



Containers

NetApp Solutions

NetApp

August 23, 2022

Table of Contents

NetApp Container Solutions	1
NVA-1165: Anthos with NetApp	1
TR-4919: DevOps with NetApp Astra	45
NVA-1160: Red Hat OpenShift with NetApp	73
NVA-1166: VMware Tanzu with NetApp	236
Archived Solutions	281

NetApp Container Solutions

:x

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 its 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.

i 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 a 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-5z451          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
```

```
tridentorchesterator.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:
/api/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.

- 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 |
+-----+-----+
+-----+-----+-----+
```

- 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
```

- 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 |
+-----+
+-----+-----+
```

- 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.template ./storage-
class-basic.yaml
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi storage-class-
basic.yaml
```

- 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.

- 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 | b90783eee-e0c9-49af-8d26-
3ea87ce2efdf | online |      0 |
+-----+-----+
+-----+-----+-----+
```

- 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.template ./storage-
class-basic.yaml
[ubuntu@gke-admin-ws-2022-05-03 trident-installer]$ vi storage-class-
basic.yaml
```

- 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.

- 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
```

- 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
```

- 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
```

- 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 MetallLB 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.

Deploy OVF Template

1 Select an OVF template

- 2 Select a name and folder
- 3 Select a compute resource
- 4 Review details
- 5 Select storage
- 6 Ready to complete

Select an OVF template

Select an OVF template from remote URL or local file system

Enter a URL to download and install the OVF package from the Internet, or browse to a location accessible from your computer, such as a local hard drive, a network share, or a CD/DVD drive.

URL

http | https://remoteserver-address/filetodeploy.ovf | .ova

Local file

BIGIP-15.0.1-0....ALL-vmware.ova

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" "ManuallLB" 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" "ManuallLB" 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 `metallLB:` 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.

- 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/>

TR-4919: DevOps with NetApp Astra

Alan Cowles and Nikhil M Kulkarni, NetApp

Use cases

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

- Easy to deploy and manage applications and development environments deployed on top of supported Kubernetes distributions.
- Discussion of real-world use cases for DevOps workflows and examples of the tools and methods that NetApp can provide to make adoption and use of these methods easier.
- Exploration of how application-consistent snapshot, backups, and clones can be used to enhance the DevOps experience.

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 so that workflows are never interrupted.
- Ease of deployment and management procedures for the end user.
- API-driven and programmable infrastructure to keep up with microservices and developer agility.
- Ability to scale infrastructure independently and in an automated fashion, based on workload demands.
- Protecting applications alongside their backing persistent data sets for DevOps workflows accelerate time to market by not having to rely on redeployments or manual copying of data.

Recognizing these capabilities and challenges, this technical report outlines the process of improving and simplifying DevOps use cases for containerized applications using the wide portfolio of NetApp products.

Technology overview

The DevOps with NetApp solution contains the following major components:

DevOps practices

DevOps practices focus on automated, repeatable, and easily manageable operations that enhance the development workflow by allowing the end user to control the environment in which they are developing their code. This solution provides several examples and use cases in which NetApp technology can be of the greatest benefit to such operations.

Container orchestration

There are numerous container orchestration platforms in use today. Although most of these platforms are based on Kubernetes, each has pros and cons. So it is important to understand feature sets and integrations when selecting a container orchestration platform for DevOps workflows. With the NetApp Astra suite of products, we support the following platforms for full-fledged DevOps use cases:

- [Red Hat OpenShift 4.6.8+](#)
- [Rancher 2.5+](#)
- [Kubernetes 1.20+](#)
- [VMware Tanzu Kubernetes Grid 1.4+](#)
- [VMware Tanzu Kubernetes Grid Integrated Edition 1.12.2+](#)

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

NetApp Astra Control Center offers a rich set of storage and application-aware data management services for stateful Kubernetes workloads deployed in an on-prem environment and powered by trusted NetApp data-protection technology.

For more information, visit the NetApp Astra website [here](#).

Astra Trident is an open-source and fully-supported storage orchestrator for containers and Kubernetes distributions like Red Hat OpenShift, VMware Tanzu, Anthos by Google Cloud etc..

For more information, visit the Astra Trident website [here](#).

[Next: DevOps Overview.](#)

DevOps Overview

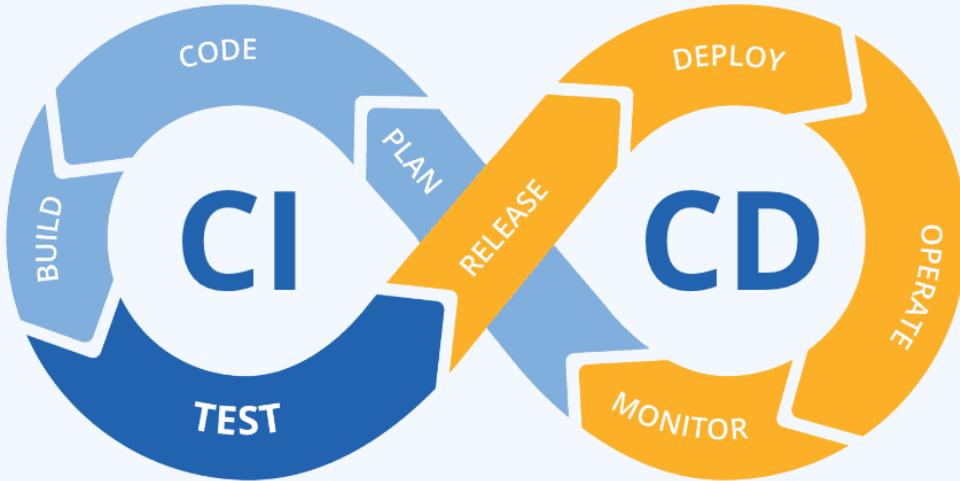
Over the past several years, organizations that build software have been embracing the concepts of DevOps. DevOps practices break down organizational barriers, bringing development and operations teams closer together. DevOps practices also empower the teams to accelerate delivery, increase availability, and make services and applications more stable, thus improving the team's productivity. In addition, adoption of an automation framework is also a key ingredient of success — from building, testing, and operating applications at scale or managing a fully automated infrastructure platform or stack. Below we discuss some primary use cases for DevOps where NetApp solutions can be implemented to help enhance the experiences that DevOps practitioners encounter during their daily practice.

DevOps use cases

Although DevOps does not have a single, universally accepted definition, solutions for DevOps practitioners typically contain similar constructs or ideologies that enable easy implementation, repetition, and management at scale. The following sections describe potential use cases for DevOps workflows enabled by NetApp solutions.

Continuous Integration, Continuous Delivery, and Continuous Deployment (CI/CD)

Continuous Integration, Continuous Delivery, and Continuous Deployment (CI/CD) is a coding philosophy that encourages developers to implement and transform their coding practices by establishing a method by which they can consistently update, test, and deploy their code in an automated fashion. The most popular method by which CI/CD is implemented in most DevOps workflows is that of the CI/CD pipeline, and there are several third-party software applications that can help achieve this.



See the following examples of popular applications that can help with CI/CD-type workflows:

[ArgoCD](#)
[Jenkins](#)
[Tekton](#)

Some of the use cases included later in this technical report have been demonstrated in Jenkins, but the primary CI/CD principles can be applied to whatever tool an organization has implemented in their own practices.

Infrastructure as code

Infrastructure as code helps provision and manage IT resources through automated commands, APIs, and software development kits (SDK). This concept greatly enhance the DevOps experience by removing physical data center or resource limitations that could prevent developers from meeting their objectives.



End users often use programming languages such as [Python](#) or automation tools such as [Ansible](#) or [Puppet](#) to create automated and repeatable infrastructure scaling actions that can be called by developers when needed.

Both NetApp ONTAP and Astra Control contain public facing APIs and ansible modules or software development toolkits that make automating operations very easy to adopt and integrate into DevOps processes.

[Next: NetApp Storage Systems Overview.](#)

NetApp storage systems overview

NetApp has several storage platforms that are qualified with Astra Trident and Astra Control to provision, protect, and manage data for containerized applications.



- 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.



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

The following pages have additional information about the NetApp storage systems validated in the DevOps with NetApp solution:

- [NetApp ONTAP](#)

Next: [NetApp storage integrations overview](#).

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, non-disruptive 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 protecting 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 non-rewritable 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 multi-protocol support.

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

For more information about NETAPP AFF/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 it 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 that can 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 Storage Integrations Overview.

NetApp Storage Integration Overview

NetApp provides a number of products to help you orchestrate, manage, protect, and migrate stateful containerized applications and their data.



NetApp Astra Control offers a rich set of storage and application-aware data management services for stateful Kubernetes workloads powered by NetApp data protection technology. The Astra Control Service is available to support stateful workloads in cloud-native Kubernetes deployments. The Astra Control Center is available to support stateful workloads in on-premises deployments of Enterprise Kubernetes platforms like Red Hat OpenShift, Rancher, VMware Tanzu etc. For more information visit the NetApp Astra Control website [here](#).

NetApp Astra Trident is an open-source and fully-supported storage orchestrator for containers and Kubernetes distributions like Red Hat OpenShift, Rancher, VMware Tanzu etc. For more information, visit the

Astra Trident website [here](#).

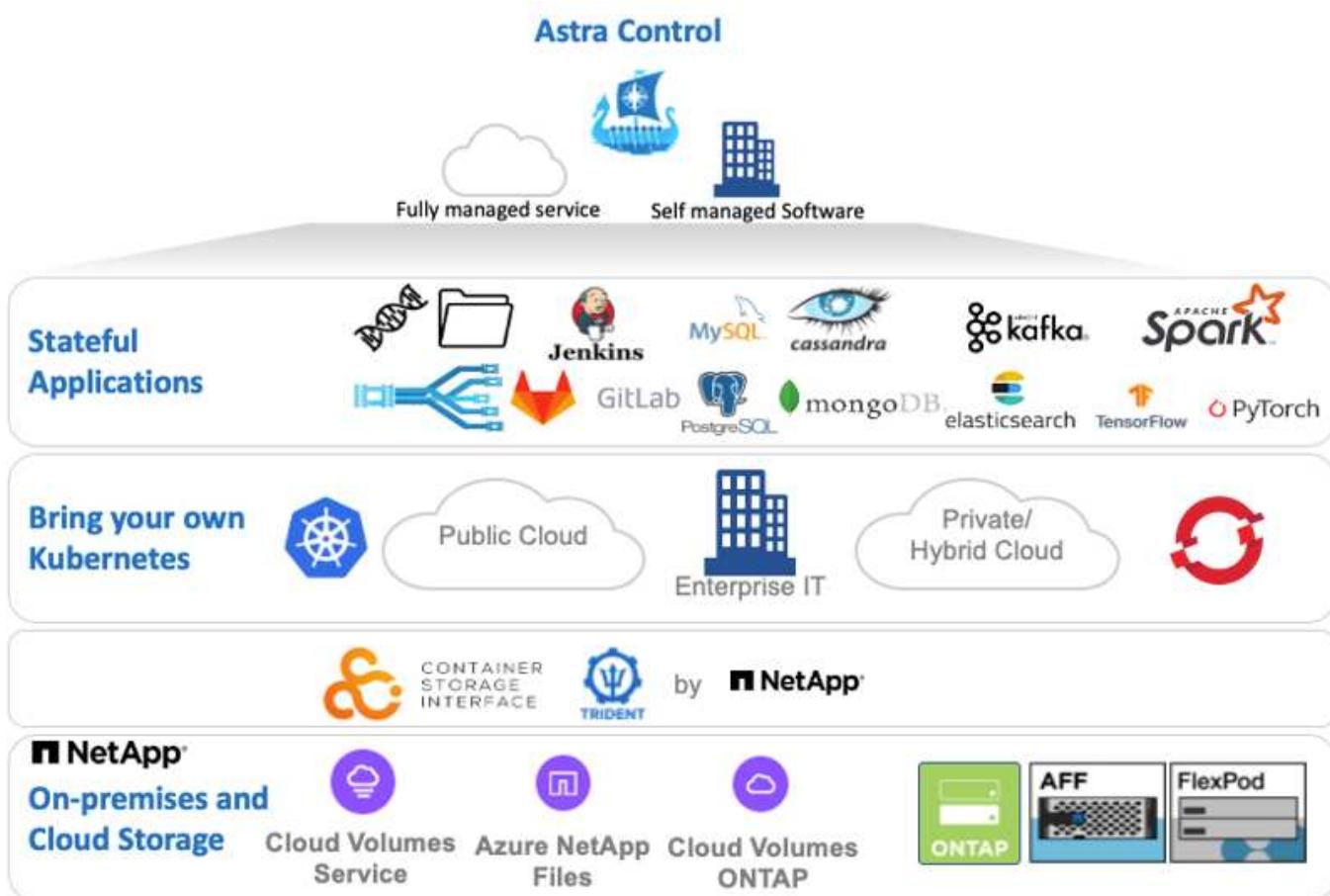
The following pages have additional information about the NetApp products that have been validated for application and persistent storage management in the DevOps with NetApp solution:

- [NetApp Astra Control Center](#)
- [NetApp Astra Trident](#)

Next: Use-case Validations: DevOps with NetApp Astra.

NetApp Astra Control overview

NetApp Astra Control Center offers a rich set of storage and application-aware data management services for stateful Kubernetes workloads deployed in an on-premises environment and powered by NetApp data protection technology.



NetApp Astra Control Center can be installed on a Kubernetes cluster that has the Astra Trident storage orchestrator deployed and configured with storage classes and storage backends to NetApp ONTAP storage systems.

For more information on Astra Trident, see [this document here](#).

In a cloud-connected environment, Astra Control Center uses Cloud Insights to provide advanced monitoring and telemetry. In the absence of a Cloud Insights connection, limited monitoring and telemetry (seven days worth of metrics) is available and exported to Kubernetes native monitoring tools (Prometheus and Grafana) through open metrics endpoints.

Astra Control Center is fully integrated into the NetApp AutoSupport and Active IQ ecosystem to provide support for users, provide assistance with troubleshooting, and display usage statistics.

In addition to the paid version of Astra Control Center, a 90-day evaluation license is also available. The evaluation version is supported through email and the community Slack channel. Customers have access to these resources, other knowledge-base articles, and documentation available from the in-product support dashboard.

To understand more about the Astra portfolio, visit the [Astra website](#).

For a detailed installation and operations guide on Astra Control Center, follow the documentation [here](#).

Astra Control Center automation

Astra Control Center has a fully functional REST API for programmatic access. Users can use any programming language or utility to interact with Astra Control REST API endpoints. To learn more about this API, see the documentation [here](#).

If you are looking for a ready-made software development toolkit for interacting with Astra Control REST APIs, NetApp provides a toolkit with Astra Control Python SDK, which you can download [here](#).

If programming is not appropriate for your situation and you would like to use a configuration management tool, you can clone and run the Ansible playbooks that NetApp publishes [here](#).

[Next: Use-case Validations: DevOps with NetApp Astra](#)

Astra Trident Overview

Astra Trident is an open-source, fully supported storage orchestrator for containers and Kubernetes distributions like Red Hat OpenShift, VMware Tanzu, Anthos by Google Cloud, Rancher etc. Trident works with the entire NetApp storage portfolio, including the 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, or 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 is 22.04 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 21.01 release, a Helm chart was made available to ease the installation of the Trident Operator.

Refer to the documentation [here](#) to install and use Astra Trident.

Next: Use-case Validations: DevOps with NetApp Astra.

Use-case validation: DevOps with NetApp Astra

The following use cases have been validated for DevOps with NetApp Astra:

- Integrate Protection into CI/CD Pipelines with NetApp Astra Control
- Leverage Astra Control to facilitate Post-mortem Analysis and Restore the Application
- Accelerating Software Development with NetApp FlexClones

Next: Videos and Demos - DevOps with NetApp Astra.

Integrate Protection into CI/CD Pipelines with NetApp Astra Control

Overview

One of the most common uses of DevOps workflows is continuous integration and continuous deployment (CI/CD) pipelines that build, integrate, and run automated test suites on applications as developers commit new code. DevOps engineers and site-reliability engineers (SREs) typically have pipelines dedicated to the various workflows for new feature development, regression testing, bug fixes, quality engineering, and other functions in the development process.

As teams increase their level of automation, the pace of change for in-production applications can feel overwhelming. Therefore, some teams prefer to protect in-production applications or services. In addition to protecting the code and container images, they also want to protect the application state, configuration data (such as Kubernetes objects and resources associated with the application), and an application's persistent data.

In this use case, we take a closer look at a promotion-to-production pipeline that deploys a new version of an application: first into a staging environment and then into a production environment. This example applies equally to the major public clouds and also to an on-premises environment. Although we show the deployment of one version of the app, the pipeline can also be used with other strategies, such as blue/green or canary deployment. As part of the CI/CD pipeline, we're going to protect the application by creating a complete application backup. An application-aware backup of the in-production application and its data, state, and configuration can be useful for numerous DevOps workflows.



The application used for validating this use-case was [Magento](#), an e-commerce solution with a web-based front end; an Elasticsearch instance for search and analysis features; and a MariaDB database that tracks all the shopping inventory and transaction details. This containerized application was installed in a Red Hat OpenShift cluster. Every pod in the application used persistent volumes to store data. The persistent volumes were automatically created by NetApp Astra Trident, the Container Storage Interface-compliant storage orchestrator for Kubernetes that enables storage to be provisioned on NetApp storage systems. Further, to utilize the Astra Control Center's application protection capabilities, the application in question was managed by Astra Control, which was then used to trigger application backups that stored the state of the application along with the data held in persistent volumes. We used the [NetApp Astra Control Python SDK](#) to automate the process of triggering application backups, which was then introduced into a CI/CD pipeline. This pipeline was created and executed using a popular CI/CD tool called [[Jenkins](#)] to automate the flow to build, protect, and deploy the application.

Let us run through the prerequisites and procedure to introduce protection in a CI/CD pipeline.

Use-case validation prerequisites

The following tools or platforms were deployed and configured as prerequisites:

1. Red Hat OpenShift Container Platform
2. NetApp Astra Trident installed on OpenShift with a backend to NetApp ONTAP system configured

3. A default storageclass configured, pointing to a NetApp ONTAP backend
4. NetApp Astra Control Center installed on an OpenShift cluster
5. OpenShift cluster added as a managed cluster to Astra Control Center
6. Jenkins installed on an OpenShift cluster and configured with an agent node with a Docker engine installed on it

Installing the application

Let's start with the initial installation of the application in the staging and production environments. For the purpose of this use case, this step is a prerequisite, so it is performed manually. The CI/CD pipeline is used for subsequent build and deploy workflows as a result of new version releases of the application.

The production environment in this use case is a namespace called `magento-prod`, and the corresponding staging environment is a namespace called `magento-staging` configured on the Red Hat OpenShift cluster. To install the application, complete the following steps:

1. Install the Magento application using bitnami helm chart on the production environment. We use RWX PVs for Magento and Mariadb pods.

```
[netapp-user@rhel7 na_astra_control_suite]$ helm install --version 14
magento bitnami/magento -n magento-prod --create-namespace --set
image.tag=2.4.1-debian-10-
r11,magentoHost=10.63.172.243,persistence.magento.accessMode=ReadWriteMa
ny,persistence.apache.accessMode=ReadWriteMany,mariadb.master.persistence.accessModes[0]=ReadWriteMany
```



Magento bitnami helm chart requires a LoadBalancer service to expose the Magento GUI service. We used [MetalLB](#) for providing an on-prem load balancer service in this example.

2. After a few minutes, verify that all pods and services are running.

```
[netapp-user@rhel7 na_astra_control_suite]$ oc get pods -n magento-prod
NAME                                         READY   STATUS
RESTARTS   AGE
magento-9d658fd96-qrxmt                     1/1    Running
0          49m
magento-elasticsearch-coordinating-only-69869cc5-768rm 1/1    Running
0          49m
magento-elasticsearch-data-0                  1/1    Running
0          49m
magento-elasticsearch-master-0                1/1    Running
0          49m
magento-mariadb-0                           1/1    Running
0          49m
```

3. Repeat the same procedure for the staging environment.

Manage the Magento application in Astra Control Center

1. Navigate to Applications and select the Discovered applications tab.
2. Click the ellipsis against the Magento application in the production environment (`magento-prod`), and click Manage.
3. The Magento application is now managed by the Astra Control Center. All operations supported by Astra Control can be performed on the application. Note the version of the application as well.

The screenshot shows the Astra Control Center interface for managing the 'magento-prod' application. At the top, there's a header with the application name and a status indicator labeled 'Available'. Below the header, there are two main sections: 'App status' and 'App protection status'. The 'App status' section indicates the application is 'Healthy'. The 'App protection status' section shows it is 'Partially Protected'. Further down, there are details about the application's images, protection schedule, group, and cluster. The images listed are docker.io/bitnami/elasticsearch:6.8.10-debian-10-r16, docker.io/bitnami/magento:2.4.1-debian-10-r11, and docker.io/bitnami/mariadb:10.3.23-debian-10-r38. The protection schedule is set to 'Disabled'. The application belongs to the 'magento-prod' group and is part of the 'ocp-vmw' cluster.

4. Repeat the steps for managing the Magento application in the staging environment (`magento-staging`).

CI/CD pipeline with integrated protection

When we work with new versions of applications, we use a CI/CD pipeline to build the container image, take backups of both the staging and production environments, deploy the new version of the application to the staging environment, wait for approval to promotion to production, and then deploy the new version of the application to the production environment. To use a CI/CD pipeline, complete the following steps:

1. Log into Jenkins, and create the required credentials: one for Magento creds, one for Mariadb admin creds, and the third for Mariadb root creds.
2. Navigate to Manage Jenkins > Manage Credentials and click the appropriate domain.
3. Click Add Credentials, and set the kind to Username with password and scope set to Global. Enter the username, password, and an ID for the credentials and click OK.

The screenshot shows the Jenkins Manage Credentials screen. The user is adding a new global credential of type 'Username with password'. The 'Scope' is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Username' field contains 'admin', and the 'Password' field contains a masked value. The 'ID' field is set to 'magento-cred'. There is also a 'Description' field which is currently empty. At the bottom of the form is a blue 'OK' button.

4. Repeat the same procedure for the other two credentials.

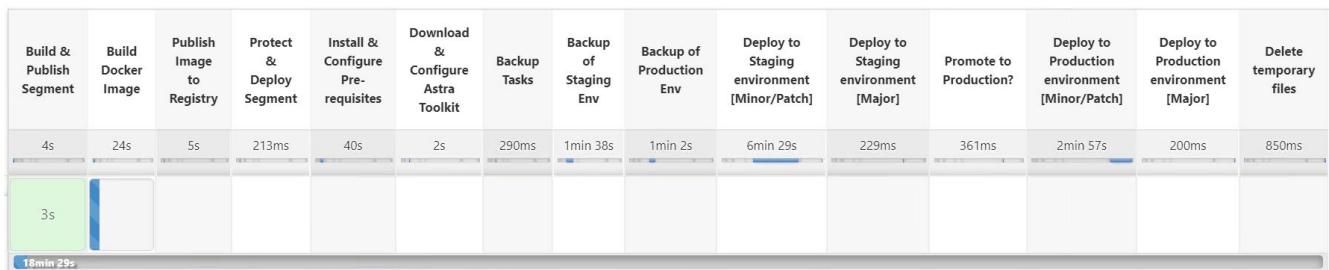
5. Go back to the Dashboard, create a pipeline by clicking New Item, and then click Pipeline.
6. Copy the pipeline from the Jenkinsfile [here](#).
7. Paste the pipeline into the Jenkins pipeline section and then click Save.
8. Fill the parameters of the Jenkins pipeline with the respective details including the helm chart version, the Magento application version to be upgraded to, the Astra toolkit version, the Astra Control Center FQDN, the API token, and its instance ID. Specify the docker registry, namespace, and Magento IP of both production and staging environments, and also specify the credential IDs of the credentials created.

```

MAGENTO_VERSION = '2.4.1-debian-10-r14'
CHART_VERSION = '14'
RELEASE_TYPE = 'MINOR'
ASTRA_TOOLKIT_VERSION = '2.0.2'
ASTRA_API_TOKEN = 'xxxxxxxxx'
ASTRA_INSTANCE_ID = 'xxx-xxx-xxx-xxx-xxx'
ASTRA_FQDN = 'netapp-astra-control-center.org.example.com'
DOCKER_REGISTRY = 'docker.io/netapp-solutions-cicd'
PROD_NAMESPACE = 'magento-prod'
PROD_MAGENTO_IP = 'x.x.x.x'
STAGING_NAMESPACE = 'magento-staging'
STAGING_MAGENTO_IP = 'x.x.x.x'
MAGENTO_CREDS = credentials('magento-cred')
MAGENTO_MARIADB_CREDS = credentials('magento-mariadb-cred')
MAGENTO_MARIADB_ROOT_CREDS = credentials('magento-mariadb-root-cred')

```

9. Click Build Now. The pipeline starts executing and progresses through the steps. The application image is first built and uploaded to the container registry.



10. The application backups are initiated via Astra Control.

The screenshot shows the Astra Control Center interface for the 'magento-prod' application. At the top, there are two cards: 'App status' (Healthy) and 'App protection status' (Partially Protected). Below these are sections for 'Images' (listing Elasticsearch, Magento, and MariaDB), 'Protection schedule' (Disabled), 'Group' (magento-prod), and 'Cluster' (ocp-vmw). The main area has tabs for 'Overview', 'Data protection' (selected), 'Storage', 'Resources', and 'Activity'. Under 'Data protection', there's a 'Configure protection policy' button and a table listing one backup entry: 'upgrade-prod-2-4-1-debian-10-r20' (Ready, On-Demand, Created: 2021/10/29 14:43 UTC, Status: Running).

11. After the backup stages have completed successful, verify the backups from the Astra Control Center.

This screenshot is identical to the one above, showing the Astra Control Center for the 'magento-prod' application. The status remains healthy and partially protected. The protection schedule is still disabled. The single backup entry ('upgrade-prod-2-4-1-debian-10-r20') is now listed as 'Available' instead of 'Running'.

12. The new version of the application is then deployed to the staging environment.

Build & Publish Segment	Build Docker Image	Publish Image to Registry	Protect & Deploy Segment	Install & Configure Pre-requisites	Download & Configure Astra Toolkit	Backup Tasks	Backup of Staging Env	Backup of Production Env	Deploy to Staging environment [Minor/Patch]	Deploy to Staging environment [Major]	Promote to Production?	Deploy to Production environment [Minor/Patch]	Deploy to Production environment [Major]	Delete temporary files
4s	47s	7s	238ms	1min 25s	2s	273ms	1min 53s	1min 18s	5min 20s	211ms	337ms	2min 39s	187ms	780ms
3s	4min 16s	30s	485ms	7s	3s	153ms	6min 9s	5min 9s						

7min 1s

13. After this step is completed, the program waits for the user to approve deployment to production. At this stage, assume that the QA team performs some manual testing and approves production. You can then click Approve to deploy the new version of the application to the production environment.



14. Verify that the production application is also upgraded to the desired version.

magento-prod Available

App status: Healthy

App protection status: Partially Protected

Images:

- docker.io/bitnami/elasticsearch:6.8.12-debian-10-r61
- docker.io/bitnami/mariadb:10.3.24-debian-10-r49
- docker.io/niksleo415/magento:2.4.1-debian-10-r14

Protection schedule: Disabled

Group: magento-prod

Cluster: ocp-vmw

As part of the CI/CD pipeline, we demonstrated the ability to protect the application by creating a complete application-aware backup. Because the entire application has been backed up as part of the promotion-to-production pipeline, you can feel more confident about highly automated application deployments. This application-aware backup containing the data, state, and configuration of the application can be useful for numerous DevOps workflows. One important workflow would be to roll back to the previous version of the application in case of unforeseen issues.

Although we demonstrated a CI/CD workflow through with Jenkins tool, the concept can easily and efficiently be extrapolated to different tools and strategies. To see this use case in action, watch the video [here](#).

Next: [Videos and Demos - DevOps with NetApp Astra](#).

Use Astra Control to facilitate post-mortem analysis and restore the application

Overview

In the [first use case](#), we demonstrated how to use NetApp Astra Control Center to protect your applications in Kubernetes. That section describes how to integrate application backups via Astra Control directly into your development workflow by using the Python SDK in the NetApp Astra toolkit. This approach allows for the protection of development and production environments by automating on-demand backups during the continuous integration and continuous deployment (CI/CD) process. With this extra layer of application-consistent data protection added to the CI/CD pipeline and the production applications, the development processes is safe if something goes wrong in the process, which promotes good business-continuity practices.

In a traditional workflow, after encountering a failure when the application is upgraded to a new version, the development team would attempt to troubleshoot the issue in real time based on bug reports being provided by customers. Alternatively, at the first sign of trouble, the team could attempt to redeploy the application to a parallel debugging environment to take that process offline. They could redeploy an older code base from a previous version into production, which would restore the application to working order.



Although this approach works, the team would have to make sure that the state of the broken production app matched that of the version seen in production when the issues occurred. They would also have to spend time promoting the known-good build into production by fetching code from their repository and redeploying the machine images to restore the application to a good running state. Also, in this scenario, we didn't consider whether the production database itself was corrupted by the faulty code. Ideally, there are separate backup processes in place for the database data, but must we assume that they're consistent with the state of the application as it was published? This is where the benefits of stateful and application-consistent backups, restores and clones with Astra Control really show their value.

First, we can use Astra Control to facilitate post-mortem analysis on the state of the application. We do this by cloning the buggy production version to a parallel testing environment in an application-consistent manner. Having this environment set aside in its bug-ridden state enable us to troubleshoot the problem in real time.

Furthermore, Astra Control supports the in-place restore capability that allows us to restore the production application to a last acceptable backup (that preceded the afflicted version of code). The restored version assumes the position of the previous, buggy production application, in an application-consistent and stateful manner, including the ingress IP previously assigned. As a result, customers accessing the front end would be unaware of the transition to the backup version.



Use-case validation prerequisites

The following tools or platforms were deployed and configured as prerequisites:

- Red Hat OpenShift Container Platform.
- NetApp Astra Trident installed on OpenShift with a backend configured to a NetApp ONTAP system.
- A default storageclass configured, pointing to a NetApp ONTAP backend.
- NetApp Astra Control Center installed on an OpenShift cluster.
- OpenShift cluster added as a managed cluster to Astra Control Center.
- Jenkins installed on an OpenShift cluster.
- Magento application installed in the production environment. The production environment in this use case is a namespace called 'magento-prod' in a Red Hat OpenShift cluster.
- Production application managed by Astra Control Center.
- Known-good backup(s) of the production application captured with Astra Control.

Clone and restore pipeline

Considering that the application has been upgraded to a new version, the application in the production environment (`magento-prod`) isn't behaving as intended after the upgrade. Let's assume that the data being returned by front-end queries doesn't match the request or that the database has in fact been corrupted. To clone and restore the pipeline, complete the following steps:



This site can't be reached

10.63.172.243 took too long to respond.

Try:

- Checking the connection
- [Checking the proxy and the firewall](#)
- [Running Windows Network Diagnostics](#)

ERR_CONNECTION_TIMED_OUT

[Reload](#)

[Details](#)

1. Log into Jenkins and create a pipeline by clicking New Item and then Pipeline.
2. Copy the pipeline from the Jenkinsfile [here](#).
3. Paste the pipeline into the Jenkins pipeline section and then click Save.
4. Fill the parameters of the Jenkins pipeline with the respective details like the current Magento application version in production, the Astra Control Center FQDN, the API token, the instance ID and application name or namespace of production and debug environments, and the source and destination cluster names. For the purpose of this use case, the production environment is a namespace called 'magento-prod' and the debug environment is a namespace called 'magento-debug' configured on a Red Hat OpenShift cluster.

```
MAGENTO_VERSION = '2.4.1-debian-10-r14'  
ASTRA_TOOLKIT_VERSION = '2.0.2'  
ASTRA_API_TOKEN = 'xxxxxx'  
ASTRA_INSTANCE_ID = 'xxx-xxx-xxx-xxx-xxx'  
ASTRA_FQDN = 'netapp-astra-control-center.org.example.com'  
PROD_APP_NAME = 'magento-prod'  
DEBUG_APP_NAME = 'magento-debug'  
DEBUG_NAMESPACE = 'magento-debug'  
PROD_KUBERNETES_CLUSTER = 'ocp-vmw'  
DEBUG_KUBERNETES_CLUSTER = 'ocp-vmw'
```

5. Click Build Now. The pipeline starts executing and progresses through the steps. The application is first cloned in the current state to a debug environment, and the application is then restored to the known-working backup.

Pipeline magento_clone-for-triage_restore-from-backup



6. Verify that the cloned application is the bug-containing version.



7. Verify that the production environment is restored to a working backup, and the application in production works as expected.



These two operations in tandem expedite the return to normal business operations. To see this use case in action, watch the video [here](#).

Next: Videos and Demos - DevOps with NetApp Astra.

Accelerating software development with NetApp FlexClone technology

Overview

Cloning a deployed application in a Kubernetes cluster is a very useful tool for developers that would like to expedite their workflows by sharing environments with partners or by testing new versions of code in a development environment without interfering with the version they are currently working on. The stateful and application-consistent cloning of a Kubernetes application is a major feature included with NetApp Astra Control, alongside the backup and restore of applications. As a bonus, if an application is cloned within the same Kubernetes cluster using the same storage backend, Astra Control defaults to using NetApp FlexClone technology for the duplication of persistent data volumes, speeding up the process significantly. By accelerating this process, the cloned environment is provisioned and available for use in a few moments, allowing developers to resume their work with just a brief pause when compared to redeploying their test or development environment. As an additional convenience, all of the functions available in NetApp Astra Control can be called with an API, which allows for easy integration into automation frameworks like Ansible. Therefore, environments can be staged even more rapidly because only minor changes are needed in a playbook or role to begin the cloning procedure.

What is NetApp FlexClone technology?

NetApp FlexClone technology is a writeable, point-in-time snapshot-based copy of a NetApp FlexVol. They are provisioned almost instantly, contain all of the data from the source volume, and consume no additional storage space until the data in the new volume begins to diverge from the source. They are often used in development or template-based environments when multiple copies of data are useful for staging purposes and storage systems have limited resources for provisioning these volumes. Compared to a traditional storage system in which data must be copied multiple times resulting in the consumption of significant storage space and time, NetApp FlexClone technology accelerates storage-dependant tasks.

Traditional Data Copies



Traditional physical copies take additional time and consume additional storage space

NetApp FlexClone Copies



NetApp FlexClone copies are near instantaneous and only consume space when written to

To find out more about NetApp FlexClone technology, visit the page on [NetApp Docs](#).

Prerequisites

1. A supported Kubernetes Distribution, such as Red Hat OpenShift 4.6.8+, Rancher 2.5+, or Kubernetes 1.19+.
2. NetApp Astra Control Center 21.12+.
3. A NetApp ONTAP system with a storage backend configured through NetApp Astra Trident.
4. Ansible 2.9+.
5. Templates for the environments that you'd like to clone as managed applications in NetApp Astra Control.

Use-case introduction

For this use case, we visualize something similar to the following workflow:



1. A user runs the ansible playbook to create a new staging environment.
2. Ansible uses the URI-API module to call out to Astra Control to execute the cloning operation.
3. Astra Control executes a cloning operation on a preprovisioned template environment, thus creating a new managed application.



This environment can be a single standalone application in development or an entire development environment like a Jenkins CI/CD pipeline.

4. The user then pulls a version of their code into the cloned dev environment from an online repository like Gitea.
5. The new version of the application is deployed and managed by NetApp Astra Control.



Both of these processes can be automated.

6. The user can develop new code in this cloned environment.
7. When the user is satisfied with their development efforts, they can push the code back to the hosted repository.

The use case presented here depends on the existence of golden templates for the particular environments or applications you would like to clone. In our environment we have created three such templates, one for a Wordpress deployment, one for a Magento deployment, and one for a Jenkins CI/CD environment with Gitea that we have titled DevTools.

The screenshot shows the HCG Solutions application management interface. On the left, there's a sidebar with various icons. The main area is titled "Applications" and contains a table with three entries:

Name	Ready	Protected	Cluster	Group	Actions
devtools-template	✓	ⓘ	ocp-vmware2	devtools-template	Available
magento-template	✓	ⓘ	ocp-vmware2	magento-template	Available
wordpress-template	✓	ⓘ	ocp-vmware2	wordpress-template	Available

Each of these environments is managed by NetApp Astra control, with persistent volumes currently stored on a NetApp ONTAP storage system with an NFS backend provided by NetApp Astra Trident.

Use-case validation

- Clone the ansible toolkit provided by the NetApp Solutions Engineering team, which includes the cloning role and the application update playbook.

```
[netapp-user@rhel7 ~]$ git clone https://github.com/NetApp-Automation/na_astra_control_suite.git
[netapp-user@rhel7 ~]$ cd na_astra_control_suite
```

- Edit `vars/clone_vars.yml` and fill in the global values that fit your Astra Control environment.

```
astra_control_fqdn: astra-control-center.example.com
astra_control_account_id: "xxxx-xxxx-xxxx-xxxx-xxxx"
astra_control_api_token: "xxxxxx"
```



The global environment values you need to fill out are available under the user profile icon in NetApp Astra Control under the API Access menu.

The screenshot shows the 'API access' section of the NetApp Astra Control interface. At the top, there's a 'API documentation' link to 'https://astra-control-center.cie.netapp.com/api'. To the right, the 'Account ID' is listed as 'fa9214eb-670d-41f1-bfcf-34cb3b69fda1'. On the far right, there are a bell icon and a user profile icon with a red circle around it, indicating notifications or updates. Below this, the 'API tokens' section is shown. It has a header with 'Actions', a 'Generate API token' button, and a search bar. A message below the table says '0-0 of 0 entries'. The table itself has columns for 'Token name' (with a checkbox), 'Created ↑', and 'Actions'. In the center of the table area is a large key icon. Below the table, a message reads 'You don't have any API token(s) right now' and 'When you have created one, it will be listed here'. A blue 'Generate new API token' button is at the bottom of this message area.

- With the global variables completed, you can choose the values for the specific application you wish to clone. To clone the devtools environment to a personal environment called `alan-devtools`, you would do the following:

```
clone_details:  
  - clone_name: alan-devtools  
    destination_namespace: alan-dev-namespace  
    source_cluster_name: ocp-vmware2  
    destination_cluster_name: ocp-vmware2  
    source_application_name: devtools-template
```



To take advantage of NetApp FlexClone technology in the cloning process, `src-cluster` and `dest-cluster` must be the same.

- You can now execute the playbook to clone the application.

```
[netapp-user@rhel7 na_astra_control_suite]$ ansible-playbook -K  
clone_app_playbook.yml]
```



The playbook as written must be run by the root user or someone that can escalate through the sudo process by passing the "-K" argument.

- When the playbook completes its run, the cloned application shows as available in the Astra Control Center console.

Name	Ready	Protected	Cluster	Group	Actions
alans-devtools	✓	⚠	ocp-vmware2	alans-dev-namespace	Available
devtools-template	✓	ⓘ	ocp-vmware2	devtools-template	Available
magento-template	✓	ⓘ	ocp-vmware2	magento-template	Available
wordpress-template	✓	ⓘ	ocp-vmware2	wordpress-template	Available

- A user can then log into the Kubernetes environment where the application was deployed, verify that the application is exposed with a new IP address, and start their development work.

For a demonstration of this use case and a example of upgrading an application, see [here](#).

Next: [Videos and Demos - DevOps with NetApp Astra](#).

Videos and demos: DevOps with NetApp Astra

The following videos demonstrate some of the capabilities described in this document:

- [Video: Integrate Data Protection in CI/CD pipeline with Astra Control](#)
- [Video: Leverage NetApp Astra Control to Perform Post-mortem Analysis and Restore Your Application](#)

- Video: Accelerate Software Development with Astra Control and NetApp FlexClone Technology

Next: Additional Information: DevOps with NetApp Astra.

Additional Information: DevOps with NetApp Astra

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

- NetApp Documentation

<https://docs.netapp.com/>

- Astra Trident Documentation

<https://docs.netapp.com/us-en/trident/>

- NetApp Astra Control Center Documentation

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

- Ansible Documentation

<https://docs.ansible.com/>

- Red Hat OpenShift Documentation

https://access.redhat.com/documentation/en-us/openshift_container_platform/4.8/

- Rancher Documentation

<https://rancher.com/docs/>

- Kubernetes Documentation

<https://kubernetes.io/docs/home/>

NVA-1160: Red Hat OpenShift with NetApp

Alan Cowles and Nikhil M Kulkarni, NetApp

This reference document provides deployment validation of the Red Hat OpenShift solution, deployed through Installer Provisioned Infrastructure (IPI) in several different data center environments as validated by NetApp. It also details storage integration with NetApp storage systems by making use of the Astra Trident storage orchestrator for the management of persistent storage. Lastly, a number of solution validations and real world use cases are explored and documented.

Use cases

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

- Easy to deploy and manage Red Hat OpenShift deployed using IPI (Installer Provisioned Infrastructure) on bare metal, Red Hat OpenStack Platform, Red Hat Virtualization, and VMware vSphere.

- Combined power of enterprise container and virtualized workloads with Red Hat OpenShift deployed virtually on OSP, RHV, or vSphere, or on bare metal with OpenShift Virtualization.
- Real world configuration and use cases highlighting the features of Red Hat OpenShift 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
- Ability to run virtualized and containerized workloads simultaneously
- Ability to scale infrastructure independently based on workload demands

Red Hat OpenShift with NetApp acknowledges these challenges and presents a solution that helps address each concern by implementing the fully automated deployment of Red Hat OpenShift IPI in the customer's choice of data center environment.

Technology overview

The Red Hat OpenShift with NetApp solution is comprised of the following major components:

Red Hat OpenShift Container Platform

Red Hat OpenShift Container Platform is a fully supported enterprise Kubernetes platform. Red Hat makes several enhancements to open-source Kubernetes to deliver an application platform with all the components fully integrated to build, deploy, and manage containerized applications.

For more information visit the OpenShift website [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

NetApp Astra Control Center offers a rich set of storage and application-aware data management services for stateful Kubernetes workloads, deployed in an on-prem environment and powered by trusted NetApp data protection technology.

For more information, visit the NetApp Astra website [here](#).

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

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 Control Center	Application Aware Data Management	21.12.60
NetApp Astra Trident	Storage Orchestration	22.01.0
Red Hat OpenShift	Container orchestration	4.6 EUS, 4.7, 4.8
Red Hat OpenStack Platform	Private Cloud Infrastructure	16.1
Red Hat Virtualization	Data center virtualization	4.4
VMware vSphere	Data center virtualization	6.7U3

Next: [Red Hat OpenShift Overview](#).

OpenShift Overview

The Red Hat OpenShift Container Platform unites development and IT operations on a single platform to build, deploy, and manage applications consistently across on-premises and hybrid cloud infrastructures. Red Hat OpenShift is built on open-source innovation and industry standards, including Kubernetes and Red Hat Enterprise Linux CoreOS, the world's leading enterprise Linux distribution designed for container-based workloads. OpenShift is part of the Cloud Native Computing Foundation (CNCF) Certified Kubernetes program, providing portability and interoperability of container workloads.

Red Hat OpenShift provides the following capabilities:

- **Self-service provisioning.** Developers can quickly and easily create applications on demand from the tools that they use most, while operations retain full control over the entire environment.
- **Persistent storage.** By providing support for persistent storage, OpenShift Container Platform allows you to run both stateful applications and cloud-native stateless applications.
- **Continuous integration and continuous development (CI/CD).** This source-code platform manages build and deployment images at scale.
- **Open-source standards.** These standards incorporate the Open Container Initiative (OCI) and Kubernetes for container orchestration, in addition to other open-source technologies. You are not

restricted to the technology or to the business roadmap of a specific vendor.

- **CI/CD pipelines.** OpenShift provides out-of-the-box support for CI/CD pipelines so that development teams can automate every step of the application delivery process and make sure it's executed on every change that is made to the code or configuration of the application.
- **Role-Based Access Control (RBAC).** This feature provides team and user tracking to help organize a large developer group.
- **Automated build and deploy.** OpenShift gives developers the option to build their containerized applications or have the platform build the containers from the application source code or even the binaries. The platform then automates deployment of these applications across the infrastructure based on the characteristic that was defined for the applications. For example, how quantity of resources that should be allocated and where on the infrastructure they should be deployed in order for them to be compliant with third-party licenses.
- **Consistent environments.** OpenShift makes sure that the environment provisioned for developers and across the lifecycle of the application is consistent from the operating system, to libraries, runtime version (for example, Java runtime), and even the application runtime in use (for example, tomcat) in order to remove the risks originated from inconsistent environments.
- **Configuration management.** Configuration and sensitive data management is built in to the platform to make sure that a consistent and environment agnostic application configuration is provided to the application no matter which technologies are used to build the application or which environment it is deployed.
- **Application logs and metrics.** Rapid feedback is an important aspect of application development. OpenShift integrated monitoring and log management provides immediate metrics back to developers in order for them to study how the application is behaving across changes and be able to fix issues as early as possible in the application lifecycle.
- **Security and container catalog.** OpenShift offers multitenancy and protects the user from harmful code execution by using established security with Security-Enhanced Linux (SELinux), CGroups, and Secure Computing Mode (seccomp) to isolate and protect containers. It also provides encryption through TLS certificates for the various subsystems and access to Red Hat certified containers (access.redhat.com/containers) that are scanned and graded with a specific emphasis on security to provide certified, trusted, and secure application containers to end users.



Physical



Virtual



Private cloud



Public cloud



Managed cloud
(Azure, AWS, IBM, Red Hat)

Deployment methods for Red Hat OpenShift

Starting with Red Hat OpenShift 4, the deployment methods for OpenShift include manual deployments using User Provisioned Infrastructure (UPI) for highly customized deployments or fully automated deployments using Installer Provisioned Infrastructure (IPI).

The IPI installation method is the preferred method in most cases because it allows for the rapid deployment of OCP clusters for dev, test, and production environments.

IPI installation of Red Hat OpenShift

The Installer Provisioned Infrastructure (IPI) deployment of OpenShift involves these high-level steps:

1. Visit the Red Hat OpenShift [website](#) and login with your SSO credentials.
2. Select the environment that you would like to deploy Red Hat OpenShift into.

Install OpenShift Container Platform 4

Select an infrastructure provider

 Run on Amazon Web Services	 Run on Microsoft Azure	 Run on Google Cloud Platform	 Run on VMware vSphere
 Red Hat OpenStack Platform Run on Red Hat OpenStack	 Red Hat Virtualization Run on Red Hat Virtualization	 Run on Bare Metal	 IBM Z IBM LinuxONE Run on IBM Z
 Power Systems Run on Power	 Run on Laptop Powered by Red Hat CodeReady Containers		

3. On the next screen download the installer, the unique pull secret, and the CLI tools for management.

Downloads

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

Linux ▾ [Download installer](#)

Pull secret
Download or copy your pull secret. The install program will prompt you for your pull secret during installation.

[Download pull secret](#)  [Copy pull secret](#)

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

Linux ▾ [Download command-line tools](#)

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

4. Follow the [installation instructions](#) provided by Red Hat to deploy to your environment of choice.

NetApp validated OpenShift deployments

NetApp has tested and validated the deployment of Red Hat OpenShift in its labs using the Installer Provisioned Infrastructure (IPI) deployment method in each of the following data center environments:

- [OpenShift on Bare Metal](#)
- [OpenShift on Red Hat OpenStack Platform](#)
- [OpenShift on Red Hat Virtualization](#)
- [OpenShift on VMware vSphere](#)

[Next: NetApp Storage Overview.](#)

OpenShift on Bare Metal

OpenShift on Bare Metal provides an automated deployment of the OpenShift Container Platform on commodity servers.

OpenShift on Bare Metal is similar to virtual deployments of OpenShift, which provide ease of deployment, rapid provisioning, and scaling of OpenShift clusters, while supporting virtualized workloads for applications that are not ready to be containerized. By deploying on bare metal, you do not require the extra overhead necessary to manage the host hypervisor environment in addition to the OpenShift environment. By deploying directly on bare metal servers, you can also reduce the physical overhead limitations of having to share resources between the host and OpenShift environment.

OpenShift on Bare Metal provides the following features:

- **IPI or assisted installer deployment.** With an OpenShift cluster deployed by Installer Provisioned Infrastructure (IPI) on bare metal servers, customers can deploy a highly versatile, easily scalable OpenShift environment directly on commodity servers, without the need to manage a hypervisor layer.
- **Compact cluster design.** To minimize the hardware requirements, OpenShift on bare metal allows for users to deploy clusters of just 3 nodes, by enabling the OpenShift control plane nodes to also act as worker nodes and host containers.
- **OpenShift virtualization.** OpenShift can run virtual machines within containers by using OpenShift Virtualization. This container-native virtualization runs the KVM hypervisor inside of a container, and attaches persistent volumes for VM storage.
- **AI/ML-optimized infrastructure.** Deploy applications like Kubeflow for machine learning applications by incorporating GPU-based worker nodes to your OpenShift environment and leveraging OpenShift Advanced Scheduling.

Network design

The Red Hat OpenShift on NetApp solution uses two data switches to provide primary data connectivity at 25Gbps. It also uses two management switches that provide connectivity at 1Gbps for in-band management for the storage nodes and out-of-band management for IPMI functionality.

For OpenShift bare-metal IPI deployment, you must create a provisioner node, a Red Hat Enterprise Linux 8 machine that must have network interfaces attached to separate networks.

- **Provisioning network.** This network is used to boot the bare-metal nodes and install the necessary images and packages to deploy the OpenShift cluster.
- **Bare-metal network.** This network is used for public-facing communication of the cluster after it is deployed.

For the setup of the provisioner node, the customer creates bridge interfaces that allow the traffic to route properly on the node itself and on the Bootstrap VM that is provisioned for deployment purposes. After the cluster is deployed, the API and ingress VIP addresses are migrated from the bootstrap node to the newly deployed cluster.

The following images depict the environment both during IPI deployment and after the deployment is complete.



7_L_OpenShift_0320



VLAN requirements

The Red Hat OpenShift with NetApp solution is designed to logically separate network traffic for different purposes by using virtual local area networks (VLANs).

VLANs	Purpose	VLAN ID
Out-of-band management network	Management for bare metal nodes and IPMI	16
Bare-metal network	Network for OpenShift services once cluster is available	181
Provisioning network	Network for PXE boot and installation of bare metal nodes via IPI	3485



Although each of these networks is virtually separated by VLANs, each physical port must be set up in Access Mode with the primary VLAN assigned, because there is no way to pass a VLAN tag during a PXE boot sequence.

Network infrastructure support resources

The following infrastructure should be in place prior to the deployment of the OpenShift container platform:

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

[Next: NetApp storage overview.](#)

OpenShift on Red Hat OpenStack Platform

The Red Hat OpenStack Platform delivers an integrated foundation to create, deploy, and scale a secure and reliable private OpenStack cloud.

OSP is an infrastructure-as-a-service (IaaS) cloud implemented by a collection of control services that manage compute, storage, and networking resources. The environment is managed using a web-based interface that allows administrators and users to control, provision, and automate OpenStack resources. Additionally, the OpenStack infrastructure is facilitated through an extensive command line interface and API enabling full automation capabilities for administrators and end-users.

The OpenStack project is a rapidly developed community project that provides updated releases every six months. Initially Red Hat OpenStack Platform kept pace with this release cycle by publishing a new release along with every upstream release and providing long term support for every third release. Recently, with the OSP 16.0 release (based on OpenStack Train), Red Hat has chosen not to keep pace with release numbers but instead has backported new features into sub-releases. The most recent release is Red Hat OpenStack Platform 16.1, which includes backported advanced features from the Ussuri and Victoria releases upstream.

For more information about OSP see the [Red Hat OpenStack Platform website](#).

OpenStack services

OpenStack Platform services are deployed as containers, which isolates services from one another and enables easy upgrades. The OpenStack Platform uses a set of containers built and managed with Kolla. The deployment of services is performed by pulling container images from the Red Hat Custom Portal. These service containers are managed using the Podman command and are deployed, configured, and maintained with Red Hat OpenStack Director.



Service	Project name	Description
Dashboard	Horizon	Web browser-based dashboard that you use to manage OpenStack services.
Identity	Keystone	Centralized service for authentication and authorization of OpenStack services and for managing users, projects, and roles.
OpenStack networking	Neutron	Provides connectivity between the interfaces of OpenStack services.
Block storage	Cinder	Manages persistent block storage volumes for virtual machines (VMs).
Compute	Nova	Manages and provisions VMs running on compute nodes.
Image	Glance	Registry service used to store resources such as VM images and volume snapshots.
Object storage	Swift	Allows users to storage and retrieve files and arbitrary data.

Telemetry	Ceilometer	Provides measurements of use of cloud resources.
Orchestration	Heat	Template-based orchestration engine that supports automatic creation of resource stacks.

Network design

The Red Hat OpenShift with NetApp solution uses two data switches to provide primary data connectivity at 25Gbps. It also uses two additional management switches that provide connectivity at 1Gbps for in-band management for the storage nodes and out-of-band management for IPMI functionality.

IPMI functionality is required by Red Hat OpenStack Director to deploy Red Hat OpenStack Platform using the Ironic bare-metal provision service.

VLAN requirements

Red Hat OpenShift with NetApp is designed to logically separate network traffic for different purposes by using virtual local area networks (VLANs). This configuration can be scaled to meet customer demands or to provide further isolation for specific network services. The following table lists the VLANs that are required to implement the solution while validating the solution at NetApp.

VLANs	Purpose	VLAN ID
Out-of-band management network	Network used for management of physical nodes and IPMI service for Ironic.	16
Storage infrastructure	Network used for controller nodes to map volumes directly to support infrastructure services like Swift.	201
Storage Cinder	Network used to map and attach block volumes directly to virtual instances deployed in the environment.	202
Internal API	Network used for communication between the OpenStack services using API communication, RPC messages, and database communication.	301
Tenant	Neutron provides each tenant with their own networks via tunneling through VXLAN. Network traffic is isolated within each tenant network. Each tenant network has an IP subnet associated with it, and network namespaces mean that multiple tenant networks can use the same address range without causing conflicts.	302

VLANs	Purpose	VLAN ID
Storage management	OpenStack Object Storage (Swift) uses this network to synchronize data objects between participating replica nodes. The proxy service acts as the intermediary interface between user requests and the underlying storage layer. The proxy receives incoming requests and locates the necessary replica to retrieve the requested data.	303
PXE	The OpenStack Director provides PXE boot as a part of the Ironic bare metal provisioning service to orchestrate the installation of the OSP Overcloud.	3484
External	Publicly available network which hosts the OpenStack Dashboard (Horizon) for graphical management and allows for public API calls to manage OpenStack services.	3485
In-band management network	Provides access for system administration functions such as SSH access, DNS traffic, and Network Time Protocol (NTP) traffic. This network also acts as a gateway for non-controller nodes.	3486

Network infrastructure support resources

The following infrastructure should be in place prior to the deployment of the OpenShift Container Platform:

- At least one DNS server which provides a full host-name resolution.
- At least three NTP servers which can keep time synchronized for the servers in the solution.
- (Optional) Outbound internet connectivity for the OpenShift environment.

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 OpenShift to an OSP private cloud with at least three compute nodes

The verified architecture described in this document presents the minimum hardware deployment suitable for HA operations by deploying three OSP controller nodes and two OSP compute nodes. This architecture ensures a fault tolerant configuration in which both compute nodes can launch virtual instances and deployed VMs can migrate between the two hypervisors.

Because Red Hat OpenShift initially deploys with three master nodes, a two-node configuration might cause at least two masters to occupy the same node, which can lead to a possible outage for OpenShift if that specific

node becomes unavailable. Therefore, it is a Red Hat best practice to deploy at least three OSP compute nodes so that the OpenShift masters can be distributed evenly and the solution receives an added degree of fault tolerance.

Configure virtual machine/host affinity

Distributing the OpenShift masters across multiple hypervisor nodes can be achieved by enabling VM/host affinity.

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. In the Red Hat OpenStack Platform, host affinity and anti-affinity rules can be created and enforced by creating server groups and configuring filters so that instances deployed by Nova in a server group deploy on different compute nodes.

A server group has a default maximum of 10 virtual instances that it can manage placement for. This can be modified by updating the default quotas for Nova.



There is a specific hard affinity/anti-affinity limit for OSP server groups; if there are not enough resources to deploy on separate nodes or not enough resources to allow sharing of nodes, the VM fails to boot.

To configure affinity groups, see [How do I configure Affinity and Anti-Affinity for OpenStack instances?](#).

Use a custom install file for OpenShift deployment

IPI makes the deployment of OpenShift clusters easy through the interactive wizard discussed earlier in this document. However, it is possible that you might need to change some default values as a part of a cluster deployment.

In these instances, you can run and task the wizard without immediately deploying a cluster; instead it creates a configuration file from which the cluster can be deployed later. This is very useful if you need to change any IPI defaults, or if you want to deploy multiple identical clusters in your environment for other uses such as multitenancy. For more information about creating a customized install configuration for OpenShift, see [Red Hat OpenShift Installing a Cluster on OpenStack with Customizations](#).

Next: [NetApp Storage Overview](#).

OpenShift on Red Hat Virtualization

Red Hat Virtualization (RHV) is an enterprise virtual data center platform that runs on Red Hat Enterprise Linux (RHEL) and uses the KVM hypervisor.

For more information about RHV, see the [Red Hat Virtualization website](#).

RHV provides the following features:

- **Centralized management of VMs and hosts.** The RHV manager runs as a physical or virtual machine (VM) in the deployment and provides a web-based GUI for the management of the solution from a central interface.
- **Self-hosted engine.** To minimize hardware requirements, RHV allows RHV Manager (RHV-M) to be deployed as a VM on the same hosts that run guest VMs.

- **High availability.** To avoid disruption in event of host failures, RHV allows VMs to be configured for high availability. The highly available VMs are controlled at the cluster level using resiliency policies.
- **High scalability.** A single RHV cluster can have up to 200 hypervisor hosts enabling it to support requirements of massive VMs to host resource-greedy, enterprise-class workloads.
- **Enhanced security.** Inherited from RHV, Secure Virtualization (sVirt) and Security Enhanced Linux (SELinux) technologies are employed by RHV for the purposes of elevated security and hardening for the hosts and VMs. The key advantage from these features is logical isolation of a VM and its associated resources.



Network design

The Red Hat OpenShift on NetApp solution uses two data switches to provide primary data connectivity at 25Gbps. It also uses two additional management switches that provide connectivity at 1Gbps for in-band management of the storage nodes and out-of-band management for IPMI functionality. OCP uses the virtual machine logical network on RHV for cluster management. This section describes the arrangement and purpose of each virtual network segment used in the solution and outlines the prerequisites for deploying the solution.

VLAN requirements

Red Hat OpenShift on RHV is designed to logically separate network traffic for different purposes by using virtual local area networks (VLANs). This configuration can be scaled to meet customer demands or to provide further isolation for specific network services. The following table lists the VLANs that are required to implement the solution while validating the solution at NetApp.

VLANs	Purpose	VLAN ID
Out-of-band management network	Management for physical nodes and IPMI	16
VM Network	Virtual guest network access	1172
In-band management network	Management for RHV-H nodes, RHV-Manager, and ovirtmgmt network	3343
Storage network	Storage network for NetApp Element iSCSI	3344

VLANs	Purpose	VLAN ID
Migration network	Network for virtual guest migration	3345

Network infrastructure support resources

The following infrastructure should be in place prior to the deployment of the OpenShift Container Platform:

- 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.
- (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 OpenShift to an RHV cluster of at least three nodes

The verified architecture described in this document presents the minimum hardware deployment suitable for HA operations by deploying two RHV-H hypervisor nodes and ensuring a fault tolerant configuration where both hosts can manage the hosted-engine and deployed VMs can migrate between the two hypervisors.

Because Red Hat OpenShift initially deploys with three master nodes, it is ensured in a two-node configuration that at least two masters will occupy the same node, which can lead to a possible outage for OpenShift if that specific node becomes unavailable. Therefore, it is a Red Hat best practice that at least three RHV-H hypervisor nodes be deployed as part of the solution so that the OpenShift masters can be distributed evenly and the solution receives an added degree of fault tolerance.

Configure virtual machine/host affinity

You can distribute the OpenShift masters across multiple hypervisor nodes by enabling VM/host affinity.

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.

The conditions defined for the parameters can be either hard enforcement or soft enforcement. Hard enforcement ensures that the VMs in an affinity group always follows the positive or negative affinity strictly without any regards to external conditions. Soft enforcement ensures that a higher preference is set for the VMs in an affinity group to follow the positive or negative affinity whenever feasible. In the two or three hypervisor configuration described in this document, soft affinity is the recommended setting. In larger clusters, hard affinity can correctly distribute OpenShift nodes.

To configure affinity groups, see the [Red Hat 6.11. Affinity Groups documentation](#).

Use a custom install file for OpenShift deployment

IPI makes the deployment of OpenShift clusters easy through the interactive wizard discussed earlier in this document. However, it is possible that there are some default values that might need to be changed as a part of cluster deployment.

In these instances, you can run and task the wizard without immediately deploying a cluster. Rather, a configuration file is created from which the cluster can be deployed later. This is very useful if you want to change any IPI defaults or if you want to deploy multiple identical clusters in your environment for other uses such as multitenancy. For more information about creating a customized install configuration for OpenShift, see [Red Hat OpenShift Installing a Cluster on RHV with Customizations](#).

Next: NetApp storage overview.

OpenShift on 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 nondisruptive 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.



Network design

The Red Hat OpenShift on NetApp solution uses two data switches to provide primary data connectivity at 25Gbps. It also uses two additional management switches that provide connectivity at 1Gbps for in-band management for the storage nodes and out-of-band management for IPMI functionality. OCP uses the VM logical network on VMware vSphere for its cluster management. This section describes the arrangement and purpose of each virtual network segment used in the solution and outlines the prerequisites for deployment of the solution.

VLAN requirements

Red Hat OpenShift on VMware vSphere is designed to logically separate network traffic for different purposes by using virtual local area networks (VLANs). This configuration can be scaled to meet customer demands or to provide further isolation for specific network services. The following table lists the VLANs that are required to implement the solution while validating the solution at NetApp.

VLANs	Purpose	VLAN ID
Out-of-band management network	Management for physical nodes and IPMI	16
VM Network	Virtual guest network access	181
Storage network	Storage network for ONTAP NFS	184
Storage network	Storage network for ONTAP iSCSI	185

VLANs	Purpose	VLAN ID
In-band management network	Management for ESXi Nodes, VCenter Server, ONTAP Select	3480
Storage network	Storage network for NetApp Element iSCSI	3481
Migration network	Network for virtual guest migration	3482

Network infrastructure support resources

The following infrastructure should be in place prior to the deployment of the OpenShift Container Platform:

- 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.
- (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 OpenShift to an ESXi cluster of at least three nodes

The verified architecture described in this document presents the minimum hardware deployment suitable for HA operations by deploying two ESXi hypervisor nodes and ensuring a fault tolerant configuration by enabling VMware vSphere HA and VMware vMotion. This configuration allows deployed VMs to migrate between the two hypervisors and reboot should one host become unavailable.

Because Red Hat OpenShift initially deploys with three master nodes, at least two masters in a two-node configuration can occupy the same node under some circumstances, which can lead to a possible outage for OpenShift if that specific node becomes unavailable. Therefore, it is a Red Hat best practice that at least three ESXi hypervisor nodes must be deployed so that the OpenShift masters can be distributed evenly, which provides an added degree of fault tolerance.

Configure virtual machine and host affinity

Ensuring the distribution of the OpenShift masters 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 [vSphere 6.7 Documentation: Using DRS Affinity Rules](#).

Use a custom install file for OpenShift deployment

IPI makes the deployment of OpenShift clusters easy through the interactive wizard discussed earlier in this document. However, it is possible that you might need to change some default values as a part of a cluster

deployment.

In these instances, you can run and task the wizard without immediately deploying a cluster, but instead the wizard creates a configuration file from which the cluster can be deployed later. This is very useful if you need to change any IPI defaults, or if you want to deploy multiple identical clusters in your environment for other uses such as multitenancy. For more information about creating a customized install configuration for OpenShift, see [Red Hat OpenShift Installing a Cluster on vSphere with Customizations](#).

Next: [NetApp Storage 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 Red Hat OpenShift.



- 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.

The following pages have additional information about the NetApp storage systems validated in the Red Hat OpenShift with NetApp solution:

- [NetApp ONTAP](#)
- [NetApp Element](#)

Next: [NetApp Storage Integrations Overview](#)

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, non-disruptive 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 non-rewritable 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 multi-protocol 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 enterprise-class speed, efficiency, and security your data fabric needs.

For more information about NETAPP AFF/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 Storage Integrations Overview](#)

NetApp Element: Red Hat OpenShift with NetApp

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 non-disruptive 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.

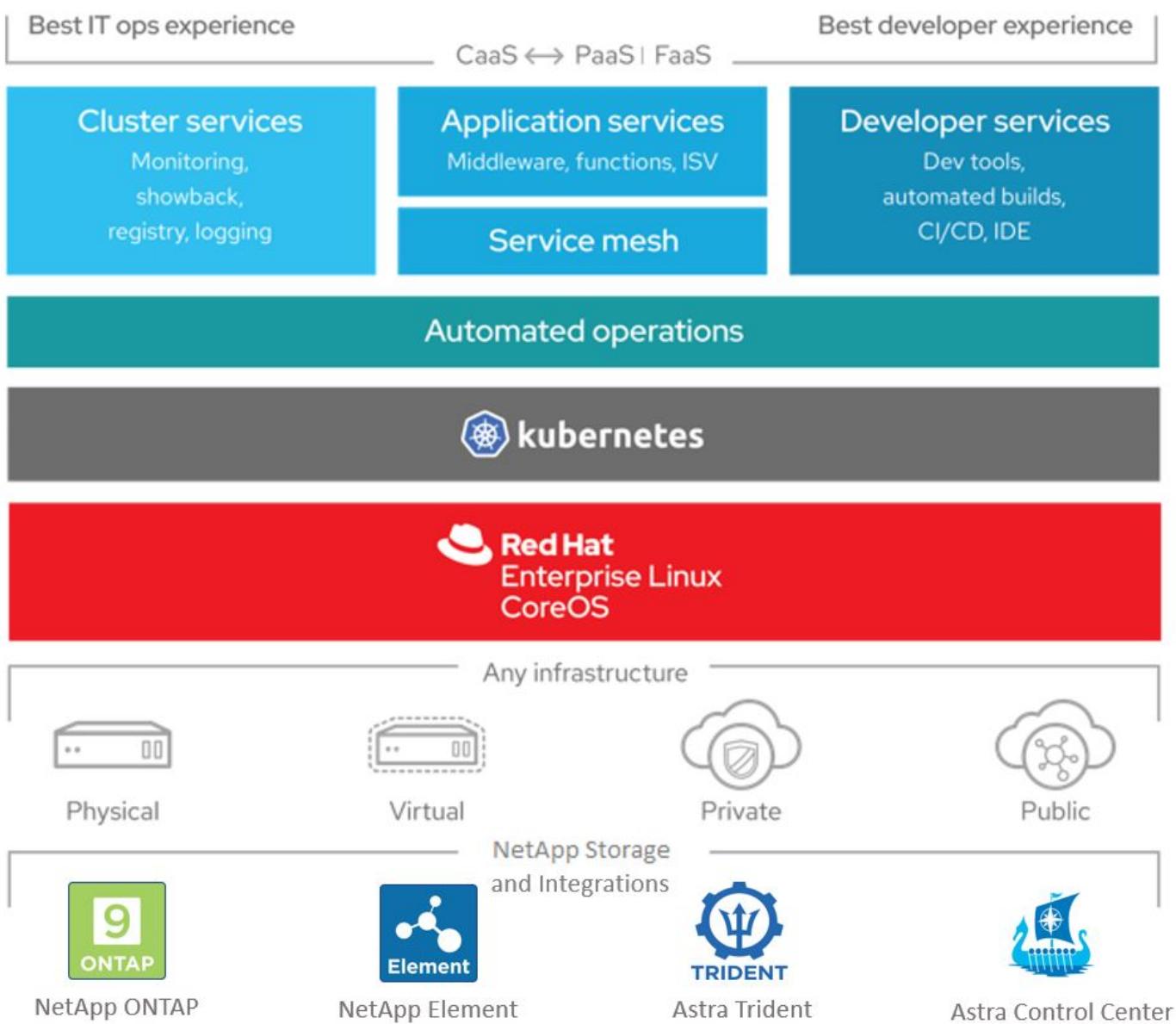


Element was designed for automation. All the storage features are available through APIs. These APIs are the only method that the UI uses to control the system.

Next: NetApp Storage Integrations Overview.

NetApp Storage Integration Overview

NetApp provides a number of products to help you with orchestrating and managing persistent data in container based environments, such as Red Hat OpenShift.



NetApp Astra Control offers a rich set of storage and application-aware data management services for stateful Kubernetes workloads, powered by NetApp data protection technology. The Astra Control Service is available to support stateful workloads in cloud-native Kubernetes deployments. The Astra Control Center is available to support stateful workloads in on-premises deployments, like Red Hat OpenShift. For more information visit the NetApp Astra Control website [here](#).

NetApp Astra Trident is an open-source and fully-supported storage orchestrator for containers and Kubernetes distributions, including Red Hat OpenShift. 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 Red Hat OpenShift with NetApp solution:

- [NetApp Astra Control Center](#)
- [NetApp Astra Trident](#)

Next: [NetApp Astra Control Center Overview](#)

NetApp Astra Control Center overview

NetApp Astra Control Center offers a rich set of storage and application-aware data management services for stateful Kubernetes workloads deployed in an on-premises environment and powered by NetApp data protection technology.



NetApp Astra Control Center can be installed on a Red Hat OpenShift cluster that has the Astra Trident storage orchestrator deployed and configured with storage classes and storage backends to NetApp ONTAP storage systems.

For the installation and configuration of Astra Trident to support Astra Control Center, see [this document here](#).

In a cloud-connected environment, Astra Control Center uses Cloud Insights to provide advanced monitoring and telemetry. In the absence of a Cloud Insights connection, limited monitoring and telemetry (7-days worth of metrics) is available and exported to Kubernetes native monitoring tools (Prometheus and Grafana) through open metrics endpoints.

Astra Control Center is fully integrated into the NetApp AutoSupport and Active IQ ecosystem to provide support for users, provide assistance with troubleshooting, and display usage statistics.

In addition to the paid version of Astra Control Center, a 90-day evaluation license is available. The evaluation version is supported through the email and community (Slack channel). Customers have access to these and other knowledge-base articles and the documentation available from the in-product support dashboard.

To get started with NetApp Astra Control Center, visit the [Astra website](#).

Astra Control Center installation prerequisites

1. One or more Red Hat OpenShift clusters. Versions 4.6 EUS and 4.7 are currently supported.
2. Astra Trident must already be installed and configured on each Red Hat OpenShift cluster.
3. One or more NetApp ONTAP storage systems running ONTAP 9.5 or greater.

 It's best practice for each OpenShift install at a site to have a dedicated SVM for persistent storage. Multi-site deployments require additional storage systems.
4. A Trident storage backend must be configured on each OpenShift cluster with an SVM backed by an ONTAP cluster.
5. A default StorageClass configured on each OpenShift cluster with Astra Trident as the storage provisioner.
6. A load balancer must be installed and configured on each OpenShift cluster for load balancing and exposing OpenShift Services.

 See the link [here](#) for information about load balancers that have been validated for this purpose.
7. A private image registry must be configured to host the NetApp Astra Control Center images.

 See the link [here](#) to install and configure an OpenShift private registry for this purpose.
8. You must have Cluster Admin access to the Red Hat OpenShift cluster.
9. You must have Admin access to NetApp ONTAP clusters.
10. An admin workstation with docker or podman, tridentctl, and oc or kubectl tools installed and added to your \$PATH.

 Docker installations must have docker version greater than 20.10 and Podman installations must have podman version greater than 3.0.

Install Astra Control Center

Using OperatorHub

1. Log into the NetApp Support Site and download the latest version of NetApp Astra Control Center. To do so requires a license attached to your NetApp account. After you download the tarball, transfer it to the admin workstation.



To get started with a trial license for Astra Control, visit the [Astra registration site](#).

2. Unpack the tar ball and change the working directory to the resulting folder.

```
[netapp-user@rhel7 ~]$ tar -vxzf astra-control-center-  
21.12.60.tar.gz  
[netapp-user@rhel7 ~]$ cd astra-control-center-21.12.60
```

3. Before starting the installation, push the Astra Control Center images to an image registry. You can choose to do this with either Docker or Podman, instructions for both are provided in this step.

Podman

- a. Export the registry FQDN with the organization/namespace/project name as a environment variable 'registry'.

```
[netapp-user@rhel7 ~]$ export REGISTRY=astra-  
registry.apps.ocp-vmw.cie.netapp.com/netapp-astra
```

- b. Log into the registry.

```
[netapp-user@rhel7 ~]$ podman login -u ocp-user -p password  
--tls-verify=false astra-registry.apps.ocp-vmw.cie.netapp.com
```

If you are using kubeadm user to log into the private registry, then use token instead of password - podman login -u ocp-user -p token --tls-verify=false astra-registry.apps.ocp-vmw.cie.netapp.com.

Alternatively, you can create a service account, assign registry-editor and/or registry-viewer role (based on whether you require push/pull access) and log into the registry using service account's token.

- c. Create a shell script file and paste the following content in it.

```
[netapp-user@rhel7 ~]$ vi push-images-to-registry.sh

for astraImageFile in $(ls images/*.tar) ; do
    # Load to local cache. And store the name of the loaded
    image trimming the 'Loaded images: '
    astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image(s): //')
    astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
    # Tag with local image repo.
    podman tag ${astraImage} ${REGISTRY}/${astraImage}
    # Push to the local repo.
    podman push ${REGISTRY}/${astraImage}
done
```

If you are using untrusted certificates for your registry, edit the shell script and use `--tls-verify=false` for the podman push command `podman push $REGISTRY/$(echo $astraImage | sed 's/[\\/]\\+\\/\\/\\/')` `--tls-verify=false`.

- d. Make the file executable.

```
[netapp-user@rhel7 ~]$ chmod +x push-images-to-registry.sh
```

- e. Execute the shell script.

```
[netapp-user@rhel7 ~]$ ./push-images-to-registry.sh
```

Docker

- Export the registry FQDN with the organization/namespace/project name as a environment variable 'registry'.

```
[netapp-user@rhel7 ~]$ export REGISTRY=astra-  
registry.apps.ocp-vmw.cie.netapp.com/netapp-astra
```

- Log into the registry.

```
[netapp-user@rhel7 ~]$ docker login -u ocp-user -p password  
astra-registry.apps.ocp-vmw.cie.netapp.com
```



If you are using kubeadmin user to log into the private registry, then use token instead of password - docker login -u ocp-user -p token astra-registry.apps.ocp-vmw.cie.netapp.com.



Alternatively, you can create a service account, assign registry-editor and/or registry-viewer role (based on whether you require push/pull access) and log into the registry using service account's token.

- Create a shell script file and paste the following content in it.

```
[netapp-user@rhel7 ~]$ vi push-images-to-registry.sh  
  
for astraImageFile in $(ls images/*.tar) ; do  
    # Load to local cache. And store the name of the loaded  
    # image trimming the 'Loaded images: '  
    astraImage=$(docker load --input ${astraImageFile} | sed  
's/Loaded image: //')  
    astraImage=$(echo ${astraImage} | sed 's!localhost/!!')  
    # Tag with local image repo.  
    docker tag ${astraImage} ${REGISTRY}/${astraImage}  
    # Push to the local repo.  
    docker push ${REGISTRY}/${astraImage}  
done
```

- Make the file executable.

```
[netapp-user@rhel7 ~]$ chmod +x push-images-to-registry.sh
```

- Execute the shell script.

```
[netapp-user@rhel7 ~]$ ./push-images-to-registry.sh
```

- When using private image registries that are not publicly trusted, upload the image registry TLS certificates to the OpenShift nodes. To do so, create a configmap in the openshift-config namespace using the TLS certificates and patch it to the cluster image config to make the certificate trusted.

```
[netapp-user@rhel7 ~]$ oc create configmap default-ingress-ca -n openshift-config --from-file=astra-registry.apps.ocp -vmw.cie.netapp.com=tls.crt
```

```
[netapp-user@rhel7 ~]$ oc patch image.config.openshift.io/cluster --patch '{"spec":{"additionalTrustedCA":{"name":"default-ingress-ca"}}}' --type=merge
```



If you are using an OpenShift internal registry with default TLS certificates from the ingress operator with a route, you still need to follow the previous step to patch the certificates to the route hostname. To extract the certificates from ingress operator, you can use the command `oc extract secret/router-ca --keys=tls.crt -n openshift-ingress-operator`.

- Create a namespace `netapp-acc-operator` for Astra Control Center.

```
[netapp-user@rhel7 ~]$ oc create ns netapp-acc-operator  
namespace/netapp-acc-operator created
```

- Create a secret with credentials to log into the image registry in `netapp-acc-operator` namespace.

```
[netapp-user@rhel7 ~]$ oc create secret docker-registry astra-registry-cred --docker-server=astra-registry.apps.ocp -vmw.cie.netapp.com --docker-username=ocp-user --docker-password=password -n netapp-acc-operator  
secret/astra-registry-cred created
```

- Log into the Red Hat OpenShift GUI console with cluster-admin access.
- Select Administrator from the Perspective drop down.
- Navigate to Operators > OperatorHub and search for Astra.



10. Select netapp-acc-operator tile and click Install.

The screenshot shows a software package named "netapp-acc-operator". It includes a pink circular icon with a white starburst, the package name, its version (21.12.63-1), and the provider (NetApp). A large blue "Install" button is prominent. Below the button, the "Latest version" is listed as 21.12.63-1 with a description of Astra Control's functionality. The "Capability level" section lists "Basic Install" (checked) and other options like "Seamless Upgrades", "Full Lifecycle", "Deep Insights", and "Auto Pilot". The "How to deploy Astra Control" section provides a link to the "Installation Procedure". The "Provider type" is listed as "Certified", and the "Documentation" section links to the "Astra Control Center Documentation".

netapp-acc-operator

21.12.63-1 provided by NetApp

Install

Latest version
21.12.63-1
Astra Control is an application-aware data management solution that manages, protects and moves data-rich Kubernetes workloads in both public clouds and on-premises.

Capability level

- Basic Install
- Seamless Upgrades
- Full Lifecycle
- Deep Insights
- Auto Pilot

How to deploy Astra Control

Refer to [Installation Procedure](#) to deploy Astra Control Center using the Operator.

Provider type
Certified

Documentation

Refer to [Astra Control Center Documentation](#) to complete the setup and start managing applications.

Provider
NetApp

11. On the Install Operator screen, accept all default parameters and click Install.

Install Operator

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic updates.

Update channel *

- alpha
- stable



netapp-acc-operator
provided by NetApp

Provided APIs

Installation mode *

- All namespaces on the cluster (default)
Operator will be available in all Namespaces.
- A specific namespace on the cluster
This mode is not supported by this Operator



AstraControlCenter is the Schema for the astracontrolcenters API

Installed Namespace *

netapp-acc-operator (Operator recommended)

⚠ Namespace already exists

Namespace **netapp-acc-operator** already exists and will be used. Other users can already have access to this namespace.

Approval strategy *

- Automatic
- Manual

Install

Cancel

12. Wait for the operator installation to complete.

netapp-acc-operator
 21.12.63-1 provided by NetApp

Installing Operator

InstallWaiting: installing; waiting for deployment acc-operator-controller-manager to become ready: Waiting for rollout to finish: 0 of 1 updated replicas are available...

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

[View installed Operators in Namespace netapp-acc-operator](#)

13. Once the operator installation succeeds, navigate to click on View Operator.



14. Then click on Create Instance in Astra Control Center tile in the operator.

A screenshot of the 'netapp-acc-operator' details page. The top navigation shows 'Installed Operators > Operator details'. Below the header, it displays the operator's name, version, and provider. A horizontal navigation bar at the top includes 'Details' (underlined), 'YAML', 'Subscription', 'Events', and 'Astra Control Center'. The main section is titled 'Provided APIs'. It features a 'ACC Astra Control Center' button with a blue background and white text. Below it, a description states 'AstraControlCenter is the Schema for the astracontrolcenters API'. At the bottom of the section is a 'Create instance' button with a plus sign icon.

15. Fill the Create AstraControlCenter form fields and click Create.

- Optionally edit the Astra Control Center instance name.
- Optionally enable or disable Auto Support. Retaining Auto Support functionality is recommended.
- Enter the FQDN for Astra Control Center.
- Enter the Astra Control Center version; the latest is displayed by default.
- Enter an account name for Astra Control Center and admin details like first name, last name and

email address.

- f. Enter the volume reclaim policy, default is Retain.
- g. In Image Registry, enter the FQDN for your registry along with the organization name as it was given while pushing the images to the registry (in this example, astra-registry.apps.ocp-vmw.cie.netapp.com/netapp-astra)
- h. If you use a registry that requires authentication, enter the secret name in Image Registry section.
- i. Configure scaling options for Astra Control Center resource limits.
- j. Enter the storage class name if you want to place PVCs on a non-default storage class.
- k. Define CRD handling preferences.

Project: netapp-acc-operator ▾

Name *
astra

Labels
app=frontend

Account Name *
HCG Solutions Engineering

Astra Control Center account name

Astra Address *
astra-control-center.cie.netapp.com

AstraAddress defines how Astra will be found in the data center. This IP address and/or DNS A record must be created prior to provisioning Astra Control Center. Example - "astra.example.com" The A record and its IP address must be allocated prior to provisioning Astra Control Center

Astra Version *
21.12.60

Version of AstraControlCenter to deploy. You are provided a Helm repository with a corresponding version. Example - 1.5.2, 1.4.2-patch

Email *
solutions_tme@netapp.com

EmailAddress will be notified by Astra as events warrant.

Auto Support * ➤

AutoSupport indicates willingness to participate in NetApp's proactive support application, NetApp Active IQ. The default election is true and indicates support data will be sent to NetApp. An empty or blank election is the same as a default election. Air gapped installations should enter false.

First Name
HCG

The first name of the SRE supporting Astra.

Last Name

The last name of the SRE supporting Astra.

Image Registry

The container image registry that is hosting the Astra application images, ACC Operator and ACC Helm Repository.

Name

The name of the image registry. For example "example.registry/astra". Do not prefix with protocol.

Secret

The name of the Kubernetes secret that will authenticate with the image registry.

Volume Reclaim Policy

Reclaim policy to be set for persistent volumes

Astra Resources Scaler

Default ▾

Scaling options for AstraControlCenter Resource limits.

Storage Class

The storage class to be used for PVCs. If not set, default storage class will be used.

Crd's

Options for how ACC should handle CRDs.

Create **Cancel**

Automated [Ansible]

1. To use Ansible playbooks to deploy Astra Control Center, you need an Ubuntu/RHEL machine with Ansible installed. Follow the procedure [here](#) for Ubuntu and the procedure [here](#) for RHEL.
2. Clone the GitHub repository that hosts the Ansible content.

```
git clone https://github.com/NetApp-
Automation/na_astra_control_suite.git
```

3. Log into the NetApp Support site and download the latest version of NetApp Astra Control Center. To do so requires a license attached to your NetApp account. After you download the tarball, transfer it to the workstation.



To get started with a trial license for Astra Control, visit the [Astra registration site](#).

4. Create or obtain the kubeconfig file with admin access to the OpenShift cluster on which Astra Control Center is to be installed.
5. Change the directory to the na_astra_control_suite.

```
cd na_astra_control_suite
```

6. Edit the vars/vars.yml file, and fill in the variables with the required information.

```
#Define whether or not to push the Astra Control Center images to  
your private registry [Allowed values: yes, no]  
push_images: yes  
  
#The directory hosting the Astra Control Center installer  
installer_directory: /home/admin/  
  
#Specify the ingress type. Allowed values - "AccTraefik" or  
"Generic"  
#"AccTraefik" if you want the installer to create a LoadBalancer  
type service to access ACC, requires MetallB or similar.  
#"Generic" if you want to create or configure ingress controller  
yourself, installer just creates a ClusterIP service for traefik.  
ingress_type: "AccTraefik"  
  
#Name of the Astra Control Center installer (Do not include the  
extension, just the name)  
astra_tar_ball_name: astra-control-center-22.04.0  
  
#The complete path to the kubeconfig file of the  
kubernetes/openshift cluster Astra Control Center needs to be  
installed to.  
hosting_k8s_cluster_kubeconfig_path: /home/admin/cluster-  
kubeconfig.yml  
  
#Namespace in which Astra Control Center is to be installed  
astra_namespace: netapp-astra-cc  
  
#Astra Control Center Resources Scaler. Leave it blank if you want  
to accept the Default setting.  
astra_resources_scaler: Default  
  
#Storageclass to be used for Astra Control Center PVCs, it must be  
created before running the playbook [Leave it blank if you want the  
PVCs to use default storageclass]  
astra_trident_storageclass: basic  
  
#Reclaim Policy for Astra Control Center Persistent Volumes [Allowed  
values: Retain, Delete]  
storageclass_reclaim_policy: Retain
```

```

#Private Registry Details
astra_registry_name: "docker.io"

#Whether the private registry requires credentials [Allowed values:
yes, no]
require_reg_creds: yes

#If require_reg_creds is yes, then define the container image
registry credentials
#Usually, the registry namespace and usernames are same for
individual users
astra_registry_namespace: "registry-user"
astra_registry_username: "registry-user"
astra_registry_password: "password"

#Kubernetes/OpenShift secret name for Astra Control Center
#This name will be assigned to the K8s secret created by the
playbook
astra_registry_secret_name: "astra-registry-credentials"

#Astra Control Center FQDN
acc_fqdn_address: astra-control-center.cie.netapp.com

#Name of the Astra Control Center instance
acc_account_name: ACC Account Name

#Administrator details for Astra Control Center
admin_email_address: admin@example.com
admin_first_name: Admin
admin_last_name: Admin

```

7. Run the playbook to deploy Astra Control Center. The playbook requires root privileges for certain configurations.

If the user running the playbook is root or has passwordless sudo configured, then run the following command to run the playbook.

```
ansible-playbook install_acc_playbook.yml
```

If the user has password-based sudo access configured, run the following command to run the playbook, and then enter the sudo password.

```
ansible-playbook install_acc_playbook.yml -K
```

Post Install Steps

1. It might take several minutes for the installation to complete. Verify that all the pods and services in the netapp-astra-cc namespace are up and running.

```
[netapp-user@rhel7 ~]$ oc get all -n netapp-astra-cc
```

2. Check the acc-operator-controller-manager logs to ensure that the installation is completed.

```
[netapp-user@rhel7 ~]$ oc logs deploy/acc-operator-controller-manager -n netapp-acc-operator -c manager -f
```



The following message indicates the successful installation of Astra Control Center.

```
{"level": "info", "ts": 1624054318.029971, "logger": "controllers.AstraControlCenter", "msg": "Successfully Reconciled AstraControlCenter in [seconds]s", "AstraControlCenter": "netapp-astra-cc/astra", "ae.Version": "[21.12.60]"}
```

3. The username for logging into Astra Control Center is the email address of the administrator provided in the CRD file and the password is a string ACC- appended to the Astra Control Center UUID. Run the following command:

```
[netapp-user@rhel7 ~]$ oc get astracontrolcenters -n netapp-astra-cc  
NAME      UUID  
astra     345c55a5-bf2e-21f0-84b8-b6f2bce5e95f
```



In this example, the password is ACC-345c55a5-bf2e-21f0-84b8-b6f2bce5e95f.

4. Get the traefik service load balancer IP.

```
[netapp-user@rhel7 ~]$ oc get svc -n netapp-astra-cc | egrep 'EXTERNAL|traefik'  
  
NAME           TYPE        CLUSTER-IP  
EXTERNAL-IP    PORT(S)  
AGE  
traefik       LoadBalancer 172.30.99.142  
10.61.186.181 80:30343/TCP,443:30060/TCP  
16m
```

5. Add an entry in the DNS server pointing the FQDN provided in the Astra Control Center CRD file to the

EXTERNAL-IP of the traefik service.

New Host

Name (uses parent domain name if blank):
astra-control-center

Fully qualified domain name (FQDN):
astra-control-center.cie.netapp.com.

IP address:
10.61.186.181

Create associated pointer (PTR) record
 Allow any authenticated user to update DNS records with the same owner name

Add Host Cancel

6. Log into the Astra Control Center GUI by browsing its FQDN.

The screenshot shows the login interface for the NetApp Astra Control Center. At the top left is the NetApp logo. Below it, the text "Log In to NetApp Astra Control Center" is displayed. There are two input fields: "Email" and "Password". Below these fields is a blue "LOGIN" button. To the right of the login form is a large, dark blue banner with white text that reads: "Astra Control Center", "Manage, protect, and migrate your Kubernetes applications with just a few clicks!".

- When you log into Astra Control Center GUI for the first time using the admin email address provided in CRD, you need to change the password.

The screenshot shows the password update page. At the top left is the NetApp logo. Below it, the text "Welcome to NetApp Astra Control Center" is displayed. A message "Update your password to proceed" is shown above two input fields. Below the fields is a light blue callout box containing password requirements: "Passwords must contain:" followed by a bulleted list: "At least 8 characters", "No more than 64 characters", "At least one uppercase letter", "At least one lowercase letter", "At least one number", and "At least one special character". At the bottom is a blue "UPDATE PASSWORD" button. To the right of the update form is a large, dark blue banner with white text that reads: "Astra Control Center", "Manage, protect, and migrate your Kubernetes applications with just a few clicks!".

- If you wish to add a user to Astra Control Center, navigate to Account > Users, click Add, enter the details of the user, and click Add.

Add user

USER DETAILS

First name Nikhil	Last name Kulkarni
Email address tme_nik@netapp.com	

PASSWORD

Temporary password *****	Confirm temporary password *****
-----------------------------	-------------------------------------

Passwords must contain:

- At least 8 characters
- No more than 64 characters
- At least one lowercase letter
- At least one uppercase letter
- At least one number
- At least one special character

USER ROLE

Role Owner

Cancel **Add ✓**

9. Astra Control Center requires a license for all of its functionalities to work. To add a license, navigate to Account > License, click Add License, and upload the license file.



If you encounter issues with the install or configuration of NetApp Astra Control Center, the knowledge base of known issues is available [here](#).

Next: Register your Red Hat OpenShift Clusters: Red Hat OpenShift with NetApp.

Register your Red Hat OpenShift Clusters with the Astra Control Center

To enable the Astra Control Center to manage your workloads, you must first register your Red Hat OpenShift cluster.

Register Red Hat OpenShift clusters

1. The first step is to add the OpenShift clusters to the Astra Control Center and manage them. Go to Clusters and click Add a Cluster, upload the kubeconfig file for the OpenShift cluster, and click Select Storage.

The screenshot shows the 'Add cluster' wizard in progress, specifically Step 1/3: CREDENTIALS. On the left, there's a 'CREDENTIALS' section with a note about providing Astra Control access via a kubeconfig credential. Below it are two buttons: 'Upload file' (underlined) and 'Paste from clipboard'. An uploaded file named 'ocp-vmw kubeconfig.txt' is shown with options to upload or remove it. To its right is a 'Credential name' input field containing 'ocp-vmw'. On the right side of the screen, there's a sidebar titled 'ADDING A CLUSTER' with instructions: 'Adding a cluster is needed for Astra Control to discover your Kubernetes applications.', 'Select a cloud provider and input credentials to get started.', and a link 'Read more in Clusters'. At the bottom of the main panel are 'Cancel' and 'Configure storage →' buttons.



The kubeconfig file can be generated to authenticate with a username and password or a token. Tokens expire after a limited amount of time and might leave the registered cluster unreachable. NetApp recommends using a kubeconfig file with a username and password to register your OpenShift clusters to Astra Control Center.

2. Astra Control Center detects the eligible storage classes. Now select the way that storageclass provisions volumes using Trident backed by an SVM on NetApp ONTAP and click Review. In the next pane, verify the details and click Add Cluster.

Set default	Storage class	Storage provisioner	Reclaim policy	Binding mode	Eligible
<input checked="" type="radio"/>	ocp-trident <small>Default</small>	csi.trident.netapp.io	Delete	Immediate	✓
<input type="radio"/>	ocp-trident-iscsi	csi.trident.netapp.io	Delete	Immediate	✓
<input type="radio"/>	project-1-sc	csi.trident.netapp.io	Delete	Immediate	⚠
<input type="radio"/>	thin	kubernetes.io/vsphere-volume	Delete	Immediate	⚠

[← Select credentials](#) [Review →](#)

3. Register both OpenShift clusters as described in step 1. When added, the clusters move to the Discovering status while Astra Control Center inspects them and installs the necessary agents. Cluster status changes to Running after they are successfully registered.

Name	Ready	Type	Version	Actions
ocp-vmw	✓	Red Hat OpenShift	v1.20.0+df9c838	Running
ocp-vmware2	✓	Red Hat OpenShift	v1.20.0+c8905da	Running



All Red Hat OpenShift clusters to be managed by Astra Control Center should have access to the image registry that was used for its installation as the agents installed on the managed clusters pull the images from that registry.

4. Import ONTAP clusters as storage resources to be managed as backends by Astra Control Center. When OpenShift clusters are added to Astra and a storageclass is configured, it automatically discovers and inspects the ONTAP cluster backing the storageclass but does not import it into the Astra Control Center to be managed.

- To import the ONTAP clusters, go to Backends, click the dropdown, and select Manage next to the ONTAP cluster to be managed. Enter the ONTAP cluster credentials, click Review Information, and then click Import Storage Backend.

- After the backends are added, the status changes to Available. These backends now have the information about the persistent volumes in the OpenShift cluster and the corresponding volumes on the ONTAP system.

Name	Status	Capacity	Type	Actions
K8s-OnTap	✓	0.11/1.07 TiB: 9.9%	ONTAP 9.8.0	Available
ONTAP-Select-02	✓	0.07/2.07 TiB: 3.3%	ONTAP 9.8.0	Available

7. For backup and restore across OpenShift clusters using Astra Control Center, you must provision an object storage bucket that supports the S3 protocol. Currently supported options are ONTAP S3, StorageGRID, and AWS S3. For the purpose of this installation, we are going to configure an AWS S3 bucket. Go to Buckets, click Add bucket, and select Generic S3. Enter the details about the S3 bucket and credentials to access it, click the checkbox "Make this bucket the default bucket for the cloud," and then click Add.

Add bucket

STORAGE BUCKET

Enter the access details of your existing object store bucket to allow Astra Control to store your application backups.

Type <input checked="" type="checkbox"/> Generic S3	Existing bucket name ocp-vmware2-astra-cc
Description (optional)	S3 server name or IP address s3.us-east-1.amazonaws.com
<input checked="" type="checkbox"/> Make this bucket the default bucket for this cloud	

SELECT CREDENTIALS

Astra Control requires S3 access credentials with the roles necessary to facilitate Kubernetes application data management.

<input checked="" type="radio"/> Add	<input type="radio"/> Use existing
Access ID AMWSTCFKDSU6HWSZXABD	Secret key
Credential name AWS-S3	

ADDING STORAGE BUCKETS

Astra Control stores backups in your existing object store buckets. The first bucket added for a selected cloud will be designated as the default bucket for backup and clone operations.

Read more in [storage buckets](#).

Next: Choose the Applications To Protect.

Choose the applications to protect

After you have registered your Red Hat OpenShift clusters, you can discover the applications that are deployed and manage them via the Astra Control Center.

Manage applications

- After the OpenShift clusters and ONTAP backends are registered with the Astra Control Center, the control center automatically starts discovering the applications in all the namespaces that are using the storageclass configured with the specified ONTAP backend.



The screenshot shows the Astra Control Center interface. On the left, there's a sidebar with navigation links: Dashboard, MANAGE YOUR APPS (which is highlighted in blue), Clusters, MANAGE YOUR STORAGE (Backend and Buckets), and MANAGE YOUR ACCOUNT (Account, Activity, Support). The main content area is titled 'Apps' and shows a table of discovered applications. The columns are: Name, Ready, Cluster, Group, Discovered, and Actions. There are 29 entries listed. Some applications are marked as 'Managed' (indicated by a star icon) and others as 'Unmanaged'. One entry, 'local-cluster', has a status of 'Discovering'.

Name	Ready	Cluster	Group	Discovered	Actions
acc-operator-system	✓	ocp-vmware2	acc-operator-system	2021/07/29 11:11 UTC	Unmanaged
acc-operator-system	✓	ocp-vmw	acc-operator-system	2021/07/29 11:09 UTC	Unmanaged
default	✓	ocp-vmw	default	2021/07/29 11:09 UTC	Unmanaged
default	✓	ocp-vmware2	default	2021/07/29 11:11 UTC	Unmanaged
hive	✓	ocp-vmware2	hive	2021/07/29 11:11 UTC	Unmanaged
local-cluster	●	ocp-vmware2	local-cluster	2021/07/29 11:45 UTC	Discovering

2. Navigate to Apps > Discovered and click the dropdown menu next to the application you would like to manage using Astra. Then click Manage.



This screenshot is similar to the previous one, showing the 'Apps' section with a list of discovered applications. However, a context menu is open over the second row, which contains the application 'wordpress-astra-fd2aa'. The menu options shown are 'Manage' and 'Ignore'.

Name	Ready	Cluster	Group	Discovered	Actions
wordpress-astra-fd2aa	●	ocp-vmware2	wordpress-astra-fd2aa	2021/07/29 11:11 UTC	Manage Ignore
wordpress-astra-5eef9	✓	ocp-vmw	wordpress-astra-5eef9	2021/07/29 11:09 UTC	Unmanaged
wordpress-astra-5eef9	●	ocp-vmware2	wordpress-astra-5eef9	2021/07/29 11:11 UTC	Discovering
wordpress-astra-5ed9e	✓	ocp-vmw	wordpress-astra-5ed9e	2021/07/29 11:09 UTC	Unmanaged
wordpress-astra	✓	ocp-vmw	wordpress-astra	2021/07/29 11:09 UTC	Unmanaged
wordpress	●	ocp-vmw	wordpress	2021/07/29 11:09 UTC	Discovering

1. The application enters the Available state and can be viewed under the Managed tab in the Apps section.

The screenshot shows the 'Apps' section of the Astra Control Center. At the top, there are buttons for 'Actions', '+ Define', 'All Clusters' (with a dropdown), 'Search', 'Managed' (with a star icon), 'Discovered' (with a count of 175), and 'Ignored'. Below this is a table header with columns: Name, Ready, Protected, Cluster, Group, Discovered, and Actions. A single row is listed: 'wordpress-astra-ff4f9' (Ready, Protected, Cluster: ocp-vmw, Group: wordpress-astra-ff4f9, Discovered: 2021/07/29 11:09 UTC, Actions: Available). A status bar at the bottom indicates '1-1 of 1 entries'.

Next: Protect Your applications.

Protect your applications

After application workloads are managed by Astra Control Center, you can configure the protection settings for those workloads.

Creating an application snapshot

A snapshot of an application creates an ONTAP Snapshot copy that can be used to restore or clone the application to a specific point in time based on that Snapshot copy.

1. To take a snapshot of the application, navigate to the Apps > Managed tab and click the application you would like to make a Snapshot copy of. Click the dropdown menu next to the application name and click Snapshot.

The screenshot shows the details for the application 'wp'. On the left, under 'APPLICATION STATUS', it says 'Healthy'. On the right, under 'APPLICATION PROTECTION S1', it says 'Unprotected'. A dropdown menu is open on the right, listing options: Running (selected), Snapshot, Backup, Clone, Restore, and Unmanage. Below the status sections, there are details: Images (docker.io/bitnami/mariadb:10.5.13-debian-10-r58, docker.io/bitnami/wordpress:5.9.0-debian-10-r1), Protection schedule (Disabled), Group (wp), and Cluster (with a refresh icon).

2. Enter the snapshot details, click Next, and then click Snapshot. It takes about a minute to create the snapshot, and the status becomes Available after the snapshot is successfully created.



Creating an application backup

A backup of an application captures the active state of the application and the configuration of its resources, converts them into files, and stores them in a remote object storage bucket.

For the backup and restore of managed applications in the Astra Control Center, you must configure superuser settings for the backing ONTAP systems as a prerequisite. To do so, enter the following commands.

```
ONTAP::> export-policy rule modify -vserver ocp-trident -policyname
default -ruleindex 1 -superuser sys
ONTAP::> export-policy rule modify -policyname default -ruleindex 1 -anon
65534 -vserver ocp-trident
```

- To create a backup of the managed application in the Astra Control Center, navigate to the Apps > Managed tab and click the application that you want to take a backup of. Click the dropdown menu next to the application name and click Backup.

- Enter the backup details, select the object storage bucket to hold the backup files, click Next, and, after reviewing the details, click Backup. Depending on the size of the application and data, the backup can take several minutes, and the status of the backup becomes Available after the backup is completed successfully.

STEP 1/2: DETAILS

BACKUP DETAILS

Name: wp-backup

Backup from an existing snapshot

BACKUP DESTINATION

Bucket: na-ocp-astra/na-ocp-acc Available

CREATING APPLICATION BACKUPS

Astra Control can take a backup of your application configuration and persistent storage. Persistent storage backups are transferred to your object store. Enter a backup name to get started.

Read more in [Application backups](#).

(⌚) Application wp
 (📁) Namespace wp
 (📦) Cluster ocp-vmw

Cancel Next →

Restoring an application

At the push of a button, you can restore an application to the originating namespace in the same cluster or to a remote cluster for application protection and disaster recovery purposes.

1. To restore an application, navigate to Apps > Managed tab and click the app in question. Click the dropdown menu next to the application name and click **Restore**.

(⌚) wp

APPLICATION STATUS
Healthy

APPLICATION PROTECTION STATUS
Partially protected

Running

Snapshot
Backup
Clone
Restore
Unmanage

Images:
docker.io/bitnami/mariadb:10.5.13-debian-10-r58
docker.io/bitnami/wordpress:5.9.0-debian-10-r1

Protection schedule: Disabled

Group: wp

Cluster: !

2. Enter the name of the restore namespace, select the cluster you want to restore it to, and choose if you want to restore it from an existing snapshot or from a backup of the application. Click **Next**.

RESTORE DETAILS

Destination cluster: ocp-vmw

Destination namespace: wp

RESTORE SOURCE

Application backup	Ready	On-Schedule/On-Demand	Created
wp-backup	Ready	On-Demand	2022/02/28 18:54 UTC

RESTORING APPLICATIONS

Astra Control can restore your application configuration and persistent storage. Select a source snapshot or backup for the restored application.

Application: wp

Namespace: wp

Cluster: ocp-vmw

3. On the review pane, enter `restore` and click Restore after you have reviewed the details.

STEP 2/2: SUMMARY

REVIEW RESTORE INFORMATION

All existing resources associated with this application will be deleted and replaced with the source backup "wp-backup" taken on 2022/02/28 18:54 UTC. Persistent volumes will be deleted and recreated. External resources with dependencies on this application may be impacted.

We recommend taking a snapshot or a backup of your application before proceeding.

Backup	Restore
wp-backup	wp

Original Group	Destination Group
wp	wp

Original Cluster	Destination Cluster
ocp-vmw	ocp-vmw

Resource Labels	Resource Labels
ClusterRole kubernetes.io/bootstrapping: rbac-defaults +1 ClusterRoleBinding	ClusterRole kubernetes.io/bootstrapping: rbac-defaults +1 ClusterRoleBinding

Are you sure you want to restore the application "wp"?

Type `restore` below to confirm.

Confirm to restore
`restore`

Back **Restore ✓**

4. The new application goes to the Restoring state while Astra Control Center restores the application on the selected cluster. After all the resources of the application are installed and detected by Astra, the application goes to the Available state.

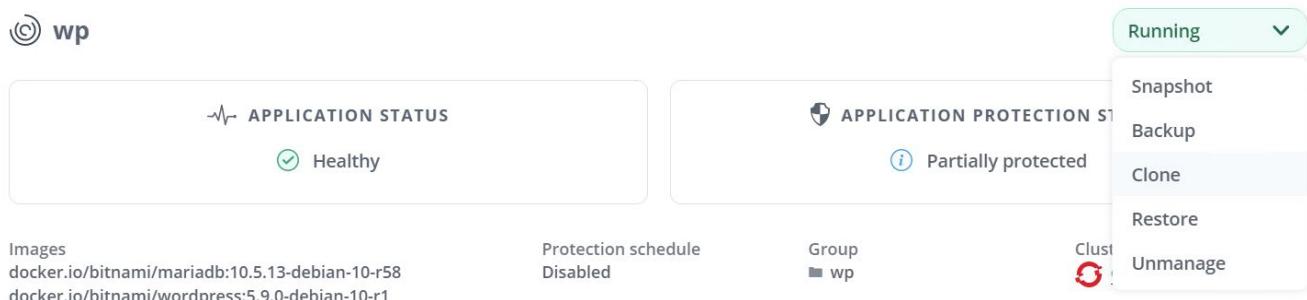
Applications

Actions		+ Define	Actions	Search	Star	Filter	110	Reset
			C	1-1 of 1 entries	<	>		
Name	Ready	Protected	Cluster	Group	Discovered	Actions		
wp	✓	ⓘ	ocp-vmw	wp	2022/02/28 18:34 UTC	Available	▼	

Cloning an application

You can clone an application to the originating cluster or to a remote cluster for dev/test or application protection and disaster recovery purposes. Cloning an application within the same cluster on the same storage backend uses NetApp FlexClone technology, which clones the PVCs instantly and saves storage space.

1. To clone an application, navigate to the Apps > Managed tab and click the app in question. Click the dropdown menu next to the application name and click Clone.



2. Enter the details of the new namespace, select the cluster you want to clone it to, and choose if you want to clone it from an existing snapshot or a backup or the current state of the application. Then click Next and click Clone on review pane once you have reviewed the details.

Clone application

STEP 1/2: DETAILS

Clone Details	Cloning Applications
Clone name wp-clone	Astra Control can create a clone of your application configuration and persistent storage. Persistent storage backups are transferred from your object store, so choosing a clone from an existing backup will complete the fastest. Enter a clone name to get started.
Destination cluster ocp-vmw	Read more in Clone applications .
<input type="checkbox"/> Clone from an existing snapshot or backup	

Cancel **Next →**

3. The new application goes to the Discovering state while Astra Control Center creates the application on the

selected cluster. After all the resources of the application are installed and detected by Astra, the application goes to the Available state.

⌚ Applications

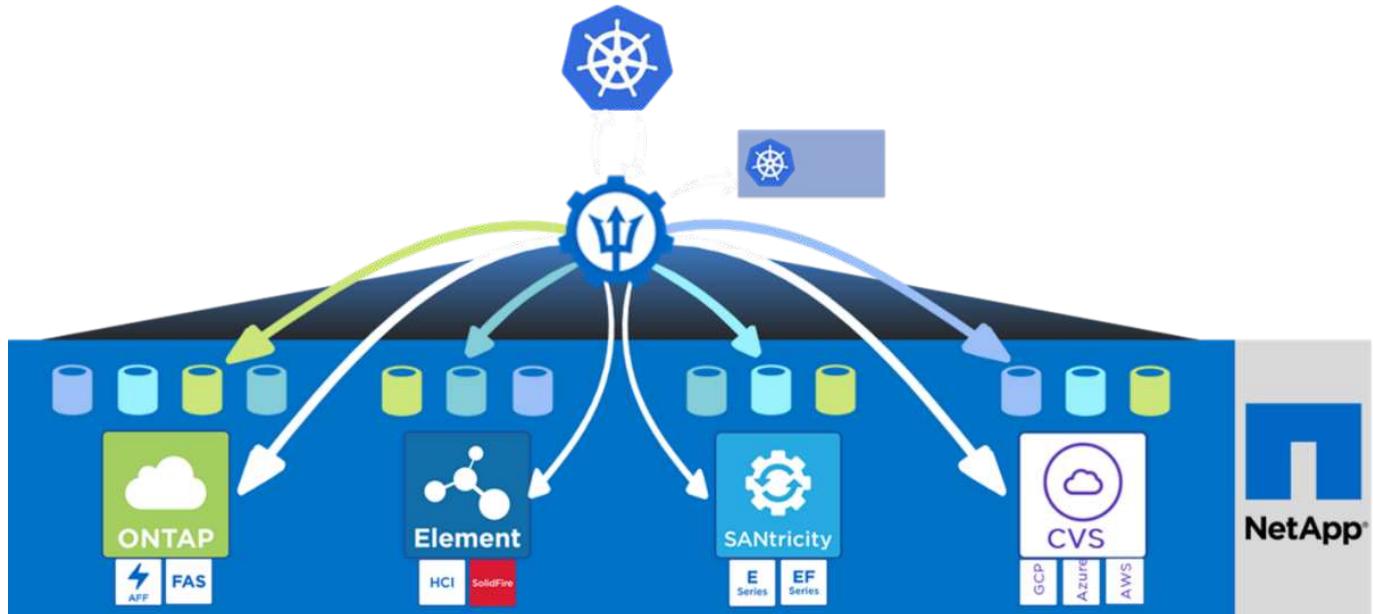
Applications									
	Name	Ready	Protected	Cluster	Group	Discovered			
<input type="checkbox"/>	wp				ocp-vmw		wp	2022/02/28 18:34 UTC	Available
<input type="checkbox"/>	wp-clone				ocp-vmw		wp-clone	2022/02/28 19:21 UTC	Available

Next: Solution Validation/Use Cases.

Astra Trident Overview

Astra Trident is an open-source and fully supported storage orchestrator for containers and Kubernetes distributions, including Red Hat OpenShift. Trident works with the entire NetApp storage portfolio, including the 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, or 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 just like Kubernetes, is released four times a year.

The latest version of Astra Trident is 22.01 released in January 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 21.01 release, a Helm chart was made available to ease the installation of the Trident Operator.

Download Astra Trident

To 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 22.01, which can be downloaded [here](#).

```
[netapp-user@rhel7 ~]$ wget
https://github.com/NetApp/trident/releases/download/v22.01.0/trident-
installer-22.01.0.tar.gz
--2021-05-06 15:17:30--
https://github.com/NetApp/trident/releases/download/v22.01.0/trident-
installer-22.01.0.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-
22.01.0.tar.gz&response-content-type=application%2Foctet-stream
[following]
--2021-05-06 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-
22.01.0.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-22.01.0.tar.gz'

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

2021-05-06 15:17:30 (88.5 MB/s) - 'trident-installer-22.01.0.tar.gz'
saved [38349341/38349341]
```

2. Extract the Trident install from the downloaded bundle.

```
[netapp-user@rhel7 ~]$ tar -xzf trident-installer-22.01.0.tar.gz
[netapp-user@rhel7 ~]$ cd trident-installer/
[netapp-user@rhel7 trident-installer]$
```

Install the Trident Operator with Helm

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.

```
[netapp-user@rhel7 trident-installer]$ export KUBECONFIG=~/ocp-install/auth/kubeconfig
```

2. 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.

```
[netapp-user@rhel7 trident-installer]$ helm install trident
helm/trident-operator-22.01.0.tgz --create-namespace --namespace trident
NAME: trident
LAST DEPLOYED: Fri May  7 12:54:25 2021
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
```

3. 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.

```
[netapp-user@rhel7 trident-installer]$ oc get pods -n trident
NAME                      READY   STATUS    RESTARTS   AGE
trident-csi-5z451          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

[netapp-user@rhel7 trident-installer]$ ./tridentctl -n trident version
+-----+-----+
| SERVER VERSION | CLIENT VERSION |
+-----+-----+
| 22.01.0        | 22.01.0       |
+-----+-----+
```

 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

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.

```
[netapp-user@rhel7 trident-installer]$ export KUBECONFIG=~/ocp-install/auth/kubeconfig
```

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

```
[netapp-user@rhel7 trident-installer]$ oc create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
customresourcedefinition.apiextensions.k8s.io/tridentorchestrators.tride
nt.netapp.io created
```

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

```
[netapp-user@rhel7 trident-installer]$ oc apply -f deploy/namespace.yaml
namespace/trident created
```

4. 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.

```
[netapp-user@rhel7 trident-installer]$ oc 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
```

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

```
[netapp-user@rhel7 trident-installer]$ oc get deployment -n trident
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
trident-operator   1/1      1          1          23s
[netapp-user@rhel7 trident-installer]$ oc get pods -n trident
NAME                           READY   STATUS    RESTARTS   AGE
trident-operator-66f48895cc-lzczk   1/1     Running   0          41s
```

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

```
[netapp-user@rhel7 trident-installer]$ oc create -f
deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created
[netapp-user@rhel7 trident-installer]$ oc describe torc trident
Name:            trident
Namespace:
Labels:          <none>
Annotations:    <none>
API Version:   trident.netapp.io/v1
Kind:           TridentOrchestrator
Metadata:
  Creation Timestamp: 2021-05-07T17:00:28Z
  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:           2021-05-07T17:00:28Z
  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:            2021-05-07T17:00:28Z
  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:21.01
    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:22.01.0
Message:              Trident installed
Namespace:            trident
Status:               Installed
Version:              v22.01.0

Events:
Type    Reason     Age   From                  Message
----  -----  ----  -----
Normal  Installing  80s  trident-operator.netapp.io  Installing
Trident
Normal  Installed   68s  trident-operator.netapp.io  Trident
installed

```

7. 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.

```

[netapp-user@rhel7 trident-installer]$ oc 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

[netapp-user@rhel7 trident-installer]$ ./tridentctl -n trident version
+-----+-----+
| SERVER VERSION | CLIENT VERSION |
+-----+-----+
| 22.01.0         | 22.01.0          |
+-----+-----+

```

Prepare worker nodes for storage

NFS

Most Kubernetes distributions come with the packages and utilities to mount NFS backends installed by default, including Red Hat OpenShift.

However, for NFSv3, there is no mechanism to negotiate concurrency between the client and the server. Hence the maximum number of client-side sunrpc slot table entries must be manually synced with supported value on the server to ensure the best performance for the NFS connection without the server having to decrease the window size of the connection.

For ONTAP, the supported maximum number of sunrpc slot table entries is 128 i.e. ONTAP can serve 128

concurrent NFS requests at a time. However, by default, Red Hat CoreOS/Red Hat Enterprise Linux has maximum of 65,536 sunrpc slot table entries per connection. We need to set this value to 128 and this can be done using Machine Config Operator (MCO) in OpenShift.

To modify the maximum sunrpc slot table entries in OpenShift worker nodes, complete the following steps:

1. Log into the OCP web console and navigate to Compute > Machine Configs. Click Create Machine Config. Copy and paste the YAML file and click Create.

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: 98-worker-nfs-rpc-slot-tables
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
      version: 3.2.0
    storage:
      files:
        - contents:
            source: data:text/plain;charset=utf-
8;base64,b3B0aW9ucyBzdW5ycGMgdGNwX21heF9zbG90X3RhYmx1X2VudHJpZXM9MTI4Cg==
=
          filesystem: root
          mode: 420
          path: /etc/modprobe.d/sunrpc.conf
```

2. After the MCO is created, the configuration needs to be applied on all worker nodes and rebooted one by one. The whole process takes approximately 20 to 30 minutes. Verify whether the machine config is applied by using `oc get mcp` and make sure that the machine config pool for workers is updated.

```
[netapp-user@rhel7 openshift-deploy]$ oc get mcp
NAME      CONFIG                      UPDATED     UPDATING
DEGRADED
master    rendered-master-a520ae930e1d135e0dee7168   True       False
False
worker    rendered-worker-de321b36eeba62df41feb7bc   True       False
False
```

iSCSI

To prepare worker nodes to allow for the mapping of block storage volumes through the iSCSI protocol, you must install the necessary packages to support that functionality.

In Red Hat OpenShift, this is handled by applying an MCO (Machine Config Operator) to your cluster after it is deployed.

To configure the worker nodes to run iSCSI services, complete the following steps:

1. Log into the OCP web console and navigate to Compute > Machine Configs. Click Create Machine Config. Copy and paste the YAML file and click Create.

When not using multipathing:

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 99-worker-element-iscsi
spec:
  config:
    ignition:
      version: 3.2.0
    systemd:
      units:
        - name: iscsid.service
          enabled: true
          state: started
  osImageURL: ""
```

When using multipathing:

```

apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: 99-worker-ontap-iscsi
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
      version: 3.2.0
    storage:
      files:
        - contents:
            source: data:text/plain;charset=utf-
8;base64,ZGVmYXVsdHMgewogICAgICAgIHZXJfZnJpZW5kbH1fbmFtZXgbm8KICAgICA
gICBmaW5kX211bHRpcGF0aHMgbm8KfQoKYmxhY2tsaXN0X2V4Y2VwdG1vbnMgewogICAgICA
gIHByb3BlcnR5ICIoU0NTSV9JREVOVF98SURfV1dOKSIKfQoKYmxhY2tsaXN0IHsKfQoK
      verification: {}
    filesystem: root
    mode: 400
    path: /etc/multipath.conf
  systemd:
    units:
      - name: iscsid.service
        enabled: true
        state: started
      - name: multipathd.service
        enabled: true
        state: started
  osImageURL: ""

```

- After the configuration is created, it takes approximately 20 to 30 minutes to apply the configuration to the worker nodes and reload them. Verify whether the machine config is applied by using `oc get mcp` and make sure that the machine config pool for workers is updated. You can also log into the worker nodes to confirm that the iscsid service is running (and the multipathd service is running if using multipathing).

```
[netapp-user@rhel7 openshift-deploy]$ oc get mcp
NAME      CONFIG                                     UPDATED     UPDATING
DEGRADED
master    rendered-master-a520ae930e1d135e0dee7168   True       False
False
worker    rendered-worker-de321b36eeba62df41feb7bc   True       False
False

[netapp-user@rhel7 openshift-deploy]$ ssh core@10.61.181.22 sudo
systemctl status iscsid
● iscsid.service - Open-iSCSI
   Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled;
   vendor preset: disabled)
     Active: active (running) since Tue 2021-05-26 13:36:22 UTC; 3 min ago
       Docs: man:iscsid(8)
              man:iscsiadm(8)
   Main PID: 1242 (iscsid)
     Status: "Ready to process requests"
      Tasks: 1
     Memory: 4.9M
        CPU: 9ms
      CGroup: /system.slice/iscsid.service
              └─1242 /usr/sbin/iscsid -f

[netapp-user@rhel7 openshift-deploy]$ ssh core@10.61.181.22 sudo
systemctl status multipathd
● multipathd.service - Device-Mapper Multipath Device Controller
   Loaded: loaded (/usr/lib/systemd/system/multipathd.service; enabled;
   vendor preset: enabled)
     Active: active (running) since Tue 2021-05-26 13:36:22 UTC; 3 min ago
   Main PID: 918 (multipathd)
     Status: "up"
      Tasks: 7
     Memory: 13.7M
        CPU: 57ms
      CGroup: /system.slice/multipathd.service
              └─918 /sbin/multipathd -d -s
```



It is also possible to confirm that the MachineConfig has been successfully applied and services have been started as expected by running the `oc debug` command with the appropriate flags.

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 links below in order to continue the setup and configuration of Astra Trident.

- [NetApp ONTAP NFS](#)
- [NetApp ONTAP iSCSI](#)
- [NetApp Element iSCSI](#)

Next: [Solution Validation/Use Cases: Red Hat OpenShift with NetApp](#).

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.

```
[netapp-user@rhel7 trident-installer]$ cp sample-input/backends-samples/ontap-nas/backend-ontap-nas.json ./
[netapp-user@rhel7 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.

```
[netapp-user@rhel7 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 |
+-----+-----+
+-----+-----+
```

- 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.

```
[netapp-user@rhel7 trident-installer]$ cp sample-input/storage-class-
samples/storage-class-csi.yaml.templ ./storage-class-basic.yaml
[netapp-user@rhel7 trident-installer]$ vi storage-class-basic.yaml
```

- 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.

- Run the `oc` command to create the storage class.

```
[netapp-user@rhel7 trident-installer]$ oc create -f storage-class-
basic.yaml
storageclass.storage.k8s.io/basic-csi created
```

- 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-input` as well.

```
[netapp-user@rhel7 trident-installer]$ cp sample-input/pvc-samples/pvc-basic.yaml ./
[netapp-user@rhel7 trident-installer]$ vi pvc-basic.yaml
```

- 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
```

- Create the PVC by issuing the `oc` 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.

```
[netapp-user@rhel7 trident-installer]$ oc create -f pvc-basic.yaml
persistentvolumeclaim/basic created

[netapp-user@rhel7 trident-installer]$ oc get pvc
NAME      STATUS      VOLUME                                     CAPACITY
ACCESS MODES     STORAGECLASS     AGE
basic     Bound      pvc-b4370d37-0fa4-4c17-bd86-94f96c94b42d   1Gi
          RWO        basic-csi       7s
```

[Next: Solution validation/use cases.](#)

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.

- 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.

```
[netapp-user@rhel7 trident-installer]$ cp sample-input/backends-samples/ontap-san/backend-ontap-san.json ./
[netapp-user@rhel7 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.

```
[netapp-user@rhel7 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.

```
[netapp-user@rhel7 trident-installer]$ cp sample-input/storage-class-samples/storage-class-csi.yaml.templ ./storage-class-basic.yaml
[netapp-user@rhel7 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, etc) or can be deleted to allow OpenShift to decide what filesystem to use.

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

```
[netapp-user@rhel7 trident-installer]$ oc 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.

```
[netapp-user@rhel7 trident-installer]$ cp sample-input/pvc-samples/pvc-
basic.yaml .
[netapp-user@rhel7 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 `oc` 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.

```
[netapp-user@rhel7 trident-installer]$ oc create -f pvc-basic.yaml
persistentvolumeclaim/basic created
```

```
[netapp-user@rhel7 trident-installer]$ oc get pvc
NAME      STATUS    VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
basic     Bound     pvc-7ceac1ba-0189-43c7-8f98-094719f7956c   1Gi
RWO          basic-csi   3s
```

[Next: Solution validation/use cases.](#)

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.

```
[netapp-user@rhel7 trident-installer]$ cp sample-input/backends-
samples/solidfire/backend-solidfire.json ./
[netapp-user@rhel7 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.

```
[netapp-user@rhel7 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 |
+-----+-----+
+-----+-----+
```

- 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.

```
[netapp-user@rhel7 trident-installer]$ cp sample-input/storage-class-
samples/storage-class-csi.yaml.templ ./storage-class-basic.yaml
[netapp-user@rhel7 trident-installer]$ vi storage-class-basic.yaml
```

- 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.

- Run the `oc` command to create the storage class.

```
[netapp-user@rhel7 trident-installer]$ oc 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-input` as well.

```
[netapp-user@rhel7 trident-installer]$ cp sample-input/pvc-samples/pvc-basic.yaml ./
[netapp-user@rhel7 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 `oc` 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.

```
[netapp-user@rhel7 trident-installer]$ oc create -f pvc-basic.yaml
persistentvolumeclaim/basic created

[netapp-user@rhel7 trident-installer]$ oc get pvc
NAME      STATUS      VOLUME                                     CAPACITY
ACCESS MODES     STORAGECLASS     AGE
basic     Bound      pvc-3445b5cc-df24-453d-a1e6-b484e874349d   1Gi
          RWO        basic-csi       5s
```

[Next: Solution validation/use cases.](#)

Advanced Configuration Options For OpenShift

Exploring load balancer options: Red Hat OpenShift with NetApp

In most cases, Red Hat OpenShift makes applications available to the outside world through routes. A service is exposed by giving it an externally reachable hostname. The defined route and the endpoints identified by its service can be consumed by an OpenShift router to provide this named connectivity to external clients.

However in some cases, applications require the deployment and configuration of customized load balancers

to expose the appropriate services. One example of this is NetApp Astra Control Center. To meet this need, we have evaluated a number of custom load balancer options. Their installation and configuration are described in this section.

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

- [MetalLB](#)
- [F5 BIG-IP](#)

[Next: Solution validation/use cases: Red Hat OpenShift with NetApp.](#)

Installing MetalLB load balancers: Red Hat OpenShift with NetApp

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

MetalLB is a self-hosted network load balancer installed on your OpenShift cluster that allows the creation of OpenShift services of type load balancer in clusters that do not run on a cloud provider. The two main features of MetalLB that work together to support LoadBalancer services are address allocation and external announcement.

MetalLB configuration options

Based on how MetalLB announces the IP address assigned to LoadBalancer services outside of the OpenShift cluster, it operates in two modes:

- **Layer 2 mode.** In this mode, one node in the OpenShift cluster takes ownership of the service and responds to ARP requests for that IP to make it reachable outside of the OpenShift cluster. Because only the node advertises the IP, it has a bandwidth bottleneck and slow failover limitations. For more information, see the documentation [here](#).
- **BGP mode.** In this mode, all nodes in the OpenShift cluster establish BGP peering sessions with a router and advertise the routes to forward traffic to the service IPs. The prerequisite for this is to integrate MetalLB with a router in that network. Owing to the hashing mechanism in BGP, it has certain limitation when IP-to-Node mapping for a service changes. For more information, refer to the documentation [here](#).



For the purpose of this document, we are configuring MetalLB in layer-2 mode.

Installing The MetalLB Load Balancer

1. Download the MetalLB resources.

```
[netapp-user@rhel7 ~]$ wget https://raw.githubusercontent.com/metallb/metallb/v0.10.2/manifests/namespace.yaml  
[netapp-user@rhel7 ~]$ wget https://raw.githubusercontent.com/metallb/metallb/v0.10.2/manifests/metallb.yaml
```

2. Edit file `metallb.yaml` and remove `spec.template.spec.securityContext` from controller Deployment and the speaker DaemonSet.

Lines to be deleted:

```
securityContext:  
  runAsNonRoot: true  
  runAsUser: 65534
```

3. Create the metallb-system namespace.

```
[netapp-user@rhel7 ~]$ oc create -f namespace.yaml  
namespace/metallb-system created
```

4. Create the MetalLB CR.

```
[netapp-user@rhel7 ~]$ oc create -f metallb.yaml  
podsecuritypolicy.policy/controller created  
podsecuritypolicy.policy/speaker created  
serviceaccount/controller created  
serviceaccount/speaker created  
clusterrole.rbac.authorization.k8s.io/metallb-system:controller created  
clusterrole.rbac.authorization.k8s.io/metallb-system:speaker created  
role.rbac.authorization.k8s.io/config-watcher created  
role.rbac.authorization.k8s.io/pod-lister created  
role.rbac.authorization.k8s.io/controller created  
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:controller  
created  
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:speaker  
created  
rolebinding.rbac.authorization.k8s.io/config-watcher created  
rolebinding.rbac.authorization.k8s.io/pod-lister created  
rolebinding.rbac.authorization.k8s.io/controller created  
daemonset.apps/speaker created  
deployment.apps/controller created
```

5. Before configuring the MetalLB speaker, grant the speaker DaemonSet elevated privileges so that it can perform the networking configuration required to make the load balancers work.

```
[netapp-user@rhel7 ~]$ oc adm policy add-scc-to-user privileged -n  
metallb-system -z speaker  
clusterrole.rbac.authorization.k8s.io/system:openshift:scc:privileged  
added: "speaker"
```

6. Configure MetalLB by creating a ConfigMap in the metallb-system namespace.

```
[netapp-user@rhel7 ~]$ vim metallb-config.yaml
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: default
      protocol: layer2
      addresses:
      - 10.63.17.10-10.63.17.200
```

```
[netapp-user@rhel7 ~]$ oc create -f metallb-config.yaml
configmap/config created
```

7. Now when loadbalancer services are created, MetallB assigns an externalIP to the services and advertises the IP address by responding to ARP requests.



If you wish to configure MetallB in BGP mode, skip step 6 above and follow the procedure in the MetallB documentation [here](#).

Next: [Solution validation/use cases: Red Hat OpenShift with NetApp](#).

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 many more. These services drastically increase the availability, security and performance of your applications.

F5 BIG-IP can be deployed and consumed in various ways, 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 as per requirement.

For efficient integration of F5 BIG-IP services with Red Hat OpenShift, F5 offers the BIG-IP Container Ingress Service (CIS). CIS is installed as a controller pod that watches OpenShift API for certain Custom Resource Definitions (CRDs) and manages the F5 BIG-IP system configuration. F5 BIG-IP CIS can be configured to control service types LoadBalancers and Routes in OpenShift.

Further, for automatic IP address allocation to service the type LoadBalancer, you can utilize the F5 IPAM controller. The F5 IPAM controller is installed as a controller pod that watches OpenShift API for LoadBalancer services with an ipamLabel annotation to allocate the IP address from a preconfigured pool.

This page lists the installation and configuration instructions for F5 BIG-IP CIS and IPAM controller. As a prerequisite, you must have an F5 BIG-IP system deployed and licensed. It must also be licensed for SDN services, which are included by default with the BIG-IP VE base license.



F5 BIG-IP can be deployed in standalone or cluster mode. For the purpose of this validation, F5 BIG-IP was deployed in standalone mode, but, for production purposes, it is preferred to have a cluster of BIG-IPs 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

Technology	Software version
Red Hat OpenShift	4.6 EUS, 4.7
F5 BIG-IP VE edition	16.1.0
F5 Container Ingress Service	2.5.1
F5 IPAM Controller	0.1.4
F5 AS3	3.30.0

Installation

1. Install the F5 Application Services 3 extension to allow BIG-IP systems to accept configurations in JSON instead of imperative commands. Go to [F5 AS3 GitHub repository](#), and download the latest RPM file.
2. Log into F5 BIG-IP system, navigate to iApps > Package Management LX and click Import.
3. Click Choose File and select the downloaded AS3 RPM file, click OK, and then click Upload.



4. Confirm that the AS3 extension is installed successfully.



5. Next configure the resources required for communication between OpenShift and BIG-IP systems. First create a tunnel between OpenShift and the BIG-IP server by creating a VXLAN tunnel interface on the BIG-IP system for OpenShift SDN. Navigate to Network > Tunnels > Profiles, click Create, and set the Parent Profile to vxlan and the Flooding Type to Multicast. Enter a name for the profile and click Finished.

Network > Tunnels > Profiles : VXLAN > New VXLAN Profile...

General Properties	
Name	vxlan-multipoint
Parent Profile	vxlan
Description	
Settings	
Port	4789
Flooding Type	Multicast <input checked="" type="checkbox"/>
<input type="button" value="Cancel"/> <input type="button" value="Repeat"/> <input type="button" value="Finished"/>	

6. Navigate to Network > Tunnels > Tunnel List, click Create, and enter the name and local IP address for the tunnel. Select the tunnel profile that was created in the previous step and click Finished.

Network > Tunnels : Tunnel List > New Tunnel...

Configuration	
Name	openshift_vxlan
Description	
Key	0
Profile	vxlan-multipoint
Local Address	10.63.172.239
Secondary Address	Any
Remote Address	Any
Mode	Bidirectional
MTU	0
Use PMTU	<input checked="" type="checkbox"/> Enabled
TOS	Preserve
Auto-Last Hop	Default
Traffic Group	None
<input type="button" value="Cancel"/> <input type="button" value="Repeat"/> <input type="button" value="Finished"/>	

7. Log into the Red Hat OpenShift cluster with cluster-admin privileges.
8. Create a hostsubnet on OpenShift for the F5 BIG-IP server, which extends the subnet from the OpenShift cluster to the F5 BIG-IP server. Download the host subnet YAML definition.

```
wget https://github.com/F5Networks/k8s-bigip-ctlr/blob/master/docs/config_examples/openshift/f5-kctlr-openshift-hostsubnet.yaml
```

9. Edit the host subnet file and add the BIG-IP VTEP (VXLAN tunnel) IP for the OpenShift SDN.

```
apiVersion: v1
kind: HostSubnet
metadata:
  name: f5-server
  annotations:
    pod.network.openshift.io/fixed-vnid-host: "0"
    pod.network.openshift.io/assign-subnet: "true"
  # provide a name for the node that will serve as BIG-IP's entry into the
  # cluster
  host: f5-server
  # The hostIP address will be the BIG-IP interface address routable to
  # the
  # OpenShift Origin nodes.
  # This address is the BIG-IP VTEP in the SDN's VXLAN.
  hostIP: 10.63.172.239
```



Change the hostIP and other details as applicable to your environment.

10. Create the HostSubnet resource.

```
[admin@rhel-7 ~]$ oc create -f f5-kctlr-openshift-hostsubnet.yaml
hostsubnet.network.openshift.io/f5-server created
```

11. Get the cluster IP subnet range for the host subnet created for the F5 BIG-IP server.

```
[admin@rhel-7 ~]$ oc get hostsubnet
```

NAME	HOST	HOST IP
SUBNET	EGRESS CIDRS	EGRESS IPS
f5-server		f5-server
10.131.0.0/23		10.63.172.239
ocp-vmw-nszws-master-0		ocp-vmw-nszws-master-0
10.128.0.0/23		10.63.172.44
ocp-vmw-nszws-master-1		ocp-vmw-nszws-master-1
10.130.0.0/23		10.63.172.47
ocp-vmw-nszws-master-2		ocp-vmw-nszws-master-2
10.129.0.0/23		10.63.172.48
ocp-vmw-nszws-worker-r8fh4		ocp-vmw-nszws-worker-r8fh4
10.130.2.0/23		10.63.172.7
ocp-vmw-nszws-worker-tvr46		ocp-vmw-nszws-worker-tvr46
10.129.2.0/23		10.63.172.11
ocp-vmw-nszws-worker-wdxhg		ocp-vmw-nszws-worker-wdxhg
10.128.2.0/23		10.63.172.24
ocp-vmw-nszws-worker-wg8r4		ocp-vmw-nszws-worker-wg8r4
10.131.2.0/23		10.63.172.15
ocp-vmw-nszws-worker-wtgef		ocp-vmw-nszws-worker-wtgef
10.128.4.0/23		10.63.172.17

12. Create a self IP on OpenShift VXLAN with an IP in OpenShift's host subnet range corresponding to the F5 BIG-IP server. Log into the F5 BIG-IP system, navigate to Network > Self IPs and click Create. Enter an IP from the cluster IP subnet created for F5 BIG-IP host subnet, select the VXLAN tunnel, and enter the other details. Then click Finished.

Network » Self IPs » New Self IP...

Configuration

Name	10.131.0.60
IP Address	10.131.0.60
Netmask	255.252.0.0
VLAN / Tunnel	openshift_vxla
Port Lockdown	Allow All
Traffic Group	<input type="checkbox"/> Inherit traffic group from current partition / path traffic-group-local-only (non-floating)
Service Policy	None

Cancel Repeat Finished

13. Create a partition in the F5 BIG-IP system to be configured and used with CIS. Navigate to System > Users > Partition List, click Create, and enter the details. Then click Finished.

System » Users : Partition List » New Partition...

Properties	
Partition Name	ocp-vmw
Partition Default Route Domain	0 ▾
Description	<input type="checkbox"/> Extend Text Area <input type="checkbox"/> Wrap Text
Redundant Device Configuration	
Device Group	<input checked="" type="checkbox"/> Inherit device group from root folder None ▾
Traffic Group	<input checked="" type="checkbox"/> Inherit traffic group from root folder traffic-group-1 (floating) ▾
<input type="button" value="Cancel"/> <input type="button" value="Repeat"/> <input type="button" value="Finished"/>	



F5 recommends that no manual configuration be done on the partition that is managed by CIS.

14. Install the F5 BIG-IP CIS using the operator from OperatorHub. Log into the Red Hat OpenShift cluster with cluster-admin privileges and create a secret with F5 BIG-IP system login credentials, which is a prerequisite for the operator.

```
[admin@rhel-7 ~]$ oc create secret generic bigip-login -n kube-system  
--from-literal=username=admin --from-literal=password=admin  
  
secret/bigip-login created
```

15. Install the F5 CIS CRDs.

```
[admin@rhel-7 ~]$ oc apply -f  
https://raw.githubusercontent.com/F5Networks/k8s-bigip-  
ctlr/master/docs/config_examples/crd/Install/customresourcedefinitions.y  
ml  
  
customresourcedefinition.apiextensions.k8s.io/virtualservers.cis.f5.com  
created  
customresourcedefinition.apiextensions.k8s.io/tlsprofiles.cis.f5.com  
created  
customresourcedefinition.apiextensions.k8s.io/transportservers.cis.f5.co  
m created  
customresourcedefinition.apiextensions.k8s.io/externaldnss.cis.f5.com  
created  
customresourcedefinition.apiextensions.k8s.io/ingresslinks.cis.f5.com  
created
```

16. Navigate to Operators > OperatorHub, search for the keyword F5, and click the F5 Container Ingress Service tile.

OperatorHub

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



The screenshot shows the OperatorHub interface. On the left, there is a sidebar with a list of categories: All Items, AI/Machine Learning, Application Runtime, Big Data, Cloud Provider, Database, Developer Tools, Development Tools, Drivers And Plugins, Integration & Delivery, Logging & Tracing, Modernization & Migration, and Monitoring. The 'All Items' tab is currently selected. In the center, there is a search bar with the text 'F5'. Below the search bar, there is a card for the 'F5 Container Ingress Services' operator. The card features the F5 logo, the text 'F5 Container Ingress Services provided by F5 Networks Inc.', and a description: 'Operator to install F5 Container Ingress Services (CIS) for BIG-IP.' To the right of the card, it says '1 items'.

17. Read the operator information and click Install.

The screenshot shows the F5 Container Ingress Services operator page. At the top, there's a logo for F5 Networks Inc. followed by the title "F5 Container Ingress Services" and the version "1.8.0 provided by F5 Networks Inc.". A large blue "Install" button is prominently displayed. Below the button, the "Latest version" is listed as 1.8.0. The "Capability level" section includes a checked radio button for "Basic Install" and other options like "Seamless Upgrades", "Full Lifecycle", "Deep Insights", and "Auto Pilot". The "Provider type" is listed as "Certified". The "Provider" is "F5 Networks Inc.". The "Repository" is "https://github.com/F5Networks/k8s-bigip-ctlr". The "Container Image" is "registry.connect.redhat.com/f5networks/k8s-bigip-ctlr". The "Introduction" section describes the operator's function of installing CIS for BIG-IP in a cluster using Helm Charts. The "F5 Container Ingress Services for BIG-IP" section provides a brief overview of how CIS integrates with container orchestration environments. The "Documentation" section links to F5 documentation and specific routes on OpenShift. The "Prerequisites" section instructs users to create BIG-IP login credentials using the command:

```
oc create secret generic <SECRET-NAME> -n kube-system --from-literal=username=<USERNAME> --from-literal=password=<PASSWORD>
```

18. On the Install operator screen, leave all default parameters, and click Install.

Install Operator

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic updates.

Update channel *

beta

F5 Container Ingress Services
provided by F5 Networks Inc.

Provided APIs

F5ContainerIngressServices

This CRD provides kind `F5ContainerIngressServices` to configure and deploy F5 Container Ingress Services.

Installation mode *

All namespaces on the cluster (default)
Operator will be available in all Namespaces.

A specific namespace on the cluster
Operator will be available in a single Namespace only.

Installed Namespace *

PR openshift-operators

Approval strategy *

Automatic

Manual

Install **Cancel**

19. It takes a while to install the operator.



20. After the operator is installed, the Installation Successful message is displayed.
21. Navigate to Operators > Installed Operators, click F5 Container Ingress Service, and then click Create Instance under the F5BigIpCtlr tile.



F5 Container Ingress Services

1.8.0 provided by F5 Networks Inc.

[Details](#)[YAML](#)[Subscription](#)[Events](#)[F5BigIpCtlr](#)

Provided APIs

FBIC F5BigIpCtlr

This CRD provides kind `F5BigIpCtlr` to configure and deploy F5 BIG-IP Controller.

 [Create instance](#)

22. Click YAML View and paste the following content after updating the necessary parameters.



Update the parameters `bigip_partition`, ``openshift_sdn_name``, `bigip_url` and `bigip_login_secret` below to reflect the values for your setup before copying the content.

```

apiVersion: cis.f5.com/v1
kind: F5BigIpCtlr
metadata:
  name: f5-server
  namespace: openshift-operators
spec:
  args:
    log_as3_response: true
    agent: as3
    log_level: DEBUG
    bigip_partition: ocp-vmw
    openshift_sdn_name: /Common/openshift_vxlan
    bigip_url: 10.61.181.19
    insecure: true
    pool-member-type: cluster
    custom_resource_mode: true
    as3_validation: true
    ipam: true
    manage_configmaps: true
    bigip_login_secret: bigip-login
  image:
    pullPolicy: Always
    repo: f5networks/cntr-ingress-svcs
    user: registry.connect.redhat.com
  namespace: kube-system
  rbac:
    create: true
  resources: {}
  serviceAccount:
    create: true
  version: latest

```

23. After pasting this content, click Create. This installs the CIS pods in the kube-system namespace.

Pods								Create Pod
Name	Status	Ready	Restarts	Owner	Memory	CPU		
f5-server-f5-bigip-ctlr-5d7578667d-qxdgj	Running	1/1	0	RS f5-server-f5-bigip-ctlr-5d7578667d	61.1 MiB	0.003 cores	⋮	



Red Hat OpenShift, by default, provides a way to expose the services via Routes for L7 load balancing. An inbuilt OpenShift router is responsible for advertising and handling traffic for these routes. However, you can also configure the F5 CIS to support the Routes through an external F5 BIG-IP system, which can run either as an auxiliary router or a replacement to the self-hosted OpenShift router. CIS creates a virtual server in the BIG-IP system that acts as a router for the OpenShift routes, and BIG-IP handles the advertisement and traffic routing. Refer to the documentation here for information on parameters to enable this feature. Note that these parameters are defined for OpenShift Deployment resource in the apps/v1 API. Therefore, when using these with the F5BigIpCtlr resource cis.f5.com/v1 API, replace the hyphens (-) with underscores (_) for the parameter names.

24. The arguments that are passed to the creation of CIS resources include `ipam: true` and `custom_resource_mode: true`. These parameters are required for enabling CIS integration with an IPAM controller. Verify that the CIS has enabled IPAM integration by creating the F5 IPAM resource.

```
[admin@rhel-7 ~]$ oc get f5ipam -n kube-system  
  
NAMESPACE      NAME          AGE  
kube-system    ipam.10.61.181.19.ocp-vmw   43s
```

25. Create the service account, role and rolebinding required for the F5 IPAM controller. Create a YAML file and paste the following content.

```
[admin@rhel-7 ~]$ vi f5-ipam-rbac.yaml

kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: ipam-ctlr-clusterrole
rules:
  - apiGroups: ["fic.f5.com"]
    resources: ["ipams","ipams/status"]
    verbs: ["get", "list", "watch", "update", "patch"]
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: ipam-ctlr-clusterrole-binding
  namespace: kube-system
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: ipam-ctlr-clusterrole
subjects:
  - apiGroup: ""
    kind: ServiceAccount
    name: ipam-ctlr
    namespace: kube-system
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: ipam-ctlr
  namespace: kube-system
```

26. Create the resources.

```
[admin@rhel-7 ~]$ oc create -f f5-ipam-rbac.yaml

clusterrole.rbac.authorization.k8s.io/ipam-ctlr-clusterrole created
clusterrolebinding.rbac.authorization.k8s.io/ipam-ctlr-clusterrole-
binding created
serviceaccount/ipam-ctlr created
```

27. Create a YAML file and paste the F5 IPAM deployment definition provided below.



Update the ip-range parameter in spec.template.spec.containers[0].args below to reflect the ipamLabels and IP address ranges corresponding to your setup.



ipamLabels [range1 and range2 in below example] are required to be annotated for the services of type LoadBalancer for the IPAM controller to detect and assign an IP address from the defined range.

```
[admin@rhel-7 ~]$ vi f5-ipam-deployment.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    name: f5-ipam-controller
    name: f5-ipam-controller
    namespace: kube-system
spec:
  replicas: 1
  selector:
    matchLabels:
      app: f5-ipam-controller
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: f5-ipam-controller
    spec:
      containers:
        - args:
            - --orchestration=openshift
            - --ip-range='{"range1":"10.63.172.242-10.63.172.249",
"range2":"10.63.170.111-10.63.170.129"}'
            - --log-level=DEBUG
          command:
            - /app/bin/f5-ipam-controller
          image: registry.connect.redhat.com/f5networks/f5-ipam-
controller:latest
          imagePullPolicy: IfNotPresent
          name: f5-ipam-controller
        dnsPolicy: ClusterFirst
        restartPolicy: Always
        schedulerName: default-scheduler
        securityContext: {}
        serviceAccount: ipam-ctlr
        serviceAccountName: ipam-ctlr
```

28. Create the F5 IPAM controller deployment.

```
[admin@rhel-7 ~]$ oc create -f f5-ipam-deployment.yaml  
deployment/f5-ipam-controller created
```

29. Verify the F5 IPAM controller pods are running.

```
[admin@rhel-7 ~]$ oc get pods -n kube-system  
  
NAME                                READY   STATUS    RESTARTS  
AGE  
f5-ipam-controller-5986cff5bd-2bvn6   1/1     Running   0  
30s  
f5-server-f5-bigip-ctlr-5d7578667d-qxdgj   1/1     Running   0  
14m
```

30. Create the F5 IPAM schema.

```
[admin@rhel-7 ~]$ oc create -f  
https://raw.githubusercontent.com/F5Networks/f5-ipam-  
controller/main/docs/_static/schemas/ipam_schema.yaml  
  
customresourcedefinition.apiextensions.k8s.io/ipams.fic.f5.com
```

Verification

1. Create a service of type LoadBalancer

```
[admin@rhel-7 ~]$ vi example_svc.yaml

apiVersion: v1
kind: Service
metadata:
  annotations:
    cis.f5.com/ipamLabel: range1
  labels:
    app: f5-demo-test
  name: f5-demo-test
  namespace: default
spec:
  ports:
  - name: f5-demo-test
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: f5-demo-test
  sessionAffinity: None
  type: LoadBalancer
```

```
[admin@rhel-7 ~]$ oc create -f example_svc.yaml

service/f5-demo-test created
```

2. Check if the IPAM controller assigns an external IP to it.

```
[admin@rhel-7 ~]$ oc get svc

NAME           TYPE      CLUSTER-IP      EXTERNAL-IP
PORT (S)       AGE
f5-demo-test   LoadBalancer 172.30.210.108  10.63.172.242
80:32605/TCP  27s
```

3. Create a deployment and use the LoadBalancer service that was created.

```
[admin@rhel-7 ~]$ vi example_deployment.yaml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: f5-demo-test
    name: f5-demo-test
spec:
  replicas: 2
  selector:
    matchLabels:
      app: f5-demo-test
  template:
    metadata:
      labels:
        app: f5-demo-test
    spec:
      containers:
        - env:
            - name: service_name
              value: f5-demo-test
          image: nginx
          imagePullPolicy: Always
          name: f5-demo-test
          ports:
            - containerPort: 80
              protocol: TCP
```

```
[admin@rhel-7 ~]$ oc create -f example_deployment.yaml
deployment/f5-demo-test created
```

4. Check if the pods are running.

```
[admin@rhel-7 ~]$ oc get pods
NAME                      READY   STATUS    RESTARTS   AGE
f5-demo-test-57c46f6f98-47wwp 1/1     Running   0          27s
f5-demo-test-57c46f6f98-cl2m8 1/1     Running   0          27s
```

5. Check if the corresponding virtual server is created in the BIG-IP system for the service of type LoadBalancer in OpenShift. Navigate to Local Traffic > Virtual Servers > Virtual Server List.



Next: Solution Validation/Use Cases: Red Hat OpenShift with NetApp.

Creating Private Image Registries

For most deployments of Red Hat OpenShift, using a public registry like [Quay.io](#) or [DockerHub](#) meets most customer's needs. However there are times when a customer may want to host their own private or customized images.

This procedure documents creating a private image registry which is backed by a persistent volume provided by Astra Trident and NetApp ONTAP.



Astra Control Center requires a registry to host the images the Astra containers require. The following section describes the steps to setup a private registry on Red Hat OpenShift cluster and pushing the images required to support the installation of Astra Control Center.

Creating A private image registry

1. Remove the default annotation from the current default storage class and annotate the Trident-backed storage class as default for the OpenShift cluster.

```
[netapp-user@rhel7 ~]$ oc patch storageclass thin -p '{"metadata": {"annotations": {"storageclass.kubernetes.io/is-default-class": "false"}}}'
storageclass.storage.k8s.io/thin patched

[netapp-user@rhel7 ~]$ oc patch storageclass ocp-trident -p '{"metadata": {"annotations": {"storageclass.kubernetes.io/is-default-class": "true"}}}'
storageclass.storage.k8s.io/ocp-trident patched
```

2. Edit the imageregistry operator by entering the following storage parameters in the `spec` section.

```
[netapp-user@rhel7 ~]$ oc edit
configs.imageregistry.operator.openshift.io

storage:
  pvc:
    claim:
```

3. Enter the following parameters in the `spec` section for creating a OpenShift route with a custom hostname.

Save and exit.

```
routes:  
  - hostname: astra-registry.apps.ocp-vmw.cie.netapp.com  
    name: netapp-astra-route
```



The above route config is used when you want a custom hostname for your route. If you want OpenShift to create a route with a default hostname, you can add the following parameters to the spec section: `defaultRoute: true`.

Custom TLS certificates

When you are using a custom hostname for the route, by default, it uses the default TLS configuration of the OpenShift Ingress operator. However, you can add a custom TLS configuration to the route. To do so, complete the following steps.

- Create a secret with the route's TLS certificates and key.

```
[netapp-user@rhel7 ~]$ oc create secret tls astra-route-tls -n openshift-image-registry -cert/home/admin/netapp-astra/tls.crt --key=/home/admin/netapp-astra/tls.key
```

- Edit the imageregistry operator and add the following parameters to the spec section.

```
[netapp-user@rhel7 ~]$ oc edit  
configs.imageregistry.operator.openshift.io  
  
routes:  
  - hostname: astra-registry.apps.ocp-vmw.cie.netapp.com  
    name: netapp-astra-route  
    secretName: astra-route-tls
```

- Edit the imageregistry operator again and change the management state of the operator to the Managed state. Save and exit.

```
oc edit configs.imageregistry/cluster  
  
managementState: Managed
```

- If all the prerequisites are satisfied, PVCs, pods, and services are created for the private image registry. In a few minutes, the registry should be up.

```
[netapp-user@rhel7 ~]$ oc get all -n openshift-image-registry
```

NAME	READY	STATUS
RESTARTS	AGE	
pod/cluster-image-registry-operator-74f6d954b6-rb7zr	1/1	Running
3 90d		
pod/image-pruner-1627257600-f5cpj	0/1	Completed
0 2d9h		
pod/image-pruner-1627344000-swqzx9	0/1	Completed
0 33h		
pod/image-pruner-1627430400-rv5nt	0/1	Completed
0 9h		
pod/image-registry-6758b547f-6pnj8	1/1	Running
0 76m		
pod/node-ca-bwb5r	1/1	Running
0 90d		
pod/node-ca-f8w54	1/1	Running
0 90d		
pod/node-ca-gjx7h	1/1	Running
0 90d		
pod/node-ca-lcx4k	1/1	Running
0 33d		
pod/node-ca-v7zmx	1/1	Running
0 7d21h		
pod/node-ca-xpppp	1/1	Running
0 89d		

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP
IP	PORT(S)	AGE	
service/image-registry	ClusterIP	172.30.196.167	<none>
5000/TCP	15h		
service/image-registry-operator	ClusterIP	None	<none>
60000/TCP	90d		

NAME	DESIRED	CURRENT	READY	UP-TO-DATE
AVAILABLE	NODE SELECTOR	AGE		
daemonset.apps/node-ca	6	6	6	6
kubernetes.io/os=linux	90d			

NAME	READY	UP-TO-DATE
AVAILABLE	AGE	
deployment.apps/cluster-image-registry-operator	1/1	1
90d		1
deployment.apps/image-registry	1/1	1
15h		1

NAME	CURRENT	READY	AGE	DESIRED
replicaset.apps/cluster-image-registry-operator-74f6d954b6	1	90d		1 1
replicaset.apps/image-registry-6758b547f	1	76m		1 1
replicaset.apps/image-registry-78bfb7f59	0	15h		0 0
replicaset.apps/image-registry-7fcc8d6cc8	0	80m		0 0
replicaset.apps/image-registry-864f88f5b	0	15h		0 0
replicaset.apps/image-registry-cb47ffffb	0	10h		0 0
NAME	COMPLETIONS	DURATION	AGE	
job.batch/image-pruner-1627257600	1/1	10s	2d9h	
job.batch/image-pruner-1627344000	1/1	6s	33h	
job.batch/image-pruner-1627430400	1/1	5s	9h	
NAME	SCHEDULE	SUSPEND	ACTIVE	LAST
SCHEDULE	AGE			
cronjob.batch/image-pruner	0 0 * * *	False	0	9h
90d				
NAME	HOST/PORT			
PATH	SERVICES	PORT	TERMINATION	WILDCARD
route.route.openshift.io/public-routes	astra-registry.apps.ocp-			
vmw.cie.netapp.com	image-registry	<all>	reencrypt	None

6. If you are using the default TLS certificates for the ingress operator OpenShift registry route, you can fetch the TLS certificates using the following command.

```
[netapp-user@rhel7 ~]$ oc extract secret/router-ca --keys=tls.crt -n openshift-ingress-operator
```

7. To allow OpenShift nodes to access and pull the images from the registry, add the certificates to the docker client on the OpenShift nodes. Create a configmap in the `openshift-config` namespace using the TLS certificates and patch it to the cluster image config to make the certificate trusted.

```
[netapp-user@rhel7 ~]$ oc create configmap astra-ca -n openshift-config  
--from-file=astra-registry.apps.ocp-vmw.cie.netapp.com=tls.crt  
  
[netapp-user@rhel7 ~]$ oc patch image.config.openshift.io/cluster  
--patch '{"spec":{"additionalTrustedCA":{"name":"astra-ca"}}}'  
--type=merge
```

8. The OpenShift internal registry is controlled by authentication. All the OpenShift users can access the OpenShift registry, but the operations that the logged in user can perform depends on the user permissions.
 - a. To allow a user or a group of users to pull images from the registry, the user(s) must have the registry-viewer role assigned.

```
[netapp-user@rhel7 ~]$ oc policy add-role-to-user registry-viewer  
ocp-user
```

```
[netapp-user@rhel7 ~]$ oc policy add-role-to-group registry-viewer  
ocp-user-group
```

- b. To allow a user or group of users to write or push images, the user(s) must have the registry-editor role assigned.

```
[netapp-user@rhel7 ~]$ oc policy add-role-to-user registry-editor  
ocp-user
```

```
[netapp-user@rhel7 ~]$ oc policy add-role-to-group registry-editor  
ocp-user-group
```

9. For OpenShift nodes to access the registry and push or pull the images, you need to configure a pull secret.

```
[netapp-user@rhel7 ~]$ oc create secret docker-registry astra-registry-  
credentials --docker-server=astra-registry.apps.ocp-vmw.cie.netapp.com  
--docker-username=ocp-user --docker-password=password
```

10. This pull secret can then be patched to serviceaccounts or be referenced in the corresponding pod definition.

- a. To patch it to service accounts, run the following command.

```
[netapp-user@rhel7 ~]$ oc secrets link <service_account_name> astra-  
registry-credentials --for=pull
```

- b. To reference the pull secret in the pod definition, add the following parameter to the spec section.

```
imagePullSecrets:  
  - name: astra-registry-credentials
```

11. To push or pull an image from workstations apart from OpenShift node, complete the following steps.

- a. Add the TLS certificates to the docker client.

```
[netapp-user@rhel7 ~]$ sudo mkdir /etc/docker/certs.d/astra-  
registry.apps.ocp-vmw.cie.netapp.com  
  
[netapp-user@rhel7 ~]$ sudo cp /path/to/tls.crt  
/etc/docker/certs.d/astra-registry.apps.ocp-vmw.cie.netapp.com
```

- b. Log into OpenShift using the oc login command.

```
[netapp-user@rhel7 ~]$ oc login --token=sha256~D49SpB_lesSrJYwrM0LIO  
-VRcjWHu0a27vKa0 --server=https://api.ocp-vmw.cie.netapp.com:6443
```

- c. Log into the registry using OpenShift user credentials with the podman/docker command.

podman

```
[netapp-user@rhel7 ~]$ podman login astra-registry.apps.ocp-  
vmw.cie.netapp.com -u kubeadmin -p $(oc whoami -t) --tls  
-verify=false
```

+

NOTE: If you are using kubeadmin user to log into the private registry, then use token instead of password.

docker

```
[netapp-user@rhel7 ~]$ docker login astra-registry.apps.ocp-  
vmw.cie.netapp.com -u kubeadmin -p $(oc whoami -t)
```

+

NOTE: If you are using kubeadmin user to log into the private registry, then use token instead of password.

- d. Push or pull the images.

podman

```
[netapp-user@rhel7 ~]$ podman push astra-registry.apps.ocp-vmw.cie.netapp.com/netapp-astra/vault-controller:latest  
[netapp-user@rhel7 ~]$ podman pull astra-registry.apps.ocp-vmw.cie.netapp.com/netapp-astra/vault-controller:latest
```

docker

```
[netapp-user@rhel7 ~]$ docker push astra-registry.apps.ocp-vmw.cie.netapp.com/netapp-astra/vault-controller:latest  
[netapp-user@rhel7 ~]$ docker pull astra-registry.apps.ocp-vmw.cie.netapp.com/netapp-astra/vault-controller:latest
```

Next: [Solution Validation/Use Cases: Red Hat OpenShift with NetApp](#).

Solution Validation and Use Cases: Red Hat OpenShift with NetApp

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

- [Deploy a Jenkins CI/CD Pipeline with Persistent Storage](#)
- [Configure Multitenancy on Red Hat OpenShift with NetApp](#)
- [Red Hat OpenShift Virtualization with NetApp ONTAP](#)
- [Advanced Cluster Management for Kubernetes on Red Hat OpenShift with NetApp](#)

Next: [Videos and Demos](#).

Deploy a Jenkins CI/CD Pipeline with Persistent Storage: Red Hat OpenShift with NetApp

This section provides the steps to deploy a continuous integration/continuous delivery or deployment (CI/CD) pipeline with Jenkins to validate solution operation.

Create the resources required for Jenkins deployment

To create the resources required for deploying the Jenkins application, complete the following steps:

1. Create a new project named Jenkins.

Create Project

Name *

Display Name

Description

Cancel

Create

2. In this example, we deployed Jenkins with persistent storage. To support the Jenkins build, create the PVC. Navigate to Storage > Persistent Volume Claims and click Create Persistent Volume Claim. Select the storage class that was created, make sure that the Persistent Volume Claim Name is jenkins, select the appropriate size and access mode, and then click Create.

Create Persistent Volume Claim

[Edit YAML](#)**Storage Class****SC basic**

Storage class for the new claim.

Persistent Volume Claim Name *

jenkins

A unique name for the storage claim within the project.

Access Mode *

-
- Single User (RWO)
-
- Shared Access (RWX)
-
- Read Only (ROX)

Permissions to the mounted drive.

Size *

100

GiB ▾

Desired storage capacity.

-
- Use label selectors to request storage

Use label selectors to define how storage is created.

Create**Cancel****Deploy Jenkins with Persistent Storage**

To deploy Jenkins with persistent storage, complete the following steps:

1. In the upper left corner, change the role from Administrator to Developer. Click +Add and select From Catalog. In the Filter by Keyword bar, search for jenkins. Select Jenkins Service with Persistent Storage.

Developer Catalog

Add shared apps, services, or source-to-image builders to your project from the Developer Catalog. Cluster admins can install additional apps which will show up here automatically.

- [All Items](#)
- [Languages](#)
- [Databases](#)
- [Middleware](#)
- [CI/CD](#)
- [Other](#)

Type

- Operator Backed (0)
- Helm Charts (0)
- Builder Image (0)
- Template (4)
- Service Class (0)

All Items

Group By: None ▾

Jenkins
Template

provided by Red Hat, Inc.

Jenkins service, with persistent storage. NOTE: You must have persistent volumes available in...

Jenkins
Template

provided by Red Hat, Inc.

Jenkins service, with persistent storage. NOTE: You must have persistent volumes available in...

Jenkins (Ephemeral)
Template

provided by Red Hat, Inc.

Jenkins service, without persistent storage. WARNING: Any data stored will be lost upon...

Jenkins (Ephemeral)
Template

provided by Red Hat, Inc.

Jenkins service, without persistent storage. WARNING:

2. Click Instantiate Template.

Jenkins

Provided by Red Hat, Inc.

[Instantiate Template](#)

Provider	Description
Red Hat, Inc.	Jenkins service, with persistent storage.
Support	NOTE: You must have persistent volumes available in your cluster to use this template.
Get support ↗	
Created At	Documentation
⌚ May 26, 3:58 am	https://docs.okd.io/latest/using_images/other_images/jenkins.html ↗

3. By default, the details for the Jenkins application are populated. Based on your requirements, modify the parameters and click Create. This process creates all the required resources for supporting Jenkins on

OpenShift.

Instantiate Template

Namespace *

Jenkins Service Name

The name of the OpenShift Service exposed for the Jenkins container.

Jenkins JNLP Service Name

The name of the service used for master/slave communication.

Enable OAuth in Jenkins

Whether to enable OAuth OpenShift integration. If false, the static account 'admin' will be initialized with the password 'password'.

Memory Limit

Maximum amount of memory the container can use.

Volume Capacity *

Volume space available for data, e.g. 512Mi, 2Gi.

Jenkins ImageStream Namespace

The OpenShift Namespace where the Jenkins ImageStream resides.

Disable memory intensive administrative monitors

Whether to perform memory intensive, possibly slow, synchronization with the Jenkins Update Center on start. If true, the Jenkins core update monitor and site warnings monitor are disabled.

Jenkins ImageStreamTag

Name of the ImageStreamTag to be used for the Jenkins image.

Fatal Error Log File

When a fatal error occurs, an error log is created with information and the state obtained at the time of the fatal error.

Allows use of Jenkins Update Center repository with invalid SSL certificate

Whether to allow use of a Jenkins Update Center that uses invalid certificate (self-signed, unknown CA). If any value other than 'false', certificate check is bypassed. By default, certificate check is enforced.

Create Cancel



Jenkins service, with persistent storage.

NOTE: You must have persistent volumes available in your cluster to use this template.

The following resources will be created:

- DeploymentConfig
- PersistentVolumeClaim
- RoleBinding
- Route
- Service
- ServiceAccount

4. The Jenkins pods take approximately 10 to 12 minutes to enter the Ready state.

Project: jenkins ▾

Pods

Create Pod

Filter by name...

1	Running	0	Pending	0	Terminating	0	CrashLoopBackOff	1	Completed	0	Failed	0	Unknown
---	---------	---	---------	---	-------------	---	------------------	---	-----------	---	--------	---	---------

Select all filters

1 of 2 Items

Name	Namespace	Status	Ready	Owner	Memory	CPU	⋮
jenkins-1-c77n9	jenkins	Running	1/1	jenkins-1	-	0.004 cores	⋮

5. After the pods are instantiated, navigate to Networking > Routes. To open the Jenkins webpage, click the URL provided for the jenkins route.

Project: jenkins ▾

Routes

Create Route

Filter by name...

1	Accepted	0	Rejected	0	Pending	Select all filters	1 Item
---	----------	---	----------	---	---------	--------------------	--------

Name	Namespace	Status	Location	Service	⋮
jenkins	jenkins	Accepted	https://jenkins-jenkins.apps.rhv-ocp-cluster.cie.netapp.com	jenkins	⋮

6. Because OpenShift OAuth was used while creating the Jenkins app, click Log in with OpenShift.



7. Authorize the Jenkins service account to access the OpenShift users.

Authorize Access

Service account jenkins in project jenkins is requesting permission to access your account (kube:admin)

Requested permissions

user:info

Read-only access to your user information (including username, identities, and group membership)

user:check-access

Read-only access to view your privileges (for example, "can I create builds?")

You will be redirected to <https://jenkins-jenkins.apps.rhv-ocp-cluster.cie.netapp.com/securityRealm/finishLogin>

[Allow selected permissions](#) [Deny](#)

8. The Jenkins welcome page is displayed. Because we are using a Maven build, complete the Maven installation first. Navigate to Manage Jenkins > Global Tool Configuration, and then, in the Maven subhead, click Add Maven. Enter the name of your choice and make sure that the Install Automatically option is selected. Click Save.

Maven

Maven installations

Add Maven

Maven

Name M3

Install automatically

Install from Apache

Version 3.6.3

Add Installer

Delete Installer

Add Maven

Delete Maven

List of Maven installations on this system

9. You can now create a pipeline to demonstrate the CI/CD workflow. On the home page, click Create New Jobs or New Item from the left-hand menu.

The screenshot shows the Jenkins home page. At the top, there's a navigation bar with the Jenkins logo, a search bar, and user information (kube:admin | log out). Below the bar, a sidebar on the left lists links: New Item, People, Build History, Manage Jenkins, My Views, Open Blue Ocean, Lockable Resources, Credentials, and New View. The main content area features a "Welcome to Jenkins!" message with a sub-instruction: "Please [create new jobs](#) to get started." Below this, there are two sections: "Build Queue" (No builds in the queue) and "Build Executor Status" (1 Idle, 2 Idle).

10. On the Create Item page, enter the name of your choice, select Pipeline, and click Ok.

The screenshot shows the "Enter an item name" dialog. The input field contains "sample-demo", which is labeled as a "Required field". Below the input field, there's a list of project types with icons:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Bitbucket Team/Project**: Scans a Bitbucket Cloud Team (or Bitbucket Server Project) for all repositories matching some defined markers.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- GitHub Organization**: Scans a GitHub organization (or user account) for all repositories matching some defined markers.

At the bottom of the dialog, there's an "OK" button and a note: "Creates a set of Pipeline projects according to detected branches in one SCM repository".

11. Select the Pipeline tab. From the Try Sample Pipeline drop-down menu, select Github + Maven. The code is automatically populated. Click Save.

General Build Triggers Advanced Project Options **Pipeline**

[Advanced...](#)

Pipeline

Definition Pipeline script

Script

```

1  node {
2      def mvnHome
3      stage('Preparation') { // for display purposes
4          // Get some code from a GitHub repository
5          git 'https://github.com/jglick/simple-maven-project-with-tests.git'
6          // Get the Maven tool.
7          // ** NOTE: This 'M3' Maven tool must be configured
8          // ** in the global configuration.
9          mvnHome = tool 'M3'
10     }
11    stage('Build') {
12        // Run the maven build
13        withEnv(["MVN_HOME=$mvnHome"]) {
14            if (isUnix()) {
15                sh '$MVN_HOME/bin/mvn' -Dmaven.test.failure.ignore clean package'
16            } else {
17                bat("%MVN_HOME%\bin\mvn" -Dmaven.test.failure.ignore clean package)
18            }
19        }
20    }
21  }

```

GitHub + Maven

Use Groovy Sandbox

[Pipeline Syntax](#)

Save **Apply**

12. Click Build Now to trigger the development through the preparation, build, and testing phase. It can take several minutes to complete the whole build process and display the results of the build.

 Jenkins

Jenkins > sample-demo >

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Build Now](#)

[Delete Pipeline](#)

[Configure](#)

[Full Stage View](#)

[Open Blue Ocean](#)

[Rename](#)

[Pipeline Syntax](#)

Pipeline sample-demo

Last Successful Artifacts

 simple-maven-project-with-tests-1.0-SNAPSHOT.jar 1.71 KB [view](#)

Recent Changes



Stage View

Average stage times:
(Average full run time: ~7s)

Preparation	Build	Results
2s	4s	69ms
2s	4s	69ms

 Latest Test Result (no failures)

Permalinks

- [Last build \(#1\), 1 min 23 sec ago](#)
- [Last stable build \(#1\), 1 min 23 sec ago](#)
- [Last successful build \(#1\), 1 min 23 sec ago](#)
- [Last completed build \(#1\), 1 min 23 sec ago](#)

13. Whenever there are any code changes, the pipeline can be rebuilt to patch the new version of software enabling continuous integration and continuous delivery. Click Recent Changes to track the changes from the previous version.

Next: Videos and Demos.

Configure Multi-tenancy on Red Hat OpenShift with NetApp ONTAP

Configuring multitenancy on Red Hat OpenShift with NetApp

Many organizations that run multiple applications or workloads on containers tend to deploy one Red Hat OpenShift cluster per application or workload. This allows them to implement strict isolation for the application or workload, optimize performance, and reduce security vulnerabilities. However, deploying a separate Red Hat OpenShift cluster for each application poses its own set of problems. It increases operational overhead having to monitor and manage each cluster on its own, increases cost owing to dedicated resources for different applications, and hinders efficient scalability.

To overcome these problems, one can consider running all the applications or workloads in a single Red Hat OpenShift cluster. But in such an architecture, resource isolation and application security vulnerabilities are one of the major challenges. Any security vulnerability in one workload could naturally spill over into another workload, thus increasing the impact zone. In addition, any abrupt uncontrolled resource utilization by one application can affect the performance of another application, because there is no resource allocation policy by default.

Therefore, organizations look out for solutions that pick up the best in both worlds, for example, by allowing them to run all their workloads in a single cluster and yet offering the benefits of a dedicated cluster for each workload.

One such effective solution is to configure multitenancy on Red Hat OpenShift. Multitenancy is an architecture that allows multiple tenants to coexist on the same cluster with proper isolation of resources, security, and so on. In this context, a tenant can be viewed as a subset of the cluster resources that are configured to be used by a particular group of users for an exclusive purpose. Configuring multitenancy on a Red Hat OpenShift cluster provides the following advantages:

- A reduction in CapEx and OpEx by allowing cluster resources to be shared
- Lower operational and management overhead
- Securing the workloads from cross-contamination of security breaches
- Protection of workloads from unexpected performance degradation due to resource contention

For a fully realized multitenant OpenShift cluster, quotas and restrictions must be configured for cluster resources belonging to different resource buckets: compute, storage, networking, security, and so on. Although we cover certain aspects of all the resource buckets in this solution, we focus on best practices for isolating and securing the data served or consumed by multiple workloads on the same Red Hat OpenShift cluster by configuring multitenancy on storage resources that are dynamically allocated by Astra Trident backed by NetApp ONTAP.

[Next: Architecture.](#)

Architecture

Although Red Hat OpenShift and Astra Trident backed by NetApp ONTAP do not provide isolation between workloads by default, they offer a wide range of features that can be used to configure multitenancy. To better understand designing a multitenant solution on a Red Hat OpenShift cluster with Astra Trident backed by NetApp ONTAP, let us consider an example with a set of requirements and outline the configuration around it.

Let us assume that an organization runs two of its workloads on a Red Hat OpenShift cluster as part of two projects that two different teams are working on. The data for these workloads reside on PVCs that are dynamically provisioned by Astra Trident on a NetApp ONTAP NAS backend. The organization has a requirement to design a multitenant solution for these two workloads and isolate the resources used for these projects to make sure that security and performance is maintained, primarily focused on the data that serves those applications.

The following figure depicts the multitenant solution on a Red Hat OpenShift cluster with Astra Trident backed by NetApp ONTAP.



Technology requirements

1. NetApp ONTAP storage cluster
2. Red Hat OpenShift cluster
3. Astra Trident

Red Hat OpenShift – Cluster resources

From the Red Hat OpenShift cluster point of view, the top-level resource to start with is the project. An OpenShift project can be viewed as a cluster resource that divides the whole OpenShift cluster into multiple virtual clusters. Therefore, isolation at project level provides a base for configuring multitenancy.

Next up is to configure RBAC in the cluster. The best practice is to have all the developers working on a single project or workload configured into a single user group in the Identity Provider (IdP). Red Hat OpenShift allows IdP integration and user group synchronization thus allowing the users and groups from the IdP to be imported into the cluster. This helps the cluster administrators to segregate access of the cluster resources dedicated to a project to a user group or groups working on that project, thereby restricting unauthorized access to any cluster resources. To learn more about IdP integration with Red Hat OpenShift, see the documentation [here](#).

NetApp ONTAP

It is important to isolate the shared storage serving as a persistent storage provider for a Red Hat OpenShift cluster to make sure that the volumes created on the storage for each project appear to the hosts as if they are

created on separate storage. To do this, create as many SVMs (storage virtual machines) on NetApp ONTAP as there are projects or workloads, and dedicate each SVM to a workload.

Astra Trident

After you have different SVMs for different projects created on NetApp ONTAP, you must map each SVM to a different Trident backend. The backend configuration on Trident drives the allocation of persistent storage to OpenShift cluster resources, and it requires the details of the SVM to be mapped to. This should be the protocol driver for the backend at the minimum. Optionally, it allows you to define how the volumes are provisioned on the storage and to set limits for the size of volumes or usage of aggregates and so on. Details concerning the definition of the Trident backends can be found [here](#).

Red Hat OpenShift – storage resources

After configuring the Trident backends, the next step is to configure StorageClasses. Configure as many storage classes as there are backends, providing each storage class access to spin up volumes only on one backend. We can map the StorageClass to a particular Trident backend by using the storagePools parameter while defining the storage class. The details to define a storage class can be found [here](#). Thus, there is a one-to-one mapping from StorageClass to Trident backend which points back to one SVM. This ensures that all storage claims via the StorageClass assigned to that project are served by the SVM dedicated to that project only.

Because storage classes are not namespaced resources, how do we ensure that storage claims to storage class of one project by pods in another namespace or project gets rejected? The answer is to use ResourceQuotas. ResourceQuotas are objects that control the total usage of resources per project. It can limit the number as well as the total amount of resources that can be consumed by objects in the project. Almost all the resources of a project can be limited using ResourceQuotas and using this efficiently can help organizations cut cost and outages due to overprovisioning or overconsumption of resources. Refer to the documentation [here](#) for more information.

For this use case, we need to limit the pods in a particular project from claiming storage from storage classes that are not dedicated to their project. To do that, we need to limit the persistent volume claims for other storage classes by setting `<storage-class-name>.storageclass.storage.k8s.io/persistentvolumeclaims` to 0. In addition, a cluster administrator must ensure that the developers in a project should not have access to modify the ResourceQuotas.

[Next: Configuration.](#)

Configuration

For any multitenant solution, no user can have access to more cluster resources than is required. So, the entire set of resources that are to be configured as part of the multitenancy configuration is divided between cluster-admin, storage-admin, and developers working on each project.

The following table outlines the different tasks to be performed by different users:

Role	Tasks
Cluster-admin	Create projects for different applications or workloads
	Create ClusterRoles and RoleBindings for storage-admin
	Create Roles and RoleBindings for developers assigning access to specific projects
	[Optional] Configure projects to schedule pods on specific nodes
Storage-admin	Create SVMs on NetApp ONTAP
	Create Trident backends
	Create StorageClasses
	Create storage ResourceQuotas
Developers	Validate access to create or patch PVCs or pods in assigned project
	Validate access to create or patch PVCs or pods in another project
	Validate access to view or edit Projects, ResourceQuotas, and StorageClasses

[Next: Prerequisites.](#)

Configuration

Prerequisites

- NetApp ONTAP cluster
- Red Hat OpenShift cluster
- Trident installed on the cluster
- Admin workstation with tridentctl and oc tools installed and added to \$PATH
- Admin access to ONTAP
- Cluster-admin access to OpenShift cluster
- Cluster is integrated with Identity Provider
- Identity provider is configured to efficiently distinguish between users in different teams

[Next: Cluster Administrator Tasks.](#)

Configuration: cluster-admin tasks

The following tasks are performed by the Red Hat OpenShift cluster-admin:

1. Log into Red Hat OpenShift cluster as the cluster-admin.
2. Create two projects corresponding to different projects.

```
oc create namespace project-1  
oc create namespace project-2
```

3. Create the developer role for project-1.

```
cat << EOF | oc create -f -  
apiVersion: rbac.authorization.k8s.io/v1  
kind: Role  
metadata:  
  namespace: project-1  
  name: developer-project-1  
rules:  
  - verbs:  
    - '*'  
    apiGroups:  
      - apps  
      - batch  
      - autoscaling  
      - extensions  
      - networking.k8s.io  
      - policy  
      - apps.openshift.io  
      - build.openshift.io  
      - image.openshift.io  
      - ingress.operator.openshift.io  
      - route.openshift.io  
      - snapshot.storage.k8s.io  
      - template.openshift.io  
    resources:  
      - '*'  
  - verbs:  
    - '*'  
    apiGroups:  
      - ''  
    resources:  
      - bindings  
      - configmaps  
      - endpoints  
      - events  
      - persistentvolumeclaims  
      - pods  
      - pods/log  
      - pods/attach  
      - podtemplates  
      - replicationcontrollers
```

```

- services
- limitranges
- namespaces
- componentstatuses
- nodes
- verbs:
  - '*'
apiGroups:
- trident.netapp.io
resources:
- tridentsnapshots
EOF

```



The role definition provided in this section is just an example. Developer roles must be defined based on end-user requirements.

4. Similarly, create developer roles for project-2.
5. All OpenShift and NetApp storage resources are usually managed by a storage admin. Access for storage administrators is controlled by the trident operator role that is created when Trident is installed. In addition to this, the storage admin also requires access to ResourceQuotas to control how storage is consumed.
6. Create a role for managing ResourceQuotas in all projects in the cluster to attach it to storage admin.

```

cat << EOF | oc create -f -
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: resource-quotas-role
rules:
- verbs:
  - '*'
apiGroups:
- ''
resources:
- resourcequotas
- verbs:
  - '*'
apiGroups:
- quota.openshift.io
resources:
- '*'
EOF

```

7. Make sure that the cluster is integrated with the organization's identity provider and that user groups are synchronized with cluster groups. The following example shows that the identity provider has been integrated with the cluster and synchronized with the user groups.

```
$ oc get groups
NAME                      USERS
ocp-netapp-storage-admins ocp-netapp-storage-admin
ocp-project-1              ocp-project-1-user
ocp-project-2              ocp-project-2-user
```

8. Configure ClusterRoleBindings for storage admins.

```
cat << EOF | oc create -f -
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: netapp-storage-admin-trident-operator
subjects:
- kind: Group
  apiGroup: rbac.authorization.k8s.io
  name: ocp-netapp-storage-admins
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-operator
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: netapp-storage-admin-resource-quotas-cr
subjects:
- kind: Group
  apiGroup: rbac.authorization.k8s.io
  name: ocp-netapp-storage-admins
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: resource-quotas-role
EOF
```



For storage admins, two roles must be bound: trident-operator and resource-quotas.

9. Create RoleBindings for developers binding the developer-project-1 role to the corresponding group (ocp-project-1) in project-1.

```

cat << EOF | oc create -f -
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: project-1-developer
  namespace: project-1
subjects:
- kind: Group
  apiGroup: rbac.authorization.k8s.io
  name: ocp-project-1
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: developer-project-1
EOF

```

10. Similarly, create RoleBindings for developers binding the developer roles to the corresponding user group in project-2.

[Next: Storage Administrator Tasks.](#)

Configuration: Storage-admin tasks

The following resources must be configured by a storage administrator:

1. Log into the NetApp ONTAP cluster as admin.
2. Navigate to Storage > Storage VMs and click Add. Create two SVMs, one for project-1 and the other for project-2, by providing the required details. Also create a vsadmin account to manage the SVM and its resources.

Add Storage VM

X

STORAGE VM NAME

project-1-svm

Access Protocol

SMB/CIFS, NFS

iSCSI

Enable SMB/CIFS

Enable NFS

Allow NFS client access

Add at least one rule to allow NFS clients to access volumes in this storage VM. [?](#)

EXPORT POLICY

Default

RULES

Rule Index	Clients	Access Protocols	Read-Only R...	Read/Wr...
	10.61.181.0/24	Any	Any	Any

[+ Add](#)

DEFAULT LANGUAGE [?](#)

c.utf_8



NETWORK INTERFACE

Use multiple network interfaces when client traffic is high.

K8s-Ontap-01

IP ADDRESS

SUBNET MASK

GATEWAY

BROADCAST DOMAIN

10.61.181.224

24

Add optional
gateway

Default-4



3. Log into the Red Hat OpenShift cluster as the storage administrator.

4. Create the backend for project-1 and map it to the SVM dedicated to the project. NetApp recommends using the SVM's vsadmin account to connect the backend to SVM instead of using the ONTAP cluster administrator.

```

cat << EOF | tridentctl -n trident create backend -f
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "nfs_project_1",
    "managementLIF": "172.21.224.210",
    "dataLIF": "10.61.181.224",
    "svm": "project-1-svm",
    "username": "vsadmin",
    "password": "NetApp123"
}
EOF

```



We are using the ontap-nas driver for this example. Use the appropriate driver when creating the backend based on the use case.



We assume that Trident is installed in the trident project.

5. Similarly create the Trident backend for project-2 and map it to the SVM dedicated to project-2.
6. Next, create the storage classes. Create the storage class for project-1 and configure it to use the storage pools from backend dedicated to project-1 by setting the storagePools parameter.

```

cat << EOF | oc create -f -
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: project-1-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
  storagePools: "nfs_project_1:.*"
EOF

```

7. Likewise, create a storage class for project-2 and configure it to use the storage pools from backend dedicated to project-2.
8. Create a ResourceQuota to restrict resources in project-1 requesting storage from storageclasses dedicated to other projects.

```
cat << EOF | oc create -f -
kind: ResourceQuota
apiVersion: v1
metadata:
  name: project-1-sc-rq
  namespace: project-1
spec:
  hard:
    project-2-sc.storageclass.storage.k8s.io/persistentvolumeclaims: 0
EOF
```

9. Similarly, create a ResourceQuota to restrict resources in project-2 requesting storage from storageclasses dedicated to other projects.

[Next: Validation.](#)

Validation

To validate the multitenant architecture that was configured in the previous steps, complete the following steps:

Validate access to create PVCs or pods in assigned project

1. Log in as ocp-project-1-user, developer in project-1.
2. Check access to create a new project.

```
oc create ns sub-project-1
```

3. Create a PVC in project-1 using the storageclass that is assigned to project-1.

```
cat << EOF | oc create -f -
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-pvc-project-1
  namespace: project-1
  annotations:
    trident.netapp.io/reclaimPolicy: Retain
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: project-1-sc
EOF
```

4. Check the PV associated with the PVC.

```
oc get pv
```

5. Validate that the PV and its volume is created in an SVM dedicated to project-1 on NetApp ONTAP.

```
volume show -vserver project-1-svm
```

6. Create a pod in project-1 and mount the PVC created in previous step.

```

cat << EOF | oc create -f -
kind: Pod
apiVersion: v1
metadata:
  name: test-pvc-pod
  namespace: project-1
spec:
  volumes:
    - name: test-pvc-project-1
      persistentVolumeClaim:
        claimName: test-pvc-project-1
  containers:
    - name: test-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
  volumeMounts:
    - mountPath: "/usr/share/nginx/html"
      name: test-pvc-project-1
EOF

```

7. Check if the pod is running and whether it mounted the volume.

```
oc describe pods test-pvc-pod -n project-1
```

Validate access to create PVCs or pods in another project or use resources dedicated to another project

1. Log in as ocp-project-1-user, developer in project-1.
2. Create a PVC in project-1 using the storageclass that is assigned to project-2.

```

cat << EOF | oc create -f -
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-pvc-project-1-sc-2
  namespace: project-1
  annotations:
    trident.netapp.io/reclaimPolicy: Retain
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: project-2-sc
EOF

```

3. Create a PVC in project-2.

```

cat << EOF | oc create -f -
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-pvc-project-2-sc-1
  namespace: project-2
  annotations:
    trident.netapp.io/reclaimPolicy: Retain
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: project-1-sc
EOF

```

4. Make sure that PVCs **test-pvc-project-1-sc-2** and **test-pvc-project-2-sc-1** were not created.

```

oc get pvc -n project-1
oc get pvc -n project-2

```

5. Create a pod in project-2.

```
cat << EOF | oc create -f -
kind: Pod
apiVersion: v1
metadata:
  name: test-pvc-pod
  namespace: project-1
spec:
  containers:
    - name: test-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
EOF
```

Validate access to view and edit Projects, ResourceQuotas, and StorageClasses

1. Log in as ocp-project-1-user, developer in project-1.
2. Check access to create new projects.

```
oc create ns sub-project-1
```

3. Validate access to view projects.

```
oc get ns
```

4. Check if the user can view or edit ResourceQuotas in project-1.

```
oc get resourcequotas -n project-1
oc edit resourcequotas project-1-sc-rq -n project-1
```

5. Validate that the user has access to view the storageclasses.

```
oc get sc
```

6. Check access to describe the storageclasses.
7. Validate the user's access to edit the storageclasses.

```
oc edit sc project-1-sc
```

[Next: Scaling.](#)

Scaling: Adding more projects

In a multitenant configuration, adding new projects with storage resources requires additional configuration to make sure that multitenancy is not violated. For adding more projects in a multitenant cluster, complete the following steps:

1. Log into the NetApp ONTAP cluster as a storage admin.
2. Navigate to Storage → Storage VMs and click Add. Create a new SVM dedicated to project-3. Also create a vsadmin account to manage the SVM and its resources.

Add Storage VM

X

STORAGE VM NAME

project-3-svm

Access Protocol

SMB/CIFS, NFS

iSCSI

Enable SMB/CIFS

Enable NFS

Allow NFS client access

Add at least one rule to allow NFS clients to access volumes in this storage VM. [?](#)

EXPORT POLICY

Default

RULES

Rule Index	Clients	Access Protocols	Read-Only R...	Read/Wr...
	10.61.181.0/24	Any	Any	Any

[+ Add](#)

DEFAULT LANGUAGE [?](#)

c.utf_8



NETWORK INTERFACE

Use multiple network interfaces when client traffic is high.

K8s-Ontap-01

IP ADDRESS

10.61.181.228

SUBNET MASK

24

GATEWAY

Add optional gateway

BROADCAST DOMAIN

Default-4



3. Log into the Red Hat OpenShift cluster as cluster admin.

4. Create a new project.

```
oc create ns project-3
```

5. Make sure that the user group for project-3 is created on IdP and synchronized with the OpenShift cluster.

```
oc get groups
```

6. Create the developer role for project-3.

```
cat << EOF | oc create -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: project-3
  name: developer-project-3
rules:
- verbs:
  - '*'
  apiGroups:
  - apps
  - batch
  - autoscaling
  - extensions
  - networking.k8s.io
  - policy
  - apps.openshift.io
  - build.openshift.io
  - image.openshift.io
  - ingress.operator.openshift.io
  - route.openshift.io
  - snapshot.storage.k8s.io
  - template.openshift.io
resources:
  - '*'
- verbs:
  - '*'
  apiGroups:
  - ''
resources:
  - bindings
  - configmaps
  - endpoints
  - events
  - persistentvolumeclaims
  - pods
  - pods/log
  - pods/attach
  - podtemplates
  - replicationcontrollers
  - services
```

```

- limitranges
- namespaces
- componentstatuses
- nodes
- verbs:
  - '*'
apiGroups:
- trident.netapp.io
resources:
- tridentsnapshots
EOF

```



The role definition provided in this section is just an example. The developer role must be defined based on the end-user requirements.

7. Create RoleBinding for developers in project-3 binding the developer-project-3 role to the corresponding group (ocp-project-3) in project-3.

```

cat << EOF | oc create -f -
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: project-3-developer
  namespace: project-3
subjects:
- kind: Group
  apiGroup: rbac.authorization.k8s.io
  name: ocp-project-3
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: developer-project-3
EOF

```

8. Login to the Red Hat OpenShift cluster as storage admin
9. Create a Trident backend and map it to the SVM dedicated to project-3. NetApp recommends using the SVM's vsadmin account to connect the backend to the SVM instead of using the ONTAP cluster administrator.

```

cat << EOF | tridentctl -n trident create backend -f
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "nfs_project_3",
    "managementLIF": "172.21.224.210",
    "dataLIF": "10.61.181.228",
    "svm": "project-3-svm",
    "username": "vsadmin",
    "password": "NetApp!23"
}
EOF

```



We are using the ontap-nas driver for this example. Use the appropriate driver for creating the backend based on the use-case.



We assume that Trident is installed in the trident project.

10. Create the storage class for project-3 and configure it to use the storage pools from backend dedicated to project-3.

```

cat << EOF | oc create -f -
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: project-3-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
  storagePools: "nfs_project_3:.*"
EOF

```

11. Create a ResourceQuota to restrict resources in project-3 requesting storage from storageclasses dedicated to other projects.

```

cat << EOF | oc create -f -
kind: ResourceQuota
apiVersion: v1
metadata:
  name: project-3-sc-rq
  namespace: project-3
spec:
  hard:
    project-1-sc.storageclass.storage.k8s.io/persistentvolumeclaims: 0
    project-2-sc.storageclass.storage.k8s.io/persistentvolumeclaims: 0
EOF

```

12. Patch the ResourceQuotas in other projects to restrict resources in those projects from accessing storage from the storageclass dedicated to project-3.

```

oc patch resourcequotas project-1-sc-rq -n project-1 --patch
'{"spec":{"hard":{"project-3-
sc.storageclass.storage.k8s.io/persistentvolumeclaims": 0}}}'
oc patch resourcequotas project-2-sc-rq -n project-2 --patch
'{"spec":{"hard":{"project-3-
sc.storageclass.storage.k8s.io/persistentvolumeclaims": 0}}}'

```

Red Hat OpenShift Virtualization with NetApp ONTAP

Red Hat OpenShift Virtualization with NetApp ONTAP

Depending on the specific use case, both containers and virtual machines (VMs) can serve as optimal platforms for different types of applications. Therefore, many organizations run some of their workloads on containers and some on VMs. Often, this leads organizations to face additional challenges by having to manage separate platforms: a hypervisor for VMs and a container orchestrator for applications.

To address this challenge, Red Hat introduced OpenShift Virtualization (formerly known as Container Native Virtualization) starting from OpenShift version 4.6. The OpenShift Virtualization feature enables you to run and manage virtual machines alongside containers on the same OpenShift Container Platform installation, providing hybrid management capability to automate deployment and management of VMs through operators. In addition to creating VMs in OpenShift, with OpenShift Virtualization, Red Hat also supports importing VMs from VMware vSphere, Red Hat Virtualization, and Red Hat OpenStack Platform deployments.



Certain features like live VM migration, VM disk cloning, VM snapshots and so on are also supported by OpenShift Virtualization with assistance from Astra Trident when backed by NetApp ONTAP. Examples of each of these workflows are discussed later in this document in their respective sections.

To learn more about Red Hat OpenShift Virtualization, see the documentation [here](#).

[Next: Deployment Prerequisites.](#)

Deployment

Deploy Red Hat OpenShift Virtualization with NetApp ONTAP

Prerequisites

- A Red Hat OpenShift cluster (later than version 4.6) installed on bare-metal infrastructure with RHCOS worker nodes
- The OpenShift cluster must be installed via installer provisioned infrastructure (IPI)
- Deploy Machine Health Checks to maintain HA for VMs
- A NetApp ONTAP cluster
- Astra Trident installed on the OpenShift cluster
- A Trident backend configured with an SVM on ONTAP cluster
- A StorageClass configured on the OpenShift cluster with Astra Trident as the provisioner
- Cluster-admin access to Red Hat OpenShift cluster
- Admin access to NetApp ONTAP cluster
- An admin workstation with tridentctl and oc tools installed and added to \$PATH

Because OpenShift Virtualization is managed by an operator installed on the OpenShift cluster, it imposes additional overhead on memory, CPU, and storage, which must be accounted for while planning the hardware requirements for the cluster. See the documentation [here](#) for more details.

Optionally, you can also specify a subset of the OpenShift cluster nodes to host the OpenShift Virtualization operators, controllers, and VMs by configuring node placement rules. To configure node placement rules for OpenShift Virtualization, follow the documentation [here](#).

For the storage backing OpenShift Virtualization, NetApp recommends having a dedicated StorageClass that requests storage from a particular Trident backend, which in turn is backed by a dedicated SVM. This maintains a level of multitenancy with regard to the data being served for VM-based workloads on the OpenShift cluster.

Next: Deploy via operator.

Deploy Red Hat OpenShift Virtualization with NetApp ONTAP

To install OpenShift Virtualization, complete the following steps:

1. Log into the Red Hat OpenShift bare-metal cluster with cluster-admin access.
2. Select Administrator from the Perspective drop down.
3. Navigate to Operators > OperatorHub and search for OpenShift Virtualization.



4. Select the OpenShift Virtualization tile and click Install.



OpenShift Virtualization

2.6.2 provided by Red Hat



Install

Latest version

2.6.2

Capability level

- Basic Install
- Seamless Upgrades
- Full Lifecycle
- Deep Insights
- Auto Pilot

Provider type

Red Hat

Provider

Red Hat

Requirements

Your cluster must be installed on bare metal infrastructure with Red Hat Enterprise Linux CoreOS workers.

Details

OpenShift Virtualization extends Red Hat OpenShift Container Platform, allowing you to host and manage virtualized workloads on the same platform as container-based workloads. From the OpenShift Container Platform web console, you can import a VMware virtual machine from vSphere, create new or clone existing VMs, perform live migrations between nodes, and more. You can use OpenShift Virtualization to manage both Linux and Windows VMs.

The technology behind OpenShift Virtualization is developed in the [KubeVirt](#) open source community. The KubeVirt project extends [Kubernetes](#) by adding additional virtualization resource types through [Custom Resource Definitions](#) (CRDs). Administrators can use Custom Resource Definitions to manage [VirtualMachine](#) resources alongside all other resources that Kubernetes provides.

- On the Install Operator screen, leave all default parameters and click **Install**.

Update channel *

- 2.1
- 2.2
- 2.3
- 2.4
- stable



OpenShift Virtualization
provided by Red Hat

Provided APIs

OpenShift Virtualization Deployment Required

Represents the deployment of OpenShift Virtualization

Installation mode *

- All namespaces on the cluster (default)
This mode is not supported by this Operator
- A specific namespace on the cluster
Operator will be available in a single Namespace only.

Installed Namespace *

- Operator recommended Namespace: **openshift-cnv**

Namespace creation

Namespace **openshift-cnv** does not exist and will be created.

- Select a Namespace

Approval strategy *

- Automatic
- Manual

Install

Cancel

6. Wait for the operator installation to complete.

The screenshot shows the OpenShift Virtualization operator page. At the top, there's a red circular icon with a white 'K8s' logo. Next to it, the text 'OpenShift Virtualization' and '2.6.2 provided by Red Hat' is displayed. A progress bar is partially visible below this section.

Installing Operator

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

[View installed Operators in Namespace openshift-cnv](#)

7. After the operator has installed, click Create HyperConverged.

The screenshot shows the same OpenShift Virtualization operator page as before, but now with a large green circular icon containing a white checkmark to the right of the status text. The rest of the interface remains the same.

Installed operator - operand required

The Operator has installed successfully. Create the required custom resource to be able to use this Operator.

This screenshot shows the details of the installed 'HyperConverged' operator. It includes a blue 'HC' icon, the name 'HyperConverged', a red exclamation mark icon with the word 'Required', and a description: 'Creates and maintains an OpenShift Virtualization Deployment'. A large green checkmark icon is positioned to the right of the status text in the previous screenshot.

[Create HyperConverged](#)

[View installed Operators in Namespace openshift-cnv](#)

8. On the Create HyperConverged screen, click Create, accepting all default parameters. This step starts the installation of OpenShift Virtualization.

Name *

Labels

Infra

infra HyperConvergedConfig influences the pod configuration (currently only placement) for all the infra components needed on the virtualization enabled cluster but not necessarily directly on each node running VMs/VMIs.

Workloads

workloads HyperConvergedConfig influences the pod configuration (currently only placement) of components which need to be running on a node where virtualization workloads should be able to run. Changes to Workloads HyperConvergedConfig can be applied only without existing workload.

Bare Metal Platform

true

BareMetalPlatform indicates whether the infrastructure is baremetal.

Feature Gates

featureGates is a map of feature gate flags. Setting a flag to `true` will enable the feature. Setting `false` or removing the feature gate, disables the feature.

Local Storage Class Name

LocalStorageClassName the name of the local storage class.

Create **Cancel**

- After all the pods move to the Running state in the openshift-cnv namespace and the OpenShift Virtualization operator is in the Succeeded state, the operator is ready to use. VMs can now be created on the OpenShift cluster.

Project: openshift-cnv ▾

Installed Operators

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

Name	Managed Namespaces	Status	Last updated	Provided APIs
 OpenShift Virtualization 2.6.2 provided by Red Hat	 openshift-cnv	✓ Succeeded Up to date	May 18, 8:02 pm	OpenShift Virtualization Deployment HostPathProvisioner deployment

[Next: Workflows: Create VM.](#)

Workflows

Workflows: Red Hat OpenShift Virtualization with NetApp ONTAP

Create VM

VMs are stateful deployments that require volumes to host the operating system and data. With CNV, because the VMs are run as pods, the VMs are backed by PVs hosted on NetApp ONTAP through Trident. These volumes are attached as disks and store the entire filesystem including the boot source of the VM.



To create a virtual machine on the OpenShift cluster, complete the following steps:

1. Navigate to Workloads > Virtualization > Virtual Machines and click Create > With Wizard.
2. Select the desired the operating system and click Next.
3. If the selected operating system has no boot source configured, you must configure it. For Boot Source, select whether you want to import the OS image from an URL or from a registry and provide the corresponding details. Expand Advanced and select the Trident-backed StorageClass. Then click Next.

Boot source

This template does not have a boot source. Provide a custom boot source for this **CentOS 8.0+** VM virtual machine.

Boot source type *

Import via URL (creates PVC)

Import URL *

<https://access.cdn.redhat.com/content/origin/files/sha256/58/588167f828001e57688ec4b9b31c11a59d532489f527488ebc89ac5e952...>

Example: For RHEL, visit the [RHEL download page](#) (requires login) and copy the download link URL of the KVM guest image

Mount this as a CD-ROM boot source ?

Persistent Volume Claim size *

5 GiB ▾

Ensure your PVC size covers the requirements of the uncompressed image and any other space requirements. More storage can be added later.

Advanced

Storage class *

basic (default)

Access mode *

Single User (RWO)

Volume mode *

Filesystem

4. If the selected operating system already has a boot source configured, the previous step can be skipped.
5. In the Review and Create pane, select the project you want to create the VM in and furnish the VM details. Make sure that the boot source is selected to be Clone and boot from CD-ROM with the appropriate PVC assigned for the selected OS.

1 Select template

2 Review and create

Review and create

You are creating a virtual machine from the Red Hat Enterprise Linux 8.0+ VM template.

Project *

PR default

Virtual Machine Name * ⓘ

rhel8-light-bat

Flavor *

Small: 1 CPU | 2 GiB Memory

Storage	Workload profile ⓘ
40 GiB	server

Boot source

Clone and boot from CD-ROM

PVC rhel8

ⓘ A new disk has been added to support the CD-ROM boot source. Edit this disk by customizing the virtual machine.

▼ Disk details

rootdisk-install - Blank - 20GiB - virtio - default Storage class

Start this virtual machine after creation

Create virtual machine [Customize virtual machine](#) [Back](#) [Cancel](#)

6. If you wish to customize the virtual machine, click Customize Virtual Machine and modify the required parameters.
7. Click Create Virtual Machine to create the virtual machine; this spins up a corresponding pod in the background.

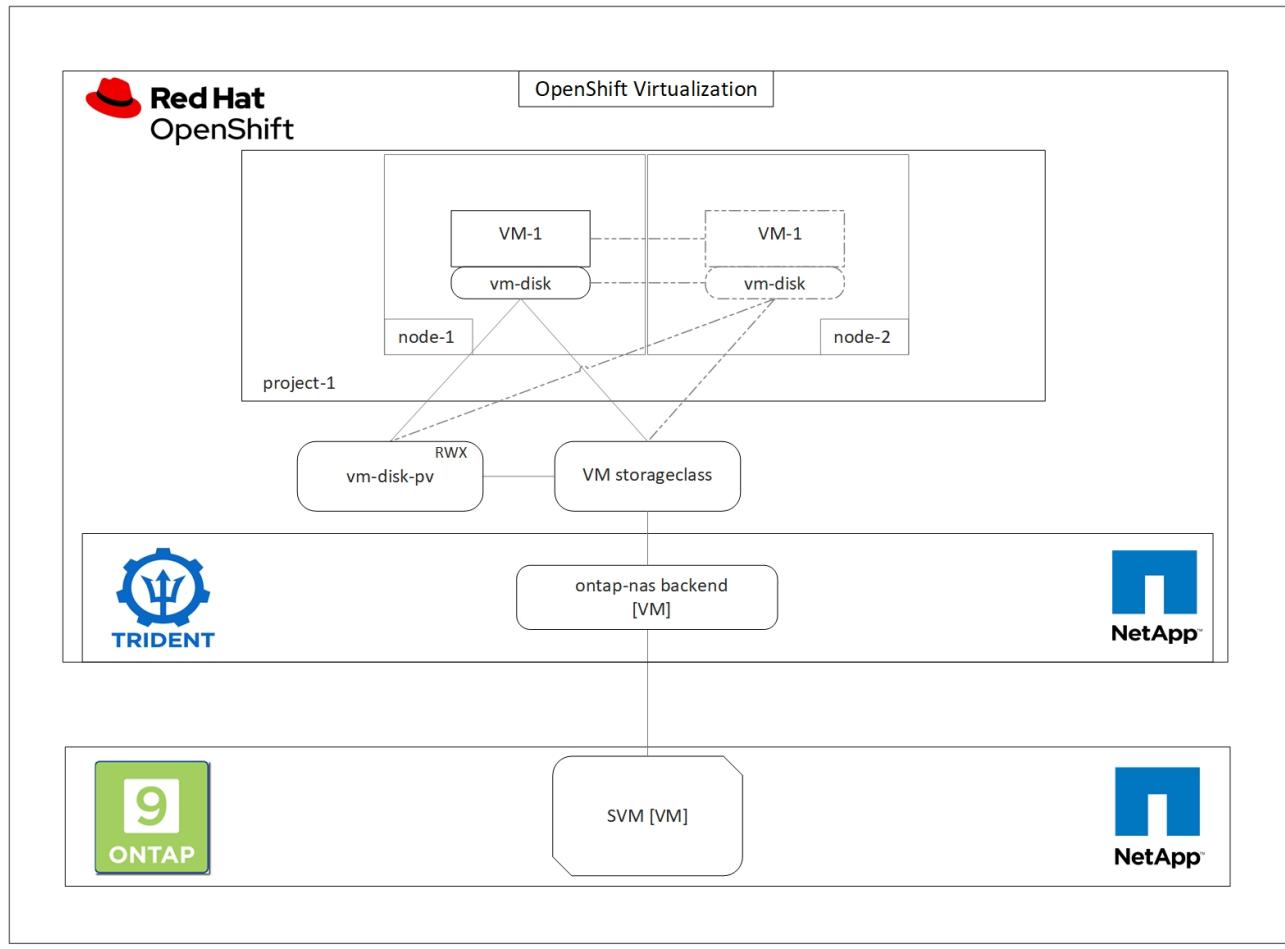
When a boot source is configured for a template or an operating system from an URL or from a registry, it creates a PVC in the `openshift-virtualization-os-images` project and downloads the KVM guest image to the PVC. You must make sure that template PVCs have enough provisioned space to accommodate the KVM guest image for the corresponding OS. These PVCs are then cloned and attached as rootdisks to virtual machines when they are created using the respective templates in any project.

[Next: Workflows: VM Live Migration.](#)

Workflows: Red Hat OpenShift Virtualization with NetApp ONTAP

VM Live Migration

Live Migration is a process of migrating a VM instance from one node to another in an OpenShift cluster with no downtime. For live migration to work in an OpenShift cluster, VMs must be bound to PVCs with shared ReadWriteMany access mode. Astra Trident backend configured with an SVM on a NetApp ONTAP cluster that is enabled for NFS protocol supports shared ReadWriteMany access for PVCs. Therefore, the VMs with PVCs that are requested from StorageClasses provisioned by Trident from NFS-enabled SVM can be migrated with no downtime.



To create a VM bound to PVCs with shared ReadWriteMany access:

1. Navigate to Workloads > Virtualization > Virtual Machines and click Create > With Wizard.
2. Select the desired the operating system and click Next. Let us assume the selected OS already had a boot source configured with it.
3. In the Review and Create pane, select the project you want to create the VM in and furnish the VM details. Make sure that the boot source is selected to be Clone and boot from CD-ROM with the appropriate PVC assigned for the selected OS.
4. Click Customize Virtual Machine and then click Storage.
5. Click the ellipsis next to rootdisk, and make sure that the storageclass provisioned using Trident is selected. Expand Advanced and select Shared Access (RWX) for Access Mode. Then click Save.

Edit Disk

Type: Disk

Interface *

virtio

Storage Class

basic (default)

Advanced

Volume Mode

Filesystem

Volume Mode is set by Source PVC

Access Mode

Shared Access (RWX) - Not recommended for basic storage class

Info Access and Volume modes should follow storage feature matrix
[Learn more ↗](#)

Cancel Save

6. Click Review and confirm and then click Create Virtual Machine.

To manually migrate a VM to another node in the OpenShift cluster, complete the following steps.

1. Navigate to Workloads > Virtualization > Virtual Machines.

2. For the VM you wish to migrate, click the ellipsis, and then click Migrate the Virtual Machine.

3. Click Migrate when the message pops up to confirm.



A VM instance in an OpenShift cluster automatically migrates to another node when the original node is placed into maintenance mode if the evictionStrategy is set to LiveMigrate.

[Next: Workflows: VM Cloning.](#)

Workflows: Red Hat OpenShift Virtualization with NetApp ONTAP

VM cloning

Cloning an existing VM in OpenShift is achieved with the support of Astra Trident's Volume CSI cloning feature. CSI volume cloning allows for creation of a new PVC using an existing PVC as the data source by duplicating its PV. After the new PVC is created, it functions as a separate entity and without any link to or dependency on the source PVC.



There are certain restrictions with CSI volume cloning to consider:

1. Source PVC and destination PVC must be in the same project.
2. Cloning is supported within the same storage class.
3. Cloning can be performed only when source and destination volumes use the same VolumeMode setting;

for example, a block volume can only be cloned to another block volume.

VMs in an OpenShift cluster can be cloned in two ways:

1. By shutting down the source VM
2. By keeping the source VM live

By Shutting down the source VM

Cloning an existing VM by shutting down the VM is a native OpenShift feature that is implemented with support from Astra Trident. Complete the following steps to clone a VM.

1. Navigate to Workloads > Virtualization > Virtual Machines and click the ellipsis next to the virtual machine you wish to clone.
2. Click Clone Virtual Machine and provide the details for the new VM.

Clone Virtual Machine

Name *

Description

Namespace *

Start virtual machine on clone

Configuration

Operating System	Red Hat Enterprise Linux 8.0 or higher
Flavor	Small: 1 CPU 2 GiB Memory
Workload Profile	server
NICs	default - virtio
Disks	cloudinitdisk - cloud-init disk rootdisk - 20Gi - basic



The VM rhel8-short-frog is still running. It will be powered off while cloning.

[Cancel](#)

[Clone Virtual Machine](#)

3. Click Clone Virtual Machine; this shuts down the source VM and initiates the creation of the clone VM.
4. After this step is completed, you can access and verify the content of the cloned VM.

By keeping the source VM live

An existing VM can also be cloned by cloning the existing PVC of the source VM and then creating a new VM using the cloned PVC. This method does not require you to shut down the source VM. Complete the following steps to clone a VM without shutting it down.

1. Navigate to Storage > PersistentVolumeClaims and click the ellipsis next to the PVC that is attached to the source VM.
2. Click Clone PVC and furnish the details for the new PVC.

Clone

Name *

rhel8-short-frog-rootdisk-28dvb-clone

Access Mode *

Single User (RWO) Shared Access (RWX) Read Only (ROX)

Size *

20

GiB



PVC details

Namespace	Requested capacity	Access mode
NS default	20 GiB	Shared Access (RWX)
Storage Class	Used capacity	Volume mode
SC basic	2.2 GiB	Filesystem

Cancel

Clone

3. Then click Clone. This creates a PVC for the new VM.
4. Navigate to Workloads > Virtualization > Virtual Machines and click Create > With YAML.
5. In the spec > template > spec > volumes section, attach the cloned PVC instead of the container disk. Provide all other details for the new VM according to your requirements.

```
- name: rootdisk
  persistentVolumeClaim:
    claimName: rhel8-short-frog-rootdisk-28dwb-clone
```

6. Click Create to create the new VM.
7. After the VM is created successfully, access and verify that the new VM is a clone of the source VM.

Next: [Workflows: Create VM from a Snapshot](#).

Workflows: Red Hat OpenShift Virtualization with NetApp ONTAP

Create VM from a Snapshot

With Astra Trident and Red Hat OpenShift, users can take a snapshot of a persistent volume on Storage Classes provisioned by it. With this feature, users can take a point-in-time copy of a volume and use it to create a new volume or restore the same volume back to a previous state. This enables or supports a variety of use-cases, from rollback to clones to data restore.

For Snapshot operations in OpenShift, the resources VolumeSnapshotClass, VolumeSnapshot, and VolumeSnapshotContent must be defined.

- A VolumeSnapshotContent is the actual snapshot taken from a volume in the cluster. It is cluster-wide resource analogous to PersistentVolume for storage.
- A VolumeSnapshot is a request for creating the snapshot of a volume. It is analogous to a PersistentVolumeClaim.
- VolumeSnapshotClass lets the administrator specify different attributes for a VolumeSnapshot. It allows you to have different attributes for different snapshots taken from the same volume.



To create Snapshot of a VM, complete the following steps:

1. Create a VolumeSnapshotClass that can then be used to create a VolumeSnapshot. Navigate to Storage > VolumeSnapshotClasses and click Create VolumeSnapshotClass.
2. Enter the name of the Snapshot Class, enter csi.trident.netapp.io for the driver, and click Create.

```
1 apiVersion: snapshot.storage.k8s.io/v1
2 kind: VolumeSnapshotClass
3 metadata:
4   name: trident-snapshot-class
5 driver: csi.trident.netapp.io
6 deletionPolicy: Delete
7
```

[Create](#)[Cancel](#) [Download](#)

3. Identify the PVC that is attached to the source VM and then create a Snapshot of that PVC. Navigate to Storage > VolumeSnapshots and click Create VolumeSnapshots.
4. Select the PVC that you want to create the Snapshot for, enter the name of the Snapshot or accept the default, and select the appropriate VolumeSnapshotClass. Then click Create.

Create VolumeSnapshot

[Edit YAML](#)**PersistentVolumeClaim *****PVC** rhel8-short-frog-rootdisk-28dvh**Name ***

rhel8-short-frog-rootdisk-28dvh-snapshot

Snapshot Class ***VSC** trident-snapshot-class[Create](#)[Cancel](#)

5. This creates the snapshot of the PVC at that point in time.

Create a new VM from the snapshot

1. First, restore the Snapshot into a new PVC. Navigate to Storage > VolumeSnapshots, click the ellipsis next to the Snapshot that you wish to restore, and click Restore as new PVC.
2. Enter the details of the new PVC and click Restore. This creates a new PVC.

Restore as new PVC

When restore action for snapshot **rhel8-short-frog-rootdisk-28dvb-snapshot** is finished a new crash-consistent PVC copy will be created.

Name *

rhel8-short-frog-rootdisk-28dvb-snapshot-restore

Storage Class *

SC basic

Access Mode *

Single User (RWO) Shared Access (RWX) Read Only (ROX)

Size *

20

GiB



VolumeSnapshot details

Created at

May 21, 12:46 am

Namespace

default

Status

Ready

API version

snapshot.storage.k8s.io/v1

Size

20 GiB

3. Next, create a new VM from this PVC. Navigate to Workloads > Virtualization > Virtual Machines and click Create > With YAML.
4. In the spec > template > spec > volumes section, specify the new PVC created from Snapshot instead of

from the container disk. Provide all other details for the new VM according to your requirements.

```
- name: rootdisk
  persistentVolumeClaim:
    claimName: rhel8-short-frog-rootdisk-28dvb-snapshot-restore
```

5. Click Create to create the new VM.
6. After the VM is created successfully, access and verify that the new VM has the same state as that of the VM whose PVC was used to create the snapshot at the time when the snapshot was created.

Advanced Cluster Management for Kubernetes on Red Hat OpenShift with NetApp

Advanced Cluster Management for Kubernetes: Red Hat OpenShift with NetApp

As a containerized application transitions from development to production, many organizations require multiple Red Hat OpenShift clusters to support the testing and deployment of that application. In conjunction with this, organizations usually host multiple applications or workloads on OpenShift clusters. Therefore, each organization ends up managing a set of clusters, and OpenShift administrators must thus face the added challenge of managing and maintaining multiple clusters across a range of environments that span multiple on-premises data centers and public clouds. To address these challenges, Red Hat introduced Advanced Cluster Management for Kubernetes.

Red Hat Advanced Cluster Management for Kubernetes enables you to perform the following tasks:

1. Create, import, and manage multiple clusters across data centers and public clouds
2. Deploy and manage applications or workloads on multiple clusters from a single console
3. Monitor and analyze health and status of different cluster resources
4. Monitor and enforce security compliance across multiple clusters

Red Hat Advanced Cluster Management for Kubernetes is installed as an add-on to a Red Hat OpenShift cluster, and it uses this cluster as a central controller for all its operations. This cluster is known as hub cluster, and it exposes a management plane for the users to connect to Advanced Cluster Management. All the other OpenShift clusters that are either imported or created via the Advanced Cluster Management console are managed by the hub cluster and are called managed clusters. It installs an agent called Klusterlet on the managed clusters to connect them to the hub cluster and serve the requests for different activities related to cluster lifecycle management, application lifecycle management, observability, and security compliance.



For more information, see the documentation [here](#).

Next: [Deployment Prerequisites](#).

Deployment

Deploy Advanced Cluster Management for Kubernetes

Prerequisites

1. A Red Hat OpenShift cluster (greater than version 4.5) for the hub cluster
2. Red Hat OpenShift clusters (greater than version 4.4.3) for managed clusters
3. Cluster-admin access to the Red Hat OpenShift cluster
4. A Red Hat subscription for Advanced Cluster Management for Kubernetes

Advanced Cluster Management is an add-on on for the OpenShift cluster, so there are certain requirements and restrictions on the hardware resources based on the features used across the hub and managed clusters. You need to take these issues into account when sizing the clusters. See the documentation [here](#) for more details.

Optionally, if the hub cluster has dedicated nodes for hosting infrastructure components and you would like to install Advanced Cluster Management resources only on those nodes, you need to add tolerations and selectors to those nodes accordingly. For more details, see the documentation [here](#).

Next: [Installation](#).

Deploy Advanced Cluster Management for Kubernetes

To install Advanced Cluster Management for Kubernetes on an OpenShift cluster, complete the following steps:

1. Choose an OpenShift cluster as the hub cluster and log into it with cluster-admin privileges.
2. Navigate to Operators > Operators Hub and search for Advanced Cluster Management for Kubernetes.

The screenshot shows the Red Hat OpenShift OperatorHub. The left sidebar is titled 'Administrator' and has sections for Home, Projects, Search, Explore, Events, Operators, OperatorHub (which is selected), and Installed Operators. The main area is titled 'Project: default' and shows 'All Items'. A search bar says 'Filter by keyword...'. There are 450 items. The 'Community' section displays three operators:

- 3scale API Management** provided by Red Hat
- Advanced Cluster Management for Kubernetes** provided by Red Hat
- Akka Cluster Operator** provided by Lightbend, Inc.

3. Select Advanced Cluster Management for Kubernetes and click Install.

Advanced Cluster Management for Kubernetes
2.2.3 provided by Red Hat

Install

Latest version	Red Hat Advanced Cluster Management for Kubernetes provides the multicluster hub, a central management console for managing multiple Kubernetes-based clusters across data centers, public clouds, and private clouds. You can use the hub to create Red Hat OpenShift Container Platform clusters on selected providers, or import existing Kubernetes-based clusters. After the clusters are managed, you can set compliance requirements to ensure that the clusters maintain the specified security requirements. You can also deploy business applications across your clusters.
2.2.3	
Capability level	<input checked="" type="checkbox"/> Basic Install <input checked="" type="checkbox"/> Seamless Upgrades <input type="radio"/> Full Lifecycle <input type="radio"/> Deep Insights <input type="radio"/> Auto Pilot
Provider type	Red Hat Advanced Cluster Management for Kubernetes also provides the following operators:
Red Hat	<ul style="list-style-type: none"> Multicluster subscriptions: An operator that provides application management capabilities including subscribing to resources from a channel and deploying those resources on MCH-managed Kubernetes clusters based on placement rules. Hive for Red Hat OpenShift: An operator that provides APIs for provisioning and performing initial configuration of OpenShift clusters. These operators are used by the multicluster hub to provide its provisioning and application-management capabilities.
Provider	Red Hat
Infrastructure features	How to Install
Disconnected	Use of this Red Hat product requires a licensing and subscription agreement.

4. On the Install Operator screen, provide the necessary details (NetApp recommends retaining the default parameters) and click Install.

Install Operator

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic updates.

Update channel *

- release-2.0
- release-2.1
- release-2.2

Installation mode *

- All namespaces on the cluster (default)
This mode is not supported by this Operator
- A specific namespace on the cluster
Operator will be available in a single Namespace only.

Installed Namespace *

- Operator recommended Namespace: **PR open-cluster-management**

 Namespace creation

Namespace **open-cluster-management** does not exist and will be created.

- Select a Namespace

Approval strategy *

- Automatic
- Manual

Install

Cancel

5. Wait for the operator installation to complete.



The screenshot shows a progress card for the 'Advanced Cluster Management for Kubernetes' operator. At the top, there's a red circular icon with a white virus-like symbol. To its right, the text reads 'Advanced Cluster Management for Kubernetes' and '2.2.3 provided by Red Hat'. Below this, a large box contains the heading 'Installing Operator' in bold. Underneath, a message says 'The Operator is being installed. This may take a few minutes.' followed by a blue link 'View installed Operators in Namespace open-cluster-management'.

6. After the operator is installed, click Create MultiClusterHub.



Advanced Cluster Management for Kubernetes

2.2.3 provided by Red Hat



Installed operator - operand required

The Operator has installed successfully. Create the required custom resource to be able to use this Operator.

MCH MultiClusterHub ! Required

Advanced provisioning and management of OpenShift and Kubernetes clusters

[Create MultiClusterHub](#)

[View installed Operators in Namespace open-cluster-management](#)

- On the Create MultiClusterHub screen, click Create after furnishing the details. This initiates the installation of a multi-cluster hub.

Project: open-cluster-management ▾

Advanced Cluster Management for Kubernetes > Create MultiClusterHub

Create MultiClusterHub

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

Configure via: Form view YAML view

i Note: Some fields may not be represented in this form view. Please select "YAML view" for full control.



MultiClusterHub

provided by Red Hat

MultiClusterHub defines the configuration for an instance of the MultiCluster Hub

Name *

multicluscherhub

Labels

app=frontend

» Advanced configuration

[Create](#)

[Cancel](#)

- After all the pods move to the Running state in the open-cluster-management namespace and the operator moves to the Succeeded state, Advanced Cluster Management for Kubernetes is installed.

Installed Operators

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

Name	Managed Namespaces	Status	Provided APIs
 Advanced Cluster Management for Kubernetes 2.2.3 provided by Red Hat	NS open-cluster-management	Succeeded Up to date	MultiClusterHub ClusterManager ClusterDeployment ClusterState View 25 more...

9. It takes some time to complete the hub installation, and, after it is done, the MultiCluster hub moves to Running state.

Installed Operators > Operator details

 Advanced Cluster Management for Kubernetes
2.2.3 provided by Red Hat

Actions ▾

Details	YAML	Subscription	Events	All instances	MultiClusterHub	ClusterManager	ClusterDeployment	ClusterSt...

MultiClusterHubs

Create MultiClusterHub

Name	Kind	Status	Labels
MCH multiclusterhub	MultiClusterHub	Phase: Running	No labels

10. It creates a route in the open-cluster-management namespace. Connect to the URL in the route to access the Advanced Cluster Management console.

Project: open-cluster-management ▾

Routes

Create Route

Filter ▾ Name mul

Name mul Clear all filters

Name	Status	Location	Service
RT multicloud-console	Accepted	https://multicloud-console.apps.ocp-vmware2.cie.netapp.com	S management-ingress

Next: Features - Cluster Lifecycle Management.

Features

Features: Advanced Cluster Management for Kubernetes on Red Hat OpenShift with NetApp

Cluster Lifecycle Management

To manage different OpenShift clusters, you can either create or import them into Advanced Cluster Management.

1. First navigate to Automate Infrastructures > Clusters.
2. To create a new OpenShift cluster, complete the following steps:
 - a. Create a provider connection: Navigate to Provider Connections and click Add a Connection, provide all the details corresponding to the selected provider type and click Add.

Select a provider and enter basic information

Provider * ②

aws Amazon Web Services

Connection name * ②

nik-hcl-aws

Namespace * ②

default

Configure your provider connection

Base DNS domain ②

cie.netapp.com

AWS access key ID * ②

AKIATCFBZDOIASDSAH

AWS secret access key * ②

.....

Red Hat OpenShift pull secret * ②

```
FuS3pNbktVaHplNFc2MkZsbmtBVGN6TktmUlZXcHcxOW9teEZwQ0lYZ1d3cjJobGxJeDBQNOxlZE0yeGM5O0ZwZk5RR2JUanlxNnNUM21Rb0FJbUFjNCIBYlpEWVZE0HitNxkTMDZPUVpoWFRHcGwtREIDQ2RSYIJRaTlxblDLT2oyQ3pVeUJfNllwcENSa2YyOsylWZGSFVfNA=,"email":"Nikhil.kulkarni@netapp.com"}, "registry.redhat.io":
```

SSH private key * ②

-----BEGIN OPENSSH PRIVATE KEY-----

```
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbasdadssadadm9uZQAAAAAAAAAABAAAAMwAAAAtzc2gtZWQyNTUxOQAAACCLcwLgAvSIHAEp+DevIRNzaG2zkNreMIZ/UHyf0UWvAAAAAJh/wa6xf8Gu
```

SSH public key * ②

```
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAltzAuAC746agdh2lcB4/4N6/VE3NobbOQ2t4zVn9QfJ/RRa8A root@nik-rhel8
```

- b. To create a new cluster, navigate to Clusters and click Add a Cluster > Create a Cluster. Provide the details for the cluster and the corresponding provider and click Create.

Configuration

Cluster name * ⓘ

Distribution

Select the type of Kubernetes distribution to use for your cluster.

Red Hat OpenShift

Select an infrastructure provider to host your Red Hat OpenShift cluster.

 AWS Amazon Web Services	 Google Cloud	 Microsoft Azure
 VMware vSphere	 Bare Metal	

Release image * ⓘ

Provider connection * ⓘ

Add a connection

- c. After the cluster is created, it appears in the cluster list with the status Ready.
3. To import an existing cluster, complete the following steps:
 - a. Navigate to Clusters and click Add a Cluster > Import an Existing Cluster.
 - b. Enter the name of the cluster and click Save Import and Generate Code. A command to add the existing cluster is displayed.
 - c. Click Copy Command and run the command on the cluster to be added to the hub cluster. This initiates the installation of the necessary agents on the cluster, and, after this process is complete, the cluster appears in the cluster list with status Ready.

Name *

Additional labels

Once you click on "Save import and generate code", the information you entered will be used to generate the code and cannot be modified anymore. If you wish to change any information, you will have to delete and re-import this cluster.

Code generated successfully Import saved

Run a command

1. Copy this command
Click the button to have the command automatically copied to your clipboard.
[Copy command](#)

2. Run this command with kubectl configured for your targeted cluster to start the import
Log in to the existing cluster in your terminal and run the command.

[View cluster](#)

[Import another](#)

4. After you create and import multiple clusters, you can monitor and manage them from a single console.

[Next: Features - Application Lifecycle Management.](#)

Features: Advanced Cluster Management for Kubernetes on Red Hat OpenShift with NetApp

Application lifecycle management

To create an application and manage it across a set of clusters,

1. Navigate to Manage Applications from the sidebar and click Create Application. Provide the details of the application you would like to create and click Save.

Applications /

Create an application

YAML: Off

[Cancel](#)

[Save](#)

Name* ⓘ

Namespace* ⓘ

Repository location for resources

Repository types

Select the type of repository where resources that you want to deploy are located

Git

URL* ⓘ

Branch ⓘ

Path ⓘ

- After the application components are installed, the application appears in the list.

Applications

Overview

Advanced configuration

⟳ Refresh every 15s

Last update: 7:36:23 PM

[Create application](#)

Search

Name	Namespace	Clusters	Resource	Time window	Created	⋮
demo-app	default	Local	Git		8 days ago	⋮

1-1 of 1 << < 1 of 1 > >>

- The application can now be monitored and managed from the console.

Next: [Features - governance and risk](#).

Features: Advanced Cluster Management for Kubernetes on Red Hat OpenShift with NetApp

Governance and risk

This feature allows you to define the compliance policies for different clusters and make sure that the clusters adhere to it. You can configure the policies to either inform or remediate any deviations or violations of the rules.

1. Navigate to Governance and Risk from the sidebar.
2. To create compliance policies, click Create Policy, enter the details of the policy standards, and select the clusters that should adhere to this policy. If you want to automatically remediate the violations of this policy, select the checkbox Enforce if Supported and click Create.

Create policy i



YAML: Off

Name *

policy-complianceoperator

Namespace * i

default

Specifications * i

1x ComplianceOperator

Cluster selector i

1x local-cluster: "true"

Standards i

1x NIST-CSF

Categories i

1x PR.IP Information Protection Processes and Procedures

Controls i

1x PR.IP-1 Baseline Configuration

 Enforce if supported i Disable policy i

3. After all the required policies are configured, any policy or cluster violations can be monitored and remediated from Advanced Cluster Management.

Governance and risk ⓘ

Filter

Refresh every 10s

Last update: 12:54:01 PM

Create policy

Summary 1

Standards ▾

NIST-CSF



No violations found

Based on the industry standards, there are no cluster or policy violations.

Policies

Cluster violations

Find policies

Policy name	Namespace	Remediation	Cluster violations	Standards	Categories	Controls	Created	⋮
policy-complianceoperator	default	inform	0/1	NIST-CSF	PR.IP Information Protection Processes and Procedures	PR.IP-1 Baseline Configuration	32 minutes ago	⋮

1 - 1 of 1 ▾

« «

1

of 1

» »

Next: Features - Observability.

Features: Advanced Cluster Management for Kubernetes on Red Hat OpenShift with NetApp

Observability

Advanced Cluster Management for Kubernetes provides a way to monitor the nodes, pods, and applications, and workloads across all the clusters.

1. Navigate to Observe Environments > Overview.

Overview

[+ Add provider connection](#)
⟳ Refresh every 1m
Last update 12:35:03 AM


2. All pods and workloads across all clusters are monitored and sorted based on a variety of filters. Click Pods to view the corresponding data.

Search

[Saved searches](#) ▼ | [Open new search tab](#)

3 Related cluster	673 Related secret	20 Related node	8 Related persistentvolumeclaim
8 Related persistentvolume	1 Related provisioning	2 Related searchcollector	3 Related iampolicycontroller

[Show all \(38\)](#)

▼ Pod (1135)

Name	Value	⋮
Name	14bbd46d68f3ddd50b9328cee6854a36807ef784dac2bded9cc20638fbpd582	⋮
Namespace	openshift-marketplace	⋮
Cluster	local-cluster	⋮
Status	Completed	⋮
Restarts	0	⋮
Host IP	10.61.186.27	⋮
Pod IP	10.129.2.215	⋮
Created	4 days ago	⋮
Labels	controller-uid=dd259738-2cce-40e2-85d3-6ccf56904ba8	⋮

3. All nodes across the clusters are monitored and analyzed based on a variety of data points. Click Nodes to get more insight into the corresponding details.

Search

Saved searches | Open new search tab

3 Related cluster	1k Related pod	12 Related service
-------------------	----------------	--------------------

Show all (3)

▼ Node (20)

Name	Cluster	Role	Architecture	OS image	CPU	Created	Labels
ocp-master-1-ocp-bare-metal.cie.netapp.com	ocp-bare-metal	master; worker	amd64	Red Hat Enterprise Linux CoreOS 47.83.202103292105-0 (Octpa)	48	a month ago	beta.kubernetes.io/arch=amd64 beta.kubernetes.io/os=linux kubernetes.io/arch=amd64 5 more
ocp-master-2-ocp-bare-metal.cie.netapp.com	ocp-bare-metal	master; worker	amd64	Red Hat Enterprise Linux CoreOS 47.83.202103292105-0 (Octpa)	48	a month ago	beta.kubernetes.io/arch=amd64 beta.kubernetes.io/os=linux kubernetes.io/arch=amd64 5 more
ocp-master-3-ocp-bare-metal.cie.netapp.com	ocp-bare-metal	master; worker	amd64	Red Hat Enterprise Linux CoreOS 47.83.202103292105-0 (Octpa)	48	a month ago	beta.kubernetes.io/arch=amd64 beta.kubernetes.io/os=linux kubernetes.io/arch=amd64 5 more

4. All clusters are monitored and organized based on different cluster resources and parameters. Click Clusters to view cluster details.

Search

Saved searches | Open new search tab

3k Related secret	787 Related pod	15 Related persistentvolumeclaim	17 Related node	1 Related application
15 Related persistentvolume	1 Related searchcollector	8 Related clusterclaim	3 Related resourcequota	5 Related identity

Show all (159)

▼ Cluster (2)

Name	Available	Hub accepted	Joined	Nodes	Kubernetes version	CPU	Memory	Console URL	Labels
local-cluster	True	True	True	8	v1.20.0+c8905da	84	418501Mi	Launch	cloud=VSphere clusterID=148632d9-69d5-4ae4-98ee-8df886463c3 installer.name=multiclusterhub 4 more
ocp-vmw	True	True	True	9	v1.20.0+df9c838	28	111981Mi	Launch	cloud=VSphere clusterID=9d76ac4e-4aae-4d45-a2e8-1b6b54282fe name=ocp-vmw 1 more

Next: Features - Create Resources.

Features: Advanced Cluster Management for Kubernetes on Red Hat OpenShift with NetApp

Create resources on multiple clusters

Advanced Cluster Management for Kubernetes allows users to create resources on one or more managed clusters simultaneously from the console. As an example, if you have OpenShift clusters at different sites backed with different NetApp ONTAP clusters and want to provision PVC's at both sites, you can click the (+) sign on the top bar. Then select the clusters on which you want to create the PVC, paste the resource YAML, and click Create.

Create resource

Cancel

Create

Clusters | Select the clusters where the resource(s) will be deployed.

2 ×

local-cluster, ▾
ocp-vmw

Resource configuration | Enter the configuration manifest for the resource(s).

YAML

```
1 kind: PersistentVolumeClaim
2 apiVersion: v1
3 metadata:
4   name: demo-pvc
5 spec:
6   accessModes:
7     - ReadWriteOnce
8   resources:
9     requests:
10    storage: 1Gi
11  storageClassName: ocp-trident
```

Videos and Demos: Red Hat OpenShift with NetApp

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

- Video: Accelerate Software Development with Astra Control and NetApp FlexClone Technology
- Video: Leverage NetApp Astra Control to Perform Post-mortem Analysis and Restore Your Application
- Video: Data Protection in CI/CD pipeline with Astra Control
- Video: Workload Migration using Astra Control Center - Red Hat OpenShift with NetApp
- Video: Workload Migration using Astra Trident and SnapMirror - Red Hat OpenShift with NetApp
- Video: Installing OpenShift Virtualization - Red Hat OpenShift with NetApp
- Video: Deploying a Virtual Machine with OpenShift Virtualization - Red Hat OpenShift with NetApp
- Video: NetApp HCI for Red Hat OpenShift on Red Hat Virtualization Deployment

Next: Additional Information: Red Hat OpenShift with NetApp.

Additional Information: Red Hat OpenShift with NetApp

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

- NetApp Documentation
<https://docs.netapp.com/>
- 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/>

- Red Hat OpenShift Documentation

https://access.redhat.com/documentation/en-us/openshift_container_platform/4.7/

- Red Hat OpenStack Platform Documentation

https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/16.1/

- Red Hat Virtualization Documentation

https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.4/

- VMware vSphere Documentation

<https://docs.vmware.com/>

NVA-1166: VMware Tanzu with NetApp

Alan Cowles and Nikhil M Kulkarni, NetApp

This reference document provides deployment validation of different flavors of VMware Tanzu Kubernetes solutions, deployed either as Tanzu Kubernetes Grid (TKG), Tanzu Kubernetes Grid Service (TKGS), or Tanzu Kubernetes Grid Integrated (TKGI) in several different data center environments as validated by NetApp. It also describes storage integration with NetApp storage systems and the Astra Trident storage orchestrator for the management of persistent storage and Astra Control Center for the backup and cloning of the stateful applications using that persistent storage. Lastly, the document provides video demonstrations of the solution integrations and validations.

Use cases

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

- Easy to deploy and manage VMware Tanzu Kubernetes Grid offerings deployed on VMware vSphere and integrated with NetApp storage systems.
- The combined power of enterprise container and virtualized workloads with VMware Tanzu Kubernetes Grid offerings.
- Real world configuration and use cases highlighting the features of VMware Tanzu when used with NetApp storage and the NetApp Astra suite of products.
- Application-consistent protection or migration of containerized workloads deployed on VMware Tanzu Kubernetes Grid clusters whose data resides on NetApp storage systems using Astra Control Center.

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
- Ability to run virtualized and containerized workloads simultaneously
- Ability to scale infrastructure independently based on workload demands
- Ability to deploy in a hybrid-cloud model with containers running in both on-premises data centers as well as in the cloud.

VMware Tanzu with NetApp acknowledges these challenges and presents a solution that helps address each concern by deploying VMware Tanzu Kubernetes offerings in the customer's choice of hybrid cloud environment.

Technology overview

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

VMware Tanzu Kubernetes platforms

VMware Tanzu comes in a variety of flavors that the solutions engineering team at NetApp has validated in our labs. Each Tanzu release successfully integrates with the NetApp storage portfolio, and each can help meet certain infrastructure demands. The following bulleted highlights describe the features and offerings of each version of Tanzu described in this document.

VMware Tanzu Kubernetes Grid (TKG)

- Standard upstream Kubernetes environment deployed in a VMware vSphere environment.
- Formerly known as Essential PKS (from Heptio acquisition, Feb 2019).
- TKG is deployed with a separate management cluster instance for support on vSphere 6.7U3 onward.
- TKG deployments can be deployed in the cloud as well with AWS or Azure.
- Allows for use of Windows or Linux worker nodes (Ubuntu/Photon).
- NSX-T, HA Proxy, AVI networking, or load balancers can be used for control plane.
- TKG supports MetalLB for the application/data plane.
- Can use vSphere CSI as well as third party CSIs like NetApp Astra Trident.

VMware Tanzu Kubernetes Grid Service (TKGS)

- Standard upstream Kubernetes environment deployed in a VMware vSphere environment.
- Formerly known as Essential PKS (from Heptio acquisition, Feb 2019).
- TKGS deployed with supervisor cluster and workload clusters only on vSphere 7.0U1 onward.
- Allows for use of Windows or Linux worker nodes (Ubuntu/Photon).
- NSX-T, HA Proxy, AVI networking, or load balancers can be used for control plane.
- TKGS supports MetalLB for application/data plane.
- Can use vSphere CSI as well as third party CSIs like NetApp Astra Trident.

- Provides support for vSphere Pods with Tanzu, allowing pods to run directly on enabled ESXi hosts in the environment.

VMWare Tanzu Kubernetes Grid Integrated (TKGI)

- Formerly known as Enterprise PKS (from Heptio acquisition, Feb 2019).
- Can use NSX-T, HA Proxy, or Avi. You can also provide your own load balancer.
- Supported from vSphere 6.7U3 onward, as well as AWS, Azure, and GCP.
- Setup via wizard to allow for ease of deployment.
- Runs Tanzu in controlled immutable VMs managed by BOSH.
- Can make use vSphere CSI as well as third party CSIs like NetApp Astra Trident (some conditions apply).

vSphere with Tanzu (vSphere Pods)

- vSphere-native pods run in a thin, photon-based layer with prescribed virtual hardware for complete isolation.
- Requires NSX-T, but that allows for additional feature support such as a Harbor image registry.
- Deployed and managed in vSphere 7.0U1 onward using a virtual Supervisor cluster like TKGS. Runs pods directly on ESXi nodes.
- Fully vSphere integrated, highest visibility and control by vSphere administration.
- Isolated CRX-based pods for the highest level of security.
- Only supports vSphere CSI for persistent storage. No third-party storage orchestrators supported.

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

NetApp Astra Control Center offers a rich set of storage and application-aware data management services for stateful Kubernetes workloads, deployed in an on-prem environment, and powered by trusted NetApp data protection technology.

For more information, visit the NetApp Astra website [here](#).

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

For more information, visit the Astra Trident website [here](#).

Current support matrix for validated releases

Technology	Purpose	Software version
NetApp ONTAP	Storage	9.9.1

NetApp Astra Control Center	Application Aware Data Management	22.04
NetApp Astra Trident	Storage Orchestration	22.04.0
VMware Tanzu Kubernetes Grid	Container orchestration	1.4+
VMware Tanzu Kubernetes Grid Service	Container orchestration	0.0.15 [vSphere Namespaces]
		1.22.6 [Supervisor Cluster Kubernetes]
VMware Tanzu Kubernetes Grid Integrated	Container orchestration	1.13.3
VMware vSphere	Data center virtualization	7.0U3
VMware NSX-T Data Center	Networking and Security	3.1.3
VMware NSX Advanced Load Balancer	Load Balancer	20.1.3

Next: [VMware Tanzu Overview](#).

VMware Tanzu overview

VMware Tanzu is a portfolio of products that enables enterprises to modernize their applications and the infrastructure they run on. VMware Tanzu's full stack of capabilities unites the development and IT operations teams on a single platform to embrace modernization in both their applications and their infrastructure consistently across on-premises and hybrid cloud environments to continuously deliver better software to production.



To understand more about the different offerings and their capabilities in the Tanzu portfolio, visit the documentation [here](#).

Regarding Tanzu's Kubernetes Operations catalog, VMware has a variety of implementations for Tanzu Kubernetes Grid, all of which provision and manage the lifecycle of Tanzu Kubernetes clusters on a variety of platforms. A Tanzu Kubernetes cluster is a full-fledged Kubernetes distribution that is built and supported by VMware.

NetApp has tested and validated the deployment and interoperability of the following products from the VMware Tanzu portfolio in its labs:

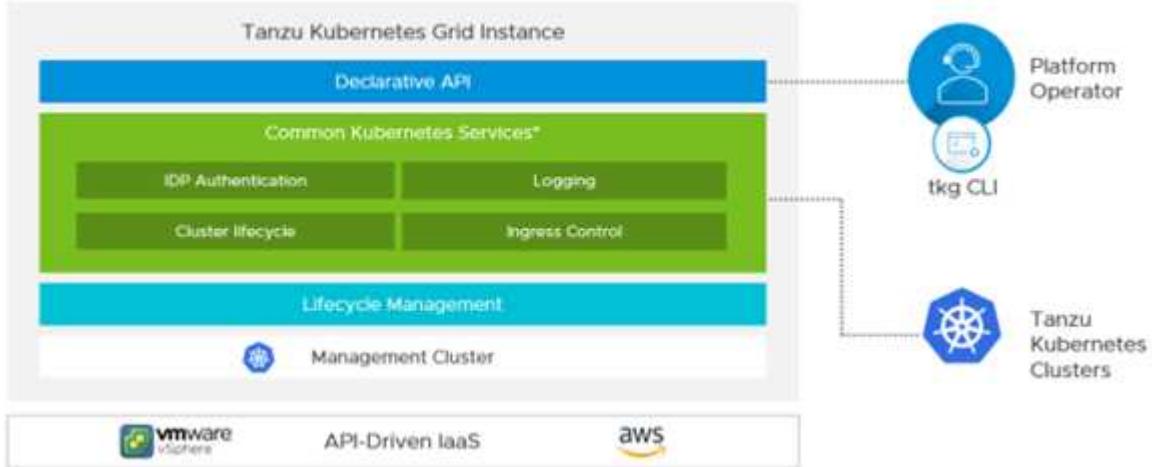
- [VMware Tanzu Kubernetes Grid \(TKG\)](#)
- [VMware Tanzu Kubernetes Grid Service \(TKGS\)](#)
- [VMware Tanzu Kubernetes Grid Integrated \(TKGI\)](#)
- [VMware vSphere with Tanzu \(vSphere Pods\)](#)

Next: [NetApp storage systems overview](#).

VMware Tanzu Kubernetes Grid (TKG) overview

VMware Tanzu Kubernetes Grid, also known as TKG, lets you deploy Tanzu Kubernetes clusters across hybrid cloud or public cloud environments. TKG is installed as a management cluster, which is a Kubernetes cluster itself, that deploys and operates the Tanzu Kubernetes clusters. These Tanzu Kubernetes clusters are the workload Kubernetes clusters on which the actual workload is deployed.

Tanzu Kubernetes Grid builds on a few of the promising upstream community projects and delivers a Kubernetes platform that is developed, marketed, and supported by VMware. In addition to Kubernetes distribution, Tanzu Kubernetes Grid provides additional add-ons that are essential production-grade services such as registry, load balancing, authentication, and so on. VMware TKG with management cluster is widely used in vSphere 6.7 environments, and, even though it is supported, it is not a recommended deployment for vSphere 7 environments because TKGS has native integration capabilities with vSphere 7.



For more information on Tanzu Kubernetes Grid, refer to the documentation [here](#).

Depending on whether the Tanzu Kubernetes Grid is being installed on-premises on vSphere cluster or in cloud environments, prepare and deploy Tanzu Kubernetes Grid by following the installation guide [here](#).

After you have installed the management cluster for Tanzu Kubernetes Grid, deploy the user clusters or workload clusters as needed by following the documentation [here](#). VMware TKG management cluster requires that an SSH key be provided for installation and operation of Tanzu Kubernetes clusters. This key can be used

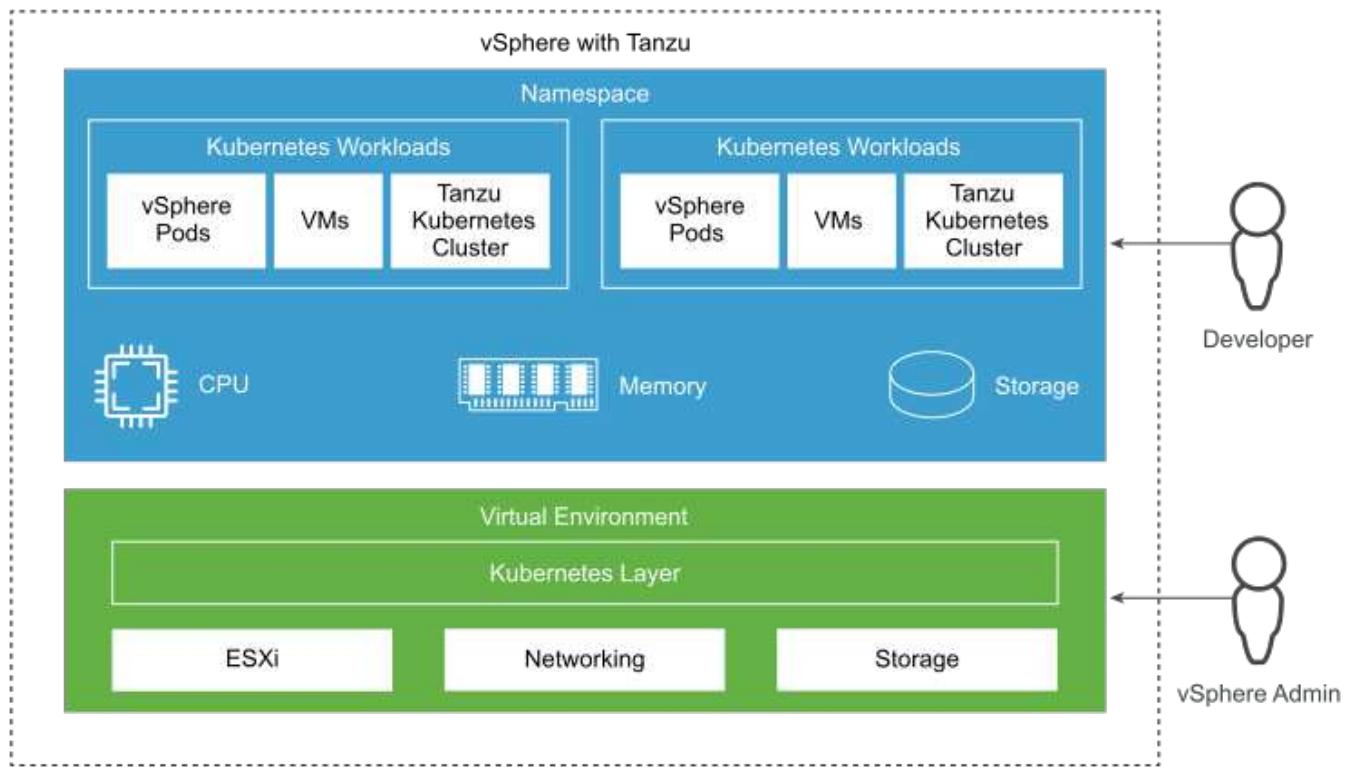
to log into the cluster nodes using the `capv` user.

Next: [NetApp storage systems overview](#).

VMware Tanzu Kubernetes Grid Service (TKGS) overview

VMware Tanzu Kubernetes Grid Service (also known as vSphere with Tanzu) lets you create and operate Tanzu Kubernetes clusters natively in vSphere and also allows you to run some smaller workloads directly on the ESXi hosts. It allows you to transform vSphere into a platform for running containerized workloads natively on the hypervisor layer. Tanzu Kubernetes Grid Service deploys a supervisor cluster on vSphere when enabled that deploys and operates the clusters required for the workloads. It is natively integrated with vSphere 7 and leverages many reliable vSphere features like vCenter SSO, Content Library, vSphere networking, vSphere storage, vSphere HA and DRS, and vSphere security for a more seamless Kubernetes experience.

vSphere with Tanzu offers a single platform for hybrid application environments where you can run your application components either in containers or in VMs, thus providing better visibility and ease of operations for developers, DevOps engineers, and vSphere administrators. VMware TKGS is only supported with vSphere 7 environments and is the only offering in Tanzu Kubernetes operations portfolio that allows you to run pods directly on ESXi hosts.



For more information on Tanzu Kubernetes Grid Service, follow the documentation [here](#).

There are a lot of architectural considerations regarding feature sets, networking, and so on. Depending on the architecture chosen, the prerequisites and the deployment process of Tanzu Kubernetes Grid Service differ. To deploy and configure Tanzu Kubernetes Grid Service in your environment, follow the guide [here](#). Furthermore, to log into the Tanzu Kubernetes cluster nodes deployed via TKGS, follow the procedure laid out in this [link](#).

NetApp recommends that all the production environments be deployed in multiple master deployments for fault tolerance with the choice of worker nodes' configuration to meet the requirements of the intended workloads. Thus, a recommended VM class for a highly intensive workload would have at least four vCPUs and 12GB of RAM.

When Tanzu Kubernetes clusters are created in a namespace, users with `owner` or `edit` permission can create pods directly in any namespace by using the user account. This is because users with the `owner` or `edit` permission are allotted the cluster administrator role. However, when creating deployments, daemon sets, stateful sets, or others in any namespace, you must assign a role with the required permissions to the corresponding service accounts. This is required because the deployments or daemon sets utilize service accounts to deploy the pods.

See the following example of `ClusterRoleBinding` to assign the cluster administrator role to all service accounts in the cluster:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: all_sa_ca
subjects:
- kind: Group
  name: system:serviceaccounts
  namespace: default
roleRef:
  kind: ClusterRole
  name: psp:vmware-system-privileged
  apiGroup: rbac.authorization.k8s.io
```

Next: [NetApp storage systems overview](#).

VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) overview

VMware Tanzu Kubernetes Grid Integrated (TKGI) Edition, formerly known as VMware Enterprise PKS, is a standalone container orchestration platform based on Kubernetes with capabilities such as life cycle management, cluster health monitoring, advanced networking, a container registry, and so on. TKGI provisions and manages Kubernetes clusters with the TKGI control plane, which consists of BOSH and Ops Manager.

TKGI can be installed and operated either on vSphere or OpenStack environments on-premises or in any of the major public clouds on their respective IaaS offerings. Furthermore, the integration of TKGI with NSX-T and Harbour enables wider use cases for enterprise workloads. To know more about TKGI and its capabilities, visit the documentation [here](#).



TKGI is installed in a variety of configurations on a variety of platforms based on different use-cases and designs. Follow the guide [here](#) to install and configure TKGI and its prerequisites. TKGI uses Bosh VMs as nodes for Tanzu Kubernetes clusters which run immutable configuration images and any manual changes on Bosh VMs do not remain persistent across reboots.

Important notes:

- NetApp Trident requires privileged container access. So, during TKGI installation, make sure to select the **Enable Privileged Containers** checkbox in the step to configure Tanzu Kubernetes cluster node plans.

The screenshot shows the configuration interface for a Node Plan in TKGI. It includes fields for Worker Node Instances (3), Worker Persistent Disk Size (50 GB), and Worker Availability Zones (az). It also shows Max Worker Node Instances (50). Under Errand VM Type, it lists medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB). A checked checkbox says "Enable Privileged Containers (Use with caution)". The Admission Plugins section has two disabled checkboxes: PodSecurityPolicy and SecurityContextDeny. The Cluster Services section contains several toggle switches for node draining: Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or Stateful Set; Force node to drain even if it has running DaemonSet managed pods; Force node to drain even if it has running pods using emptyDir; and Force node to drain even if pods are still running after timeout.

Worker Node Instances ⓘ 3	Worker Persistent Disk Size ⓘ 50 GB	Worker Availability Zones ⓘ az
Worker VM Type ⓘ medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)	Max Worker Node Instances ⓘ 50	
Errand VM Type ⓘ medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)	<input checked="" type="checkbox"/> Enable Privileged Containers (Use with caution) ⓘ	
Admission Plugins <input type="checkbox"/> PodSecurityPolicy ⓘ <input type="checkbox"/> SecurityContextDeny ⓘ		
Cluster Services		
Node Drain Timeout (minutes, min: 0, max: 1440) ⓘ 0	<input type="checkbox"/> Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or Stateful Set ⓘ	
Pod Shutdown Grace Period (seconds, min: -1, max: 86400) ⓘ 10	<input type="checkbox"/> Force node to drain even if it has running DaemonSet managed pods ⓘ <input type="checkbox"/> Force node to drain even if it has running pods using emptyDir ⓘ <input type="checkbox"/> Force node to drain even if pods are still running after timeout ⓘ	
SAVE PLAN		DELETE

- NetApp recommends that all production environments be deployed in multiple master deployments for fault tolerance with the choice of worker nodes' configuration to meet the requirements of the intended

workloads. Thus, a recommended TKGI cluster plan would consist of at least three masters and three workers with at least four vCPUs and 12GB of RAM for a highly intensive workload.

Next: [NetApp storage systems overview](#).

NetApp storage systems overview

NetApp has several storage platforms that are qualified with Astra Trident and Astra Control to provision, protect, and manage data for containerized applications.



- 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.



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

The following pages have additional information about the NetApp storage systems validated in the VMware Tanzu with NetApp solution:

- [NetApp ONTAP](#)

Next: [NetApp storage integrations overview](#).

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, non-disruptive 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 protecting 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 non-rewritable 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 multi-protocol support.

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

For more information about NETAPP AFF/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 it 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 that can 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 storage integrations overview.

NetApp storage integration overview

NetApp provides a number of products to help you orchestrate, manage, protect, and migrate stateful containerized applications and their data.



NetApp Astra Control offers a rich set of storage and application-aware data management services for stateful Kubernetes workloads powered by NetApp data protection technology. The Astra Control Service is available to support stateful workloads in cloud-native Kubernetes deployments. The Astra Control Center is available to support stateful workloads in on-premises deployments of Enterprise Kubernetes platforms like Red Hat OpenShift, Rancher, VMware Tanzu etc. For more information visit the NetApp Astra Control website [here](#).

NetApp Astra Trident is an open-source and fully-supported storage orchestrator for containers and Kubernetes distributions like Red Hat OpenShift, Rancher, VMware Tanzu etc. For more information, visit the

Astra Trident website [here](#).

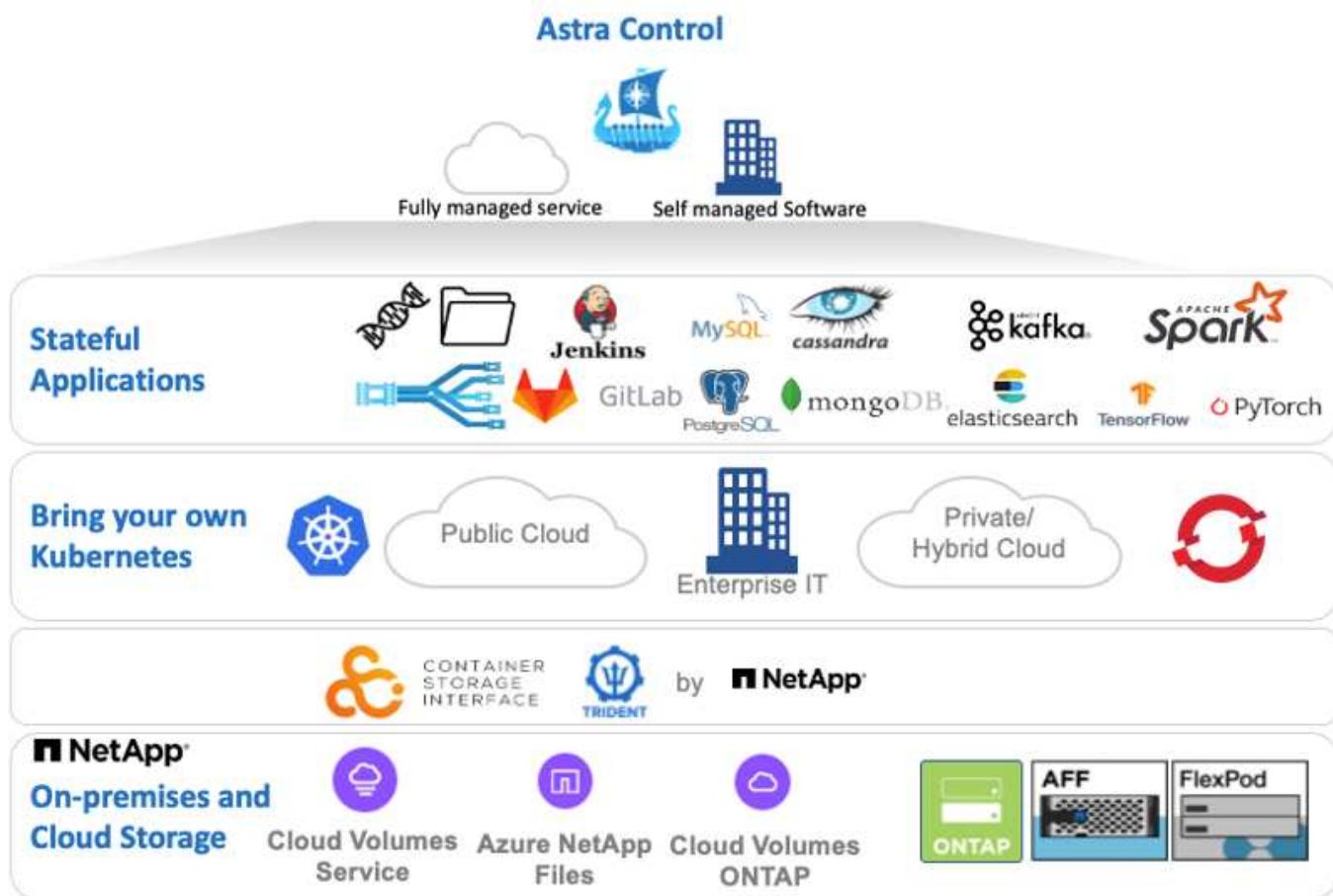
The following pages have additional information about the NetApp products that have been validated for application and persistent storage management in the VMware Tanzu with NetApp solution:

- [NetApp Astra Control Center](#)
- [NetApp Astra Trident](#)

Next: [NetApp Astra Control overview](#).

NetApp Astra Control overview

NetApp Astra Control Center offers a rich set of storage and application-aware data management services for stateful Kubernetes workloads deployed in an on-premises environment and powered by NetApp data protection technology.



NetApp Astra Control Center can be installed on a VMware Tanzu cluster that has the Astra Trident storage orchestrator deployed and configured with storage classes and storage backends to NetApp ONTAP storage systems.

For more information on Astra Trident, see [this document here](#).

In a cloud-connected environment, Astra Control Center uses Cloud Insights to provide advanced monitoring and telemetry. In the absence of a Cloud Insights connection, limited monitoring and telemetry (seven days worth of metrics) is available and exported to Kubernetes native monitoring tools (Prometheus and Grafana) through open metrics endpoints.

Astra Control Center is fully integrated into the NetApp AutoSupport and Active IQ ecosystem to provide support for users, provide assistance with troubleshooting, and display usage statistics.

In addition to the paid version of Astra Control Center, a 90-day evaluation license is also available. The evaluation version is supported through email and the community Slack channel. Customers have access to these resources, other knowledge-base articles, and documentation available from the in-product support dashboard.

To understand more about the Astra portfolio, visit the [Astra website](#).

Astra Control Center automation

Astra Control Center has a fully functional REST API for programmatic access. Users can use any programming language or utility to interact with Astra Control REST API endpoints. To learn more about this API, see the documentation [here](#).

If you are looking for a ready-made software development toolkit for interacting with Astra Control REST APIs, NetApp provides a toolkit with the Astra Control Python SDK that you can download [here](#).

If programming is not appropriate for your situation and you would like to use a configuration management tool, you can clone and run the Ansible playbooks that NetApp publishes [here](#).

Astra Control Center installation prerequisites

Astra Control Center installation requires the following prerequisites:

- One or more Tanzu Kubernetes clusters, managed either by a management cluster or TKGS or TKGI. TKG workload clusters 1.4+ and TKGI user clusters 1.12.2+ are supported.
- Astra Trident must already be installed and configured on each of the Tanzu Kubernetes clusters.
- One or more NetApp ONTAP storage systems running ONTAP 9.5 or greater.



It's a best practice for each Tanzu Kubernetes install at a site to have a dedicated SVM for persistent storage. Multi-site deployments require additional storage systems.

- A Trident storage backend must be configured on each Tanzu Kubernetes cluster with an SVM backed by an ONTAP cluster.
- A default StorageClass configured on each Tanzu Kubernetes cluster with Astra Trident as the storage provisioner.
- A load balancer must be installed and configured on each Tanzu Kubernetes cluster for load balancing and exposing Astra Control Center if you are using ingressType `AccTraefik`.
- An ingress controller must be installed and configured on each Tanzu Kubernetes cluster for exposing Astra Control Center if you are using ingressType `Generic`.
- A private image registry must be configured to host the NetApp Astra Control Center images.
- You must have Cluster Admin access to the Tanzu Kubernetes cluster where Astra Control Center is being installed.
- You must have Admin access to NetApp ONTAP clusters.
- A RHEL or Ubuntu admin workstation.

Install Astra Control Center

This solution describes an automated procedure for installing Astra Control Center using Ansible playbooks. If you are looking for a manual procedure to install Astra Control Center, follow the detailed installation and operations guide [here](#).

1. To use the Ansible playbooks that deploy Astra Control Center, you must have an Ubuntu/RHEL machine with Ansible installed. Follow this [procedure](#) for Ubuntu and this [procedure](#) for RHEL.
2. Clone the GitHub repository that hosts the Ansible content.

```
git clone https://github.com/NetApp-Automation/na_astra_control_suite.git
```

3. Log into the NetApp Support Site and download the latest version of NetApp Astra Control Center. To do so requires a license attached to your NetApp account. After you download the tarball, transfer it to the workstation.



To get started with a trial license for Astra Control, visit the [Astra registration site](#).

4. Create or obtain the kubeconfig file with admin access to the user or workload Tanzu Kubernetes cluster on which Astra Control Center is to be installed.
5. Change the directory to na_astra_control_suite.

```
cd na_astra_control_suite
```

6. Edit the vars/vars.yml file and fill the variables with the required information.

```
#Define whether or not to push the Astra Control Center images to your
private registry [Allowed values: yes, no]
push_images: yes

#The directory hosting the Astra Control Center installer
installer_directory: /home/admin/

#Specify the ingress type. Allowed values - "AccTraefik" or "Generic"
#"AccTraefik" if you want the installer to create a LoadBalancer type
service to access ACC, requires MetalLB or similar.
#"Generic" if you want to create or configure ingress controller
yourself, installer just creates a ClusterIP service for traefik.
ingress_type: "AccTraefik"

#Name of the Astra Control Center installer (Do not include the
extension, just the name)
astra_tar_ball_name: astra-control-center-22.04.0

#The complete path to the kubeconfig file of the kubernetes/openshift
```

```

cluster Astra Control Center needs to be installed to.
hosting_k8s_cluster_kubeconfig_path: /home/admin/cluster-kubeconfig.yml

#Namespace in which Astra Control Center is to be installed
astra_namespace: netapp-astra-cc

#Astra Control Center Resources Scaler. Leave it blank if you want to
accept the Default setting.
astra_resources_scaler: Default

#Storageclass to be used for Astra Control Center PVCs, it must be
created before running the playbook [Leave it blank if you want the PVCs
to use default storageclass]
astra_trident_storageclass: basic

#Reclaim Policy for Astra Control Center Persistent Volumes [Allowed
values: Retain, Delete]
storageclass_reclaim_policy: Retain

#Private Registry Details
astra_registry_name: "docker.io"

#Whether the private registry requires credentials [Allowed values: yes,
no]
require_reg_creds: yes

#If require_reg_creds is yes, then define the container image registry
credentials
#Usually, the registry namespace and usernames are same for individual
users
astra_registry_namespace: "registry-user"
astra_registry_username: "registry-user"
astra_registry_password: "password"

#Kubernetes/OpenShift secret name for Astra Control Center
#This name will be assigned to the K8s secret created by the playbook
astra_registry_secret_name: "astra-registry-credentials"

#Astra Control Center FQDN
acc_fqdn_address: astra-control-center.cie.netapp.com

#Name of the Astra Control Center instance
acc_account_name: ACC Account Name

#Administrator details for Astra Control Center
admin_email_address: admin@example.com
admin_first_name: Admin

```

```
admin_last_name: Admin
```

7. Run the playbook to deploy Astra Control Center. The playbook requires root privileges for certain configurations.

Run the following command to run the playbook if the user running the playbook is root or has passwordless sudo configured.

```
ansible-playbook install_acc_playbook.yml
```

If the user has password-based sudo access configured, then run the following command to run the playbook and then enter the sudo password.

```
ansible-playbook install_acc_playbook.yml -K
```

Post Install Steps

1. It might take several minutes for the installation to complete. Verify that all the pods and services in the `netapp-astra-cc` namespace are up and running.

```
[netapp-user@rhel7 ~]$ kubectl get all -n netapp-astra-cc
```

2. Check the `acc-operator-controller-manager` logs to ensure that the installation is completed.

```
[netapp-user@rhel7 ~]$ kubectl logs deploy/acc-operator-controller-
manager -n netapp-acc-operator -c manager -f
```



The following message indicates the successful installation of Astra Control Center.

```
{"level": "info", "ts": 1624054318.029971, "logger": "controllers.AstraContro-
lCenter", "msg": "Successfully Reconciled AstraControlCenter in
[seconds]s", "AstraControlCenter": "netapp-astra-
cc/astra", "ae.Version": "[22.04.0]"}
```

3. The username for logging into Astra Control Center is the email address of the administrator provided in the CRD file and the password is a string `ACC-` appended to the Astra Control Center UUID. Run the following command:

```
[netapp-user@rhel7 ~]$ oc get astracontrolcenters -n netapp-astra-cc
NAME      UUID
astra    345c55a5-bf2e-21f0-84b8-b6f2bce5e95f
```



In this example, the password is ACC-345c55a5-bf2e-21f0-84b8-b6f2bce5e95f.

4. Get the traefik service load balancer IP if the ingressType is AccTraefik.

```
[netapp-user@rhel7 ~]$ oc get svc -n netapp-astra-cc | egrep
'EXTERNAL|traefik'

NAME           TYPE        CLUSTER-IP
EXTERNAL-IP   PORT(S)
AGE
traefik       LoadBalancer 172.30.99.142
10.61.186.181 80:30343/TCP,443:30060/TCP
16m
```

5. Add an entry in the DNS server pointing the FQDN provided in the Astra Control Center CRD file to the EXTERNAL-IP of the traefik service.

New Host

X

Name (uses parent domain name if blank):

astra-control-center

Fully qualified domain name (FQDN):

astra-control-center.cie.netapp.com.

IP address:

10.61.186.181

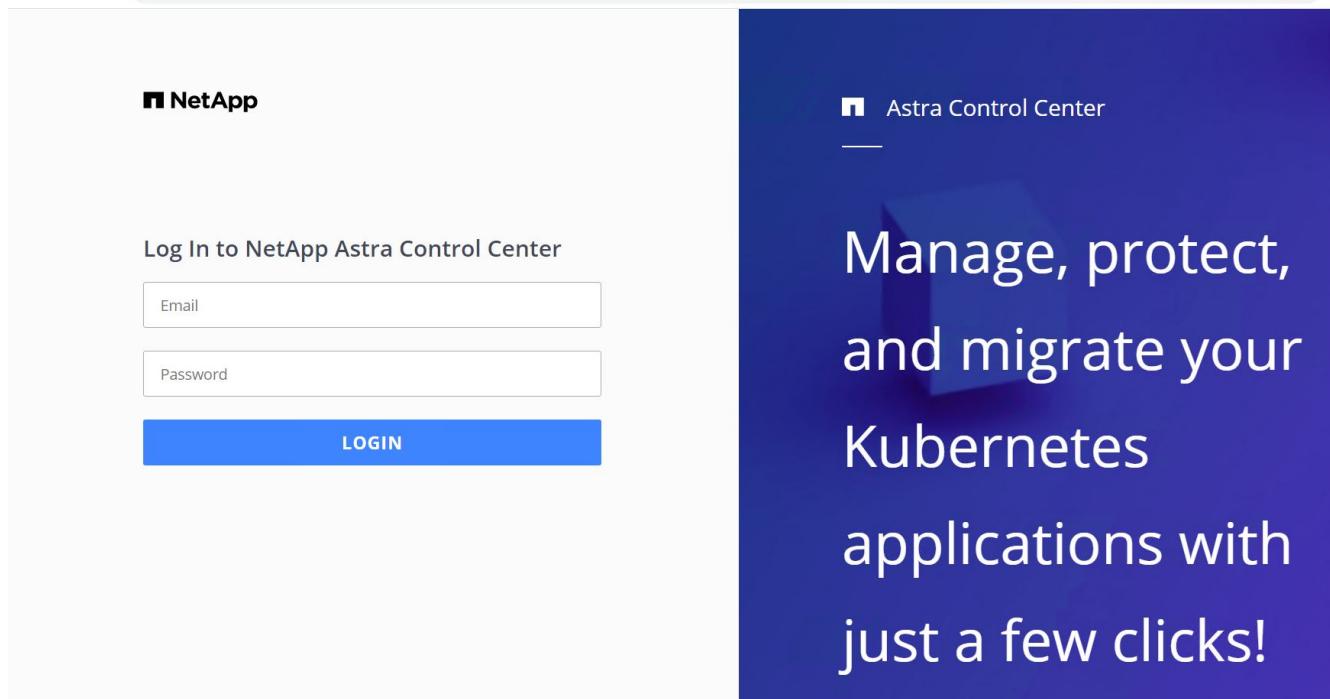
Create associated pointer (PTR) record

Allow any authenticated user to update DNS records with the same owner name

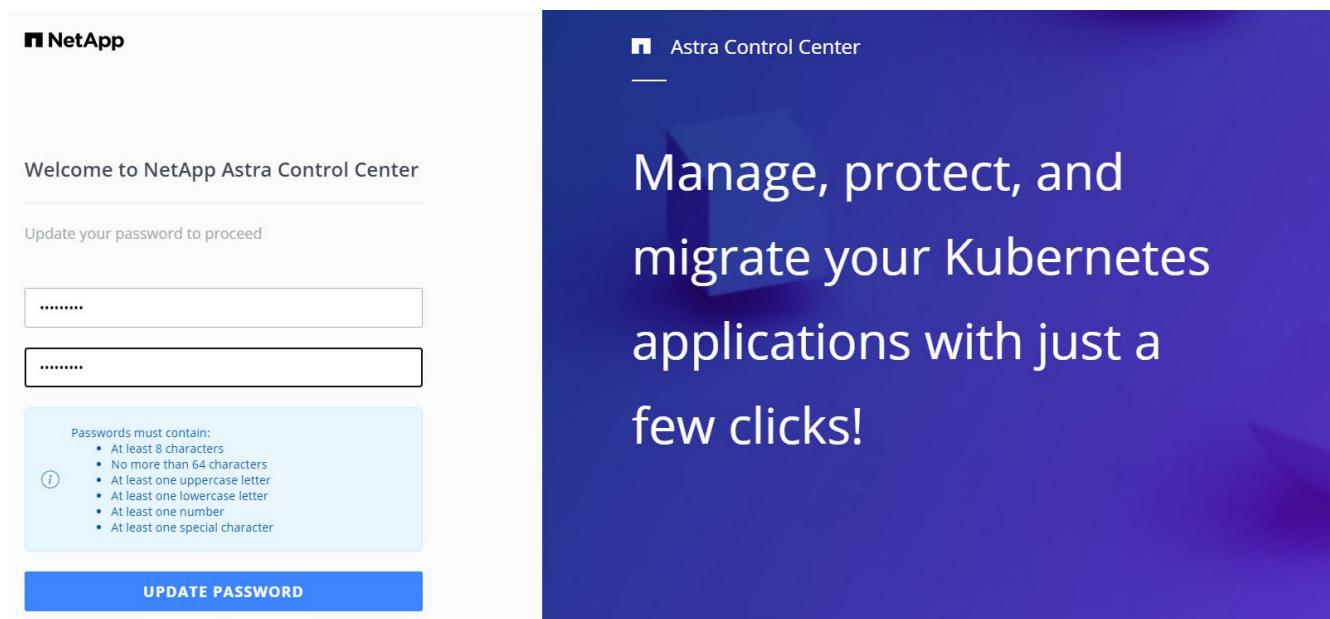
Add Host

Cancel

6. Log into the Astra Control Center GUI by browsing its FQDN.



- When you log into Astra Control Center GUI for the first time using the admin email address provided in CRD, you need to change the password.



- If you wish to add a user to Astra Control Center, navigate to Account > Users, click Add, enter the details of the user, and click Add.

Add user

USER DETAILS

First name Nikhil	Last name Kulkarni
Email address tme_nik@netapp.com	

PASSWORD

Temporary password *****	Confirm temporary password *****
-----------------------------	-------------------------------------

Passwords must contain:

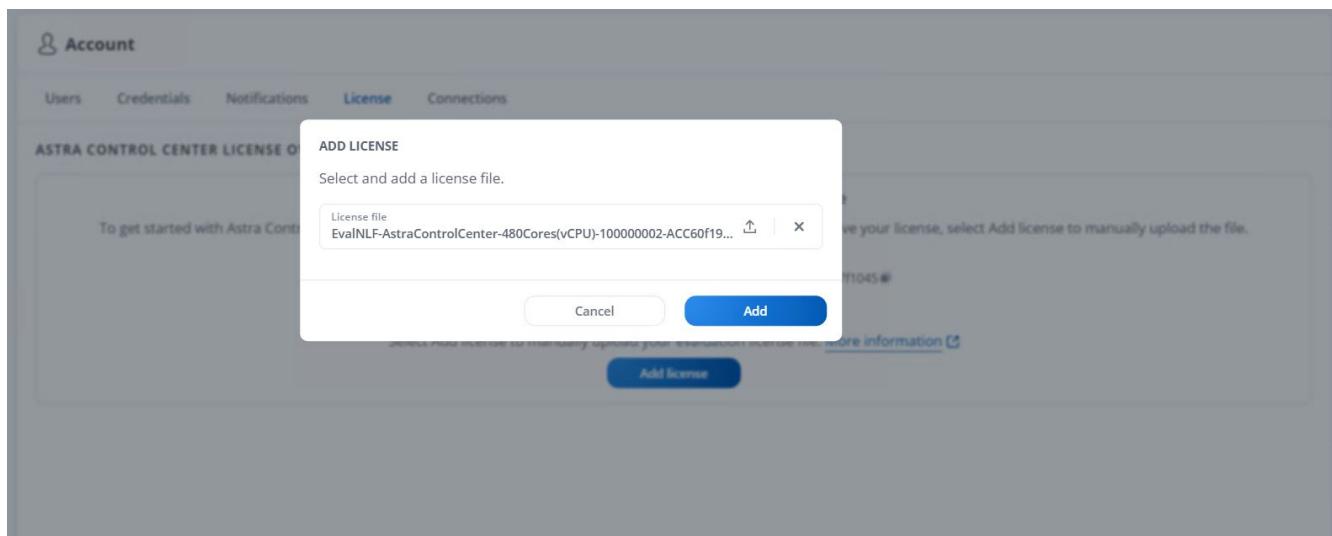
- At least 8 characters
- No more than 64 characters
- At least one lowercase letter
- At least one uppercase letter
- At least one number
- At least one special character

USER ROLE

Role Owner

Cancel
Add ✓

9. Astra Control Center requires a license for all of its functionalities to work. To add a license, navigate to Account > License, click Add License, and upload the license file.



If you encounter issues with the install or configuration of NetApp Astra Control Center, the knowledge base of known issues is available [here](#).

Next: Register your Tanzu Kubernetes clusters.

Register your VMware Tanzu Kubernetes Clusters with the Astra Control Center

To enable the Astra Control Center to manage your workloads, you must first register your Tanzu Kubernetes clusters.

Register VMware Tanzu Kubernetes clusters

1. The first step is to add the Tanzu Kubernetes clusters to the Astra Control Center and manage them. Go to Clusters and click Add a Cluster, upload the kubeconfig file for the Tanzu Kubernetes cluster, and click Select Storage.

STEP 1/3: CREDENTIALS

CREDENTIALS

Provide Astra Control access to your Kubernetes and OpenShift clusters by entering a kubeconfig credential.

Follow [instructions](#) on how to create a dedicated admin-role kubeconfig.

Upload file Paste from clipboard

Kubeconfig YAML file
tkgi-kubeconfig.txt

Credential name
tkgi-acc

ADDING CLUSTERS

Adding a cluster allows Astra Control to install its storage services, and enable data management operations on your containerized applications.

For more details on required versions or cloud specific setup refer to the documentation.

Read more in [Adding clusters](#).

Cancel Next →

2. Astra Control Center detects the eligible storage classes. Now select the way that storageclass provisions volumes using Trident backed by an SVM on NetApp ONTAP and click Review. In the next pane, verify the details and click Add Cluster.
3. When the cluster is added, it moves to the Discovering status while Astra Control Center inspects it and installs the necessary agents. The cluster status changes to Healthy after it is successfully registered.

Name	Type	Version	Actions
tkgi-acc	Kubernetes	v1.22.6+vmware.1	



All Tanzu Kubernetes clusters to be managed by Astra Control Center should have access to the image registry that was used for its installation as the agents installed on the managed clusters pull the images from that registry.

4. Import ONTAP clusters as storage resources to be managed as backends by Astra Control Center. When Tanzu Kubernetes clusters are added to Astra and a storageclass is configured, it automatically discovers and inspects the ONTAP cluster backing the storageclass but does not import it into the Astra Control Center to be managed.

Backend

Name	State	Capacity	Throughput	Type	Cluster	Cloud	Actions
172.21.224.201(trident)	(i) Discovered	Not available yet	Not available yet	ONTAP	Not applicable	Not applicable	⋮

- To import the ONTAP clusters, navigate to Backends, click the dropdown, and select Manage next to the ONTAP cluster to be managed. Enter the ONTAP cluster credentials, click Review Information, and then click Import Storage Backend.

Manage ONTAP storage backend

STEP 1/2: CREDENTIALS

CREDENTIALS

Enter cluster administrator credentials for the ONTAP storage backend you want to manage.

Cluster management IP address 172.21.224.201	User name admin	Password *****	
---	--------------------	-------------------	--

MANAGING STORAGE BACKENDS

Storage backends provide storage to your Kubernetes applications.

Managing storage clusters in Astra Control as a storage backend will allow you to get linkages between PVs and the storage backend. You will also see capacity and health details of the storage backend, including performance metrics if Astra Control is connected to Cloud Insights.

Read more in [Storage type](#).

ONTAP

Cancel Next →

- After the backends are added, the status changes to Available. These backends now have the information about the persistent volumes in the Tanzu Kubernetes cluster and the corresponding volumes on the ONTAP system.

Backends

Name	State	Capacity	Throughput	Type	Cluster	Cloud	Actions
K8s-OnTap	 Available	Not available yet	Not available yet	ONTAP 9.9.1	Not applicable	Not applicable	

7. For backup and restore across Tanzu Kubernetes clusters using Astra Control Center, you must provision an object storage bucket that supports the S3 protocol. Currently supported options are ONTAP S3, StorageGRID, AWS S3, and Microsoft Azure Blob storage. For the purpose of this installation, we are going to configure an AWS S3 bucket. Go to Buckets, click Add bucket, and select Generic S3. Enter the details about the S3 bucket and credentials to access it, click the checkbox Make this Bucket the Default Bucket for the Cloud, and then click Add.

Add bucket X

Enter the access details of your existing object store bucket to allow Astra Control to store your application backups.

Type  Generic S3	Existing bucket name na-tanzu-astra/na-astra-tkgj
Description (optional)	S3 server name or IP address s3.us-east-1.amazonaws.com
<input checked="" type="checkbox"/> Make this bucket the default bucket for this cloud ?	

BUCKETS
Astra Control stores backups in your existing object store buckets. The first bucket added for a selected cloud will be designated as the default bucket for backup and clone operations.
[Read more in Storage buckets](#)

SELECT CREDENTIALS
Astra Control requires S3 access credentials with the roles necessary to facilitate Kubernetes application data management.

Add Use existing

Select credential
AWS Creds

[Cancel](#) Add 

Next: Choose the Applications To Protect.

Choose the applications to protect

After you have registered your Tanzu Kubernetes clusters, you can discover the applications that are deployed and manage them via the Astra Control Center.

Manage applications

- After the Tanzu Kubernetes clusters and ONTAP backends are registered with the Astra Control Center, the control center automatically starts discovering the applications in all the namespaces that are using the storageclass configured with the specified ONTAP backend.

The screenshot shows the 'Applications' section of the Astra Control Center. The left sidebar includes 'Dashboard', 'MANAGE YOUR APPLICATIONS' (with 'Applications' selected), 'Clusters', 'MANAGE YOUR STORAGE' (with 'Backends' and 'Buckets'), and 'MANAGE YOUR ACCOUNT' (with 'Account', 'Activity', and 'Support'). The main area is titled 'Applications' and displays a table of discovered applications. The columns are: Actions, Name, State, Cluster, Group, Discovered, and Actions. The table contains six entries:

Actions	Name	State	Cluster	Group	Discovered	Actions
<input type="checkbox"/>	magento-5295b	Healthy	tkgi-acc	magento-5295b	2022/05/11 09:52 UTC	<input type="button" value="⋮"/>
<input type="checkbox"/>	magento	Healthy	tkgi-acc	magento	2022/05/09 18:20 UTC	<input type="button" value="⋮"/>
<input type="checkbox"/>	pks-system	Healthy	tkgi-acc	pks-system	2022/05/04 06:40 UTC	<input type="button" value="⋮"/>
<input type="checkbox"/>	netapp-acc-operator	Healthy	tkgi-acc	netapp-acc-operator	2022/05/04 06:40 UTC	<input type="button" value="⋮"/>
<input type="checkbox"/>	netapp-astra-cc	Healthy	tkgi-acc	netapp-astra-cc	2022/05/04 06:40 UTC	<input type="button" value="⋮"/>

- Navigate to Apps > Discovered and click the dropdown menu next to the application you would like to manage using Astra. Then click Manage.

The screenshot shows the same 'Applications' section as before, but with a callout highlighting the 'Manage' option in the dropdown menu for the 'magento' application. The 'Manage' button is located at the bottom right of the application row.

- The application enters the Available state and can be viewed under the Managed tab in the Apps section.

The screenshot shows the 'Applications' section of the Astra Control Center. At the top, there are buttons for 'Actions' (dropdown), '+ Define', 'All clusters' (dropdown), 'Search' (input field), 'Managed' (button), 'Discovered 60' (button), and 'Ignored' (button). Below the header is a table with the following columns: Name, State, Protection, Cluster, Group, Discovered, and Actions. The table contains one row for the application 'magento'. The 'Name' column shows a checkbox and the name 'magento'. The 'State' column shows a green checkmark and the word 'Healthy'. The 'Protection' column shows an orange warning icon and the word 'Unprotected'. The 'Cluster' column shows a gear icon and 'tkgi-acc'. The 'Group' column shows a blue square and 'magento'. The 'Discovered' column shows the date '2022/05/09 18:20 UTC'. The 'Actions' column has a three-dot menu icon.

Next: Protect Your applications.

Protect your applications

After application workloads are managed by Astra Control Center, you can configure the protection settings for those workloads.

Create an application snapshot

A snapshot of an application creates an ONTAP Snapshot copy and a copy of the application metadata that can be used to restore or clone the application to a specific point in time based on that Snapshot copy.

- To take a snapshot of the application, navigate to the Apps > Managed tab and click the application you would like to make a Snapshot copy of. Click the dropdown menu next to the application name and click Snapshot.

The screenshot shows the details for the 'magento' application. On the left, under 'APPLICATION STATUS', it says 'Healthy'. On the right, under 'APPLICATION PROTECTION STATUS', it says 'Unprotected'. Below these are sections for 'Images' (listing docker.io/bitnami/elasticsearch:6.8.12-debian-10-r61, docker.io/bitnami/magento:2.4.1-debian-10-r14, docker.io/bitnami/mariadb:10.3.24-debian-10-r49), 'Protection schedule' (Disabled), 'Group' (magento), and 'Cluster' (tkgi-acc). To the right, a dropdown menu titled 'Actions' is open, showing options: Snapshot, Backup, Clone, Restore, and Unmanage.

- Enter the snapshot details, click Next, and then click Snapshot. It takes about a minute to create the snapshot, and the status becomes Available after the snapshot is successfully created.

Create an application backup

A backup of an application captures the active state of the application and the configuration of its resources, converts them into files, and stores them in a remote object storage bucket.

- For the backup and restore of managed applications in the Astra Control Center, you must configure superuser settings for the backing ONTAP systems as a prerequisite. To do so, enter the following commands.

```
ONTAP::> export-policy rule modify -vserver ocp-trident -policyname default -ruleindex 1 -superuser sys
ONTAP::> export-policy rule modify -policyname default -ruleindex 1 -anon 65534 -vserver ocp-trident
```

- To create a backup of the managed application in the Astra Control Center, navigate to the Apps > Managed tab and click the application that you want to take a backup of. Click the dropdown menu next to the application name and click Backup.

- Enter the backup details, select the object storage bucket to hold the backup files, click Next, and, after reviewing the details, click Backup. Depending on the size of the application and data, the backup can take several minutes, and the status of the backup becomes Available after the backup is completed

successfully.

The screenshot shows the 'Back up namespace application' interface. At the top, it says 'STEP 1/2: DETAILS'. On the left, there's a 'BACKUP DETAILS' section with a 'Name' field containing 'magento-backup-20220516212622' and a checkbox for 'Back up from an existing snapshot'. Below that is a 'BACKUP DESTINATION' section with a 'Bucket' dropdown showing 'na-tanzu-astra/na-astra-tkgi' (Available) and 'Default'. On the right, there's a sidebar titled 'CREATING APPLICATION BACKUPS' with instructions about Astra Control taking backups of application configuration and persistent storage. It also lists the namespace application 'magento' under 'Namespace application' and the cluster 'tkgi-acc' under 'Cluster'. At the bottom are 'Cancel' and 'Next →' buttons.

Restoring an application

At the push of a button, you can restore an application to the originating namespace in the same cluster or to a remote cluster for application protection and disaster recovery purposes.

1. To restore an application, navigate to the Apps > Managed tab and click the app in question. Click the dropdown menu next to the application name and click Restore.

The screenshot shows the 'APPLICATION STATUS' and 'APPLICATION PROTECTION STATUS' sections for the 'magento' application. In the 'APPLICATION STATUS' section, it says 'Healthy'. In the 'APPLICATION PROTECTION STATUS' section, it says 'Unprotected'. On the right, there's an 'Actions' dropdown menu with options: Snapshot, Backup, Clone, Restore, and Unmanage. The 'Cluster' dropdown shows 'tkgi-acc'.

2. Enter the name of the restore namespace, select the cluster you want to restore it to, and choose if you want to restore it from an existing snapshot or from a backup of the application. Click Next.

Restore namespace application

STEP 1/2: DETAILS

RESTORE DETAILS

Destination cluster	tkgi-acc	Destination namespace	magento
---------------------	----------	-----------------------	---------

RESTORE SOURCE

Application backup	State	On-Schedule/On-Demand	Created ↑
magento-backup-20220516212730	Healthy	On-Demand	2022/05/16 21:27 UTC

RESTORING APPLICATIONS

Astra Control can restore your application configuration and persistent storage. Select a source snapshot or backup for the restored application.

- Namespace application magento
- Namespace magento
- Cluster tkgi-acc

Cancel **Next →**

3. On the review pane, enter `restore` and click Restore after you have reviewed the details.

Restore namespace application

STEP 2/2: SUMMARY

REVIEW RESTORE INFORMATION

All existing resources associated with this namespace application will be deleted and replaced with the source backup "magento-backup-20220516212730" taken on 2022/05/16 21:27 UTC. Persistent volumes will be deleted and recreated. External resources with dependencies on this namespace application might be impacted.

We recommend taking a snapshot or a backup of your namespace application before proceeding.

BACKUP magento-backup-20220516212730	RESTORE magento
ORIGINAL GROUP magento	DESTINATION GROUP magento
ORIGINAL CLUSTER tkgi-acc	DESTINATION CLUSTER tkgi-acc
RESOURCE LABELS Config Maps app.kubernetes.io/name: elasticsearch +9 Deployments	RESOURCE LABELS Config Maps app.kubernetes.io/name: elasticsearch +9 Deployments

Are you sure you want to restore the namespace application "magento"?

Type `restore` below to confirm.

Confirm to restore
`restore`

Back **Restore ✓**

4. The new application goes to the Restoring state while Astra Control Center restores the application on the selected cluster. After all the resources of the application are installed and detected by Astra, the application goes to the Available state.

⌚ Applications

Actions		+ Define	All clusters	Search	Managed	Discovered 60	Ignored
⌚ 1-1 of 1 entries < >							
Name	State	Protection	Cluster	Group	Discovered	Actions	
<input type="checkbox"/> magento	Healthy	Unprotected	 tkgi-acc	 magento	2022/05/09 18:20 UTC		

Cloning an application

You can clone an application to the originating cluster or to a remote cluster for dev/test or application protection and disaster recovery purposes. Cloning an application within the same cluster on the same storage backend uses NetApp FlexClone technology, which clones the PVCs instantly and saves storage space.

1. To clone an application, navigate to the Apps > Managed tab and click the app in question. Click the dropdown menu next to the application name and click Clone.



The screenshot shows the application details for 'magento'. On the left, the 'APPLICATION STATUS' tab displays 'Healthy'. On the right, the 'APPLICATION PROTECTION STATUS' tab shows 'Unprotected'. A dropdown menu is open on the right side, listing actions: Snapshot, Backup, Clone, and Restore. Below the tabs, there are sections for 'Images' (listing docker.io/bitnami/elasticsearch:6.8.12-debian-10-r61, docker.io/bitnami/magento:2.4.1-debian-10-r14, docker.io/bitnami/mariadb:10.3.24-debian-10-r49), 'Protection schedule' (Disabled), 'Group' (magento), and 'Cluster' (tkg).

2. Enter the details of the new namespace, select the cluster you want to clone it to, and choose if you want to clone it from an existing snapshot, from a backup, or from the current state of the application. Click Next and then click Clone on the review pane after you have reviewed the details.

Clone namespace application

STEP 1/2: DETAILS

CLONE DETAILS

Clone namespace
magento-bef7f

Destination cluster
tkgi-acc

Clone from an existing snapshot or backup

CLONING APPLICATIONS

Astra Control can create a clone of your application configuration and persistent storage. Persistent storage backups are transferred from your object store, so choosing a clone from an existing backup will complete the fastest. Enter a clone name to get started.

Not all applications may support cloning.

Read more in [Clone applications](#).

- Namespace application
magento
- Namespace
magento
- Cluster
tkgi-acc

Cancel **Next →**

- The new application goes to the Discovering state while Astra Control Center creates the application on the selected cluster. After all the resources of the application are installed and detected by Astra, the application goes to the Available state.

Applications

<input type="checkbox"/>	Name	State	Protection	Cluster	Group	Discovered	Actions
<input type="checkbox"/>	magento-bef7f	Healthy	Unprotected	tkgi-acc	magento-bef7f	2022/05/16 21:31 UTC	⋮
<input type="checkbox"/>	magento	Healthy	Partially protected	tkgi-acc	magento	2022/05/09 18:20 UTC	⋮

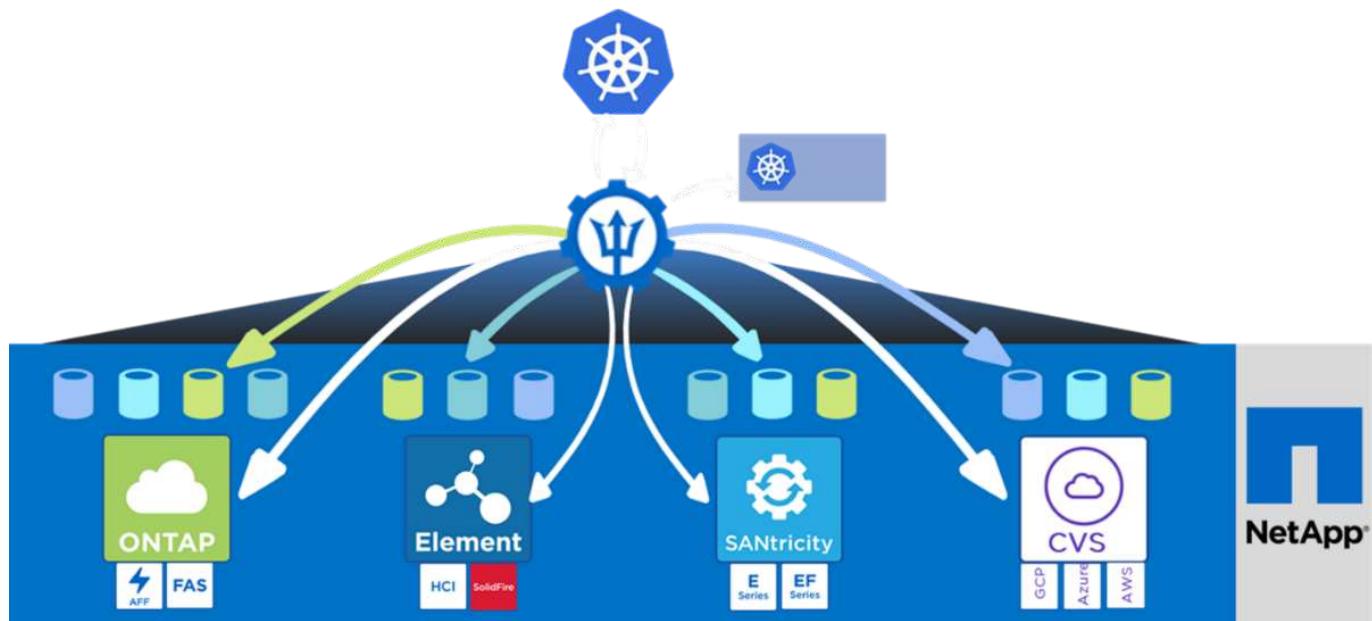
Next: Videos and demos: VMware Tanzu with NetApp.

Astra Trident overview

Astra Trident is an open-source, fully supported storage orchestrator for containers and Kubernetes distributions like Red Hat OpenShift, VMware Tanzu, Anthos by Google Cloud, Rancher etc. Trident works with the entire NetApp storage portfolio, including the 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, or 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 is 22.04 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 21.01 release, a Helm chart was made available to ease the installation of the Trident Operator.

Deploy Trident operator using Helm

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.

```
<<<<< HEAD
[netapp-user@rhel7]$ export KUBECONFIG=~/tanzu-install/auth/kubeconfig
=====
[netapp-user@rhel7]$ export KUBECONFIG=~/Tanzu-install/auth/kubeconfig
>>>>> eba1007b77b1ef6011dadd158f1df991acc5299f
```

2. Add the NetApp Astra Trident helm repository.

```
[netapp-user@rhel7]$ helm repo add netapp-trident
https://netapp.github.io/trident-helm-chart
"netapp-trident" has been added to your repositories
```

3. Update the helm repositories.

```
[netapp-user@rhel7]$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "netapp-trident" chart repository
...Successfully got an update from the "bitnami" chart repository
Update Complete. □Happy Helming!□
```

4. Create a new namespace for the installation of Trident.

```
[netapp-user@rhel7]$ kubectl create ns trident
```

5. Create a secret with DockerHub credentials to download the Astra Trident images.

```
[netapp-user@rhel7]$ kubectl create secret docker-registry docker-
registry-cred --docker-server=docker.io --docker-username=netapp
--solutions-tme --docker-password=xxxxxx -n trident
```

6. For user or workload clusters managed by TKGS (vSphere with Tanzu) or TKG with management cluster deployments, complete the following procedure to install Astra Trident:

- Ensure that the logged in user has the permissions to create service accounts in trident namespace and that the service accounts in trident namespace have the permissions to create pods.
- Run the below helm command to install Trident operator in the namespace created.

```
[netapp-user@rhel7]$ helm install trident netapp-trident/trident-
operator -n trident --set imagePullSecrets[0]=docker-registry-cred
```

7. For a user or workload cluster managed by TKGI deployments, run the following helm command to install Trident operator in the namespace created.

```
[netapp-user@rhel7]$ helm install trident netapp-trident/trident-
operator -n trident --set imagePullSecrets[0]=docker-registry-
cred,kubeletDir="/var/vcap/data/kubelet"
```

8. Verify that the Trident pods are up and running.

NAME	READY	STATUS	RESTARTS
AGE			
trident-csi-6vv62	2/2	Running	0
14m			
trident-csi-cfd844bcc-sqhcg	6/6	Running	0
12m			
trident-csi-dfcmz	2/2	Running	0
14m			
trident-csi-pb2n7	2/2	Running	0
14m			
trident-csi-qsw6z	2/2	Running	0
14m			
trident-operator-67c94c4768-xw978	1/1	Running	0
14m			
[netapp-user@rhel17]\$./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 links below to continue the setup and configuration of Astra Trident.

- [NetApp ONTAP NFS](#)
- [NetApp ONTAP iSCSI](#)

Next: [Videos and demos: VMware Tanzu with NetApp](#).

NetApp ONTAP NFS configuration

To enable Trident integration with the NetApp ONTAP storage system via NFS, you must create a backend that enables communication with the storage system. We configure a basic backend in this solution, but if you are looking for more customized options, visit the documentation [here](#).

Create an SVM in ONTAP

1. Log into ONTAP System Manager, navigate to Storage > Storage VMs, and click Add.
2. Enter a name for the SVM, enable the NFS protocol, check the Allow NFS Client Access checkbox, and add the subnets that your worker nodes are on in the export policy rules for allowing the volumes to be mounted as PVs in your workload clusters.

Add Storage VM

X

STORAGE VM NAME

trident_svm

Access Protocol

SMB/CIFS, NFS, S3

iSCSI

Enable SMB/CIFS

Enable NFS

Allow NFS client access

Add at least one rule to allow NFS clients to access volumes in this storage VM. [?](#)

EXPORT POLICY

Default

RULES

Rule Index	Clients	Access Protocols	Read-Only Rule	Read/Wr
	0.0.0.0/0	Any	Any	Any



If you are using NAT'ed deployment of user clusters or workload clusters with NSX-T, you need to add the Egress subnet (in the case of TKGS0 or the Floating IP subnet (in the case of TKG1) to the export policy rules.

3. Provide the details for data LIFs and the details for SVM administration account, and then click Save.

NETWORK INTERFACE

Use multiple network interfaces when client traffic is high.

K8s-Ontap-01

IP ADDRESS	SUBNET MASK	GATEWAY	BROADCAST DOMAIN
172.21.252.180	24	172.21.252.1 	Default 

Storage VM Administration

Manage administrator account

USER NAME

vsadmin

PASSWORD

CONFIRM PASSWORD

Add a network interface for storage VM management.

-
4. Assign the aggregates to an SVM. Navigate to Storage > Storage VMs, click the ellipsis next to the newly created SVM and then click Edit. Check the Limit Volume Creation to Preferred Local Tiers checkbox and attach the required aggregates to it.

Edit Storage VM

X

STORAGE VM NAME

DEFAULT LANGUAGE

 ▼

DELETED VOLUME RETENTION PERIOD ?

HOURS

Resource Allocation

Limit volume creation to preferred local tiers

LOCAL TIERS

 X

[Cancel](#)

[Save](#)

5. In case of NAT'ed deployments of user or workload clusters on which Trident is to be installed, the storage mount request might arrive from a non-standard port due to SNAT. By default, ONTAP only allows the volume mount requests when originated from root port. Thus, log into ONTAP CLI and modify the setting to

allow mount requests from non-standard ports.

```
ontap-01> vserver nfs modify -vserver tanzu_svm -mount-rootonly disabled
```

Create backends and StorageClasses

1. For NetApp ONTAP systems serving NFS, create a backend config file on the jumphost with the backendName, managementLIF, dataLIF, svm, username, password, and other details.

```
{  
    "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": "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.

2. Create the Trident backend by running the following command.

```
[netapp-user@rhel7]$ ./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 |  
+-----+-----+  
+-----+-----+-----+
```

3. With the backend created, you must next create a storage class. The following sample storage class definition highlights the required and basic fields. The parameter `backendType` should reflect the storage driver from the newly created Trident backend.

```

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

```

4. Create the storage class by running the kubectl command.

```
[netapp-user@rhel7 trident-installer]$ kubectl create -f storage-class-nfs.yaml
storageclass.storage.k8s.io/ontap-nfs created
```

5. With the storage class created, you must then create the first persistent volume claim (PVC). A sample PVC definition is given below. Make sure that the storageClassName field matches the name of the storage class just created. The PVC definition can be further customized as required depending upon the workload to be provisioned.

```

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

```

6. 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.

```
[netapp-user@rhel7 trident-installer]$ kubectl create -f pvc-basic.yaml
persistentvolumeclaim/basic created
```

NAME	STATUS	VOLUME	CAPACITY
ACCESS MODES		STORAGECLASS	AGE
basic	Bound	pvc-b4370d37-0fa4-4c17-bd86-94f96c94b42d	1Gi
RWO		ontap-nfs	7s

Next: Videos and demos: VMware Tanzu with NetApp.

NetApp ONTAP iSCSI configuration

To integrate NetApp ONTAP storage system with VMware Tanzu Kubernetes clusters for persistent volumes via iSCSI, the first step is to prepare the nodes by logging into each node and configuring the iSCSI utilities or packages to mount iSCSI volumes. To do so, follow the procedure laid out in this [link](#).



NetApp does not recommend this procedure for NAT'ed deployments of VMware Tanzu Kubernetes clusters.



TKGI uses Bosh VMs as nodes for Tanzu Kubernetes clusters that run immutable configuration images, and any manual changes of iSCSI packages on Bosh VMs do not remain persistent across reboots. Therefore, NetApp recommends using NFS volumes for persistent storage for Tanzu Kubernetes clusters deployed and operated by TKGI.

After the cluster nodes are prepared for iSCSI volumes, you must create a backend that enables communication with the storage system. We configured a basic backend in this solution, but, if you are looking for more customized options, visit the documentation [here](#).

Create an SVM in ONTAP

To create an SVM in ONTAP, complete the following steps:

1. Log into ONTAP System Manager, navigate to Storage > Storage VMs, and click Add.
2. Enter a name for the SVM, enable the iSCSI protocol, and then provide details for the data LIFs.

Add Storage VM

X

STORAGE VM NAME

trident_svm_iscsi

Access Protocol

SMB/CIFS, NFS, S3

iSCSI

Enable iSCSI

NETWORK INTERFACE

K8s-Ontap-01

IP ADDRESS

SUBNET MASK

GATEWAY

BROADCAST DOMAIN

10.61.181.231

24

10.61.181.1 

Defa... 

Use the same subnet mask, gateway, and broadcast domain for all of the following interfaces

IP ADDRESS

SUBNET MASK

GATEWAY

BROADCAST DOMAIN

10.61.181.232

24

10.61.181.1 

Defa... 

3. Enter the details for the SVM administration account, and then click Save.

Storage VM Administration

Manage administrator account

USER NAME

vsadmin

PASSWORD

CONFIRM PASSWORD

Add a network interface for storage VM management.

Save

Cancel

4. To assign the aggregates to the SVM, navigate to Storage > Storage VMs, click the ellipsis next to the newly created SVM, and then click Edit. Check the Limit Volume Creation to Preferred Local Tiers checkbox, and attach the required aggregates to it.

Edit Storage VM

X

STORAGE VM NAME

DEFAULT LANGUAGE



DELETED VOLUME RETENTION PERIOD [?](#)

HOURS

Resource Allocation

Limit volume creation to preferred local tiers

LOCAL TIERS

Cancel

Save

Create backends and StorageClasses

1. For NetApp ONTAP systems serving NFS, create a backend config file on the jumphost with the backendName, managementLIF, dataLIF, svm, username, password, and other details.

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "ontap-san+10.61.181.231",
  "managementLIF": "172.21.224.201",
  "dataLIF": "10.61.181.231",
  "svm": "trident_svm_iscsi",
  "username": "admin",
  "password": "password"
}
```

2. Create the Trident backend by running the following command.

```
[netapp-user@rhel7 trident-installer]$ ./tridentctl -n trident create
backend -f backend-ontap-san.json
+-----+
+-----+-----+
|           NAME           | STORAGE DRIVER |          UUID
| STATE | VOLUMES |
+-----+-----+
+-----+-----+-----+
| ontap-san+10.61.181.231 | ontap-san       | 6788533c-7fea-4a35-b797-
fb9bb3322b91 | online |      0 |
+-----+-----+
+-----+-----+
```

3. After you create a backend, you must next create a storage class. The following sample storage class definition highlights the required and basic fields. The parameter `backendType` should reflect the storage driver from the newly created Trident backend. Also note the `name`-field value, which must be referenced in a later step.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-iscsi
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 can be deleted to allow Tanzu Kubernetes clusters to decide what filesystem to use.

4. Create the storage class by running the kubectl command.

```
[netapp-user@rhel7 trident-installer]$ kubectl create -f storage-class-iscsi.yaml
storageclass.storage.k8s.io/ontap-iscsi created
```

5. With the storage class created, you must then create the first persistent volume claim (PVC). A sample PVC definition is given below. Make sure that the storageClassName field matches the name of the storage class just created. The PVC definition can be further customized as required depending upon the workload to be provisioned.

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

6. 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.

```
[netapp-user@rhel7 trident-installer]$ kubectl create -f pvc-basic.yaml
persistentvolumeclaim/basic created

[netapp-user@rhel7 trident-installer]$ kubectl get pvc
  NAME      STATUS      VOLUME                                     CAPACITY
  ACCESS MODES   STORAGECLASS      AGE
  basic     Bound      pvc-7ceac1ba-0189-43c7-8f98-094719f7956c   1Gi
  RWO                  ontap-iscsi        3s
```

Next: [Solution validation/use cases](#).

Videos and demos: VMware Tanzu with NetApp

The following videos demonstrate some of the capabilities described in this document:

- [Use Astra Trident to Provision Persistent Storage in VMware Tanzu](#)
- [Use Astra Control Center to Clone Applications in VMWare Tanzu](#)

Next: [Additional Information: VMware Tanzu with NetApp](#).

Additional Information: VMware Tanzu with NetApp

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

- NetApp Documentation

<https://docs.netapp.com/>

- Astra Trident Documentation

<https://docs.netapp.com/us-en/trident/>

- NetApp Astra Control Center Documentation

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

- Ansible Documentation

<https://docs.ansible.com/>

- VMware Tanzu Documentation

<https://docs.vmware.com/en/VMware-Tanzu/index.html>

- VMware Tanzu Kubernetes Grid Documentation

<https://docs.vmware.com/en/VMware-Tanzu-Kubernetes-Grid/1.5/vmware-tanzu-kubernetes-grid-15/GUID-index.html>

- VMware Tanzu Kubernetes Grid Service Documentation

<https://docs.vmware.com/en/VMware-vSphere/7.0/vmware-vsphere-with-tanzu/GUID-152BE7D2-E227-4DAA-B527-557B564D9718.html>

- VMware Tanzu Kubernetes Grid Integrated Edition Documentation

<https://docs.vmware.com/en/VMware-Tanzu-Kubernetes-Grid-Integrated-Edition/index.html>

Archived Solutions

WP-7337: Anthos on Bare Metal

Alan Cowles and Nikhil M Kulkarni, NetApp

NetApp and Google Cloud have had a strong relationship for several years now, with NetApp first introducing cloud data services for Google Cloud with Cloud Volumes ONTAP and the Cloud Volumes Service. This relationship was then expanded by validating the NetApp HCI platform for use with Google Cloud Anthos on-premises, a hypervisor-based hybrid multi-cloud Kubernetes solution deployed on VMware vSphere. NetApp then passed Anthos Ready qualification for NetApp Astra Trident, ONTAP, and the NFS protocol to provide dynamic persistent storage for containers.

Anthos can now be directly installed on bare-metal servers in a customer's environment, which adds an additional option for customers to extend Google Cloud into their local data centers without a hypervisor. Additionally, by leveraging the capabilities of NetApp ONTAP storage operating system and NetApp Astra

Trident, you can extend your platform's capabilities by integrating persistent storage for containers.

This combination allows you to realize the full potential of your servers, storage, and networking combined with the support, service levels, monthly billing, and on-demand flexibility that Google Cloud provides. Because you are using your own hardware, network, and storage, you have direct control over application scale, security, and network latency, as well as having the benefit of managed and containerized applications with Anthos on bare metal.

[Next: Solution overview.](#)

Solution overview

NetApp ONTAP on NetApp AFF/FAS

NetApp AFF is a robust all-flash storage platform that provides low-latency performance, integrated data protection, multiprotocol support, and nondisruptive operations. Powered by NetApp ONTAP data management software, NetApp AFF ensures nondisruptive operations, from maintenance to upgrades to complete replacement of your storage system.

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.

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 non-rewritable 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.



Across the NetApp data fabric, you can count on a common set of features and fast, efficient replication across platforms. You can use the same interface and the same data management tools.

NetApp Astra Trident

NetApp Astra Trident is an open-source and fully supported storage orchestrator for containers and Kubernetes distributions, including Google Cloud Anthos. It works with the entire NetApp storage portfolio, including NetApp ONTAP software. Trident is fully CSI-compliant, and it accelerates the DevOps workflow by allowing you to provision and manage storage from your NetApp storage systems, without intervention from a storage administrator. Trident is deployed as an operator that communicates directly with the Kubernetes API endpoint to serve containers' storage requests in the form of persistent volume claims (PVCs) by creating and managing volumes on the NetApp storage system.

Persistent volumes (PVs) are provisioned based on storage classes defined in the Kubernetes environment. They use storage backends created by a storage administrator (which can be customized based on project needs) and storage system models to allow for any number of advanced storage features, such as compression, specific disk types, or QoS levels that guarantee performance.

For more information about NetApp Astra Trident, see the [Trident](#) page.

Trident orchestrates storage from each system and service in the NetApp portfolio.



Google Cloud's Anthos

Google Cloud's Anthos is a cloud-based 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 bare metal extends the capability of Anthos to run on-premises directly on physical servers without a hypervisor layer and interoperate with Anthos GKE clusters 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.

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. Figure 3 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](#).

The following figure presents Google Cloud's Anthos architecture.



Anthos on bare metal

Anthos on bare metal is an extension of GKE that is deployed in a customer's private data center. An organization can deploy the same applications designed to run in containers in Google Cloud in Anthos clusters on-premises. Anthos on bare metal runs directly on physical servers with the user's choice of underlying Linux operating system and provides customers with a full-fledged hybrid cloud environment with the capability to run at the core or edge of their data centers.

Anthos on bare metal offers the following benefits:

- **Hardware agnostic.** Customers can run Anthos on their choice of optimized hardware platform in their existing data centers.
- **Cost savings.** You can realize significant cost savings by using your own physical resources for application deployments instead of provisioning resources in the Google Cloud environment.
- **Develop then publish.** You can use on-premises deployments while applications are in development, which allows for the testing of applications in the privacy of your local data center before you make them publicly available in the cloud.

- **Better performance.** Intensive applications that demand low latency and the highest levels of performance can be run closer to the hardware.
- **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.
- **Management and operations.** Anthos on bare metal comes with a wide range of facilities that increase operational efficiency such as built-in networking, lifecycle management, diagnostics, health checks, logging, and monitoring.

[Next: Solution requirements.](#)

Solution requirements

Hardware requirements

Compute: bring your own server

The hardware-agnostic capabilities of Anthos on bare metal allow you to select a compute platform optimized for your use case. Therefore, you can match your existing infrastructure and reduce capital expenditure.

The following table lists the minimum number of compute hardware components that are required to implement this solution, although the hardware models used can vary based on customer requirements.

Usage	Hardware and model	Quantity
Admin nodes	Cisco UCS B200	3
Worker nodes	HP Proliant DL360	4

Storage: NetApp ONTAP

The following table lists the minimum number of storage-hardware components needed to implement the solution, although the hardware models used can vary based on customer requirements.

Hardware	Model	Quantity
NetApp AFF	NetApp AFF A300	2 (1 HA pair)

Software requirements

The software versions identified in the following table were used by NetApp and our partners to validate the solution with NetApp, although the software components used can vary based on customer requirements.

Software	Purpose	Version
Ubuntu	OS on 3 Admins	20.04
	OS on Worker4	20.04
	OS on Worker3	18.04
CentOS	OS on Worker2	8.2
Red Hat Enterprise Linux	OS on Worker1	8.1
Anthos on bare metal	Container Orchestration	1.6.0

Software	Purpose	Version
NetApp ONTAP	Storage OS	9.7P8
NetApp Astra Trident	Container Storage Management	20.10



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

For Anthos on bare metal hardware and software requirements, see the [Anthos on bare metal documentation page](#).

[Next: Deployment summary.](#)

Deployment summary

For the initial validation of this solution, NetApp partnered with World Wide Technology (WWT) to establish an environment at WWT's Advanced Technology Center (ATC). Anthos was deployed on a bare metal infrastructure using the `bmctl` tool provided by Google Cloud. The following section details the deployment used for validation purposes.

The Anthos on bare metal with NetApp solution was built as a highly available hybrid cluster with three Anthos control-plane nodes and four Anthos worker nodes.

The control-plane nodes used were Cisco UCS B200M3 blade servers hosted in a chassis and configured with a single virtual network interface card (vNIC) on each, which allowed for A/B failover at the Cisco UCS platform level for fault tolerance. The Cisco UCS chassis connected upstream to a pair of Cisco UCS 6248 fabric interconnects providing disparate paths for the separation of traffic along fabric A and fabric B. Those fabric interconnects connected upstream to a pair of Cisco Nexus 5548 data center switches that tied back to the core network at WWT.

The worker nodes were HP Proliant DL360 nodes, each running one of the supported Linux distributions for Anthos on bare metal: Red Hat Enterprise Linux 8.2, CentOS 8.2, Ubuntu 20.04 LTS, or Ubuntu 18.04 LTS. The Red Hat Enterprise Linux 8 and CentOS 8 nodes were configured with NIC teams running in LACP mode and cabled to two Nexus 9k C93180YC-FX switches for fault tolerance. The Ubuntu servers were configured for network bonding in LACP mode and cabled to the same pair of Nexus 9k switches for fault tolerance.

The NetApp AFF A300 storage system running ONTAP 9.7 software was installed and connected physically to the same pair of Nexus 9k switches as the Anthos worker nodes. These network uplinks were aggregated into an interface group (a0a), and the appropriate data network VLAN was tagged to allow the worker nodes to interact with the storage system. A storage virtual machine (SVM) was created with data LIFs supporting the NFS protocol and dedicated to storage operations for Trident to provide persistent storage to the containers deployed in the Anthos on bare metal cluster. These persistent volumes were provided by NetApp Astra Trident 20.10, the latest release of the fully supported NetApp open-source storage orchestrator for Kubernetes.

The following figure depicts a physical cabling diagram of the solution to the top of rack data center switches.



The next figure presents a logical view of the solution as deployed and validated on the hardware in the lab at the NetApp partner WWT.



Next: Solution validation.

Solution validation

The current deployment of this solution was put through two rigorous validation processes using tools provided by the Google Cloud team. These validations include a subset of the following tests:

- Partner validation of the Anthos-ready platform:
 - Confirm that all Anthos on bare metal platform services are installed and running.
 - Scale down the physical Anthos on bare metal cluster from four worker nodes to three and then back to four.
 - Create and delete a custom namespace.
 - Create a deployment of the Nginx web server, scaling that deployment by increasing the number of replicas.

- Create an ingress for the Nginx application and verify connectivity by curling the index.html.
- Successfully clean up all test suite activities and return the cluster to a pretest state.
- Partner validation of Anthos-ready storage:
 - Create a deployment with a persistent volume claim.
 - Use NetApp Astra Trident to provision and attach the requested persistent volume from NetApp ONTAP.
 - Validate the detach-and-reattach capability of persistent volumes.
 - Validate multi-attach, read-only access of persistent volumes from other pods on the node.
 - Validate the offline volume resize operation.
 - Verify that the persistent volume survives a cluster-scaling operation.

Next: Conclusion.

Conclusion

Anthos on bare metal with NetApp provides a robust platform to run container-based workloads efficiently by allowing for the customization of deployed infrastructure. Customers can use the server infrastructure and supported operating system of their choice or even deploy the solution within their existing infrastructure. The power and flexibility of these environments increases greatly through the integration of NetApp ONTAP and NetApp Astra Trident, supporting stateful application workloads by efficiently provisioning and managing persistent storage for containers. By extending the potential of Google Cloud into their data center powered by NetApp, a customer can realize the benefits of a fully supported, highly available, easily scalable, and fully managed Kubernetes solution for development and production of their application workloads.

Next: Where to find additional information.

Where to find additional information

To learn more about the information that is described in this document, review the following documents and/or websites:

- NetApp ONTAP Documentation Center
<https://docs.netapp.com/ontap-9/index.jsp>
- NetApp Astra Trident
<https://netapp-trident.readthedocs.io/en/stable-v20.10/>
- Google Cloud's Anthos
<https://cloud.google.com/anthos>
- Anthos on bare metal
<https://cloud.google.com/anthos/gke/docs/bare-metal>

Copyright Information

Copyright © 2022 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means-graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system-without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademark Information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.