98e8e573d4ae?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIWNJYAX4CSVEH53A%2F20210506%2Fus-east-
1%2Fs3%2Faws4_request&X-Amz-Date=20210506T191643Z&X-Amz-Expires=300&X-
Amz-
Signature=8a49a2a1e08c147d1ddd8149ce45a5714f9853fee19bb1c507989b9543eb36
30&X-Amz-
SignedHeaders=host&actor_id=0&key_id=0&repo_id=77179634&response-
content-disposition=attachment%3B%20filename%3Dtrident-installer-
22.04.0.tar.gz&response-content-type=application%2Foctet-stream
Resolving github-releases.githubusercontent.com (github-
releases.githubusercontent.com)... 185.199.108.154, 185.199.109.154,
185.199.110.154, ...
Connecting to github-releases.githubusercontent.com (github-
releases.githubusercontent.com)|185.199.108.154|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 38349341 (37M) [application/octet-stream]
Saving to: 'trident-installer-22.04.0.tar.gz'

100%[=====================================================================
==========================================>] 38,349,341  88.5MB/s
in 0.4s

2022-05-06 15:17:30 (88.5 MB/s) - 'trident-installer-22.04.0.tar.gz'
saved [38349341/38349341]
```

2. Extract the Trident install from the downloaded bundle.

```
[ubuntu@gke-admin-ws-2022-05-03 ~]$ tar -xzf trident-installer-
22.04.0.tar.gz
[ubuntu@gke-admin-ws-2022-05-03 ~]$ cd trident-installer/
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$
```

3. Set the location of the user cluster's `kubeconfig` file as an environment variable so that you don't have to reference it, because Trident has no option to pass this file.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ KUBECONFIG=~/user-
cluster-1/user-cluster-1-kubeconfig
```

4. The `trident-installer` directory contains manifests for defining all the required resources. Using the appropriate manifests, create the `TridentOrchestrator` custom resource definition.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
customresourcedefinition.apiextensions.k8s.io/tridentorchestrators.tride
nt.netapp.io created
```

5. If one does not exist, create a Trident namespace in your cluster using the provided manifest.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl apply -f
deploy/namespace.yaml
namespace/trident created
```

6. Create the resources required for the Trident operator deployment, such as a `ServiceAccount` for the operator, a `ClusterRole` and `ClusterRoleBinding` to the `ServiceAccount`, a dedicated `PodSecurityPolicy`, or the operator itself.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl create -f
deploy/bundle.yaml
serviceaccount/trident-operator created
clusterrole.rbac.authorization.k8s.io/trident-operator created
clusterrolebinding.rbac.authorization.k8s.io/trident-operator created
deployment.apps/trident-operator created
podsecuritypolicy.policy/tridentoperatorpods created
```

7. You can check the status of the operator after it's deployed with the following commands:

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl get
deployment -n trident
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
trident-operator   1/1     1            1           23s
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl get pods -n
trident
NAME                                READY   STATUS    RESTARTS   AGE
trident-operator-66f48895cc-lzczk   1/1     Running   0          41s
```

8. With the operator deployed, we can now use it to install Trident. This requires creating a `TridentOrchestrator`.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl create -f
deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl describe
torc trident
Name:         trident
Namespace:
Labels:       <none>
Annotations:  <none>
API Version:  trident.netapp.io/v1
Kind:         TridentOrchestrator
Metadata:
  Creation Timestamp:  2022-05-06T17:00:28Z
  Generation:          1
  Managed Fields:
    API Version:  trident.netapp.io/v1
    Fields Type:  FieldsV1
    fieldsV1:
      f:spec:
        .:
        f:debug:
        f:namespace:
    Manager:      kubectl-create
    Operation:    Update
    Time:         2022-05-06T17:00:28Z
    API Version:  trident.netapp.io/v1
    Fields Type:  FieldsV1
    fieldsV1:
      f:status:
        .:
        f:currentInstallationParams:
          .:
          f:IPv6:
          f:autosupportHostname:
          f:autosupportImage:
          f:autosupportProxy:
          f:autosupportSerialNumber:
          f:debug:
          f:enableNodePrep:
          f:imagePullSecrets:
          f:imageRegistry:
          f:k8sTimeout:
          f:kubeletDir:
          f:logFormat:
          f:silenceAutosupport:
```

```
          f:tridentImage:
        f:message:
        f:namespace:
        f:status:
        f:version:
    Manager:         trident-operator
    Operation:       Update
    Time:            2022-05-06T17:00:28Z
  Resource Version:  931421
  Self Link:
/apis/trident.netapp.io/v1/tridentorchestrators/trident
  UID:               8a26a7a6-dde8-4d55-9b66-a7126754d81f
Spec:
  Debug:      true
  Namespace:  trident
Status:
  Current Installation Params:
    IPv6:                      false
    Autosupport Hostname:
    Autosupport Image:         netapp/trident-autosupport:22.04
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                     true
    Enable Node Prep:          false
    Image Pull Secrets:
    Image Registry:
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Silence Autosupport:  false
    Trident Image:        netapp/trident:22.04.0
  Message:                Trident installed
  Namespace:              trident
  Status:                 Installed
  Version:                v22.04.0
Events:
  Type    Reason      Age   From                           Message
  ----    ------      ----  ----                           -------
  Normal  Installing  80s   trident-operator.netapp.io  Installing
Trident
  Normal  Installed   68s   trident-operator.netapp.io  Trident
installed
```

9. You can verify that Trident is successfully installed by checking the pods that are running in the namespace or by using the tridentctl binary to check the installed version.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl get pods -n
trident
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-bb64c6cb4-lmd6h         6/6     Running   0          82s
trident-csi-gn59q                   2/2     Running   0          82s
trident-csi-m4szj                   2/2     Running   0          82s
trident-csi-sb9k9                   2/2     Running   0          82s
trident-operator-66f48895cc-lzczk   1/1     Running   0          2m39s

[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ ./tridentctl -n
trident version
+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 22.04.0        | 22.04.0        |
+----------------+----------------+
```

Create storage-system backends

After completing the Astra Trident Operator install, you must configure the backend for the specific NetApp storage platform you are using. Follow the link below in order to continue the setup and configuration of Astra Trident.

Next: NetApp ONTAP NFS.

NetApp ONTAP NFS configuration

To enable Trident integration with the NetApp ONTAP storage system, you must create a backend that enables communication with the storage system.

1. There are sample backend files available in the downloaded installation archive in the `sample-input` folder hierarchy. For NetApp ONTAP systems serving NFS, copy the `backend-ontap-nas.json` file to your working directory and edit the file.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ cp sample-
input/backends-samples/ontap-nas/backend-ontap-nas.json ./
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi backend-ontap-
nas.json
```

2. Edit the backendName, managementLIF, dataLIF, svm, username, and password values in this file.

```
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "ontap-nas+10.61.181.221",
    "managementLIF": "172.21.224.201",
    "dataLIF": "10.61.181.221",
    "svm": "trident_svm",
    "username": "cluster-admin",
    "password": "password"
}
```

It is a best practice to define the custom backendName value as a combination of the storageDriverName and the dataLIF that is serving NFS for easy identification.

3. With this backend file in place, run the following command to create your first backend.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ ./tridentctl -n
trident create backend -f backend-ontap-nas.json
+------------------------+----------------
+--------------------------------------+--------+---------+
|          NAME          | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+------------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas+10.61.181.221 | ontap-nas      | be7a619d-c81d-445c-b80c-
5c87a73c5b1e | online |       0 |
+------------------------+----------------
+--------------------------------------+--------+---------+
```

4. With the backend created, you must next create a storage class. Just as with the backend, there is a sample storage class file that can be edited for the environment available in the sample-inputs folder. Copy it to the working directory and make necessary edits to reflect the backend created.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ cp sample-
input/storage-class-samples/storage-class-csi.yaml.templ ./storage-
class-basic.yaml
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi storage-class-
basic.yaml
```

5. The only edit that must be made to this file is to define the `backendType` value to the name of the storage driver from the newly created backend. Also note the name-field value, which must be referenced in a later step.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: basic-csi
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
```

There is an optional field called `fsType` that is defined in this file. This line can be deleted in NFS backends.

6. Run the `kubectl` command to create the storage class.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl create -f
storage-class-basic.yaml
storageclass.storage.k8s.io/basic-csi created
```

7. With the storage class created, you must then create the first persistent volume claim (PVC). There is a sample `pvc-basic.yaml` file that can be used to perform this action located in sample-inputs as well.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ cp sample-input/pvc-
samples/pvc-basic.yaml ./
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi pvc-basic.yaml
```

8. The only edit that must be made to this file is ensuring that the `storageClassName` field matches the one just created. The PVC definition can be further customized as required by the workload to be provisioned.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: basic
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: basic-csi
```

9. Create the PVC by issuing the `kubectl` command. Creation can take some time depending on the size of the backing volume being created, so you can watch the process as it completes.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl create -f
pvc-basic.yaml
persistentvolumeclaim/basic created

[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl get pvc
NAME    STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
basic   Bound    pvc-b4370d37-0fa4-4c17-bd86-94f96c94b42d   1Gi
RWO            basic-csi      7s
```

Next: NetApp ONTAP iSCSI.

NetApp ONTAP iSCSI configuration

To enable Trident integration with the NetApp ONTAP storage system, you must create a backend that enables communication with the storage system.

1. There are sample backend files available in the downloaded installation archive in the `sample-input` folder hierarchy. For NetApp ONTAP systems serving iSCSI, copy the `backend-ontap-san.json` file to your working directory and edit the file.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ cp sample-
input/backends-samples/ontap-san/backend-ontap-san.json ./
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi backend-ontap-
san.json
```

2. Edit the managementLIF, dataLIF, svm, username, and password values in this file.

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "managementLIF": "172.21.224.201",
  "dataLIF": "10.61.181.240",
  "svm": "trident_svm",
  "username": "admin",
  "password": "password"
}
```

3. With this backend file in place, run the following command to create your first backend.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ ./tridentctl -n
trident create backend -f backend-ontap-san.json
+------------------------+----------------
+--------------------------------------+--------+---------+
|          NAME          | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+------------------------+----------------
+--------------------------------------+--------+---------+
| ontapsan_10.61.181.241 | ontap-san      | 6788533c-7fea-4a35-b797-
fb9bb3322b91 | online |       0 |
+------------------------+----------------
+--------------------------------------+--------+---------+
```

4. With the backend created, you must next create a storage class. Just as with the backend, there is a sample storage class file that can be edited for the environment available in the sample-inputs folder. Copy it to the working directory and make necessary edits to reflect the backend created.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ cp sample-
input/storage-class-samples/storage-class-csi.yaml.templ ./storage-
class-basic.yaml
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi storage-class-
basic.yaml
```

5. The only edit that must be made to this file is to define the `backendType` value to the name of the storage driver from the newly created backend. Also note the name-field value, which must be referenced in a later step.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: basic-csi
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
```

> There is an optional field called `fsType` that is defined in this file. In iSCSI backends, this value can be set to a specific Linux filesystem type (XFS, ext4, and so on), or it can be deleted to allow the worker node OS to decide which filesystem to use.

6. Run the `kubectl` command to create the storage class.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl create -f
storage-class-basic.yaml
storageclass.storage.k8s.io/basic-csi created
```

7. With the storage class created, you must then create the first persistent volume claim (PVC). There is a sample `pvc-basic.yaml` file that can be used to perform this action located in sample-inputs as well.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ cp sample-input/pvc-
samples/pvc-basic.yaml ./
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi pvc-basic.yaml
```

8. The only edit that must be made to this file is ensuring that the `storageClassName` field matches the one just created. The PVC definition can be further customized as required by the workload to be provisioned.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: basic
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: basic-csi
```

9. Create the PVC by issuing the `kubectl` command. Creation can take some time depending on the size of the backing volume being created, so you can watch the process as it completes.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl create -f
pvc-basic.yaml
persistentvolumeclaim/basic created

[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl get pvc
NAME    STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
basic   Bound    pvc-7ceac1ba-0189-43c7-8f98-094719f7956c   1Gi
RWO            basic-csi      3s
```

Next: NetApp Element iSCSI.

NetApp Element iSCSI configuration

To enable Trident integration with the NetApp Element storage system, you must create a backend that enables communication with the storage system using the iSCSI protocol.

1. There are sample backend files available in the downloaded installation archive in the `sample-input` folder hierarchy. For NetApp Element systems serving iSCSI, copy the `backend-solidfire.json` file to your working directory and edit the file.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ cp sample-
input/backends-samples/solidfire/backend-solidfire.json ./
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi ./backend-
solidfire.json
```

   a. Edit the user, password, and MVIP value on the `EndPoint` line.

   b. Edit the `SVIP` value.

```
 {
    "version": 1,
    "storageDriverName": "solidfire-san",
    "Endpoint": "https://trident:password@172.21.224.150/json-
rpc/8.0",
    "SVIP": "10.61.180.200:3260",
    "TenantName": "trident",
    "Types": [{"Type": "Bronze", "Qos": {"minIOPS": 1000, "maxIOPS":
2000, "burstIOPS": 4000}},
              {"Type": "Silver", "Qos": {"minIOPS": 4000, "maxIOPS":
6000, "burstIOPS": 8000}},
              {"Type": "Gold", "Qos": {"minIOPS": 6000, "maxIOPS":
8000, "burstIOPS": 10000}}]
}
```

2. With this back-end file in place, run the following command to create your first backend.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ ./tridentctl -n
trident create backend -f backend-solidfire.json
+------------------------+----------------
+--------------------------------------+--------+---------+
|          NAME          | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+------------------------+----------------
+--------------------------------------+--------+---------+
| solidfire_10.61.180.200 | solidfire-san  | b90783ee-e0c9-49af-8d26-
3ea87ce2efdf | online |       0 |
+------------------------+----------------
+--------------------------------------+--------+---------+
```

3. With the backend created, you must next create a storage class. Just as with the backend, there is a sample storage class file that can be edited for the environment available in the sample-inputs folder. Copy it to the working directory and make necessary edits to reflect the backend created.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ cp sample-
input/storage-class-samples/storage-class-csi.yaml.templ ./storage-
class-basic.yaml
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi storage-class-
basic.yaml
```

4. The only edit that must be made to this file is to define the `backendType` value to the name of the storage driver from the newly created backend. Also note the name-field value, which must be referenced in a later step.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: basic-csi
provisioner: csi.trident.netapp.io
parameters:
  backendType: "solidfire-san"
```

> There is an optional field called `fsType` that is defined in this file. In iSCSI backends, this value can be set to a specific Linux filesystem type (XFS, ext4, and so on), or it can be deleted to allow OpenShift to decide what filesystem to use.

5. Run the `kubectl` command to create the storage class.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl create -f
storage-class-basic.yaml
storageclass.storage.k8s.io/basic-csi created
```

6. With the storage class created, you must then create the first persistent volume claim (PVC). There is a sample `pvc-basic.yaml` file that can be used to perform this action located in sample-inputs as well.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ cp sample-input/pvc-
samples/pvc-basic.yaml ./
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi pvc-basic.yaml
```

7. The only edit that must be made to this file is ensuring that the `storageClassName` field matches the one just created. The PVC definition can be further customized as required by the workload to be provisioned.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: basic
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: basic-csi
```

8. Create the PVC by issuing the `kubectl` command. Creation can take some time depending on the size of the backing volume being created, so you can watch the process as it completes.

```
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl create -f
pvc-basic.yaml
persistentvolumeclaim/basic created

[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ kubectl get pvc
NAME    STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
basic   Bound    pvc-3445b5cc-df24-453d-a1e6-b484e874349d   1Gi
RWO            basic-csi      5s
```

Next: Advanced Configuration Options.

Advanced configuration options

Typically, the easiest-to-deploy solution is best, but, in some cases, advanced customizations are required to meet the requirements or specifications of a specific application or the environment that solution is being deployed to. To this end, the Red Hat OpenShift with NetApp solution allows for the following customizations to meet these needs.

In this section we have documented some advanced configuration options such as using third-party load balancers or creating a private registry for hosting customized container images, both of which are prerequisites for installing the NetApp Astra Control Center.

The following pages have additional information about the advanced configuration options validated in the Red Hat OpenShift with NetApp solution:

Next: Exploring Load Balancer Options.

Exploring load balancer options

An application deployed in Anthos is exposed to the world by a service that is delivered by a load balancer deployed in the Anthos on-prem environment.

The following pages have additional information about load balancer options validated in the Anthos with NetApp solution:

- Installing F5 BIG-IP load balancers
- Installing MetalLB load balancers
- Installing SeeSaw load balancers

Next: Installing F5 BIG-IP load balancer.

Installing F5 BIG-IP load balancers

F5 BIG-IP is an Application Delivery Controller (ADC) that offers a broad set of advanced, production-grade traffic management and security services like L4-L7 load balancing, SSL/TLS offload, DNS, firewall, and more. These services dramatically increase the availability, security, and performance of your applications.

F5 BIG-IP can be deployed and consumed in various ways, including on dedicated hardware, in the cloud, or as a virtual appliance on-premises. Refer to the documentation here to explore and deploy F5 BIG-IP.

F5 BIG-IP was the first of the bundled load balancer solutions available with Anthos On-Prem and was used in a number of the early Anthos Ready partner validations for the Anthos with NetApp solution.

F5 BIG-IP can be deployed in standalone mode or in cluster mode. For the purpose of this validation, F5 BIG-IP was deployed in standalone mode. However, for production purposes, NetApp recommends creating a cluster of BIG-IP instances to avoid a single point of failure.

An F5 BIG-IP system can be deployed on dedicated hardware, in the cloud, or as a virtual appliance on-premises with versions greater than 12.x for it to be integrated with F5 CIS. For the purpose of this document, the F5 BIG-IP system was validated as a virtual appliance, for example using the BIG-IP VE edition.

Validated releases

This solution makes use of the virtual appliance deployed in VMware vSphere. Networking for the F5 Big-IP virtual appliance can be configured in a two-armed or three-armed configuration based on your network environment. The deployment in this document is based on the two-armed configuration. Additional details on configuring the virtual appliance for use with Anthos can be found here.

The Solutions Engineering Team at NetApp have validated the releases in the following table in our lab to work with deployments of Anthos On-Prem:

Make	Type	Version
F5	BIG-IP VE	15.0.1-0.0.11
F5	BIG-IP VE	16.1.0-0.0.19

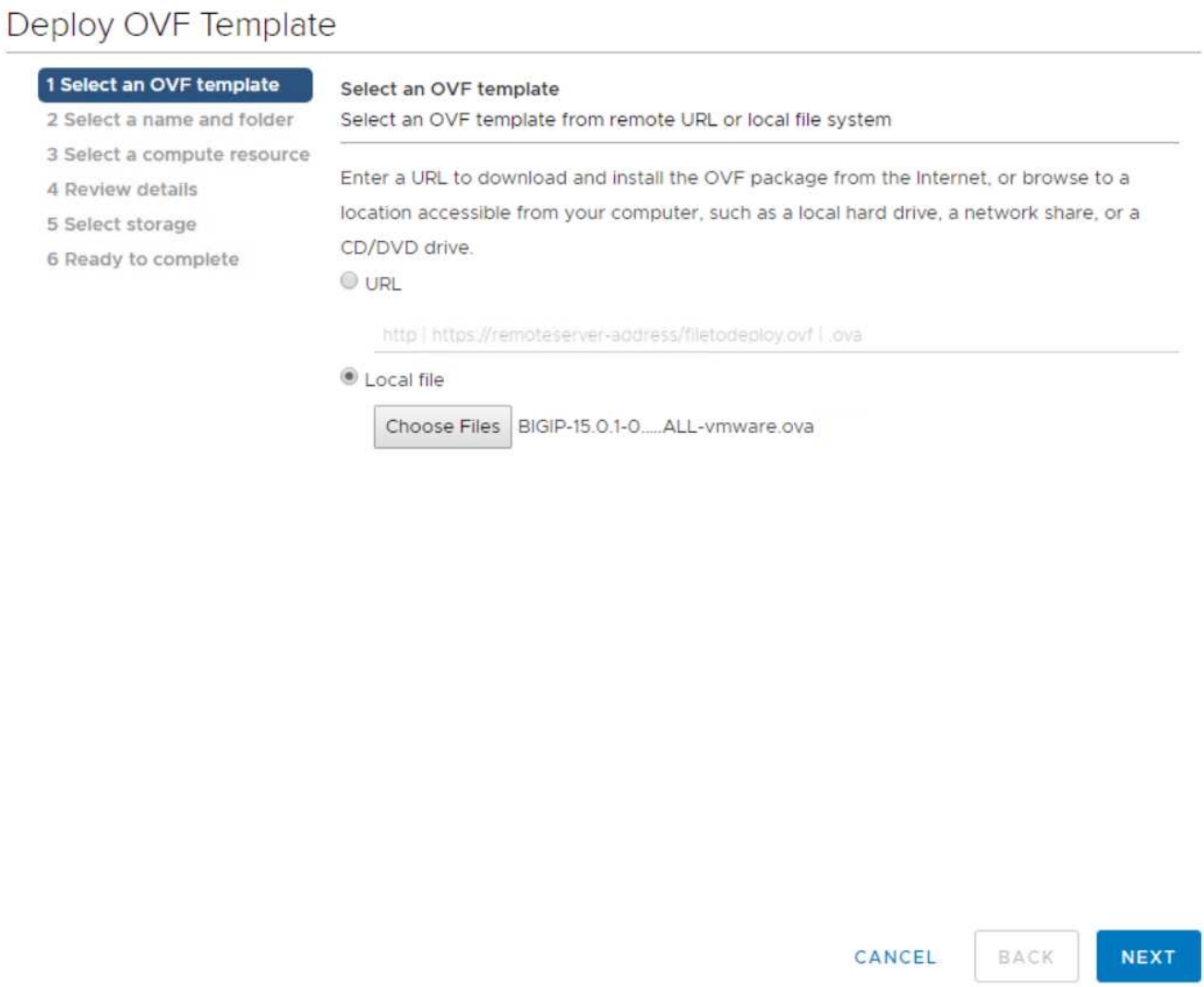
Installation

To install F5 BIG-IP, complete the following steps:

1. Download the virtual application Open Virtual Appliance (OVA) file from F5 here.

To download the appliance, a user must register with F5. They provide a 30-day demo license for the Big-IP Virtual Edition Load Balancer. NetApp recommends a permanent 10Gbps license for the production deployment of an appliance.

2. Right-click the Infrastructure Resource Pool and select Deploy OVF Template. A wizard launches that allows you to select the OVA file that you just downloaded in Step 1. Click Next.



3. Click Next to continue through each step and accept the default values for each screen presented until you reach the storage selection screen. Select the VM_Datastore that you would like to deploy the virtual machine to, and then click Next.
4. The next screen presented by the wizard allows you to customize the virtual networks for use in the environment. Select VM_Network for the External field and select Management_Network for the Management field. Internal and HA are used for advanced configurations for the F5 Big-IP appliance and are not configured. These parameters can be left alone, or they can be configured to connect to non-infrastructure, distributed port groups. Click Next.
5. Review the summary screen for the appliance, and, if all the information is correct, click Finish to start the deployment.
6. After the virtual appliance is deployed, right-click it and power it up. It should receive a DHCP address on the management network. The appliance is Linux-based, and it has VMware Tools deployed, so you can view the DHCP address it receives in the vSphere client.
7. Open a web browser and connect to the appliance at the IP address from the previous step. The default login is admin/admin, and, after the first login, the appliance immediately prompts you to change the admin password. It then returns you to a screen where you must log in with the new credentials.

8. The first screen prompts the user to complete the Setup Utility. Begin the utility by clicking Next.

9. The next screen prompts for activation of the license for the appliance. Click Activate to begin. When prompted on the next page, paste either the 30-day evaluation license key you received when you registered for the download or the permanent license you acquired when you purchased the appliance. Click Next.

> For the device to perform activation, the network defined on the management interface must be able to reach the internet.

10. On the next screen, the End User License Agreement (EULA) is presented. If the terms in the license are acceptable, click Accept.
11. The next screen counts the elapsed time as it verifies the configuration changes that have been made so far. Click Continue to resume with the initial configuration.

12. The Configuration Change window closes, and the Setup Utility displays the Resource Provisioning menu. This window lists the features that are currently licensed and the current resource allocations for the virtual appliance and each running service.

13. Clicking the Platform menu option on the left enables additional modification of the platform. Modifications include setting the management IP address configured with DHCP, setting the host name and the time zone the appliance is installed in, and securing the appliance from SSH accessibility.

14. Next click the Network menu, which enables you to configure standard networking features. Click Next to begin the Standard Network Configuration wizard.

15. The first page of the wizard configures redundancy; leave the defaults and click Next. The next page enables you to configure an internal interface on the load balancer. Interface 1.1 maps to the VMNIC labeled Internal in the OVF deployment wizard.

> ⓘ The spaces in this page for Self IP Address, Netmask, and Floating IP address can be filled with a non-routable IP for use as a placeholder. They can also be filled with an internal network that has been configured as a distributed port group for virtual guests if you are deploying the three-armed configuration. They must be completed to continue with the wizard.

16. The next page enables you to configure an external network that is used to map services to the pods deployed in Kubernetes. Select a static IP from the VM_Network range, the appropriate subnet mask, and a floating IP from that same range. Interface 1.2 maps to the VMNIC labeled External in the OVF deployment wizard.

17. On the next page, you can configure an internal-HA network if you are deploying multiple virtual appliances in the environment. To proceed, you must fill the Self-IP Address and the Netmask fields, and you must select interface 1.3 as the VLAN Interface, which maps to the HA network defined by the OVF template wizard.

18. The next page enables you to configure the NTP servers. Then click Next to continue to the DNS setup. The DNS servers and domain search list should already be populated by the DHCP server. Click Next to accept the defaults and continue.
19. For the remainder of the wizard, click Next to continue through the advanced peering setup, the configuration of which is beyond the scope of this document. Then click Finish to exit the wizard.
20. Create individual partitions for the Anthos admin cluster and each user cluster deployed in the environment. Click System in the menu on the left, navigate to Users, and click Partition List.

21. The displayed screen only shows the current common partition. Click Create on the right to create the first additional partition, and name it `GKE-Admin`. Then click Repeat, and name the partition `User-Cluster-1`. Click the Repeat button again to name the next partition `User-Cluster-2`. Finally click Finished to complete the wizard. The Partition list screen returns with all the partitions now listed.

Integration with Anthos

There is a section in each configuration file, respectively for the admin cluster, and each user cluster that you choose to deploy to configure the load balancer so that it is managed by Anthos On Prem.

The following script is a sample from the configuration of the partition for the GKE-Admin cluster. The values that need to be uncommented and modified are placed in bold text below:

```
# (Required) Load balancer configuration
loadBalancer:
  # (Required) The VIPs to use for load balancing
  vips:
    # Used to connect to the Kubernetes API
    controlPlaneVIP: "10.61.181.230"
    # # (Optional) Used for admin cluster addons (needed for multi cluster
features). Must
    # # be the same across clusters
    # # addonsVIP: ""
  # (Required) Which load balancer to use "F5BigIP" "Seesaw" or
"ManualLB". Uncomment
  # the corresponding field below to provide the detailed spec
  kind: F5BigIP
  # # (Required when using "ManualLB" kind) Specify pre-defined nodeports
  # manualLB:
  #   # NodePort for ingress service's http (only needed for user cluster)
  #   ingressHTTPNodePort: 0
  #   # NodePort for ingress service's https (only needed for user
cluster)
  #   ingressHTTPSNodePort: 0
  #   # NodePort for control plane service
  #   controlPlaneNodePort: 30968
  #   # NodePort for addon service (only needed for admin cluster)
  #   addonsNodePort: 31405
  # # (Required when using "F5BigIP" kind) Specify the already-existing
partition and
  # # credentials
  f5BigIP:
    address: "172.21.224.21"
    credentials:
      username: "admin"
      password: "admin-password"
    partition: "GKE-Admin"
  #   # # (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.
    #  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: 8192
    # (Optional) Network that the LB interface of Seesaw runs in (default:
cluster
    # network)
    #   vCenter:
      # vSphere network name
      #     networkName: VM_Network
    # (Optional) Run two LB VMs to achieve high availability (default:
false)
    #   enableHA: false
```

Next: Installing MetalLB load balancers.

Installing MetalLB load balancers

This page lists the installation and configuration instructions for the MetalLB managed load balancer.

Installing The MetalLB Load Balancer

The MetalLB load balancer is fully integrated with Anthos Clusters on VMware and has automated deployment performed as part of the Admin and User cluster setups starting with the 1.11 release. There are blocks of text in the respective `cluster.yaml` configuration files that you must modify to provide load balancer info. It is self-hosted on your Anthos cluster instead of requiring the deployment of external resources like the other supported load balancer solutions. It also allows you to create an ip-pool that automatically assigns addresses with the creation of Kubernetes services of type load balancer in clusters that do not run on a cloud provider.

Integration with Anthos

When enabling the MetalLB load balancer for Anthos admin, you must modify a few lines in the `loadBalancer:` section that exists in the `admin-cluster.yaml` file. The only values that you must modify are to set the `controlPlaneVIP:` address and then set the `kind:` as MetalLB. See the following code snippet for an example:

```
# (Required) Load balancer configuration
loadBalancer:
  # (Required) The VIPs to use for load balancing
  vips:
    # Used to connect to the Kubernetes API
    controlPlaneVIP: "10.61.181.230"
    # # (Optional) Used for admin cluster addons (needed for multi cluster
features). Must
    # # be the same across clusters
    # addonsVIP: ""
  # (Required) Which load balancer to use "F5BigIP" "Seesaw" "ManualLB" or
"MetalLB".
  # Uncomment the corresponding field below to provide the detailed spec
  kind: MetalLB
```

When enabling the MetalLB load balancer for Anthos user clusters, there are two areas in each `user-cluster.yaml` file that you must update. First, in a manner similar to the `admin-cluster.yaml` file, you must modify the `controlPlaneVIP:`, `ingressVIP:`, and `kind:` values in the `loadBalancer:` section. See the following code snippet for an example:

```
loadBalancer:
  # (Required) The VIPs to use for load balancing
  vips:
    # Used to connect to the Kubernetes API
    controlPlaneVIP: "10.61.181.240"
    # Shared by all services for ingress traffic
    ingressVIP: "10.61.181.244"
  # (Required) Which load balancer to use "F5BigIP" "Seesaw" "ManualLB" or
"MetalLB".
  # Uncomment the corresponding field below to provide the detailed spec
  kind: MetalLB
```

The ingressVIP IP address must exist within the pool of IP addresses assigned to the MetalLB load balancer later in the configuration.

You then need to navigate to the `metalLB:` subsection and modify the `addressPools:` section by naming the pool in the `- name:` variable. You must also create a pool of ip-addresses that MetalLB can assign to services of type LoadBalancer by providing a range to the `addresses:` variable.

```
# # (Required when using "MetalLB" kind in user clusters) Specify the
MetalLB config
 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: "default"
 #     # (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:
      - "10.61.181.244-10.61.181.249"
```

The address pool can be provided as a range like in the example, limiting it to a number of addresses in a particular subnet, or it can be provided as a CIDR notation if the entire subnet is made available.

1. When Kubernetes services of type LoadBalancer are created, MetalLB automatically assigns an externalIP to the services and advertises the IP address by responding to ARP requests.

Next: Installing SeeSaw load balancers.

Installing SeeSaw load balancers

This page lists the installation and configuration instructions for the SeeSaw managed load balancer.

Seesaw is the default managed network load balancer installed in an Anthos Clusters on VMware environment from versions 1.6 to 1.10.

Installing The SeeSaw load balancer

The SeeSaw load balancer is fully integrated with Anthos Clusters on VMware and has automated deployment performed as part of the Admin and User cluster setups. There are blocks of text in the `cluster.yaml` configuration files that must be modified to provide load balancer info, and then there is an additional step prior to cluster deployment to deploy the load balancer using the built in `gkectl` tool.

SeeSaw load balancers can be deployed in HA or non-HA mode. For the purpose of this validation, the SeeSaw load balancer was deployed in non-HA mode, which is the default setting. For production purposes, NetApp recommends deploying SeeSaw in an HA configuration for fault tolerance and reliability.

Integration with Anthos

There is a section in each configuration file, respectively for the admin cluster, and in each user cluster that you choose to deploy to configure the load balancer so that it is managed by Anthos On-Prem.

The following text is a sample from the configuration of the partition for the GKE-Admin cluster. The values that

need to be uncommented and modified are placed in bold text below:

```
loadBalancer:
  # (Required) The VIPs to use for load balancing
  vips:
    # Used to connect to the Kubernetes API
    controlPlaneVIP: "10.61.181.230"
    # # (Optional) Used for admin cluster addons (needed for multi cluster
features). Must
    # # be the same across clusters
    # # addonsVIP: ""
  # (Required) Which load balancer to use "F5BigIP" "Seesaw" or
"ManualLB". Uncomment
  # the corresponding field below to provide the detailed spec
  kind: Seesaw
  # # (Required when using "ManualLB" kind) Specify pre-defined nodeports
  # manualLB:
  #   # NodePort for ingress service's http (only needed for user cluster)
  #   ingressHTTPNodePort: 0
  #   # NodePort for ingress service's https (only needed for user
cluster)
  #   ingressHTTPSNodePort: 0
  #   # NodePort for control plane service
  #   controlPlaneNodePort: 30968
  #   # NodePort for addon service (only needed for admin cluster)
  #   addonsNodePort: 31405
  # # (Required when using "F5BigIP" kind) Specify the already-existing
partition and
  # # credentials
  # f5BigIP:
  #   address:
  #   credentials:
  #     username:
  #     password:
  #   partition:
  #   # # (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.
  ipBlockFilePath: "admin-seesaw-block.yaml"
  #   (Required) The Virtual Router IDentifier of VRRP for the Seesaw
group. Must
  #   be between 1-255 and unique in a VLAN.
```

```
    vrid: 100
  #   (Required) The IP announced by the master of Seesaw group
    masterIP: "10.61.181.236"
  #   (Required) The number CPUs per machine
    cpus: 1
  #   (Required) Memory size in MB per machine
    memoryMB: 2048
  #   (Optional) Network that the LB interface of Seesaw runs in (default:
cluster
  #   network)
    vCenter:
  #   vSphere network name
      networkName: VM_Network
  #   (Optional) Run two LB VMs to achieve high availability (default:
false)
    enableHA: false
```

The SeeSaw load balancer also has a separate static `seesaw-block.yaml` file that you must provide for each cluster deployment. This file must be located in the same directory relative to the `cluster.yaml` deployment file, or the full path must be specified in the section above.

A sample of the `admin-seesaw-block.yaml` file looks like the following script:

```
blocks:
  - netmask: "255.255.255.0"
    gateway: "10.63.172.1"
    ips:
    - ip: "10.63.172.152"
      hostname: "admin-seesaw-vm"
```

This file provides the gateway and netmask for the network that the load balancer provides to the underlying cluster, as well as the management IP and hostname for the virtual machine that is deployed to run the load balancer.

Next: Solution validation/use cases.

Solution Validation and Use Cases

The examples provided on this page are solution validations and use cases for Anthos with NetApp.

Install an application using the Google Cloud Console

Next: Videos and Demos.

Videos and Demos

The following video demonstrates some of the capabilities documented in this document:

Video: Deployment of Anthos on bare metal

Next: Where to find additional information.

Where to find additional information

To learn more about the information described in this document, review the following websites:

- NetApp Documentation

  https://docs.netapp.com/

- NetApp Astra Trident Documentation

  https://docs.netapp.com/us-en/trident/index.html

- NetApp Astra Control Center Documentation

  https://docs.netapp.com/us-en/astra-control-center/

- Anthos Clusters on VMware Documentation

  https://cloud.google.com/anthos/clusters/docs/on-prem/1.10/overview

- Anthos on bare metal Documentation

  https://cloud.google.com/anthos/clusters/docs/bare-metal/latest

- VMware vSphere Documentation

  https://docs.vmware.com/

Copyright information

Trademark information