具有提供商管理组件的混合云
NetApp public and hybrid cloud solutions

NetApp
August 18, 2025

This PDF was generated from https://docs.netapp.com/zh-cn/netapp-solutions-cloud/openshift/os-pm-solution.html on August 18, 2025. Always check docs.netapp.com for the latest.

目录

具有提供商管理组件的混合云 1
- 具有托管 Red Hat OpenShift 容器平台工作负载的NetApp解决方案 1
- 在 AWS 上部署和配置托管 Red Hat OpenShift 容器平台 1
- 使用Google Cloud NetApp Volumes在 Google Cloud 上部署和配置 OpenShift Dedicated 4
- 数据保护 6
  - 备份/从备份恢复 6
  - 快照/从快照恢复 6
  - 博客 6
  - 创建快照并从中恢复的详细步骤 6
- 数据迁移 21
  - 数据迁移 22
- 适用于 Red Hat OpenShift 工作负载的其他NetApp混合多云解决方案 23
  - 其他解决方案 23

具有提供商管理组件的混合云

具有托管 Red Hat OpenShift 容器平台工作负载的NetApp解决方案

客户可能“诞生于云端”，或者正处于现代化历程的某个阶段，准备将部分选定的工作负载或所有工作负载从数据中心迁移到云端。他们可以选择在云中使用提供商管理的 OpenShift 容器和提供商管理的NetApp存储来运行他们的工作负载。他们应该在云中规划和部署托管的 Red Hat OpenShift 容器集群，以便为他们的容器工作负载提供成功的生产就绪环境。 NetApp为三大领先公共云中的托管 Red Hat 解决方案提供完全托管的存储产品。

Amazon FSx for NetApp ONTAP (FSx ONTAP)

FSx ONTAP为 AWS 中的容器部署提供数据保护、可靠性和灵活性。 Trident作为动态存储供应器，为客户的有状态应用程序使用持久性 FSx ONTAP存储。

由于 ROSA 可以在 HA 模式下部署，并且控制平面节点分布在多个可用区域，因此 FSx ONTAP还可以配置多可用区选项，以提供高可用性并防止可用区故障。

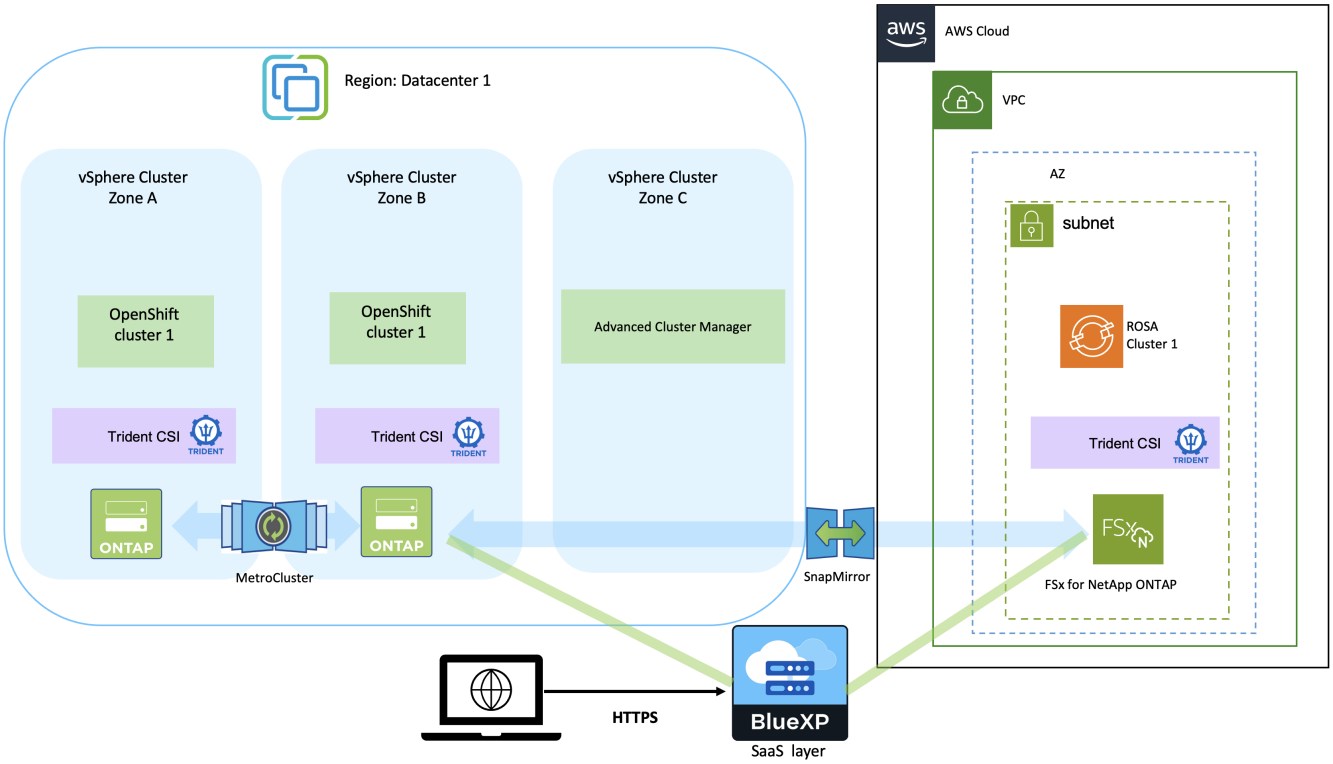
- Google Cloud NetApp Volumes*

Red Hat OpenShift Dedicated 是一个完全托管的应用程序平台，使您能够在混合云中快速构建、部署和扩展应用程序。 Google Cloud NetApp Volumes提供持久卷，将 ONTAP 的全套企业数据管理功能带入 Google Cloud 中的 OpenShift 部署。

在 AWS 上部署和配置托管 Red Hat OpenShift 容器平台

本节介绍在 AWS（ROSA）上设置托管 Red Hat OpenShift 集群的高级工作流程。它展示了Trident使用托管的Amazon FSx for NetApp ONTAP (FSx ONTAP) 作为存储后端来提供持久卷。提供了有关使用BlueXP在 AWS 上部署 FSx ONTAP的详细信息。此外，还提供了有关使用BlueXP和 OpenShift GitOps（Argo CD）为 ROSA 集群上的有状态应用程序执行数据保护和迁移活动的详细信息。

下图描述了部署在 AWS 上并使用 FSx ONTAP作为后端存储的 ROSA 集群。



该解决方案通过在AWS的两个VPC中使用两个ROSA集群进行了验证。每个 ROSA 集群都使用Trident与 FSx ONTAP集成。在 AWS 中部署 ROSA 集群和 FSx ONTAP有几种方法。此设置的高级描述提供了所使用的特定方法的文档链接。您可以参考"资源部分"。

设置过程可分为以下步骤：

安装**ROSA**集群

- 创建两个 VPC 并在 VPC 之间建立 VPC 对等连接。
- 参考"此处"有关安装 ROSA 集群的说明。

安装 **FSx ONTAP**

- 从BlueXP在 VPC 上安装 FSx ONTAP 。参考"此处"用于创建BlueXP帐户并开始使用。参考"此处"用于安装 FSx ONTAP。参考"此处"用于在 AWS 中创建连接器来管理 FSx ONTAP。
- 使用 AWS 部署 FSx ONTAP 。参考"此处"使用 AWS 控制台进行部署。

在 **ROSA** 集群上安装**Trident** （使用 **Helm** 图表）

- 使用 Helm chart 在 ROSA 集群上安装Trident 。请参阅文档链接： https://docs.netapp.com/us-en/trident/trident-get-started/kubernetes-deploy-helm.html [此处]。

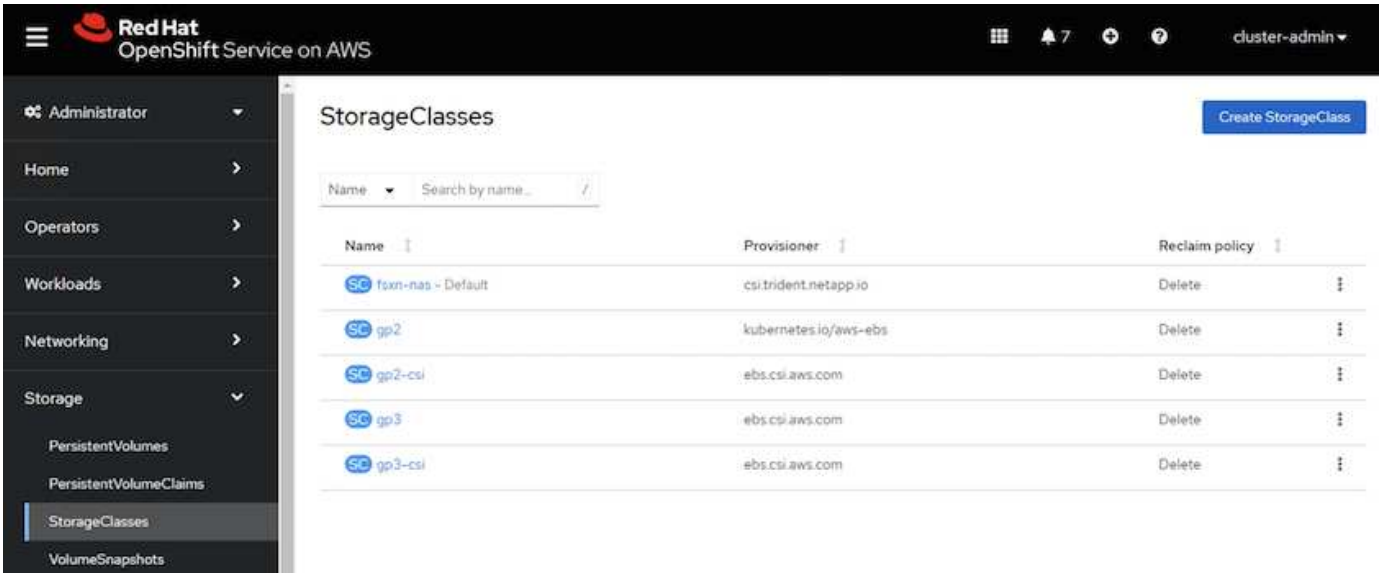
FSx ONTAP与Trident集成用于 ROSA 集群

OpenShift GitOps 可用于将Trident CSI 部署到所有托管集群，因为它们使用 ApplicationSet 注册到 ArgoCD。

```yaml
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
```

使用**Trident**创建后端和存储类（适用于 **FSx ONTAP**）

- 参考"此处"有关创建后端和存储类的详细信息。
- 从 OpenShift 控制台将使用Trident CSI 为 FsxN 创建的存储类设为默认。请参阅下面的截图：



使用 **OpenShift GitOps**（**Argo CD**）部署应用程序

- 在集群上安装 OpenShift GitOps 操作员。参考说明"此处"。
- 为集群设置一个新的 Argo CD 实例。参考说明"此处"。

打开Argo CD的控制台并部署一个应用程序。例如，您可以使用带有 Helm Chart 的 Argo CD 部署 Jenkins 应用程序。创建应用程序时，提供了以下详细信息：项目：默认集群：'https://kubernetes.default.svc' （不带引号）

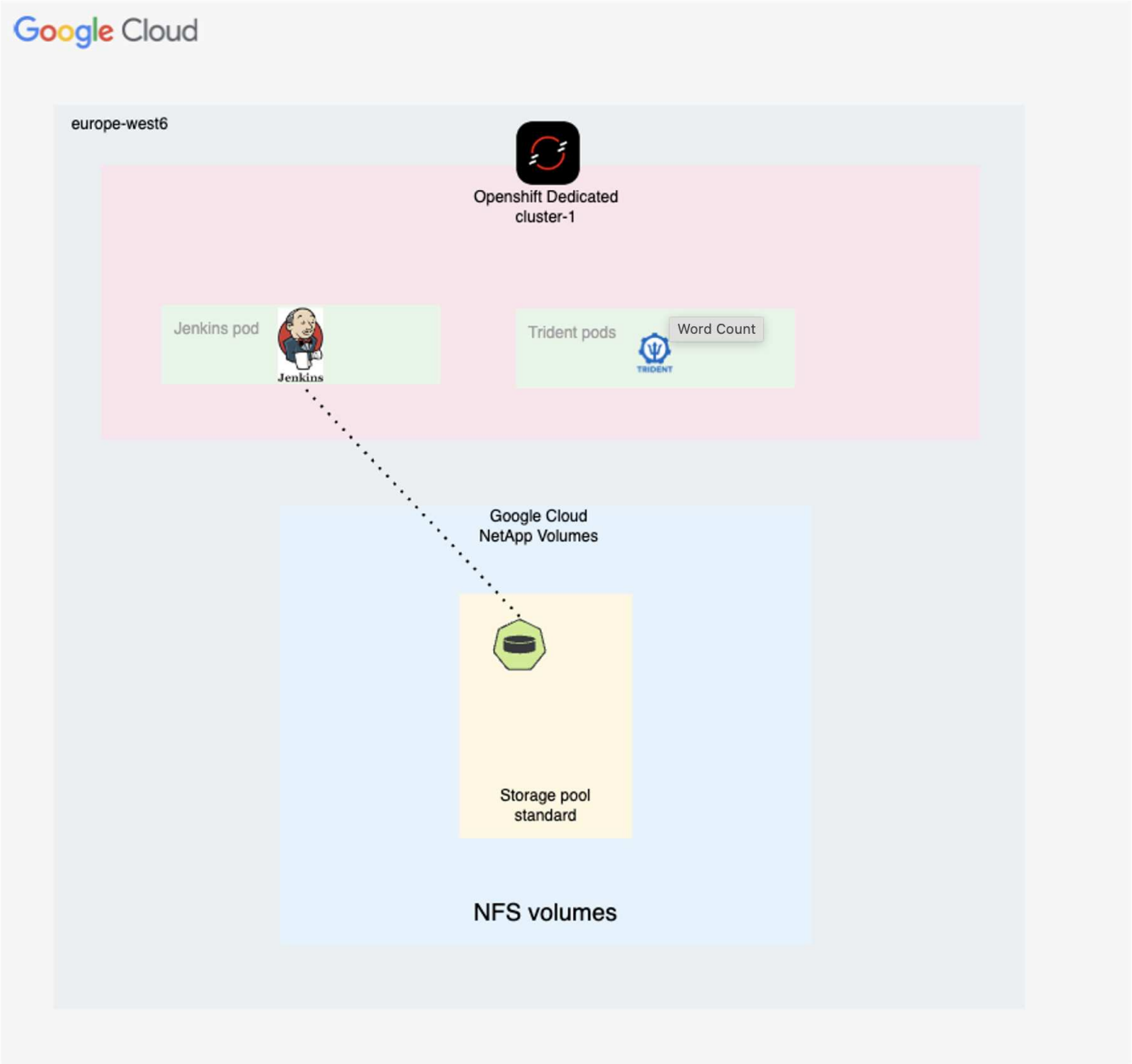
命名空间：Jenkins Helm Chart 的 URL：'https://charts.bitnami.com/bitnami' （不带引号）

Helm 参数：global.storageClass: fsxn-nas

使用Google Cloud NetApp Volumes在 Google Cloud 上部署和配置 OpenShift Dedicated

本节介绍在 Google Cloud 平台上设置 OpenShift Dedicated (OSD) 集群的高级工作流程。它显示NetApp Trident使用Google Cloud NetApp Volumes作为存储后端，为使用 Kubernetes 运行的有状态应用程序提供持久卷。

下图描述了部署在 Google Cloud 上并使用NetApp Volumes 作为后端存储的 OSD 集群。



设置过程可分为以下步骤：

在 **Google Cloud** 中安装 **OSD** 集群

- 如果您希望为集群使用现有的 VPC，则必须为 OSD 集群创建 VPC、两个子网、一个云路由器和两个 GCP 云 NAT。参考"此处"以获取说明。
- 参考"此处"有关使用客户云订阅 (CCS) 计费模型在 GCP 上安装 OSD 集群的说明。 OSD 也包含在 Google Cloud Marketplace 中。演示如何使用 Google Cloud Marketplace 解决方案安装 OSD 的视频位于"此处"。

启用**Google Cloud NetApp Volumes**

- 参考"此处"有关设置对Google Cloud NetApp Volumes 的访问权限的信息。遵循所有步骤，包括
- 创建存储池。参考"此处"有关如何在Google Cloud NetApp Volumes上设置存储池的信息。将在存储池内创建在 OSD 上运行的有状态 Kubernetes 应用程序的卷。

在 **OSD** 集群上安装**Trident** （使用 **Helm** 图表）

- 使用 Helm 图表在 OSD 集群上安装Trident 。参考"此处"有关如何安装 Helm Chart 的说明。舵图可能位于"此处"。

将**NetApp Volumes** 与**NetApp Trident**集成用于 **OSD** 集群

使用Trident创建后端和存储类（适用于Google Cloud NetApp Volumes）

- 有关创建后端的详细信息，请参阅此处。
- 如果 kubernetes 中的任何当前存储类被标记为默认，请通过编辑存储类来删除该注释。
- 使用Trident CSI 配置程序为NetApp卷创建至少一个存储类。使用注释将其中一个存储类设为默认存储类。当 PVC 清单中未明确调用时，这将允许 PVC 使用此存储类。下面显示了一个带有注释的示例。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-standard-k8s
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "nfs"
allowVolumeExpansion: true
```

使用 **OpenShift GitOps**（**Argo CD**）部署应用程序

- 在集群上安装 OpenShift GitOps 操作员。参考说明"此处"。
- 为集群设置一个新的 Argo CD 实例。参考说明"此处"。

打开Argo CD的控制台并部署一个应用程序。例如，您可以使用带有 Helm Chart 的 Argo CD 部署 Jenkins 应用程序。创建应用程序时，提供了以下详细信息：项目：默认集群：'https://kubernetes.default.svc' （不带引号）命名空间：Jenkins Helm Chart 的 URL：'https://charts.bitnami.com/bitnami' （不带引号）

数据保护

此页面显示使用Astra Control Service 的托管 Red Hat OpenShift on AWS (ROSA) 集群的数据保护选项。 Astra Control Service (ACS) 提供了一个易于使用的图形用户界面，您可以使用它来添加集群、定义在其上运行的应用程序以及执行应用程序感知数据管理活动。还可以使用允许工作流程自动化的 API 访问 ACS 功能。

为Astra Control（ACS 或 ACC）提供支持的是NetApp Trident。 Trident集成了多种类型的 Kubernetes 集群，例如 Red Hat OpenShift、EKS、AKS、SUSE Rancher、Anthos 等，以及各种类型的NetApp ONTAP存储，例如FAS/ AFF、 ONTAP Select、CVO、Google Google Cloud NetApp Volumes、 Azure NetApp Files和Amazon FSx ONTAP。

本节详细介绍了使用 ACS 的以下数据保护选项：

- 展示在一个区域运行的 ROSA 应用程序的备份和恢复以及恢复到另一个区域的视频。
- 展示 ROSA 应用程序的快照和恢复的视频。
- 安装 ROSA 集群、 Amazon FSx ONTAP、使用NetApp Trident与存储后端集成、在 ROSA 集群上安装 postgresql 应用程序、使用 ACS 创建应用程序的快照以及从中恢复应用程序的分步详细信息。
- 这篇博客逐步展示了使用 ACS 在带有 FSx ONTAP的 ROSA 集群上为 mysql 应用程序创建和恢复快照的过程。

备份/从备份恢复

以下视频展示了在一个区域运行的 ROSA 应用程序的备份以及恢复到另一个区域的过程。

AWS 上的 FSx NetApp ONTAP for Red Hat OpenShift 服务

快照/从快照恢复

以下视频展示了如何拍摄 ROSA 应用程序的快照以及如何从快照中恢复。

使用Amazon FSx ONTAP存储为 AWS (ROSA) 集群上的 Red Hat OpenShift Service 应用程序进行快照/还原

博客

- "使用Astra Control Service 和Amazon FSx存储对 ROSA 集群上的应用程序进行数据管理"

创建快照并从中恢复的详细步骤

先决条件设置

- "AWS 账户"
- "Red Hat OpenShift 帐户"
- IAM 用户"适当的权限"创建并访问ROSA集群
- "AWS CLI"
- "ROSA CLI"

- "OpenShift CLI"（oc）
- 具有子网和适当网关及路由的 VPC
- "ROSA集群安装"进入 VPC
- "Amazon FSx ONTAP"在同一个 VPC 中创建
- 从以下位置访问 ROSA 集群"OpenShift 混合云控制台"

后续步骤

1. 创建管理员用户并登录集群。
2. 为集群创建一个 kubeconfig 文件。
3. 在集群上安装Trident 。
4. 使用Trident CSI 配置器创建后端、存储类和快照类配置。
5. 在集群上部署postgresql应用程序。
6. 创建数据库并添加记录。
7. 将集群添加到 ACS。
8. 在 ACS 中定义应用程序。
9. 使用 ACS 创建快照。
10. 删除 postgresql 应用程序中的数据库。
11. 使用 ACS 从快照恢复。
12. 验证您的应用程序已从快照恢复。

1.创建管理员用户并登录集群

使用以下命令创建管理员用户来访问 ROSA 集群：（仅当您在安装时未创建管理员用户时才需要创建管理员用户）

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

该命令将提供如下所示的输出。使用 `oc login`输出中提供的命令。

```
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-00000-00000
```

您还可以使用令牌登录集群。如果您在创建集群时已经创建了管理员用户，则可以使用管理员用户凭据从 Red Hat OpenShift Hybrid Cloud 控制台登录集群。然后通过点击右上角显示登录用户的名称，您可以获取 `oc login`命令行的命令（令牌登录）。

2.为集群创建 **kubeconfig** 文件

遵循程序"此处"为 ROSA 集群创建 kubeconfig 文件。当您将集群添加到 ACS 时，稍后将使用此 kubeconfig 文件。

3.在集群上安装**Trident**

在 ROSA 集群上安装Trident （最新版本）。为此，您可以按照以下任一程序进行操作"此处"。要从集群控制台使用 helm 安装Trident ，首先创建一个名为Trident的项目。

然后从开发人员视图创建一个 Helm 图表存储库。对于 URL 字段使用 `'https://netapp.github.io/trident-helm-chart'`。然后为Trident操作员创建一个掌舵版本。

Project: trident

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

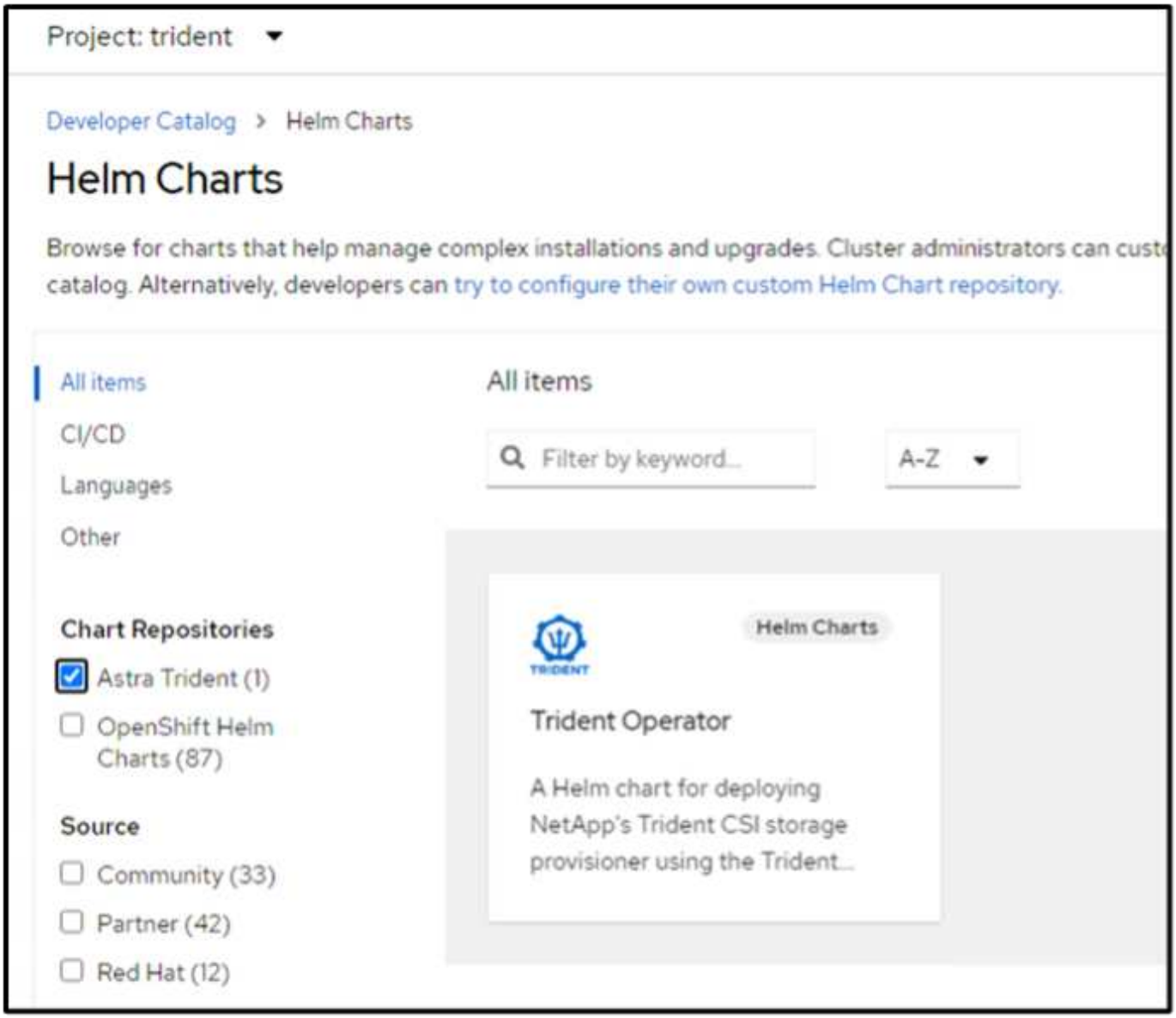
NetApp Astra Trident

A description for the Helm Chart repository.

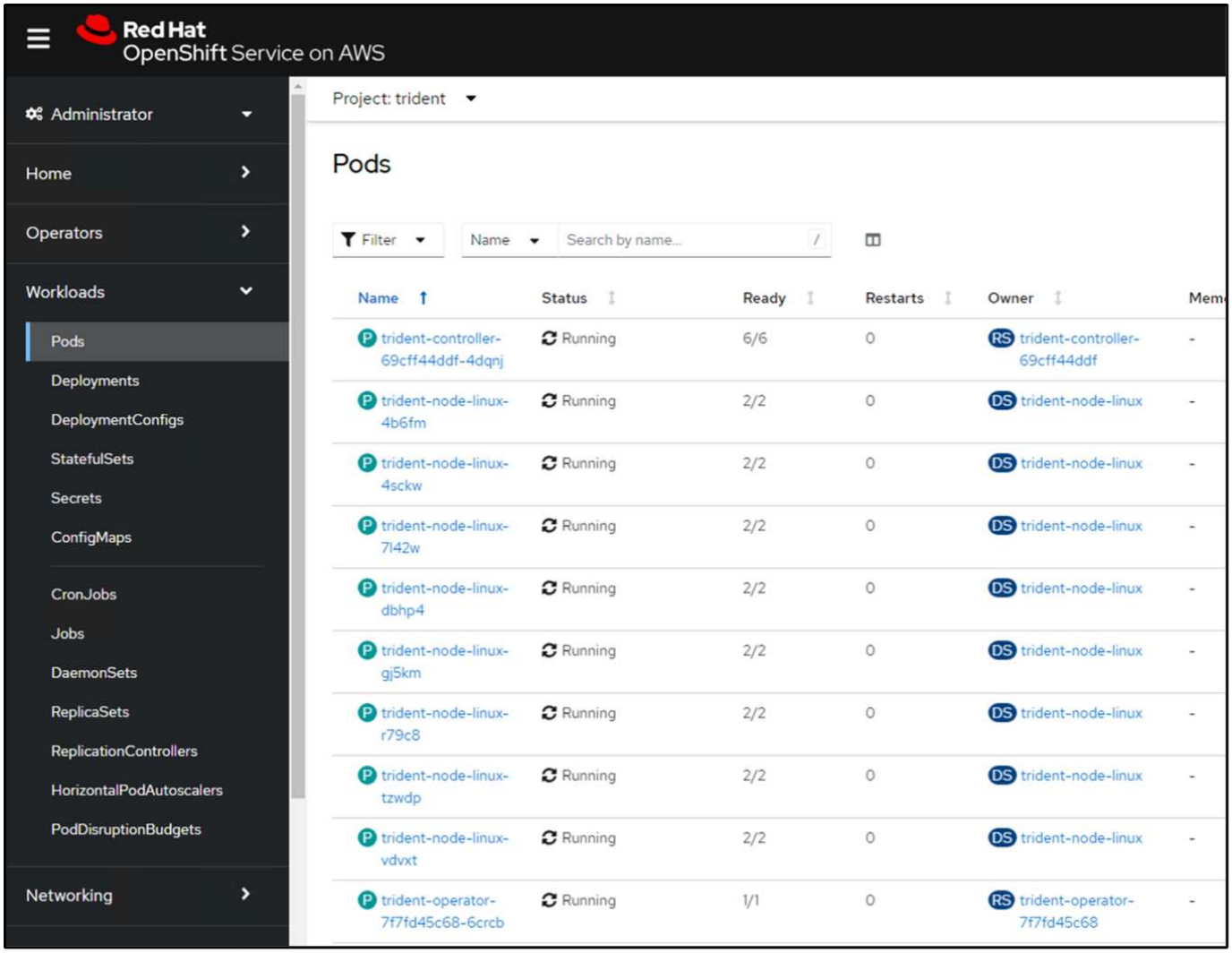
[ ] Disable usage of the repo in the developer catalog.

URL *

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



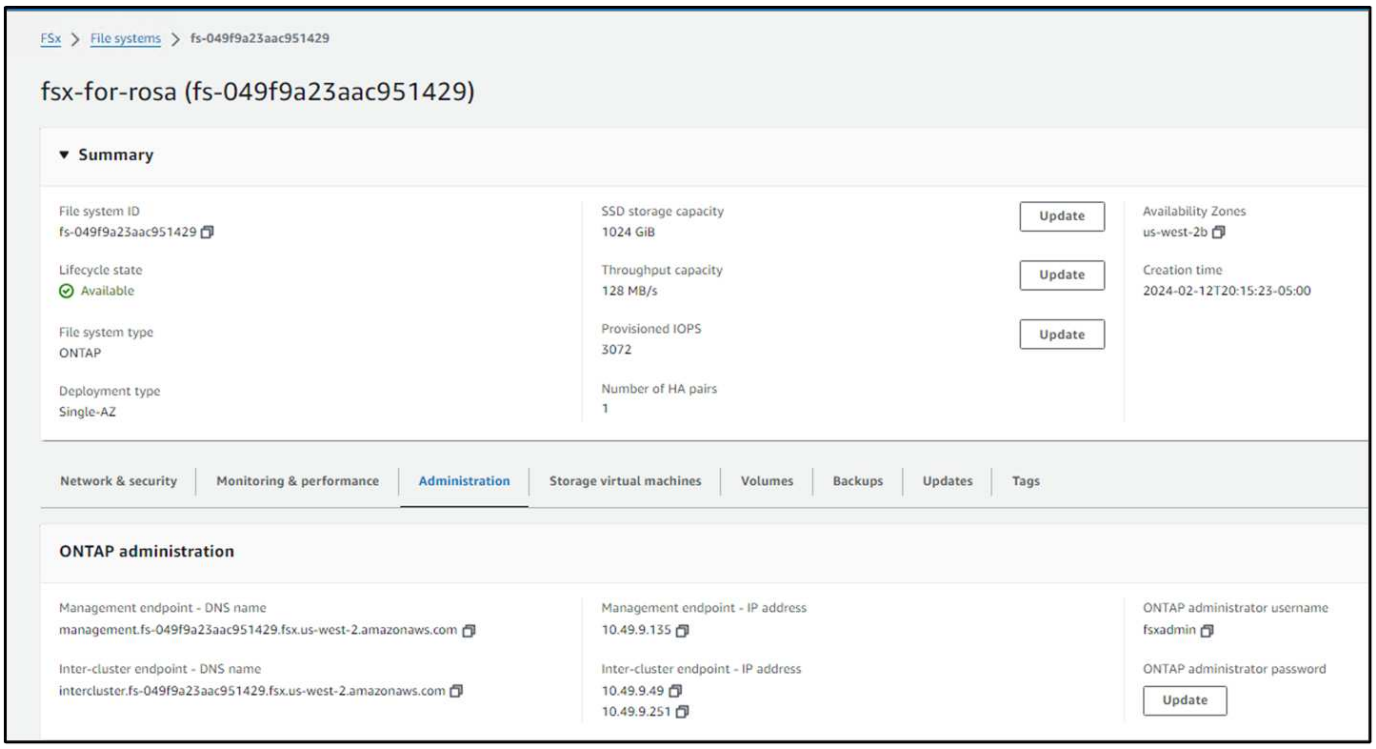
通过返回控制台上的管理员视图并选择 trident 项目中的 pod，验证所有 trident pod 是否正在运行。



4.使用**Trident CSI** 配置器创建后端、存储类和快照类配置

使用下面显示的 yaml 文件创建 trident 后端对象、存储类对象和 Volumesnapshot 对象。确保在后端的配置 yaml 中提供您创建的Amazon FSx ONTAP文件系统、管理 LIF 和文件系统的 vserver 名称的凭证。要获取这些详细信息，请转到Amazon FSx的 AWS 控制台并选择文件系统，导航到“管理”选项卡。另外，单击更新以设置 `fsxadmin`用户。

> 您可以使用命令行创建对象，也可以使用混合云控制台中的 yaml 文件创建对象。



- Trident后端配置**

```yaml
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
```

存储类别

```yaml
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
```

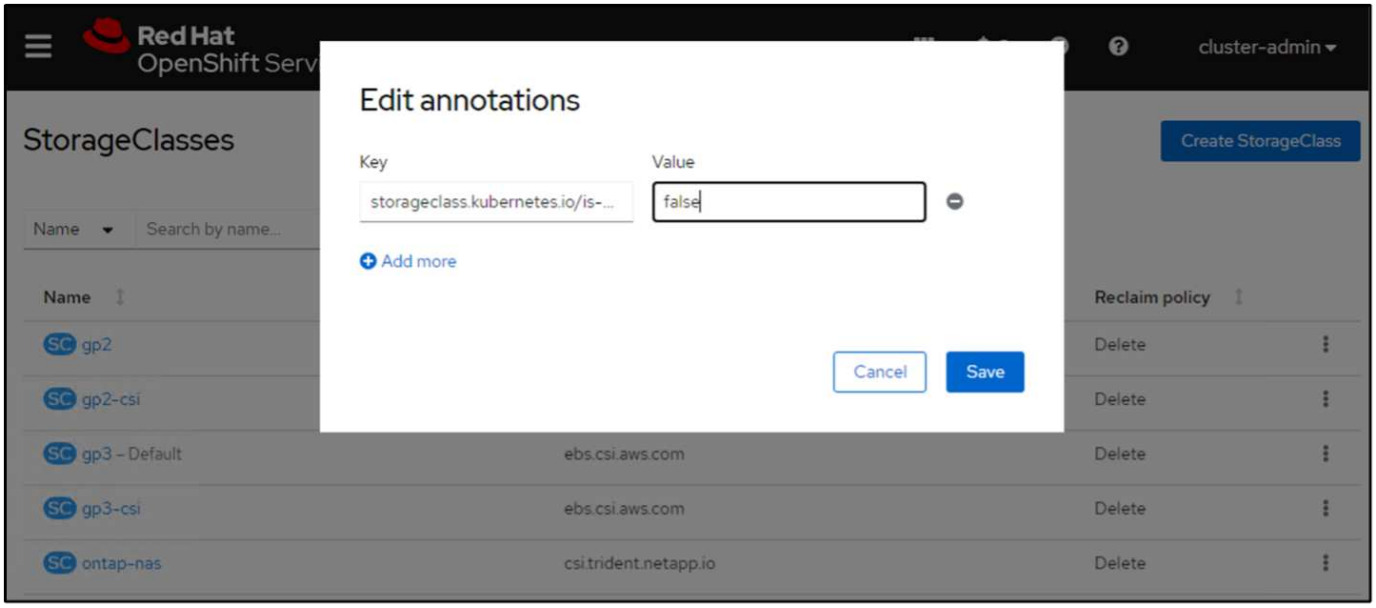
快照类

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

通过发出下面显示的命令来验证后端、存储类和 trident-snapshotclass 对象是否已创建。

```
[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]$
```

这时候你需要做的一个重要的修改就是将 ontap-nas 设置为默认存储类，而不是 gp3，这样你后面部署的 postgresql app 就可以使用默认存储类。在集群的 Openshift 控制台中，在“存储”下选择“StorageClasses”。将当前默认类的注释编辑为 false，并为 ontap-nas 存储类添加注释 storageclass.kubernetes.io/is-default-class 设置为 true。



5.在集群上部署 **postgresql** 应用程序

您可以从命令行部署应用程序，如下所示：

```
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="PG
PASSWORD=$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]$
```

如果您没有看到应用程序 pod 正在运行，那么可能是由于安全上下文约束导致的错误。

```
[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]$
```

通过编辑 `runAsUser`和 `fsGroup`中的字段 `statefuleset.apps/postgresql`具有输出中的 uid 的对象 `oc get project`命令如下所示。

```
[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 应用程序应该正在运行并使用由Amazon FSx ONTAP存储支持的持久卷。

```
[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]$
```

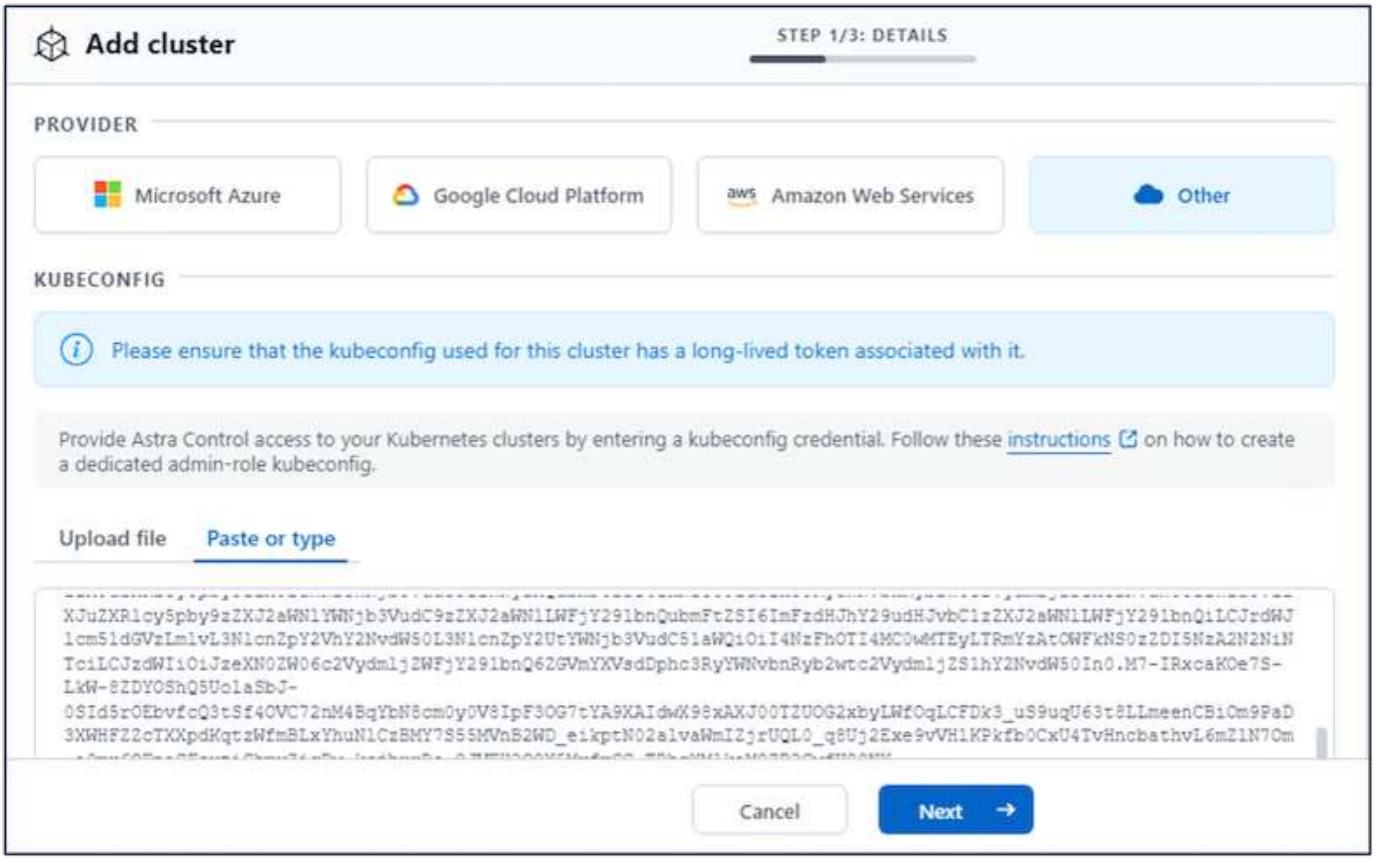
```
[ec2-user@ip-10-49-11-132 storage]$ kubectl get pvc -n postgresql
NAME                STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
data-postgresql-0   Bound    pvc-dd09524a-de75-4825-9424-03a9b91195ca   8Gi        RWO            ontap-nas      4m2s
[ec2-user@ip-10-49-11-132 storage]$
```

6.创建数据库并添加记录

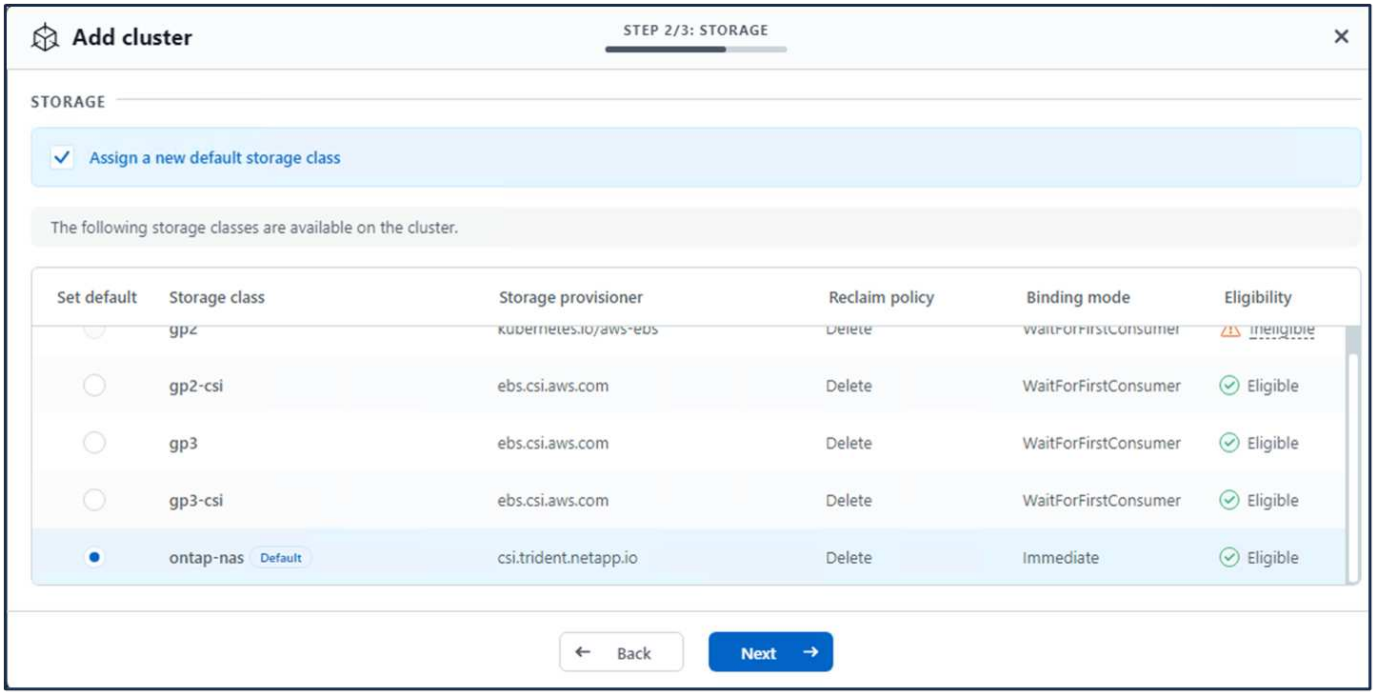
```
[ec2-user@ip-10-49-11-132 astra]$ export POSTGRES_PASSWORD=$(kubectl get secret --namespace postgresql postgresql -o jsonpath="
{.data.postgres-password}" | base64 -d)
[ec2-user@ip-10-49-11-132 astra]$ kubectl run postgresql-client --rm --tty -i --restart='Never' --namespace postgresql --image
docker.io/bitnami/postgresql:16.2.0-debian-11-r1 --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 se
t securityContext.allowPrivilegeEscalation=false), unrestricted capabilities (container "postgresql-client" must set securityCo
ntext.capabilities.drop=["ALL"]), runAsNonRoot != true (pod or container "postgresql-client" must set securityContext.runAsNonR
oot=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.将集群添加到 **ACS**

登录 ACS。选择集群并单击添加。选择其他并上传或粘贴 kubeconfig 文件。

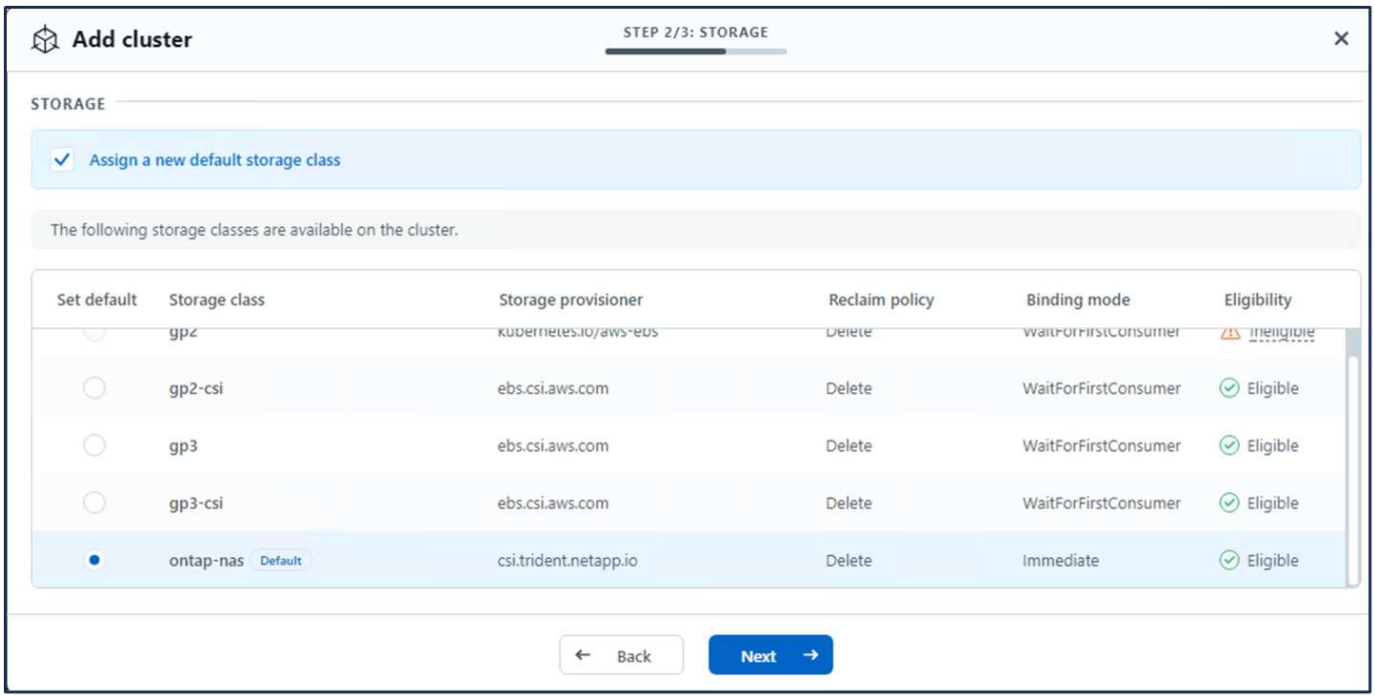


单击“下一步”并选择 ontap-nas 作为 ACS 的默认存储类。单击“下一步”，查看详细信息并“添加”集群。



8.在 **ACS** 中定义应用程序

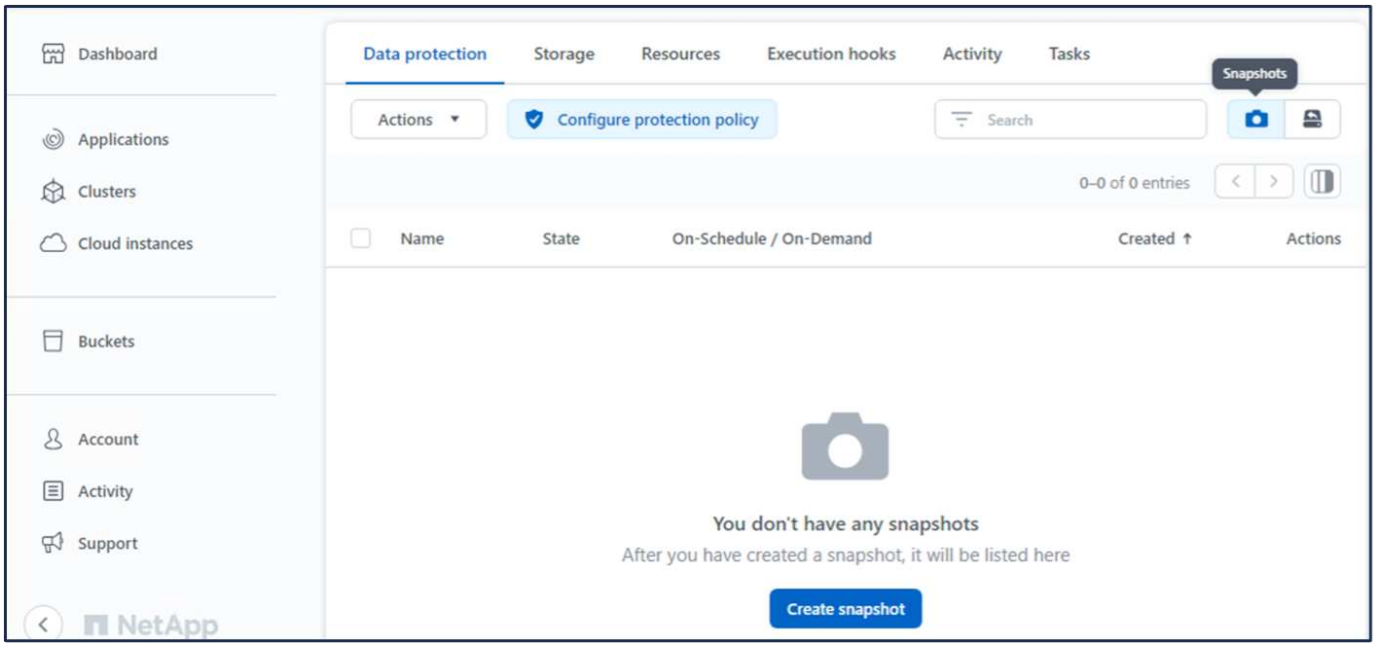
在 ACS 中定义 postgresql 应用程序。从登陆页面，选择*应用程序*、*定义*并填写适当的详细信息。单击“下一步”几次，查看详细信息，然后单击“定义”。应用程序被添加到 ACS。

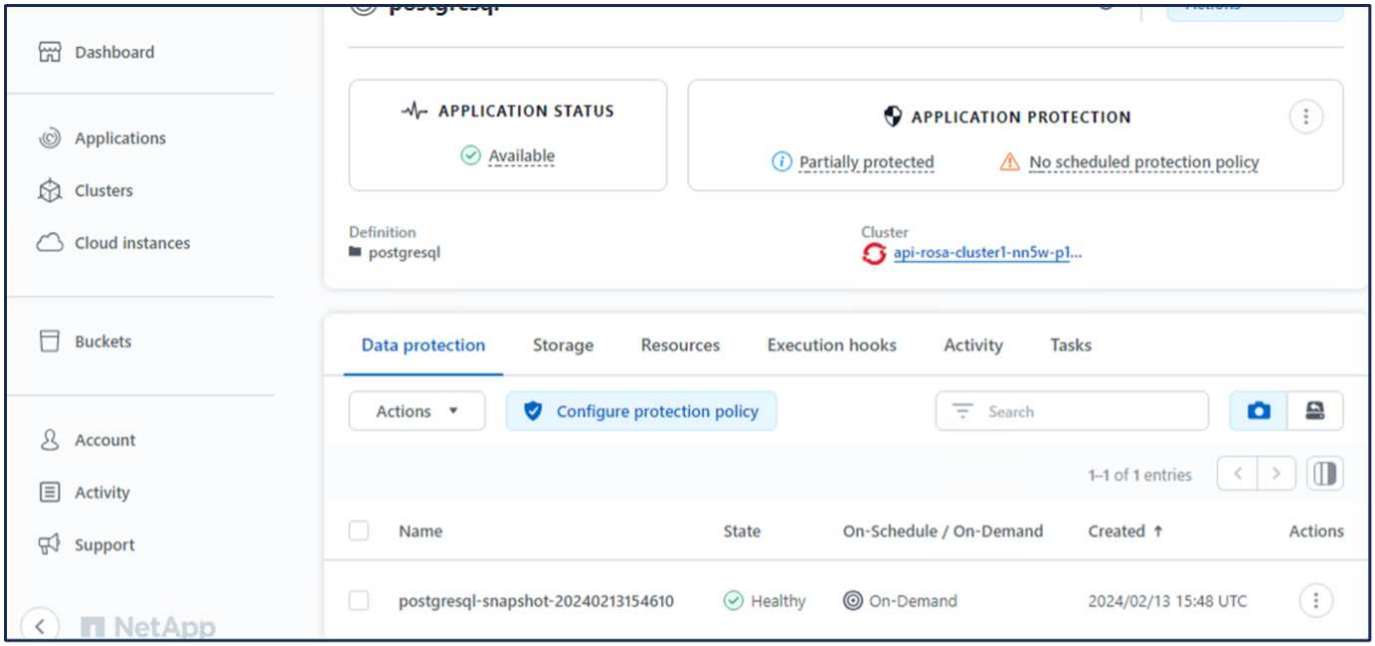


9.使用 **ACS** 创建快照

在 ACS 中创建快照的方法有很多种。您可以从显示应用程序详细信息的页面中选择应用程序并创建快照。您可以点击创建快照来创建按需快照或配置保护策略。

只需单击“创建快照”，提供名称，查看详细信息，然后单击“快照”即可创建按需快照。操作完成后，快照状态变为“健康”。





10.删除 **postgresql** 应用程序中的数据库

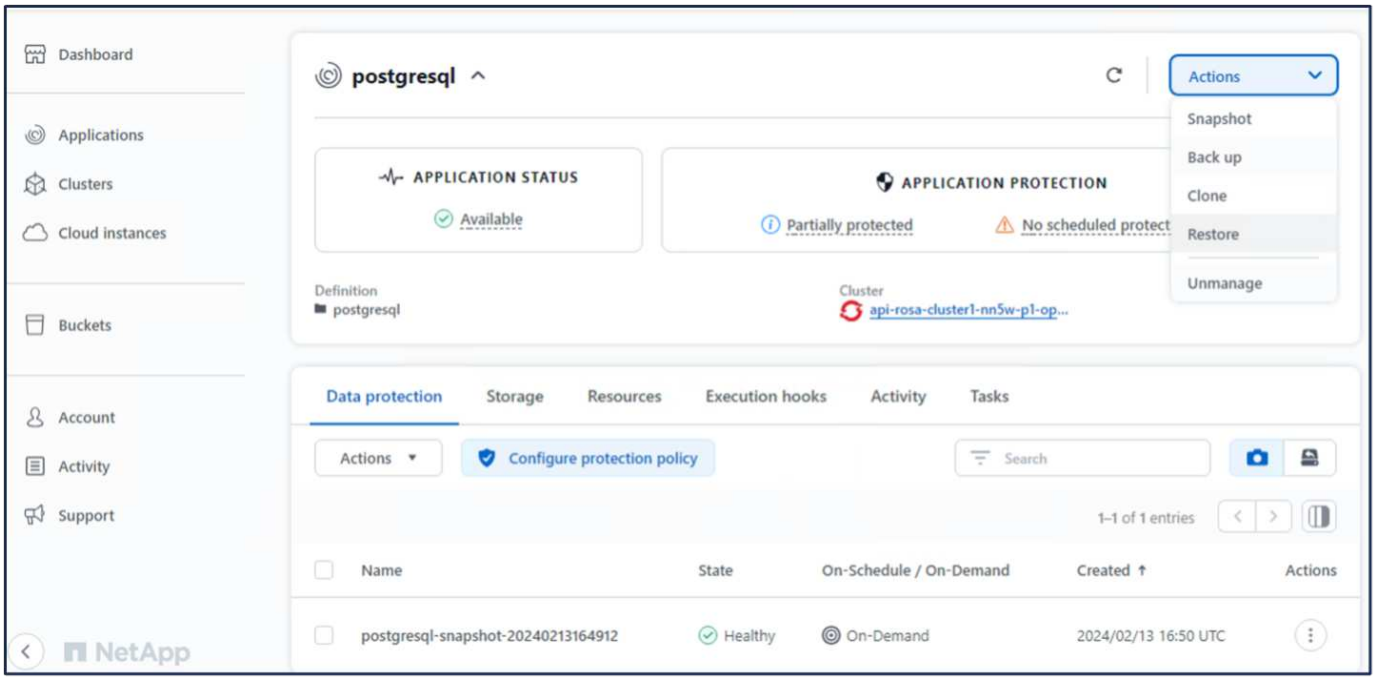
重新登录 postgresql，列出可用的数据库，删除之前创建的数据库，然后再次列出以确保该数据库已被删除。

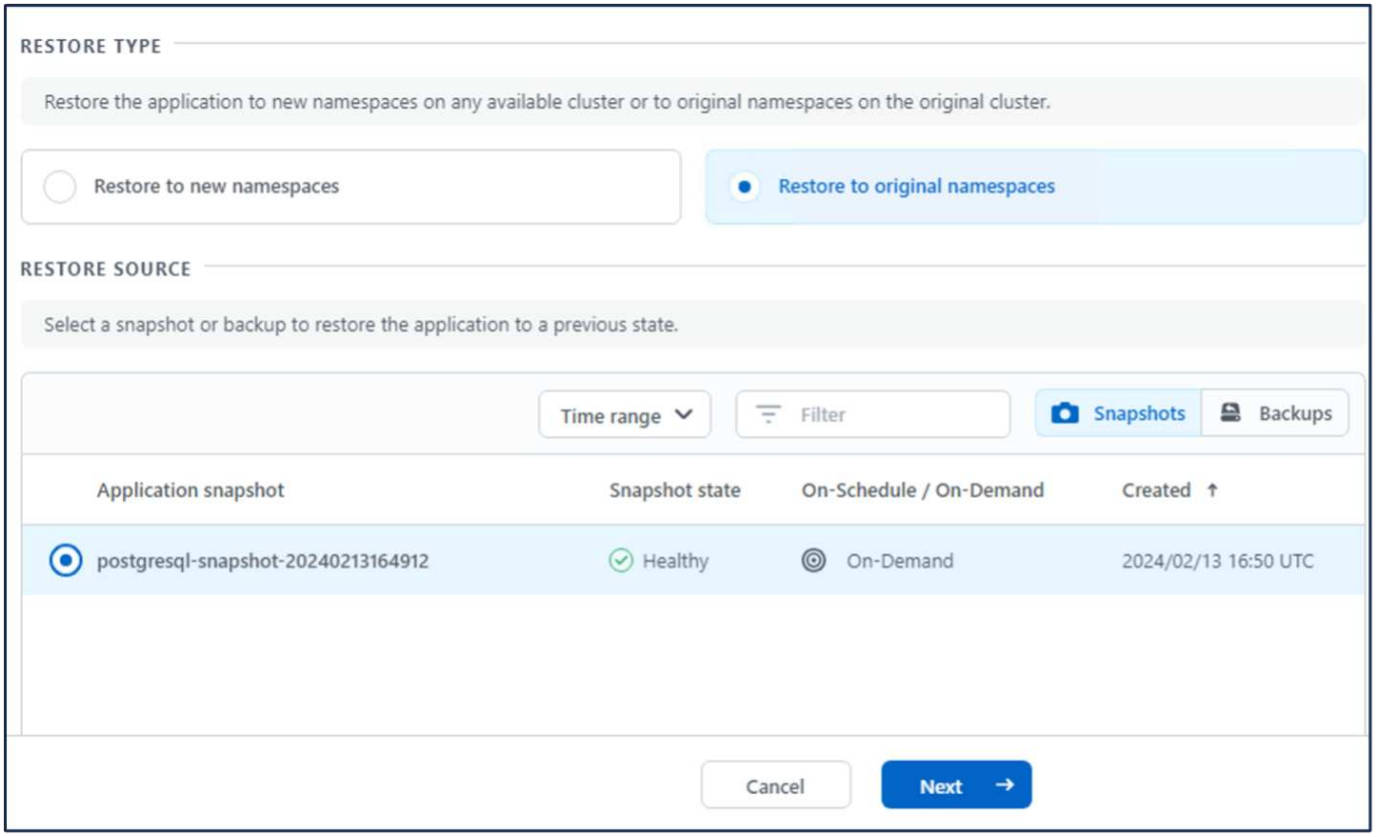
```
postgres=# \l
                                                   List of databases
   Name    |  Owner   | Encoding | Locale Provider |   Collate   |    Ctype    | ICU Locale | ICU Rules |   Access priv
-----------+----------+----------+-----------------+-------------+-------------+------------+-----------+---------------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

postgres=# DROP DATABASE erp;
DROP DATABASE
postgres=# \l
                                                   List of databases
   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
+          |          |          |                 |             |             |            |           | postgres=CTc/
 template1 | postgres | UTF8     | libc            | en_US.UTF-8 | en_US.UTF-8 |            |           | =c/postgres
+          |          |          |                 |             |             |            |           | postgres=CTc/
(3 rows)
```

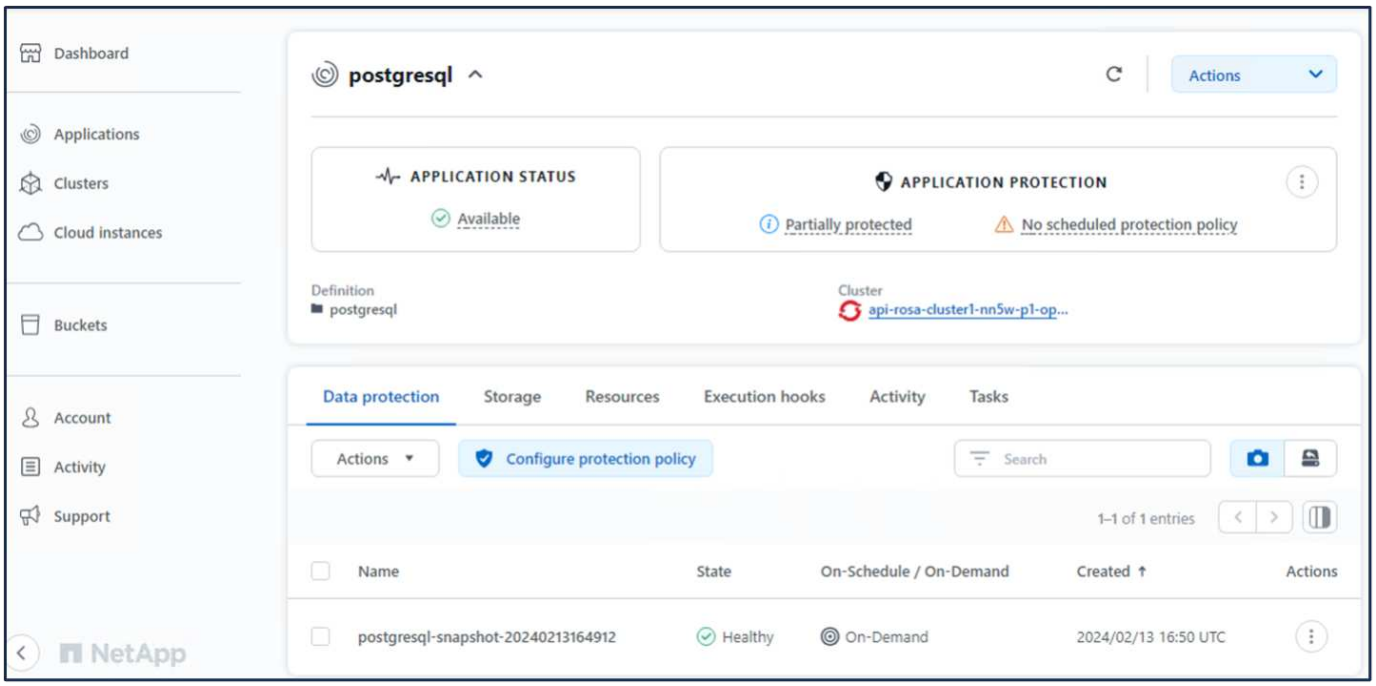
11.使用 **ACS** 从快照还原

要从快照恢复应用程序，请转到 ACS UI 登录页面，选择应用程序并选择恢复。您需要选择一个快照或备份来从中恢复。 （通常，您会根据已配置的策略创建多个）。在接下来的几个屏幕中做出适当的选择，然后单击“恢复

”。从快照恢复后，应用程序状态从“正在恢复”变为“可用”。







12.验证您的应用已从快照恢复

登录到 postgresql 客户端，您现在应该看到您之前拥有的表和表中的记录。就是这样。只需单击一个按钮，您的应用程序就会恢复到以前的状态。这就是我们利用Astra Control 为客户提供的便利。

```
[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.allowPrivi
legeEscalation=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 securityCon
text.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 |            |           | =c/postgres          +
           |          |          |                 |             |             |            |           | postgres=CTc/postgres
 template1 | postgres | UTF8     | libc            | en_US.UTF-8 | en_US.UTF-8 |            |           | =c/postgres          +
           |          |          |                 |             |             |            |           | postgres=CTc/postgres
(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)
```

数据迁移

此页面显示使用 FSx ONTAP进行持久存储的托管 Red Hat OpenShift 集群上的容器工作负载的数据迁移选项。

数据迁移

AWS 上的 Red Hat OpenShift 服务以及Amazon FSx for NetApp ONTAP （FSx ONTAP）都是 AWS 服务组合的一部分。 FSx ONTAP可在单可用区或多可用区选项上使用。 Multi-Az 选项提供可用区域故障时的数据保护。 FSx ONTAP可以与Trident集成，为 ROSA 集群上的应用程序提供持久存储。

使用 **Helm chart** 将 **FSx ONTAP**与**Trident**集成

ROSA 集群与Amazon FSx ONTAP集成

容器应用的迁移主要包括：

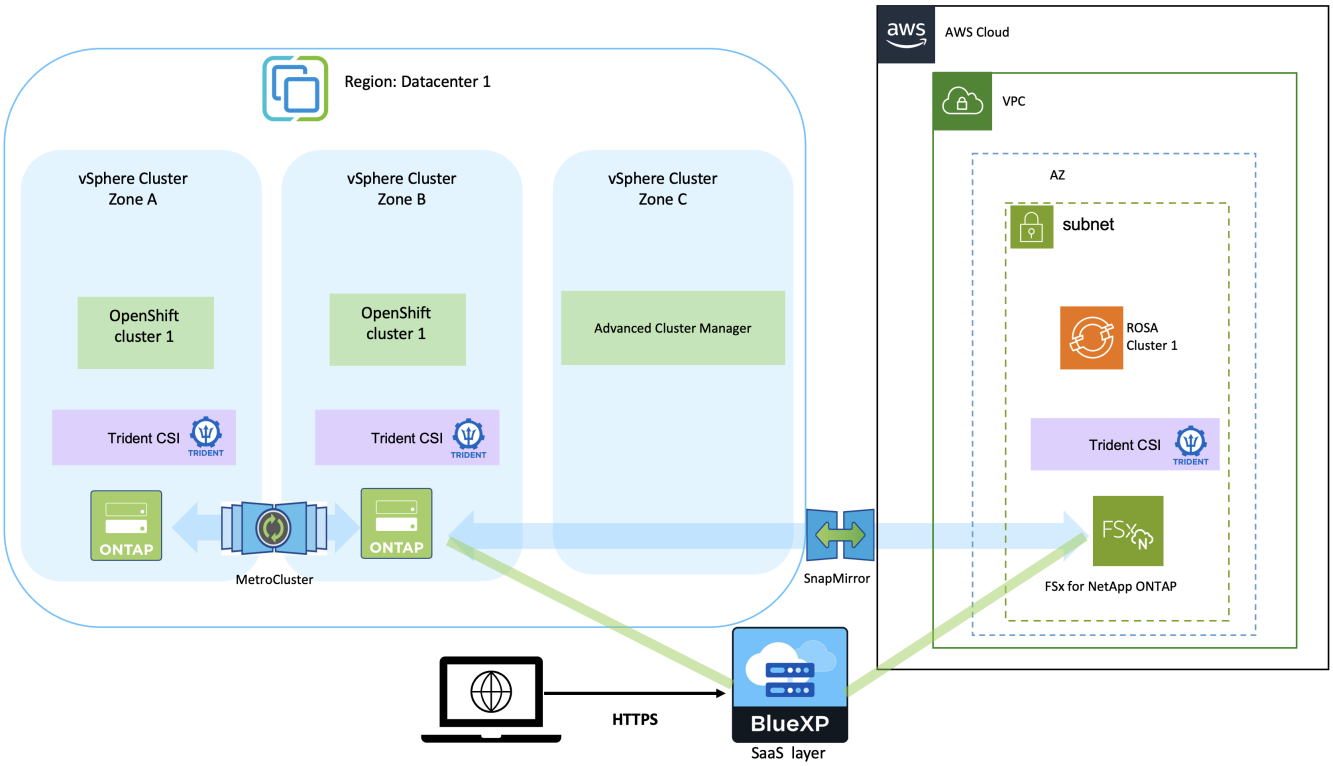
- 持久卷：这可以使用BlueXP来实现。另一种选择是使用Trident Protect 来处理从本地到云环境的容器应用程序迁移。自动化可以用于相同的目的。
- 应用程序元数据：这可以使用 OpenShift GitOps（Argo CD）来完成。

使用 **FSx ONTAP**进行 **ROSA** 集群上应用程序的故障转移和故障恢复，以实现持久存储

以下视频演示了使用BlueXP和 Argo CD 的应用程序故障转移和故障恢复场景。

ROSA 集群上应用程序的故障转移和故障恢复

OpenShift Container 工作负载的数据保护和迁移解决方案



适用于 Red Hat OpenShift 工作负载的其他NetApp混合多云解决方案

其他解决方案

其他部分提供了其他解决方案，如下所示：

对于 Red Hat OpenShift Container 解决方案，请参阅"此处"。

对于 Red Hat OpenShift 虚拟化解决方案，请参阅"此处"。

版权信息

版权所有 © 2025 NetApp, Inc.。保留所有权利。中国印刷。未经版权所有者事先书面许可，本文档中受版权保护的任何部分不得以任何形式或通过任何手段（图片、电子或机械方式，包括影印、录音、录像或存储在电子检索系统中）进行复制。

从受版权保护的 NetApp 资料派生的软件受以下许可和免责声明的约束：

本软件由 NetApp 按“原样”提供，不含任何明示或暗示担保，包括但不限于适销性以及针对特定用途的适用性的隐含担保，特此声明不承担任何责任。在任何情况下，对于因使用本软件而以任何方式造成的任何直接性、间接性、偶然性、特殊性、惩罚性或后果性损失（包括但不限于购买替代商品或服务；使用、数据或利润方面的损失；或者业务中断），无论原因如何以及基于何种责任理论，无论出于合同、严格责任或侵权行为（包括疏忽或其他行为），NetApp 均不承担责任，即使已被告知存在上述损失的可能性。

NetApp 保留在不另行通知的情况下随时对本文档所述的任何产品进行更改的权利。除非 NetApp 以书面形式明确同意，否则 NetApp 不承担因使用本文档所述产品而产生的任何责任或义务。使用或购买本产品不表示获得 NetApp 的任何专利权、商标权或任何其他知识产权许可。

本手册中描述的产品可能受一项或多项美国专利、外国专利或正在申请的专利的保护。

有限权利说明：政府使用、复制或公开本文档受 DFARS 252.227-7013（2014 年 2 月）和 FAR 52.227-19（2007 年 12 月）中“技术数据权利 — 非商用”条款第 (b)(3) 条规定的限制条件的约束。

本文档中所含数据与商业产品和/或商业服务（定义见 FAR 2.101）相关，属于 NetApp, Inc. 的专有信息。根据本协议提供的所有 NetApp 技术数据和计算机软件具有商业性质，并完全由私人出资开发。 美国政府对这些数据的使用权具有非排他性、全球性、受限且不可撤销的许可，该许可既不可转让，也不可再许可，但仅限在与交付数据所依据的美国政府合同有关且受合同支持的情况下使用。除本文档规定的情形外，未经 NetApp, Inc. 事先书面批准，不得使用、披露、复制、修改、操作或显示这些数据。美国政府对国防部的授权仅限于 DFARS 的第 252.227-7015(b)（2014 年 2 月）条款中明确的权利。

商标信息

NetApp、NetApp 标识和 http://www.netapp.com/TM 上所列的商标是 NetApp, Inc. 的商标。其他公司和产品名称可能是其各自所有者的商标。