



Hybrid Cloud with Provider-Managed Components

NetApp Solutions

NetApp
April 25, 2024

This PDF was generated from <https://docs.netapp.com/us-en/netapp-solutions/rhhc/rhhc-pm-solution.html> on April 25, 2024. Always check docs.netapp.com for the latest.

Table of Contents

- NetApp Hybrid Multicloud solutions for Red Hat OpenShift Container workloads 1
 - Overview 1
 - Deploy and configure the Managed Red Hat OpenShift Container platform on AWS 4
 - Data protection 6
 - Data migration 22

NetApp Hybrid Multicloud solutions for Red Hat OpenShift Container workloads

Overview

NetApp is seeing a significant increase in customers modernizing their legacy enterprise applications and building new applications using containers and orchestration platforms built around Kubernetes. Red Hat OpenShift Container Platform is one example that we see adopted by many of our customers.

As more and more customers begin adopting containers within their enterprises, NetApp is perfectly positioned to help serve the persistent storage needs of their stateful applications and classic data management needs such as data protection, data security, and data migration. However, these needs are met using different strategies, tools, and methods.

NetApp ONTAP based storage options listed below, deliver security, data protection, reliability, and flexibility for containers and Kubernetes deployments.

- Self-managed storage in on-premises:
 - NetApp Fabric Attached Storage (FAS), NetApp All Flash FAS Arrays (AFF), NetApp All SAN Array (ASA) and ONTAP Select
- Provider-managed storage in on-premises:
 - NetApp Keystone provides Storage as a Service (STaaS)
- Self-managed storage in the cloud:
 - NetApp Cloud Volumes ONTAP(CVO) provide self managed storage in the hyperscalers
- Provider-managed storage in the cloud:
 - Cloud Volumes Service for Google Cloud (CVS), Azure NetApp Files (ANF), Amazon FSx for NetApp ONTAP offer fully managed storage in the hyperscalers

ONTAP feature highlights



Storage Administration

- Multi-tenancy
- FlexVol & FlexGroup
- LUN
- Quotas
- ONTAP CLI & API
- System Manager & BlueXP

Performance & Scalability

- FlexCache
- FlexClone
- nconnect, session trunking, multipathing
- Scale-out clusters

Availability & Resilience

- Multi-AZ HA deployment (MetroCluster)
- SnapShot & SnapRestore
- SnapMirror
- SnapMirror Business Continuity
- SnapMirror Cloud

Access Protocols

- NFS –v3, v4, v4.1, v4.2
- SMB – v2, v3
- iSCSI
- Multi-protocol access

Storage Efficiency

- Deduplication & Compression
- Compaction
- Thin provisioning
- Data Tiering (Fabric Pool)

Security & Compliance

- Fpolicy & Vscan
- Active Directory integration
- LDAP & Kerberos
- Certificate based authentication

NetApp BlueXP enables you to manage all of your storage and data assets from a single control plane/interface.

You can use BlueXP to create and administer cloud storage (for example, Cloud Volumes ONTAP and Azure NetApp Files), to move, protect, and analyze data, and to control many on-prem and edge storage devices.

NetApp Astra Trident is a CSI Compliant Storage Orchestrator that enable quick and easy consumption of persistent storage backed by a variety of the above-mentioned NetApp storage options. It is an open-source software maintained and supported by NetApp.

Astra Trident CSI feature highlights



CSI specific <ul style="list-style-type: none">• CSI NetApp® Snapshot™ copies and volume creation from CSI Snapshot copies• CSI topology• Volume expansion	Security <ul style="list-style-type: none">• Dynamic-export policy management• iSCSI initiator-groups dynamic management• iSCSI bidirectional CHAP
Control <ul style="list-style-type: none">• Storage and performance consumption• Monitoring• Volume Import• Cross Namespace Volume Access	Installation methods <ul style="list-style-type: none">• Binary• Helm chart• Operator• GitOps
Choose your access mode <ul style="list-style-type: none">• RWO (ReadWriteOnce, i.e 1↔1)• RWX (ReadWriteMany, i.e 1↔n)• ROX (ReadOnlyMany)• RWOP (ReadWriteOnce POD)	Choose your protocol <ul style="list-style-type: none">• NFS• SMB• iSCSI

Business critical container workloads need more than just persistent volumes. Their data management requirements require protection and migration of the application kubernetes objects as well.



Application data includes kubernetes objects in addition to the user data: Some examples are as follows:

- kubernetes objects such as pods specs, PVCs, deployments, services
- custom config objects such as config maps and secrets
- persistent data such as Snapshot copies, backups, clones
- custom resources such as CRs and CRDs

NetApp Astra Control, available as both fully-managed and self-managed software, provides orchestration for robust application data management. Refer to the [Astra documentation](#) for additional details on the Astra family of products.

This reference documentation provides validation of migration and protection of container-based applications, deployed on RedHat OpenShift container platform, using NetApp Astra Control Center. In addition, the solution provides high-level details for the deployment and the use of Red Hat Advanced Cluster Management (ACM) for managing the container platforms. The document also highlights the details for the integration of NetApp storage with Red Hat OpenShift container platforms using Astra Trident CSI provisioner. Astra Control Center is deployed on the hub cluster and is used to manage the container applications and their persistent storage lifecycle. Finally, it provides a solution for replication and failover and fail-back for container workloads on managed Red Hat OpenShift clusters in AWS (ROSA) using Amazon FSx for NetApp ONTAP (FSxN) as

persistent storage.

NetApp Solution with Managed Red Hat OpenShift Container platform workloads on AWS

Customers may be "born in the cloud" or may be at a point in their modernization journey when they are ready to move some select workloads or all workloads from their data centers to the cloud. They may choose to use provider-managed OpenShift containers and provider-managed NetApp storage in the cloud for running their workloads. They should plan and deploy the Managed Red Hat OpenShift container clusters (ROSA) in the cloud for a successful production-ready environment for their container workloads. When they are in AWS cloud, they could also deploy FSx for NetApp ONTAP for the storage needs.

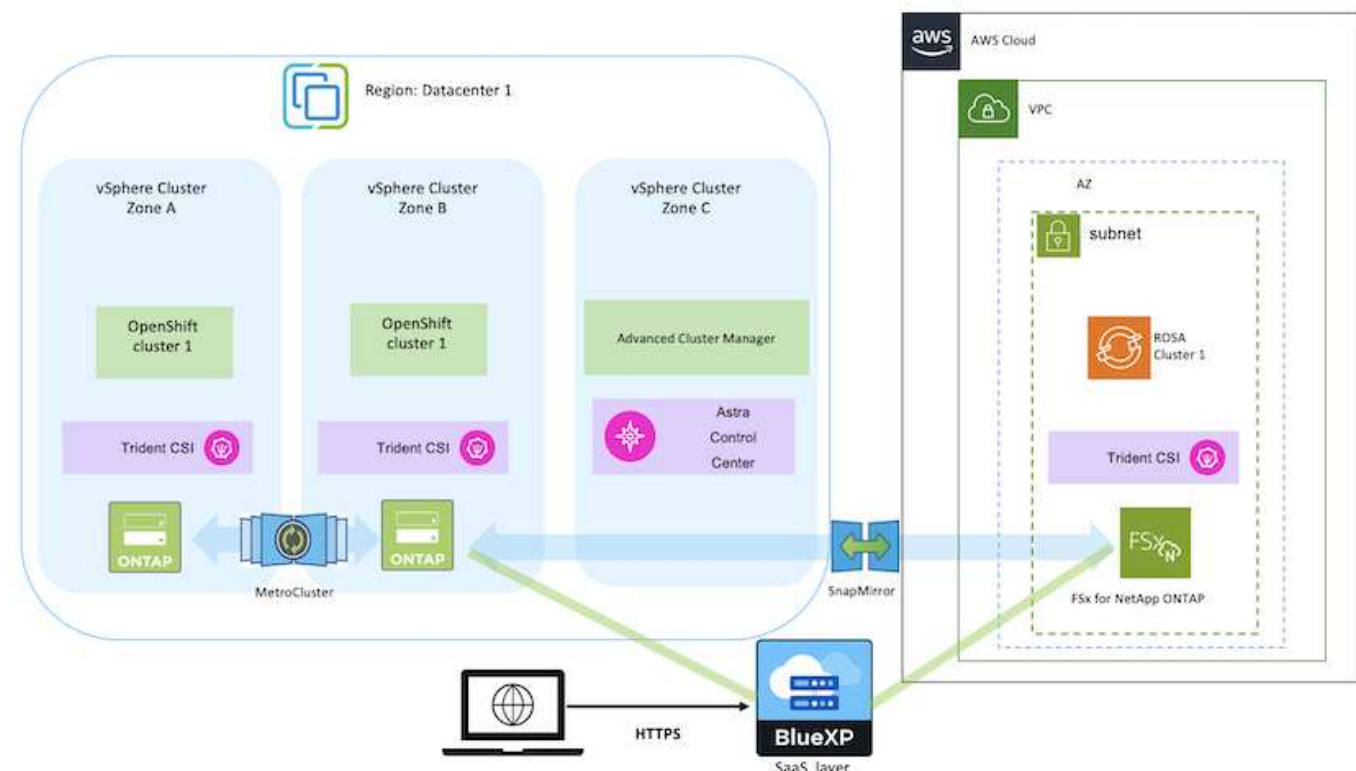
FSx for NetApp ONTAP delivers data protection, reliability, and flexibility for container deployments in AWS. Astra Trident serves as the dynamic storage provisioner to consume the persistent FSxN storage for customers' stateful applications.

As ROSA can be deployed in HA mode with control plane nodes spread across multiple availability zones, FSx ONTAP can also be provisioned with Multi-AZ option which provides high availability and protect against AZ failures.



There are no data transfer charges when accessing an Amazon FSx file system from the file system's preferred Availability Zone (AZ). For more info on pricing, refer [here](#).

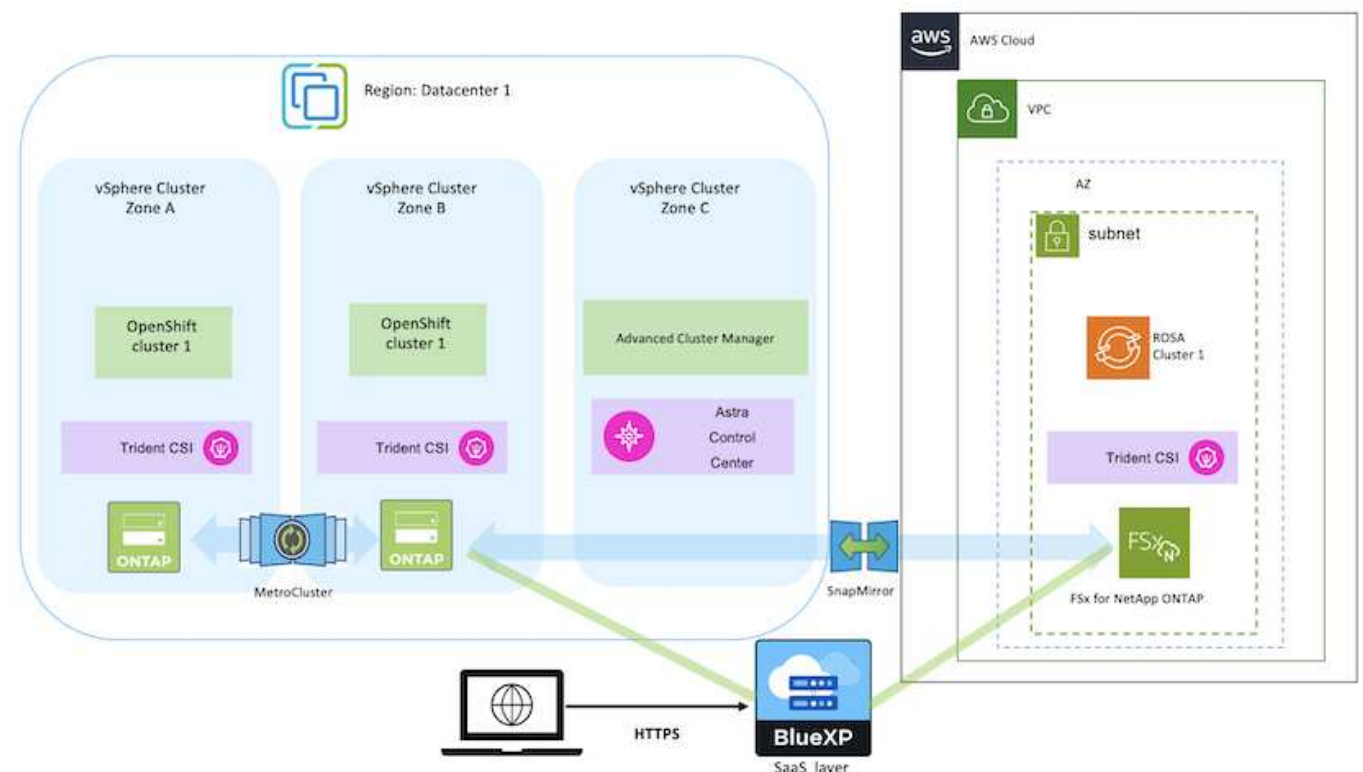
Data protection and migration solution for OpenShift Container workloads



Deploy and configure the Managed Red Hat OpenShift Container platform on AWS

This section describes a high-level workflow of setting up the Managed Red Hat OpenShift clusters on AWS (ROSA). It shows the use of Managed FSx for NetApp ONTAP (FSxN) as the storage backend by Astra Trident to provide persistent volumes. Details are provided about the deployment of FSxN on AWS using BlueXP. Also, details are provided about the use of BlueXP and OpenShift GitOps (Argo CD) to perform data protection and migration activities for the stateful applications on ROSA clusters.

Here is a diagram that depicts the ROSA clusters deployed on AWS and using FSxN as the backend storage.



This solution was verified by using two ROSA clusters in two VPCs in AWS. Each ROSA cluster was integrated with FSxN using Astra Trident. There are several ways of deploying ROSA clusters and FSxN in AWS. This high-level description of the setup provides documentation links for the specific method that was used. You can refer to the other methods in the relevant links provided in the [resources section](#).

The setup process can be broken down into the following steps:

Install ROSA clusters

- Create two VPCs and set up VPC peering connectivity between the VPCs.
- Refer [here](#) for instructions to install ROSA clusters.

Install FSxN

- Install FSxN on the VPCs from BlueXP.
Refer [here](#) for BlueXP account creation and to get started.
Refer [here](#) for installing FSxN.
Refer [here](#) for creating a connector in AWS to manage the FSxN.
- Deploy FSxN using AWS.
Refer [here](#) for deployment using AWS console.

Install Trident on ROSA clusters (using Helm chart)

- Use Helm chart to install Trident on ROSA clusters.
url for the Helm chart: <https://netapp.github.io/trident-helm-chart>

Integration of FSxN with Astra Trident for ROSA clusters



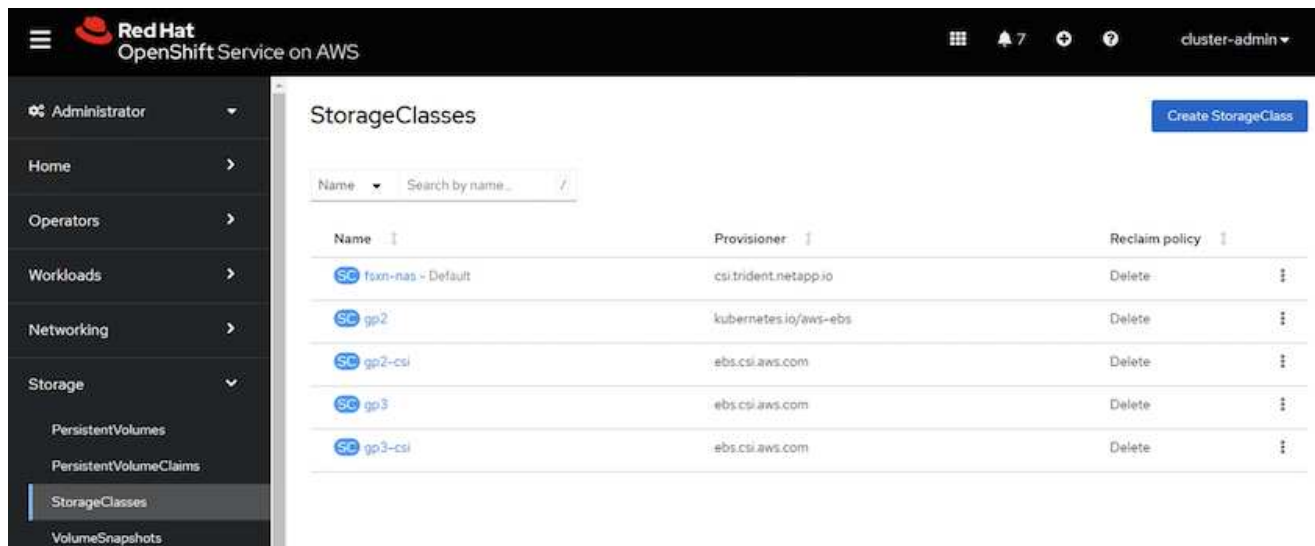
OpenShift GitOps can be utilized to deploy Astra Trident CSI to all managed clusters as they get registered to ArgoCD using ApplicationSet.

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet
metadata:
  name: trident-operator
spec:
  generators:
  - clusters: {}
    # selector:
    #   matchLabels:
    #     tridentversion: '23.04.0'
  template:
    metadata:
      name: '{{nameNormalized}}-trident'
    spec:
      destination:
        namespace: trident
        server: '{{server}}'
      source:
        repoURL: 'https://netapp.github.io/trident-helm-chart'
        targetRevision: 23.04.0
        chart: trident-operator
        project: default
        syncPolicy:
          syncOptions:
            - CreateNamespace=true
```



Create backend and storage classes using Trident (for FSxN)

- Refer [here](#) for details about creating backend and storage class.
- Make the storage class created for FsxN with Trident CSI as default from OpenShift Console. See screenshot below:



Deploy an application using OpenShift GitOps (Argo CD)

- Install OpenShift GitOps operator on the cluster. Refer to instructions [here](#).
- SetUp a new Argo CD instance for the cluster. Refer to instructions [here](#).

Open the console of Argo CD and deploy an app.

As an example, you can deploy a Jenkins App using Argo CD with a Helm Chart.

When creating the application, the following details were provided:

Project: default

cluster: <https://kubernetes.default.svc>

Namespace: Jenkins

The url for the Helm Chart: <https://charts.bitnami.com/bitnami>

Helm Parameters:

global.storageClass: fsxn-nas

Data protection

This page shows the data protection options for Managed Red Hat OpenShift on AWS (ROSA) clusters using Astra Control Service. Astra Control Service (ACS) provides an easy-to-use graphical user-interface with which you can add clusters, define applications running on them, and perform application aware data management activities. ACS functions can also be accessed using an API that allows for automation of workflows.

Powering Astra Control (ACS or ACC) is NetApp Astra Trident. Astra Trident integrates several types of Kubernetes clusters such as Red Hat OpenShift, EKS, AKS, SUSE Rancher, Anthos etc., with various flavors

of NetApp ONTAP storage such as FAS/AFF, ONTAP Select, CVO, Google Cloud Volumes Service, Azure NetApp Files and Amazon FSx for NetApp ONTAP.

This section provides details for the following data protection options using ACS:

- A video showing Backup and Restore of a ROSA application running in one region and restoring to another region.
- A video showing Snapshot and Restore of a ROSA application.
- Step-by-step details of installing a ROSA cluster, Amazon FSx for NetApp ONTAP, using NetApp Astra Trident to integrate with storage backend, installing a postgresql application on ROSA cluster, using ACS to create a snapshot of the application and restoring the application from it.
- A blog showing step-by-step details of creating and restoring from a snapshot for a mysql application on a ROSA cluster with FSx for ONTAP using ACS.

Backup/Restore from Backup

The following video shows the backup of a ROSA application running in one region and restoring to another region.

[FSx NetApp ONTAP for Red Hat OpenShift Service on AWS](#)

Snapshot/Restore from snapshot

The following video shows taking a snapshot of a ROSA application and restoring from the snapshot after.

[Snapshot/Restore for Applications on Red Hat OpenShift Service on AWS \(ROSA\)clusters with Amazon FSx for NetApp ONTAP storage](#)

Blog

- [Using Astra Control Service for data management of apps on ROSA clusters with Amazon FSx storage](#)

Step-by-Step Details to create snapshot and restore from it

Prerequisite setup

- [AWS account](#)
- [Red Hat OpenShift account](#)
- IAM user with [appropriate permissions](#) to create and access ROSA cluster
- [AWS CLI](#)
- [ROSA CLI](#)
- [OpenShift CLI\(oc\)](#)
- VPC with subnets and appropriate gateways and routes
- [ROSA Cluster installed](#) into the VPC
- [Amazon FSx for NetApp ONTAP](#) created in the same VPC
- Access to the ROSA cluster from [OpenShift Hybrid Cloud Console](#)

Next Steps

1. Create an admin user and login to the cluster.
2. Create a kubeconfig file for the cluster.
3. Install Astra Trident on the cluster.
4. Create a backend, storage class and snapshot class configuration using the Trident CSI provisioner.
5. Deploy a postgresql application on the cluster.
6. Create a database and add a record.
7. Add the cluster into ACS.
8. Define the application in ACS.
9. Create a snapshot using ACS.
10. Delete the database in the postgresql application.
11. Restore from a snapshot using ACS.
12. Verify your app has been restored from the snapshot.

1. Create an admin user and login to the cluster

Access the ROSA cluster by creating an admin user with the following command : (You need to create an admin user only if you did not create one at the time of installation)

```
rosa create admin --cluster=<cluster-name>
```

The command will provide an output that will look like the following. Login to the cluster using the `oc login` command provided in the output.

```
W: It is recommended to add an identity provider to login to this cluster.
See 'rosa create idp --help' for more information.
I: Admin account has been added to cluster 'my-rosa-cluster'. It may take up
to a minute for the account to become active.
I: To login, run the following command:
oc login https://api.my-rosa-cluster.abcd.p1.openshiftapps.com:6443 \
--username cluster-admin \
--password FWGYL-2mkJI-000000-000000
```



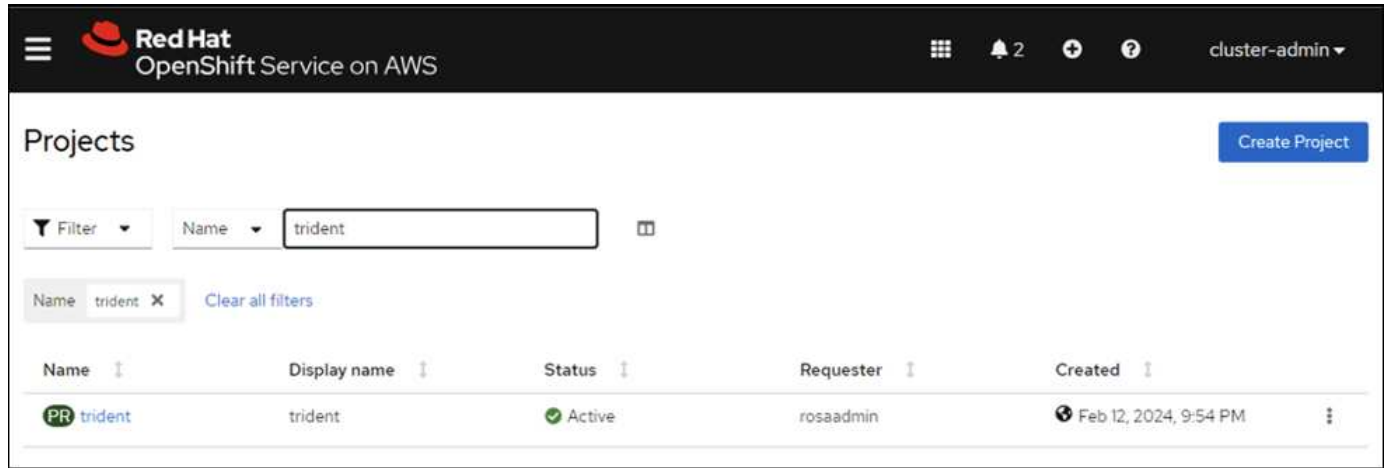
You can also login to the cluster using a token. If you already created an admin-user at the time of cluster creation, you can login to the cluster from the Red Hat OpenShift Hybrid Cloud console with the admin-user credentials. Then by clicking on the top right corner where it displays the name of the logged in user, you can obtain the `oc login` command (token login) for the command line.

2. Create a kubeconfig file for the cluster

Follow the procedures [here](#) to create a kubeconfig file for the ROSA cluster. This kubeconfig file will be used later when you add the cluster into ACS.

3. Install Astra Trident on the cluster

Install Astra Trident (latest version) on the ROSA cluster. To do this, you can follow any one of the procedures given [here](#). To install Trident using helm from the console of the cluster, first create a project called Trident.



Then from the Developer view, create a Helm chart repository. For the URL field use 'https://netapp.github.io/trident-helm-chart'. Then create a helm release for Trident operator.

Create Helm Chart Repository

Add helm chart repository.

Configure via: ☒ Form view ☐ YAML view

Scope type

☐ Namespaced scoped (ProjectHelmChartRepository)

Add Helm Chart Repository in the selected namespace.

☒ Cluster scoped (HelmChartRepository)

Add Helm Chart Repository at the cluster level and in all namespaces.

Name *

trident

A unique name for the Helm Chart repository.

Display name

Astra Trident

A display name for the Helm Chart repository.

Description

NetApp Astra Trident

A description for the Helm Chart repository.

☐ Disable usage of the repo in the developer catalog.

URL *

https://netapp.github.io/trident-helm-chart

Project: trident ▼

[Developer Catalog](#) > [Helm Charts](#)

Helm Charts

Browse for charts that help manage complex installations and upgrades. Cluster administrators can customize the catalog. Alternatively, developers can [try to configure their own custom Helm Chart repository](#).

All items

CI/CD

Languages

Other

Chart Repositories

☒ Astra Trident (1)

☐ OpenShift Helm Charts (87)

Source

☐ Community (33)


☐ Partner (42)

☐ Red Hat (12)

All items

Q Filter by keyword...

A-Z ▼



Helm Charts

Trident Operator

A Helm chart for deploying NetApp's Trident CSI storage provisioner using the Trident...

Verify all trident pods are running by going back to the Administrator view on the console and selecting pods in the trident project.

Red Hat
 OpenShift Service on AWS

Administrator

Home

Operators

Workloads

Podsnetworking

Deployments

DeploymentConfigs

StatefulSets

Secrets

ConfigMaps

CronJobs

Jobs

DaemonSets

ReplicaSets

ReplicationControllers

HorizontalPodAutoscalers

PodDisruptionBudgets

Project: trident

Filter

Name

Search by name...

Name	Status	Ready	Restarts	Owner	Memory
trident-controller-69cff44ddf-4dqnj	Running	6/6	0	trident-controller-69cff44ddf	-
trident-node-linux-4b6fm	Running	2/2	0	trident-node-linux	-
trident-node-linux-4sckw	Running	2/2	0	trident-node-linux	-
trident-node-linux-7l42w	Running	2/2	0	trident-node-linux	-
trident-node-linux-dbhp4	Running	2/2	0	trident-node-linux	-
trident-node-linux-gj5km	Running	2/2	0	trident-node-linux	-
trident-node-linux-r79c8	Running	2/2	0	trident-node-linux	-
trident-node-linux-tzwdp	Running	2/2	0	trident-node-linux	-
trident-node-linux-vdvxt	Running	2/2	0	trident-node-linux	-
trident-operator-7f7fd45c68-6crcb	Running	1/1	0	trident-operator-7f7fd45c68	-

4. Create a backend, storage class and snapshot class configuration using the Trident CSI provisioner

Use the yaml files shown below to create a trident backend object, storage class object and the Volumesnapshot object. Be sure to provide the credentials to your Amazon FSx for NetApp ONTAP file system you created, the management LIF and the vserver name of your file system in the configuration yaml for the backend. To get those details, go to the AWS console for Amazon FSx and select the file system, navigate to the Administration tab. Also, click on update to set the password for the `fsxadmin` user.



You can use the command line to create the objects or create them with the yaml files from the hybrid cloud console.

FSx > File systems > fs-049f9a23aac951429

fsx-for-rosa (fs-049f9a23aac951429)

▼ Summary

File system ID fs-049f9a23aac951429	SSD storage capacity 1024 GiB	<input type="button" value="Update"/>	Availability Zones us-west-2b
Lifecycle state Available	Throughput capacity 128 MB/s	<input type="button" value="Update"/>	Creation time 2024-02-12T20:15:23-05:00
File system type ONTAP	Provisioned IOPS 3072	<input type="button" value="Update"/>	
Deployment type Single-AZ	Number of HA pairs 1		

Network & security | Monitoring & performance | **Administration** | Storage virtual machines | Volumes | Backups | Updates | Tags

ONTAP administration

Management endpoint - DNS name management.fs-049f9a23aac951429.fsx.us-west-2.amazonaws.com	Management endpoint - IP address 10.49.9.135	ONTAP administrator username fsxadmin
Inter-cluster endpoint - DNS name intercluster.fs-049f9a23aac951429.fsx.us-west-2.amazonaws.com	Inter-cluster endpoint - IP address 10.49.9.49	ONTAP administrator password <input type="button" value="Update"/>
	10.49.9.251	

Trident Backend Configuration

```

apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-nas-secret
type: Opaque
stringData:
  username: fsxadmin
  password: <password>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: ontap-nas
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: <management lif>
  backendName: ontap-nas
  svm: fsx
  credentials:
    name: backend-tbc-ontap-nas-secret

```

Storage Class

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  media: "ssd"
  provisioningType: "thin"
  snapshots: "true"
allowVolumeExpansion: true

```

snapshot class

```

apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: trident-snapshotclass
driver: csi.trident.netapp.io
deletionPolicy: Delete

```


Verify that the backend, storage class and the trident-snapshotclass objects are created by issuing the commands shown below.

```

[ec2-user@ip-10-49-11-132 storage]$ kubectl get tbc -n trident
NAME          BACKEND NAME      BACKEND UUID          PHASE    STATUS
ontap-nas     ontap-nas         8a5e4583-2dac-46bb-b01e-fa7c3816f121  Bound    Success
[ec2-user@ip-10-49-11-132 storage]$ kubectl get sc
NAME          PROVISIONER          RECLAIMPOLICY    VOLUMEBINDINGMODE    ALLOWVOLUMEEXPANSION    AGE
gp2           kubernetes.io/aws-ebs Delete            WaitForFirstConsumer  true                    3h23m
gp2-csi       ebs.csi.aws.com      Delete            WaitForFirstConsumer  true                    3h19m
gp3 (default) ebs.csi.aws.com      Delete            WaitForFirstConsumer  true                    3h23m
gp3-csi       ebs.csi.aws.com      Delete            WaitForFirstConsumer  true                    3h19m
ontap-nas     csi.trident.netapp.io Delete            Immediate              true                    141m
[ec2-user@ip-10-49-11-132 storage]$ kubectl get Volumesnapshotclass
NAME          DRIVER          DELETIONPOLICY    AGE
csi-aws-vsc   ebs.csi.aws.com Delete            3h19m
trident-snapshotclass csi.trident.netapp.io Delete            6m56s
[ec2-user@ip-10-49-11-132 storage]$

```

At this time, an important modification you need to make is to set ontap-nas as the default storage class instead of gp3 so that the postgresql app you deploy later can use the default storage class. In the Openshift console of your cluster, under Storage select StorageClasses. Edit the annotation of the current default class to be false and add the annotation storageclass.kubernetes.io/is-default-class set to true for the ontap-nas storage class.



The screenshot shows the Red Hat OpenShift StorageClasses page. A modal titled "Edit annotations" is open, allowing editing of annotations for a selected StorageClass. The modal contains two input fields: "Key" with the value "storageclass.kubernetes.io/is-..." and "Value" with the value "false". There is a "+ Add more" link below the inputs. The background shows a table of StorageClasses with columns for Name, Reclaim policy, and actions (Delete and a menu icon). The StorageClasses listed are gp2, gp2-csi, gp3 - Default, gp3-csi, and ontap-nas.

Name	Reclaim policy	Actions
SC gp2		Delete
SC gp2-csi		Delete
SC gp3 - Default	ebs.csi.aws.com	Delete
SC gp3-csi	ebs.csi.aws.com	Delete
SC ontap-nas	csi.trident.netapp.io	Delete

StorageClasses

Create StorageClass

Name

Search by name...

Name	Provisioner	Reclaim policy
 gp2	kubernetes.io/aws-ebs	Delete
 gp2-csi	ebs.csi.aws.com	Delete
 gp3	ebs.csi.aws.com	Delete
 gp3-csi	ebs.csi.aws.com	Delete
 ontap-nas - Default	csi.trident.netapp.io	Delete

5. Deploy a postgresql application on the cluster

You can deploy the application from the command line as follows:

```
helm install postgresql bitnami/postgresql -n postgresql --create-namespace
```

```
[ec2-user@ip-10-49-11-132 astra]$ helm install postgresql bitnami/postgresql -n postgresql --create-namespace
NAME: postgresql
LAST DEPLOYED: Tue Feb 13 14:46:16 2024
NAMESPACE: postgresql
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
**CHART NAME: postgresql
**CHART VERSION: 14.0.4
**APP VERSION: 16.2.0

** Please be patient while the chart is being deployed **

PostgreSQL can be accessed via port 5432 on the following DNS names from within your cluster:

    postgresql.postgresql.svc.cluster.local - Read/Write connection

To get the password for "postgres" run:

    export POSTGRES_PASSWORD=$(kubectl get secret --namespace postgresql postgresql -o jsonpath="{.data.postgres-password}" | base64 -d)

To connect to your database run the following command:

    kubectl run postgresql-client --rm --tty -i --restart='Never' --namespace postgresql --image docker.io/bitnami/postgresql:16.2.0-debian-11-r1 --env="PGPASSWORD=$POSTGRES_PASSWORD" \
    --command -- psql --host postgresql -U postgres -d postgres -p 5432

    > NOTE: If you access the container using bash, make sure that you execute "/opt/bitnami/scripts/postgresql/entrypoint.sh /bin/bash" in order to avoid
the error "psql: local user with ID 1001} does not exist"

To connect to your database from outside the cluster execute the following commands:

    kubectl port-forward --namespace postgresql svc/postgresql 5432:5432 &
    PGPASSWORD=$POSTGRES_PASSWORD psql --host 127.0.0.1 -U postgres -d postgres -p 5432

WARNING: The configured password will be ignored on new installation in case when previous PostgreSQL release was deleted through the helm command. In that
case, old PVC will have an old password, and setting it through helm won't take effect. Deleting persistent volumes (PVs) will solve the issue.
[ec2-user@ip-10-49-11-132 astra]$
```

If you do not see the application pods running, then there might be an error caused due to security context constraints.

```
[ec2-user@ip-10-49-11-132 astra]$ kubectl get all -n postgresql
NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
service/postgresql                  ClusterIP      172.30.245.50  <none>          5432/TCP    12m
service/postgresql-hl               ClusterIP      None           <none>          5432/TCP    12m

NAME                                READY    AGE
statefulset.apps/postgresql          0/1      12m
[ec2-user@ip-10-49-11-132 astra]$ kubectl get events -n postgresql
LAST SEEN   TYPE      REASON              OBJECT                                          MESSAGE
2m39s       Normal    WaitForFirstConsumer persistentvolumeclaim/data-postgresql-0        waiting for first consumer to be created before binding
12m         Normal    SuccessfulCreate     statefulset/postgresql                        create Claim data-postgresql-0 Pod postgresql-0 in StatefulSet postg
resql success
107s        Warning   FailedCreate         statefulset/postgresql                        create Pod postgresql-0 in StatefulSet postgresql failed error: pods
"postgresql-0" is forbidden: unable to validate against any security context constraint: [provider "trident-controller": Forbidden: not usable by user or
serviceaccount, provider "anyuid": Forbidden: not usable by user or serviceaccount, provider restricted-v2: .spec.securityContext.fsGroup: Invalid value: [
int64{1001}: 1001 is not an allowed group, provider restricted-v2: .containers[0].runAsUser: Invalid value: 1001: must be in the ranges: [1001010000, 1001
019999], provider "restricted": Forbidden: not usable by user or serviceaccount, provider "nonroot-v2": Forbidden: not usable by user or serviceaccount, pr
ovider "nonroot": Forbidden: not usable by user or serviceaccount, provider "pcap-dedicated-admins": Forbidden: not usable by user or serviceaccount, provi
der "hostmount-anyuid": Forbidden: not usable by user or serviceaccount, provider "machine-api-termination-handler": Forbidden: not usable by user or servi
ceaccount, provider "hostnetwork-v2": Forbidden: not usable by user or serviceaccount, provider "hostnetwork": Forbidden: not usable by user or serviceacco
unt, provider "hostaccess": Forbidden: not usable by user or serviceaccount, provider "splunkforwarder": Forbidden: not usable by user or serviceaccount, p
rovider "trident-node-linux": Forbidden: not usable by user or serviceaccount, provider "node-exporter": Forbidden: not usable by user or serviceaccount, p
rovider "privileged": Forbidden: not usable by user or serviceaccount]
[ec2-user@ip-10-49-11-132 astra]$
```

Fix the error by editing the runAsUser and fsGroup fields in statefulset.apps/postgresql object with the uid that is in the output of the oc get project command as shown below.

```
[ec2-user@ip-10-49-11-132 astra]$ oc get project postgresql -o yaml | grep uid-range
openshift.io/sa.scc.uid-range: 1001010000/10000
[ec2-user@ip-10-49-11-132 astra]$ oc edit -n postgresql statefulset.apps/postgresql
statefulset.apps/postgresql edited
[ec2-user@ip-10-49-11-132 astra]$
```

postgresql app should be running and using persistent volumes backed by Amazon FSx for NetApp ONTAP storage.

```
[ec2-user@ip-10-49-11-132 astra]$ oc get pods -n postgresql
```

NAME	READY	STATUS	RESTARTS	AGE
postgresql-0	1/1	Running	0	2m46s

```
[ec2-user@ip-10-49-11-132 astra]$
```

```
[ec2-user@ip-10-49-11-132 storage]$ kubectl get pvc -n postgresql
```

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES	STORAGECLASS	AGE
data-postgresql-0	Bound	pvc-dd09524a-de75-4825-9424-03a9b91195ca	8Gi	RWO	ontap-nas	4m2s

```
[ec2-user@ip-10-49-11-132 storage]$
```

6. Create a database and add a record

```
[ec2-user@ip-10-49-11-132 astra]$ export POSTGRES_PASSWORD=$(kubectl get secret --namespace postgresql postgresql -o jsonpath="{.data.postgres-password}" | base64 -d)
[ec2-user@ip-10-49-11-132 astra]$ kubectl run postgresql-client --rm --tty -i --restart='Never' --namespace postgresql --image docker.io/bitnami/postgresql:16.2.0-debian-11-r1 --env="PGPASSWORD=$POSTGRES_PASSWORD" \
> --command -- psql --host postgresql -U postgres -d postgres -p 5432
Warning: would violate PodSecurity "restricted:v1.24": allowPrivilegeEscalation != false (container "postgresql-client" must set securityContext.allowPrivilegeEscalation=false), unrestricted capabilities (container "postgresql-client" must set securityContext.capabilities.drop=["ALL"]), runAsNonRoot != true (pod or container "postgresql-client" must set securityContext.runAsNonRoot=true), seccompProfile (pod or container "postgresql-client" must set securityContext.seccompProfile.type to "RuntimeDefault" or "Localhost")
If you don't see a command prompt, try pressing enter.

postgres=# CREATE DATABASE erp;
CREATE DATABASE
postgres=# \c erp
You are now connected to database "erp" as user "postgres".
erp=# CREATE TABLE PERSONS(ID INT PRIMARY KEY NOT NULL, FIRSTNAME TEXT NOT NULL, LASTNAME TEXT NOT NULL);
CREATE TABLE
erp=# INSERT INTO PERSONS VALUES(1,'John','Doe');
INSERT 0 1
erp=# \dt
          List of relations
 Schema | Name   | Type  | Owner
-----+-----+-----+-----
 public | persons | table | postgres
(1 row)

erp=# SELECT * FROM persons;
 id | firstname | lastname
----+-----+-----
  1 | John     | Doe
(1 row)
```

7. Add the cluster into ACS

Log in to ACS. Select cluster and click on Add. Select other and upload or paste the kubeconfig file.

Click **Next** and select `ontap-nas` as the default storage class for ACS. Click **Next**, review the details and **Add** the cluster.

8. Define the application in ACS

Define the postgresql application in ACS. From the landing page, select **Applications, Define** and fill in the appropriate details. Click **Next** a couple of times, Review the details and click **Define**. The application gets

added to ACS.

Add cluster

STEP 2/3: STORAGE

STORAGE

Assign a new default storage class

The following storage classes are available on the cluster.

Set default	Storage class	Storage provisioner	Reclaim policy	Binding mode	Eligibility
<input type="radio"/>	gp2	kubernetes.io/aws-ebs	Delete	WaitForFirstConsumer	<div></div> Ineligible
<input type="radio"/>	gp2-csi	ebs.csi.aws.com	Delete	WaitForFirstConsumer	<div></div> Eligible
<input type="radio"/>	gp3	ebs.csi.aws.com	Delete	WaitForFirstConsumer	<div></div> Eligible
<input type="radio"/>	gp3-csi	ebs.csi.aws.com	Delete	WaitForFirstConsumer	<div></div> Eligible
<input checked="" type="radio"/>	ontap-nas <div>Default</div>	csi.trident.netapp.io	Delete	Immediate	<div></div> Eligible

← Back

Next →

9. Create a snapshot using ACS

There are many ways to create a snapshot in ACS. You can select the application and create a snapshot from the page that shows the details of the application. You can click on Create snapshot to create an on-demand snapshot or configure a protection policy.

Create an on-demand snapshot by simply clicking on **Create snapshot**, providing a name, reviewing the details, and clicking on **Snapshot**. The snapshot state changes to Healthy after the operation is completed.

Dashboard

Applications

Clusters

Cloud instances

Buckets

Account

Activity

Support

Data protection

Storage

Resources

Execution hooks

Activity

Tasks

Actions

Configure protection policy

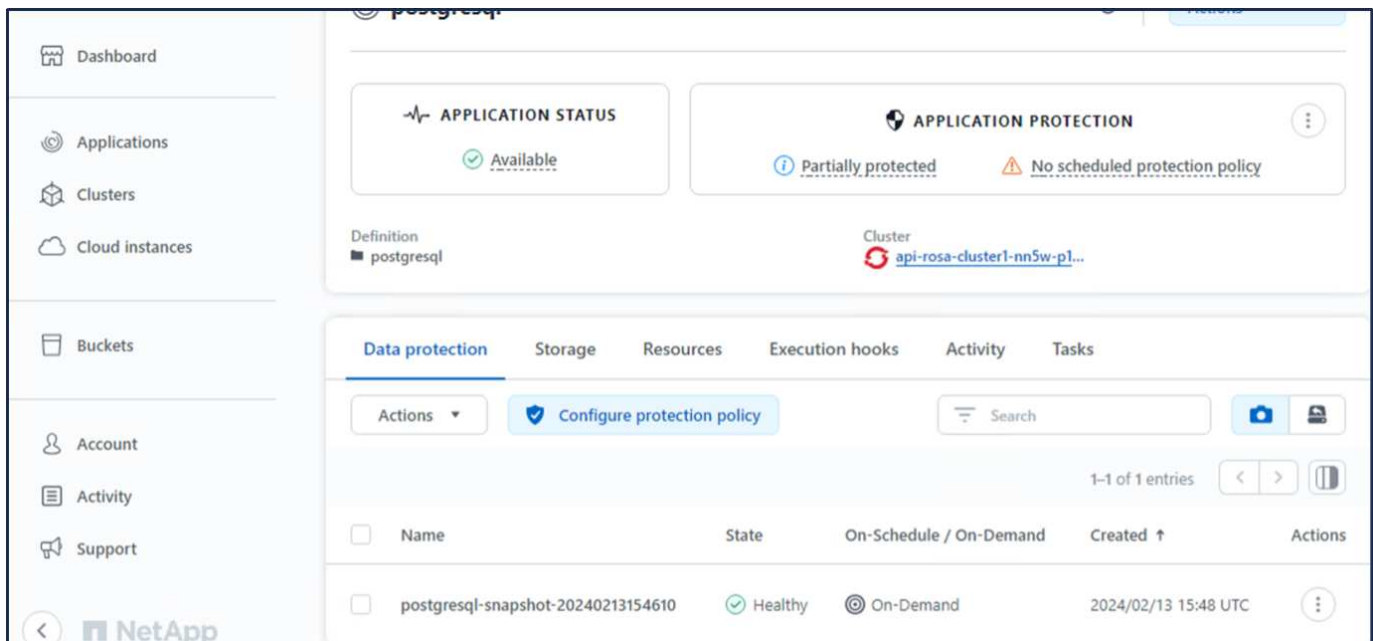
Search

0-0 of 0 entries

<

>

<input type="checkbox"/>	Name	State	On-Schedule / On-Demand	Created ↑	Actions
<div><div></div><div>You don't have any snapshots</div><div>After you have created a snapshot, it will be listed here</div><div>Create snapshot</div></div>					



10. Delete the database in the postgresql application

Log back into postgresql, list the available databases, delete the one you created previously and list again to ensure that the database has been deleted.

```
postgres=# \l
```

Name	Owner	Encoding	Locale Provider	Collate	Ctype	ICU Locale	ICU Rules	Access priv
erp	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			
postgres	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			
template0	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=c/postgres
template1	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			postgres=Ct...

(4 rows)

```
postgres=# DROP DATABASE erp;
DROP DATABASE
postgres=# \l
```

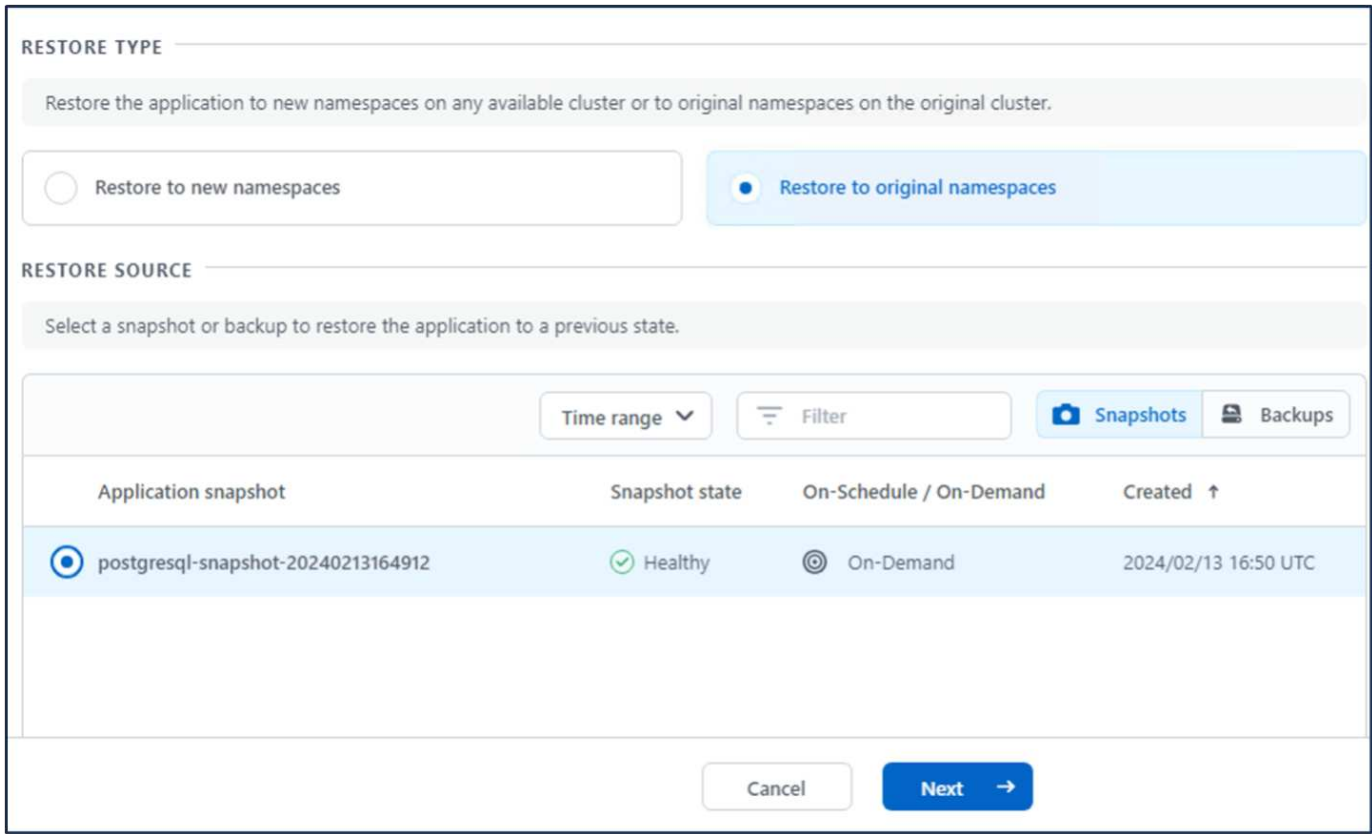
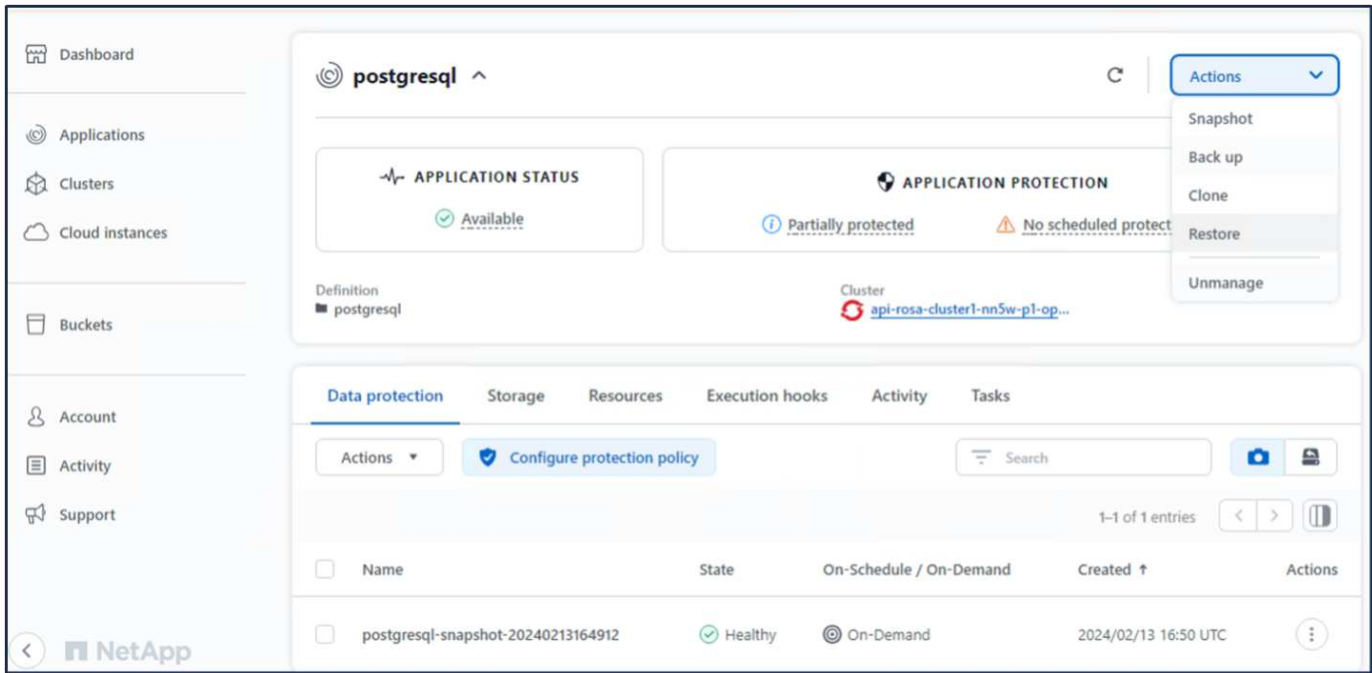
Name	Owner	Encoding	Locale Provider	Collate	Ctype	ICU Locale	ICU Rules	Access priv
postgres	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			
template0	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			=c/postgres
template1	postgres	UTF8	libc	en_US.UTF-8	en_US.UTF-8			postgres=Ct...

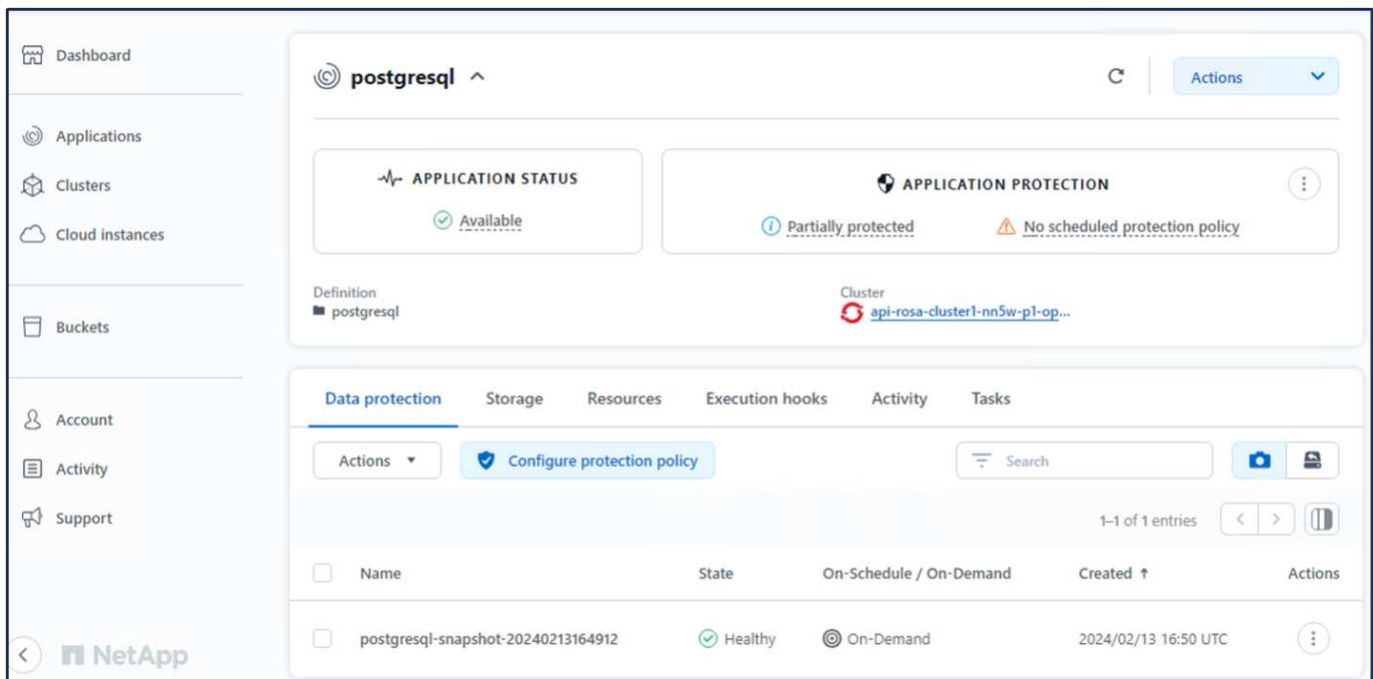
(3 rows)

11. Restore from a snapshot using ACS

To restore the application from a snapshot, go to ACS UI landing page, select the application and select Restore. You need to pick a snapshot or a backup from which to restore. (Typically, you would have multiple

created based on a policy that you have configured). Make appropriate choices in the next couple of screens and then click on **Restore**. The application status moves from Restoring to Available after it has been restored from the snapshot.





12. Verify your app has been restored from the snapshot

Login to the postgresql client and you should now see the table and the record in the table that you previously had. That's it. Just by clicking a button, your application has been restored to a previous state. That is how easy we make it for our customers with Astra Control.

```
[ec2-user@ip-10-49-11-132 ~]$ kubectl run postgresql-client --rm --tty -i --restart='Never' --namespace postgresql --image docker.io/bitnami/postgresql:16.2.0-debian-11-r1 --env="PGPASSWORD=$POSTGRES_PASSWORD" --command -- psql --host postgresql -U postgres -d postgres -p 5432
Warning: would violate PodSecurity "restricted:v1.24": allowPrivilegeEscalation != false (container "postgresql-client" must set securityContext.allowPrivilegeEscalation=false), unrestricted capabilities (container "postgresql-client" must set securityContext.capabilities.drop=["ALL"]), runAsNonRoot != true (pod or container "postgresql-client" must set securityContext.runAsNonRoot=true), seccompProfile (pod or container "postgresql-client" must set securityContext.seccompProfile.type to "RuntimeDefault" or "Localhost")
If you don't see a command prompt, try pressing enter.

postgres=# \l
          List of databases
  Name | Owner  | Encoding | Locale Provider | Collate | Ctype  | ICU Locale | ICU Rules | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----+-----
erp    | postgres | UTF8     | libc            | en_US.UTF-8 | en_US.UTF-8 |             |           |
postgres | postgres | UTF8     | libc            | en_US.UTF-8 | en_US.UTF-8 |             |           |
template0 | postgres | UTF8     | libc            | en_US.UTF-8 | en_US.UTF-8 |             |           |
template1 | postgres | UTF8     | libc            | en_US.UTF-8 | en_US.UTF-8 |             |           |
(4 rows)

postgres=# \c erp
You are now connected to database "erp" as user "postgres".
erp=# \dt
          List of relations
 Schema | Name  | Type  | Owner
-----+-----+-----+-----
 public | persons | table | postgres
(1 row)

erp=# SELECT * from PERSONS;
 id | firstname | lastname
----+-----+-----
  1 | John      | Doe
(1 row)
```

Data migration

This page shows the data migration options for container workloads on Managed Red Hat OpenShift clusters using FSx for NetApp ONTAP for persistent storage.

Data Migration

Red Hat OpenShift service on AWS as well as FSx for NetApp ONTAP (FSxN) are part of their service portfolio by AWS. FSxN is available on Single AZ or Multi-AZ options.

Multi-AZ option provides data protection from availability zone failure.

FSxN can be integrated with Astra Trident to provide persistent storage for applications on ROSA clusters.

Integration of FSxN with Trident using Helm chart

[ROSA Cluster Integration with Amazon FSx for ONTAP](#)

The migration of container applications involves:

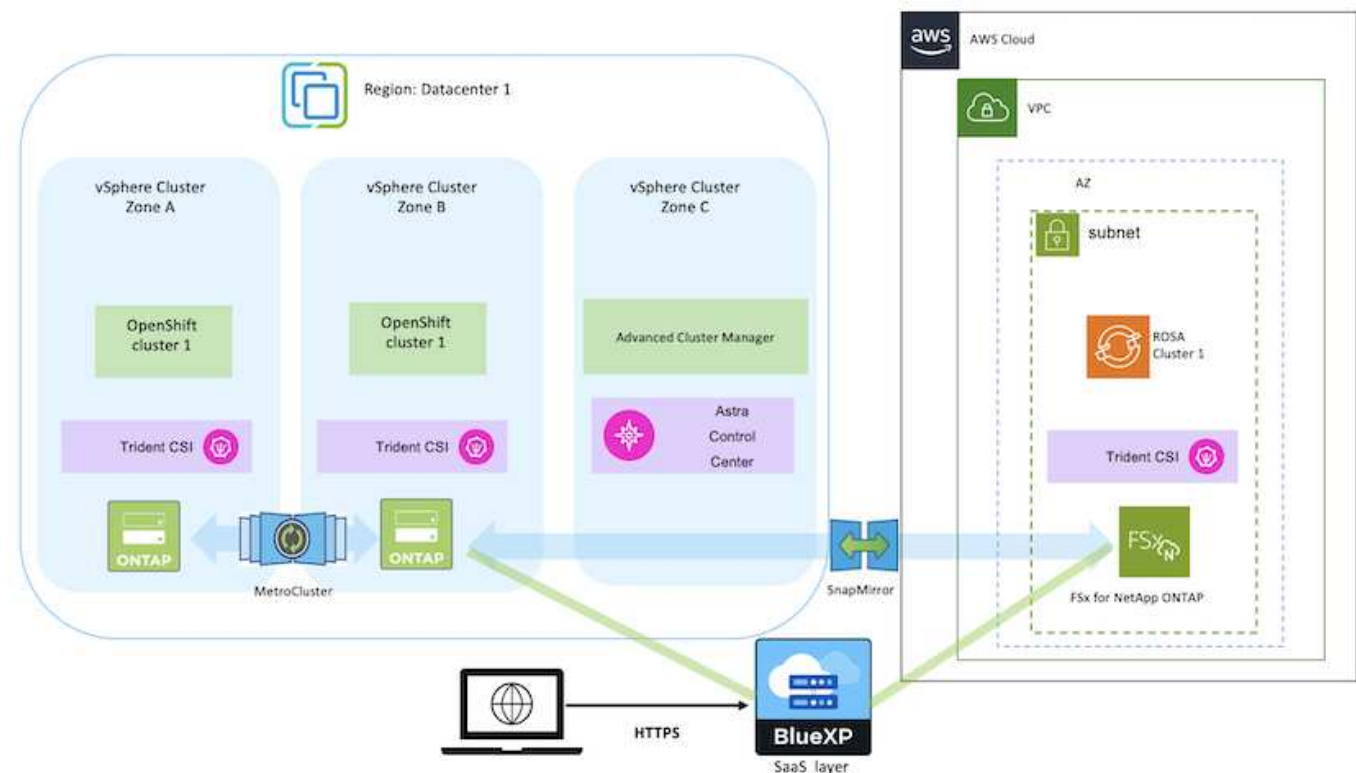
- Persistent volumes: this can be accomplished using BlueXP.
Another option is to use Astra Control Center to handle container application migrations from on-premises to the cloud environment. Automation can be used for the same purpose.
- Application metadata: this can be accomplished using OpenShift GitOps (Argo CD).

Failover and Fail-back of applications on ROSA cluster using FSxN for persistent storage

The following video is a demonstration of application failover and fail-back scenarios using BlueXP and Argo CD.

[Failover and Fail-back of applications on ROSA cluster](#)

Data protection and migration solution for OpenShift Container workloads



Copyright information

Copyright © 2024 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.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

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.