# Cloud ibrido con componenti gestiti dal provider

NetApp Solutions

NetApp
April 26, 2024

# Sommario

# Soluzioni NetApp ibride multicloud per i carichi di lavoro dei container Red Hat OpenShift

## Panoramica

NetApp sta assistendo a un significativo aumento dei clienti nella modernizzazione delle applicazioni aziendali legacy e nella creazione di nuove applicazioni utilizzando container e piattaforme di orchestrazione basate su Kubernetes. Red Hat OpenShift Container Platform è un esempio che vediamo adottato da molti dei nostri clienti.

Man mano che un numero sempre maggiore di clienti inizia ad adottare container all'interno delle proprie aziende, NetApp si trova nella posizione ideale per soddisfare le esigenze di storage persistenti delle proprie applicazioni stateful e le esigenze di gestione dei dati classiche, come protezione dei dati, sicurezza dei dati e migrazione dei dati. Tuttavia, queste esigenze vengono soddisfatte utilizzando strategie, strumenti e metodi diversi.

**Le opzioni di storage basate su NetApp ONTAP** elencate di seguito offrono sicurezza, protezione dei dati, affidabilità e flessibilità per le implementazioni di container e Kubernetes.

- Storage autogestita on-premise:
  - NetApp Fabric Attached Storage (FAS), NetApp All Flash FAS Array (AFF), NetApp All SAN Array (ASA) e ONTAP Select
- Storage gestito dal provider on-premise:
  - NetApp Keystone offre storage as a service (STaaS)
- Storage autogestita nel cloud:
  - NetApp Cloud Volumes ONTAP (CVO) offre storage autogestiti negli hyperscaler
- Storage gestito dal provider nel cloud:
  - Cloud Volumes Service per Google Cloud (CVS), Azure NetApp Files (ANF) e Amazon FSX per NetApp ONTAP offrono storage completamente gestito negli hyperscaler

## 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** consente di gestire tutte le risorse di storage e dati da un singolo piano di controllo/interfaccia.

È possibile utilizzare BlueXP per creare e amministrare lo storage cloud (ad esempio, Cloud Volumes ONTAP e Azure NetApp Files), per spostare, proteggere e analizzare i dati e per controllare molti dispositivi storage on-premise e edge.

**NetApp Astra Trident** è un orchestratore di storage conforme a CSI che consente un consumo rapido e semplice dello storage persistente supportato da una serie di opzioni di storage NetApp sopra menzionate. Si tratta di un software open-source gestito e supportato da NetApp.

## Astra Trident CSI feature highlights

### CSI specific

- CSI NetApp® Snapshot™ copies and volume creation from CSI Snapshot copies
- CSI topology
- Volume expansion

### Security

- Dynamic-export policy management
- iSCSI initiator-groups dynamic management
- iSCSI bidirectional CHAP

### Control

- Storage and performance consumption
- Monitoring
- Volume Import
- Cross Namespace Volume Access

### Installation methods

- Binary
- Helm chart
- Operator
- GitOps

### Choose your access mode

- RWO (ReadWriteOnce, i.e 1⇔1)
- RWX (*ReadWriteMany*, i.e 1⇔n)
- ROX (*ReadOnlyMany*)
- RWOP (ReadWriteOnce POD)

### Choose your protocol

- NFS
- SMB
- iSCSI

I carichi di lavoro dei container business-critical richiedono molto di più dei semplici volumi persistenti. I loro requisiti di gestione dei dati richiedono anche la protezione e la migrazione degli oggetti di kubernetes dell'applicazione.

> I dati dell'applicazione includono oggetti kubernetes oltre ai dati dell'utente: Alcuni esempi sono i seguenti: - Kubernetes oggetti come specifiche di pod, PVC, implementazioni, servizi - oggetti di configurazione personalizzati come mappe di configurazione e segreti - dati persistenti come copie Snapshot, backup, cloni - risorse personalizzate come CRS e CRD

**NetApp Astra Control**, disponibile sia come software completamente gestito che autogestita, offre un'orchestrazione per una solida gestione dei dati applicativi. Fare riferimento a. "Documentazione Astra" Per ulteriori informazioni sulla famiglia di prodotti Astra.

Questa documentazione di riferimento fornisce la convalida della migrazione e della protezione delle applicazioni basate su container, implementate sulla piattaforma container RedHat OpenShift, utilizzando NetApp Astra Control Center. Inoltre, la soluzione fornisce dettagli di alto livello per l'implementazione e l'utilizzo di Red Hat Advanced Cluster Management (ACM) per la gestione delle piattaforme container. Il documento evidenzia inoltre i dettagli per l'integrazione dello storage NetApp con le piattaforme container Red Hat OpenShift che utilizzano Astra Trident CSI Provisioner. Astra Control Center viene implementato nel cluster dell'hub e viene utilizzato per gestire le applicazioni container e il loro ciclo di vita dello storage persistente. Infine, offre una soluzione per la replica, il failover e il fail-back per i carichi di lavoro dei container su cluster Red Hat OpenShift gestiti in AWS (ROSA) utilizzando Amazon FSX per NetApp ONTAP (FSxN) come storage persistente.

# Soluzione NetApp con workload gestiti della piattaforma container Red Hat OpenShift su AWS

## Soluzione NetApp con workload gestiti della piattaforma container Red Hat OpenShift su AWS

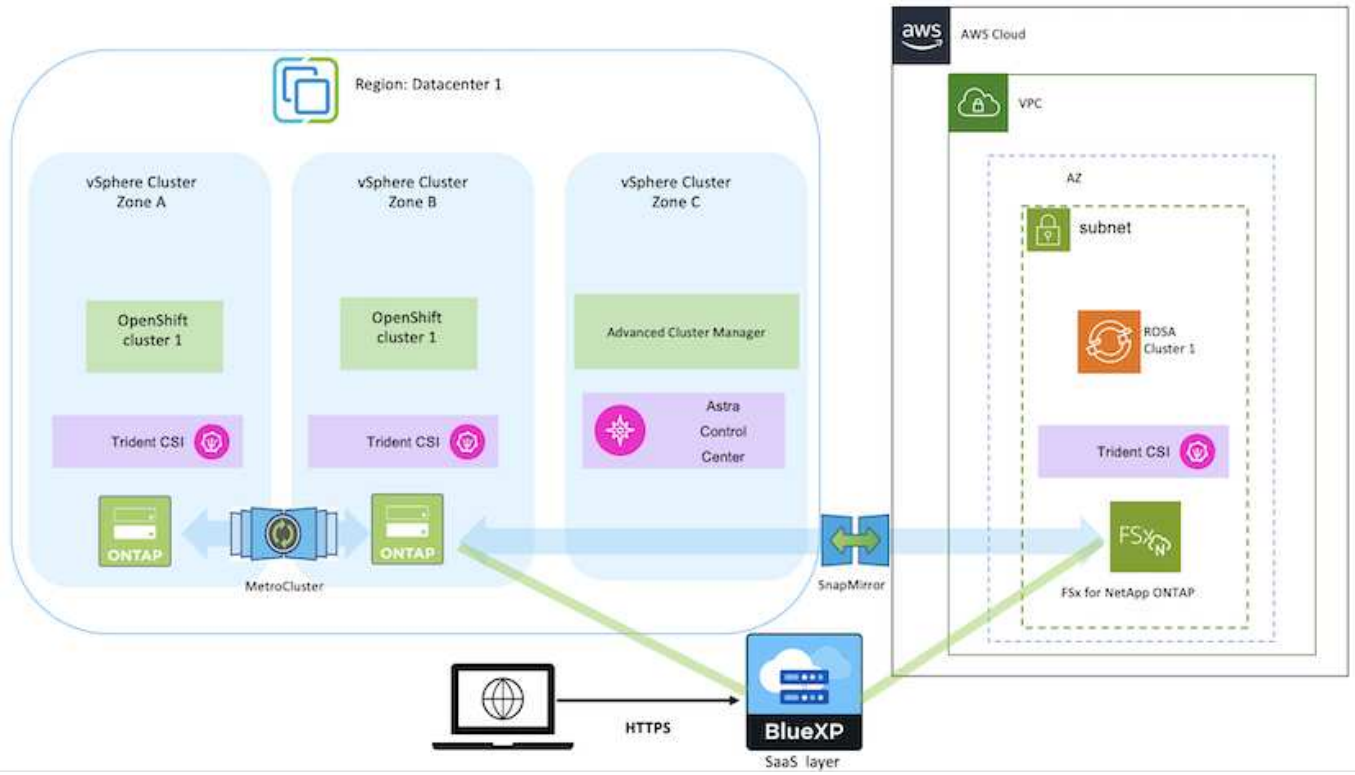
I clienti possono "nascere nel cloud" o trovarsi in un punto del loro percorso di modernizzazione quando sono pronti a spostare alcuni carichi di lavoro selezionati o tutti i carichi di lavoro dai data center al cloud. Possono scegliere di utilizzare container OpenShift gestiti da provider e storage NetApp gestito da provider nel cloud per l'esecuzione dei carichi di lavoro. Devono pianificare e implementare i cluster di container gestiti Red Hat OpenShift (ROSA) nel cloud per un ambiente pronto per la produzione di successo per i carichi di lavoro dei container. Quando si trovano nel cloud AWS, potrebbero anche implementare FSX per NetApp ONTAP per le esigenze di storage.

FSX per NetApp ONTAP offre protezione dei dati, affidabilità e flessibilità per le implementazioni di container in AWS. Astra Trident funge da provider di storage dinamico per consumare lo storage FSxN persistente per le applicazioni stateful dei clienti.

Poiché ROSA può essere implementato in modalità ha con nodi del piano di controllo distribuiti in più zone di disponibilità, FSX ONTAP può anche essere fornito con l'opzione Multi-AZ che fornisce alta disponibilità e protegge dai guasti AZ.

> Non sono previsti costi per il trasferimento dei dati quando si accede a un file system Amazon FSX dalla zona di disponibilità preferita (AZ) del file system. Per ulteriori informazioni sui prezzi, fare riferimento a. "qui".

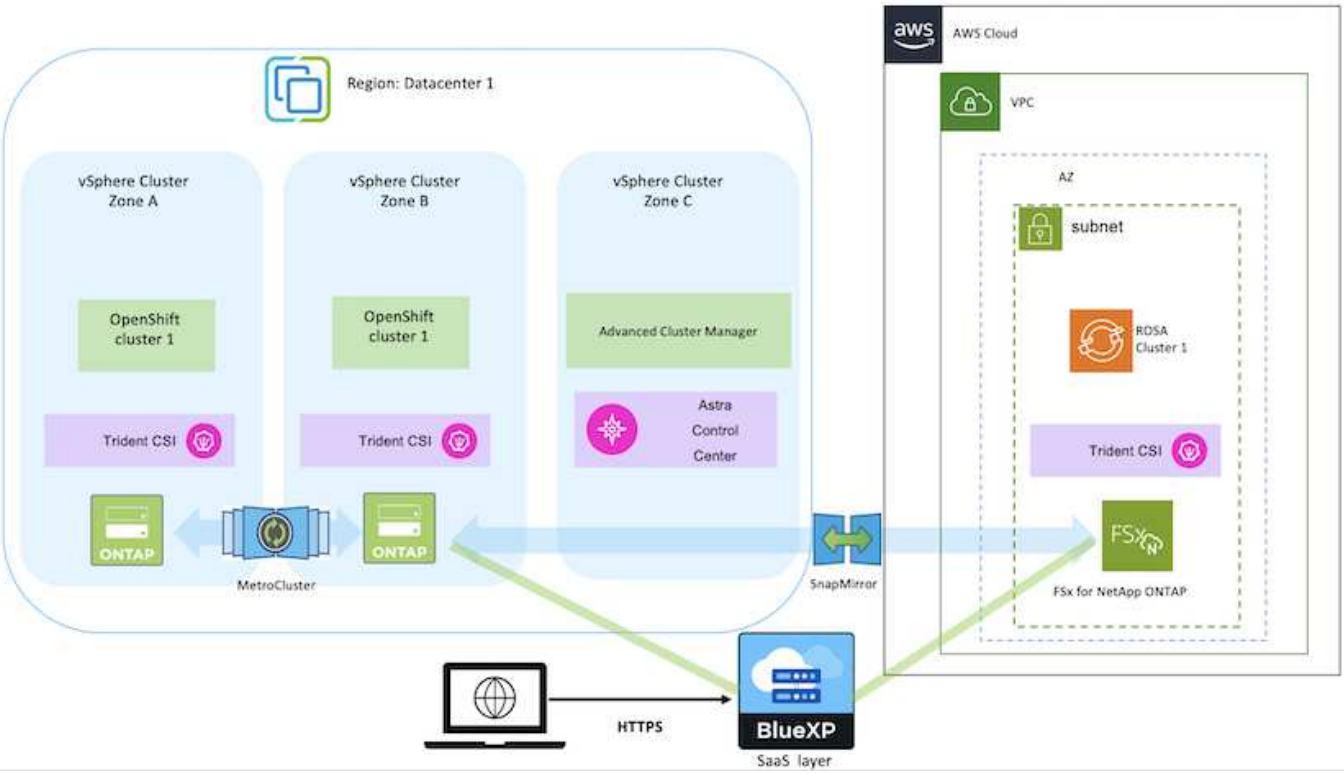
**Soluzione per la migrazione e la protezione dei dati per i carichi di lavoro dei container OpenShift**




# Implementa e configura la piattaforma container Managed Red Hat OpenShift su AWS

Questa sezione descrive un workflow di alto livello per la configurazione dei cluster Managed Red Hat OpenShift su AWS(ROSA). Mostra l'utilizzo di FSX gestito per NetApp ONTAP (FSxN) come back-end di storage di Astra Trident per fornire volumi persistenti. Vengono forniti dettagli sull'implementazione di FSxN su AWS utilizzando BlueXP. Inoltre, vengono forniti dettagli sull'utilizzo di BlueXP e OpenShift GitOps (Argo CD) per eseguire attività di migrazione e protezione dei dati per le applicazioni stateful sui cluster ROSA.

Di seguito è riportato un diagramma che illustra i cluster ROSA implementati su AWS e che utilizzano FSxN come storage back-end.




Questa soluzione è stata verificata utilizzando due cluster ROSA in due VPC in AWS. Ogni cluster ROSA è stato integrato con FSxN utilizzando Astra Trident. Esistono diversi modi per implementare I cluster ROSA e FSxN in AWS. Questa descrizione di alto livello dell'installazione fornisce collegamenti alla documentazione per il metodo specifico utilizzato. È possibile fare riferimento agli altri metodi nei relativi collegamenti forniti in "sezione risorse".

Il processo di installazione può essere suddiviso nei seguenti passaggi:

**Installare I cluster ROSA**

- Creare due VPC e configurare la connettività di peering VPC tra i VPC.
- Fare riferimento a. "qui" Per istruzioni sull'installazione dei cluster ROSA.

**Installare FSxN**

- Installare FSxN sui VPC da BlueXP. Fare riferimento a. "qui" Per la creazione di un account BlueXP e per iniziare. Fare riferimento a. "qui" Per l'installazione di FSxN. Fare riferimento a. "qui" Per creare un connettore in AWS per gestire FSxN.
- Implementare FSxN utilizzando AWS. Fare riferimento a. "qui" Per l'implementazione utilizzando la console AWS.

**Installare Trident sui cluster ROSA (utilizzando il grafico Helm)**

- USA il grafico Helm per installare Trident sui cluster ROSA. url del grafico Helm: https://netapp.github.io/trident-helm-chart

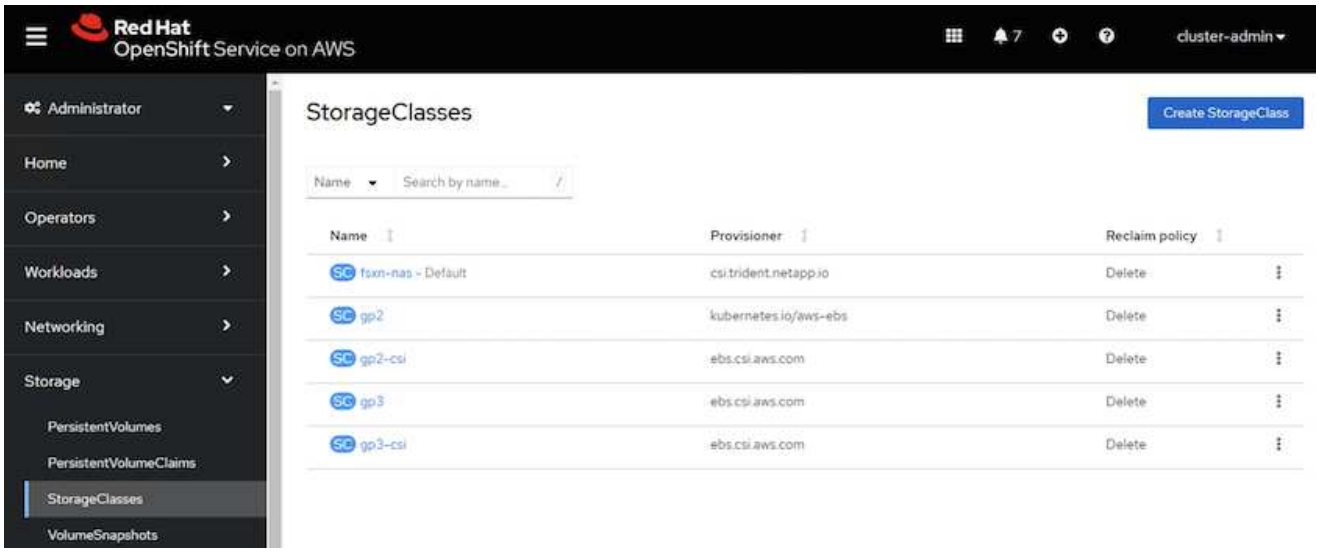
Integrazione di FSxN con Astra Trident per i cluster ROSA

OpenShift GitOps può essere utilizzato per implementare Astra Trident CSI su tutti i cluster gestiti, man mano che vengono registrati su ArgoCD utilizzando 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
```

**Creare classi di storage e backend utilizzando Trident (per FSxN)**

- Fare riferimento a. "qui" per informazioni dettagliate sulla creazione di classe di storage e backend.
- Rendere la classe di storage creata per FsxN con Trident CSI come predefinita da OpenShift Console. Vedere la schermata riportata di seguito:




**Implementare un'applicazione utilizzando OpenShift GitOps (CD Argo)**

- Installare l'operatore OpenShift GitOps sul cluster. Fare riferimento alle istruzioni "qui".
- Configurare una nuova istanza del CD Argo per il cluster. Fare riferimento alle istruzioni "qui".

Aprire la console del CD Argo e implementare un'applicazione. Ad esempio, puoi implementare un'applicazione Jenkins utilizzando il CD Argo con Helm Chart. Durante la creazione dell'applicazione, sono stati forniti i seguenti dettagli: Progetto: Cluster predefinito: https://kubernetes.default.svcSpazio dei nomi: Jenkins l'URL per il grafico Helm: https://charts.bitnami.com/bitnami

Parametri Helm: Global.storageClass: Fsxn-nas

# Protezione dei dati

Questa pagina mostra le opzioni di protezione dei dati per i cluster Managed Red Hat OpenShift on AWS (ROSA) utilizzando Astra Control Service. Astra Control Service (ACS) offre un'interfaccia grafica utente di facile utilizzo che consente di aggiungere cluster, definire le applicazioni in esecuzione ed eseguire attività di gestione dei dati integrate con le applicazioni. È possibile accedere alle funzioni ACS anche utilizzando un'API che consente l'automazione dei workflow.

L'alimentazione di Astra Control (ACS o ACC) è NetApp Astra Trident. Astra Trident integra diversi tipi di cluster Kubernetes come Red Hat OpenShift, EKS, AKS, SUSE Rancher, anthos ecc., con varie soluzioni di storage NetApp ONTAP come FAS/AFF, ONTAP Select, CVO, Google Cloud Volumes Service, Azure NetApp Files e Amazon FSX per NetApp ONTAP.

Questa sezione fornisce dettagli sulle seguenti opzioni di protezione dei dati con ACS:

- Un video che mostra il backup e il ripristino di un'applicazione ROSA in esecuzione in una regione e il ripristino in un'altra regione.
- Un video che mostra l'istantanea e il ripristino di un'applicazione ROSA.
- Dettagli dettagliati sull'installazione di un cluster ROSA, Amazon FSX per NetApp ONTAP, utilizzando NetApp Astra Trident per l'integrazione con il backend di storage, installazione di un'applicazione postgresql su un cluster ROSA, utilizzando ACS per creare una snapshot dell'applicazione e il ripristino dell'applicazione da esso.
- Un blog che mostra i dettagli passo per passo della creazione e del ripristino da uno snapshot per un'applicazione mysql su un cluster ROSA con FSX per ONTAP usando ACS.

## Backup/Ripristino da backup

Il video seguente mostra il backup di un'applicazione ROSA in esecuzione in una regione e il ripristino in un'altra regione.

FSX NetApp ONTAP per il servizio Red Hat OpenShift su AWS

## Snapshot/Ripristina da snapshot

Il video seguente mostra come scattare un'istantanea di un'applicazione ROSA e come eseguire il ripristino dall'istantanea dopo.

Snapshot/ripristino per le applicazioni su Red Hat OpenShift Service su cluster AWS (ROSA) con Amazon FSX per lo storage NetApp ONTAP

## Blog in inglese

- "Utilizzo di Astra Control Service per la gestione dei dati delle app su cluster ROSA con storage Amazon FSX"

## Dettagli dettagliati per creare snapshot e ripristinarle

### Impostazione dei prerequisiti

- "Account AWS"
- "Account Red Hat OpenShift"
- Utente IAM con "autorizzazioni appropriate" Per creare e accedere al cluster ROSA
- "CLI AWS"
- "ROSA CLI"
- "CLI OpenShift"(oc)
- VPC con subnet e gateway e percorsi appropriati
- "ROSA Cluster installato" Nel VPC
- "Amazon FSX per NetApp ONTAP" Creato nello stesso VPC
- Accesso al gruppo ROSA da "Console di cloud ibrido OpenShift"

**Passi successivi**

1. Creare un utente amministratore e accedere al cluster.
2. Creare un file kubeconfig per il cluster.
3. Installare Astra Trident nel cluster.
4. Creare una configurazione backend, di classe storage e di classe Snapshot utilizzando il provisioner Trident CSI.
5. Implementare un'applicazione postgresql nel cluster.
6. Creare un database e aggiungere un record.
7. Aggiungere il cluster in ACS.
8. Definire l'applicazione in ACS.
9. Creare uno snapshot utilizzando ACS.
10. Eliminare il database nell'applicazione postgresql.
11. Ripristino da uno snapshot utilizzando ACS.
12. Verifica che l'app sia stata ripristinata dall'istantanea.

#### 1. Creare un utente amministratore e accedere al cluster

Accedere al cluster ROSA creando un utente amministratore con il seguente comando: (È necessario creare un utente amministratore solo se non è stato creato uno al momento dell'installazione)

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

Il comando fornirà un output simile a quello riportato di seguito. Accedere al cluster utilizzando `oc login` comando fornito nell'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-00000-00000
```

> È inoltre possibile accedere al cluster utilizzando un token. Se hai già creato un utente admin al momento della creazione del cluster, puoi accedere al cluster dalla console Red Hat OpenShift Hybrid Cloud con le credenziali admin-user. Quindi, facendo clic sull'angolo in alto a destra in cui viene visualizzato il nome dell'utente connesso, è possibile ottenere `oc login` comando (accesso token) per la riga di comando.

#### 2. Creare un file kubeconfig per il cluster

Seguire le procedure "qui" Per creare un file kubeconfig per il cluster ROSA. Questo file kubeconfig verrà utilizzato in seguito quando si aggiunge il cluster in ACS.

**3. Installare Astra Trident sul cluster**

Installare Astra Trident (versione più recente) sul cluster ROSA. A tale scopo, è possibile seguire una qualsiasi delle procedure indicate "qui". Per installare Trident utilizzando helm dalla console del cluster, creare prima un progetto chiamato Trident.


Quindi, dalla vista sviluppatore, creare un archivio grafico Helm. Per il campo URL utilizzare `'https://netapp.github.io/trident-helm-chart'`. Quindi, creare una release helm per l'operatore Trident.

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**

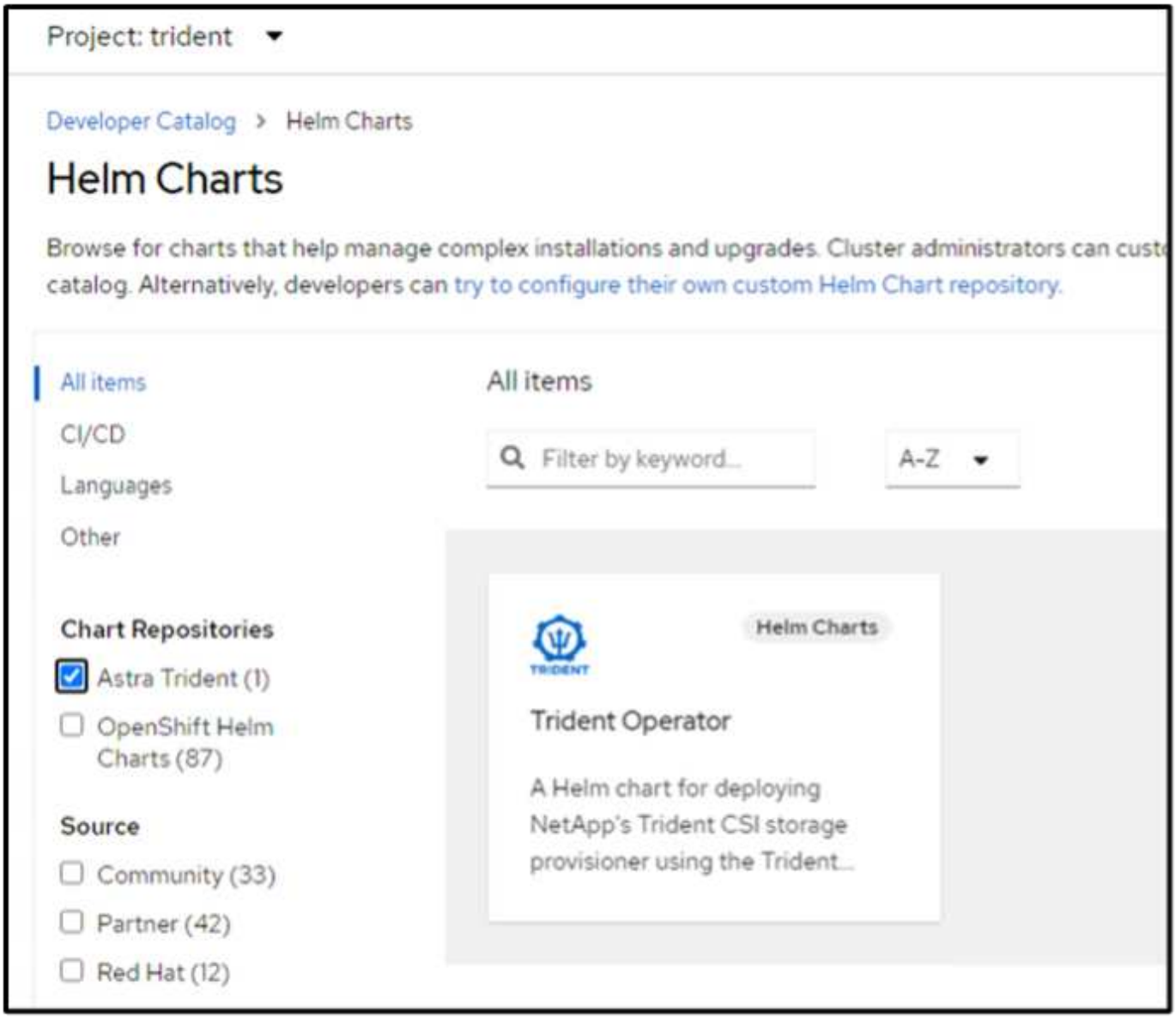
NetApp Astra Trident

A description for the Helm Chart repository.

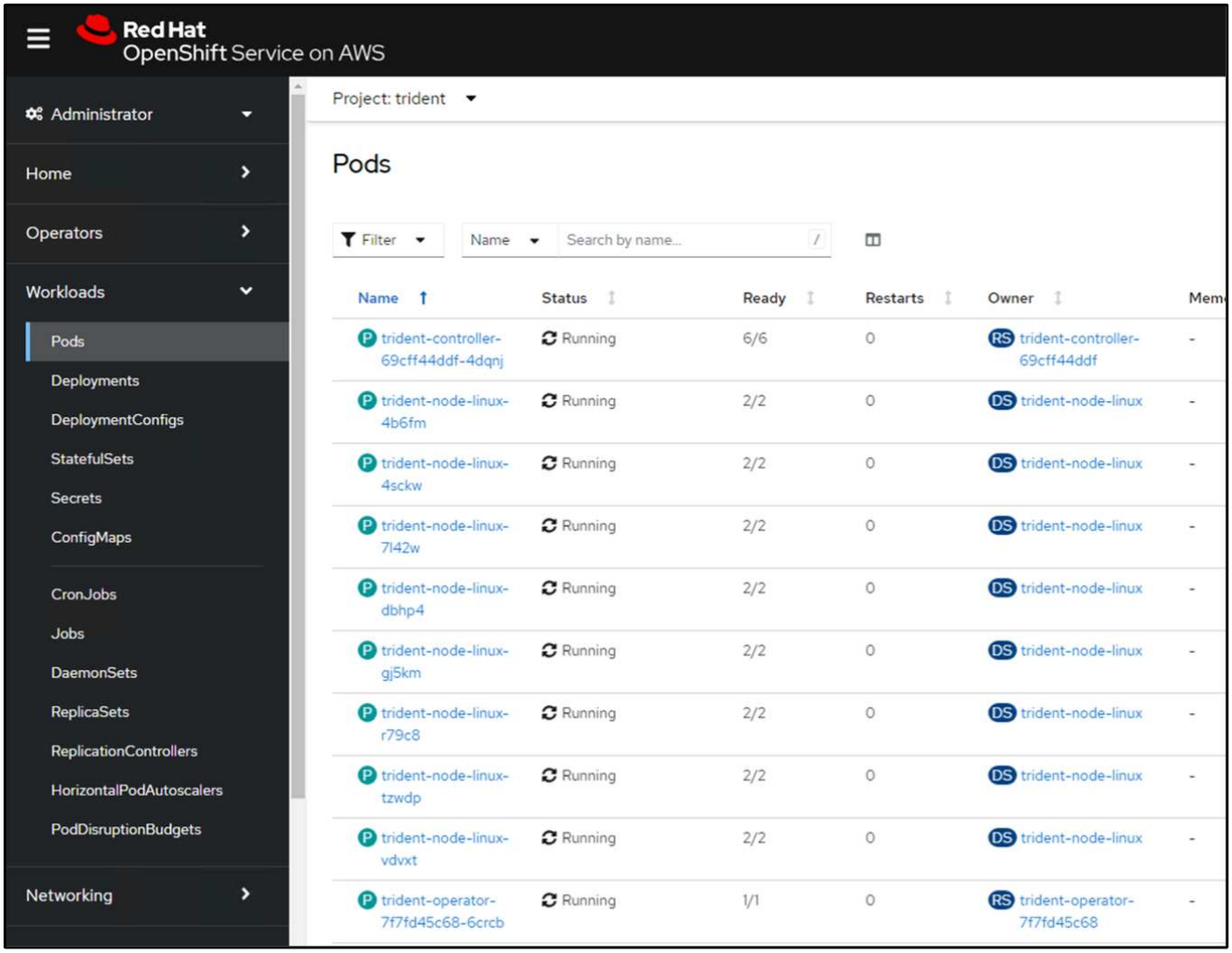
[ ] Disable usage of the repo in the developer catalog.

**URL** *

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



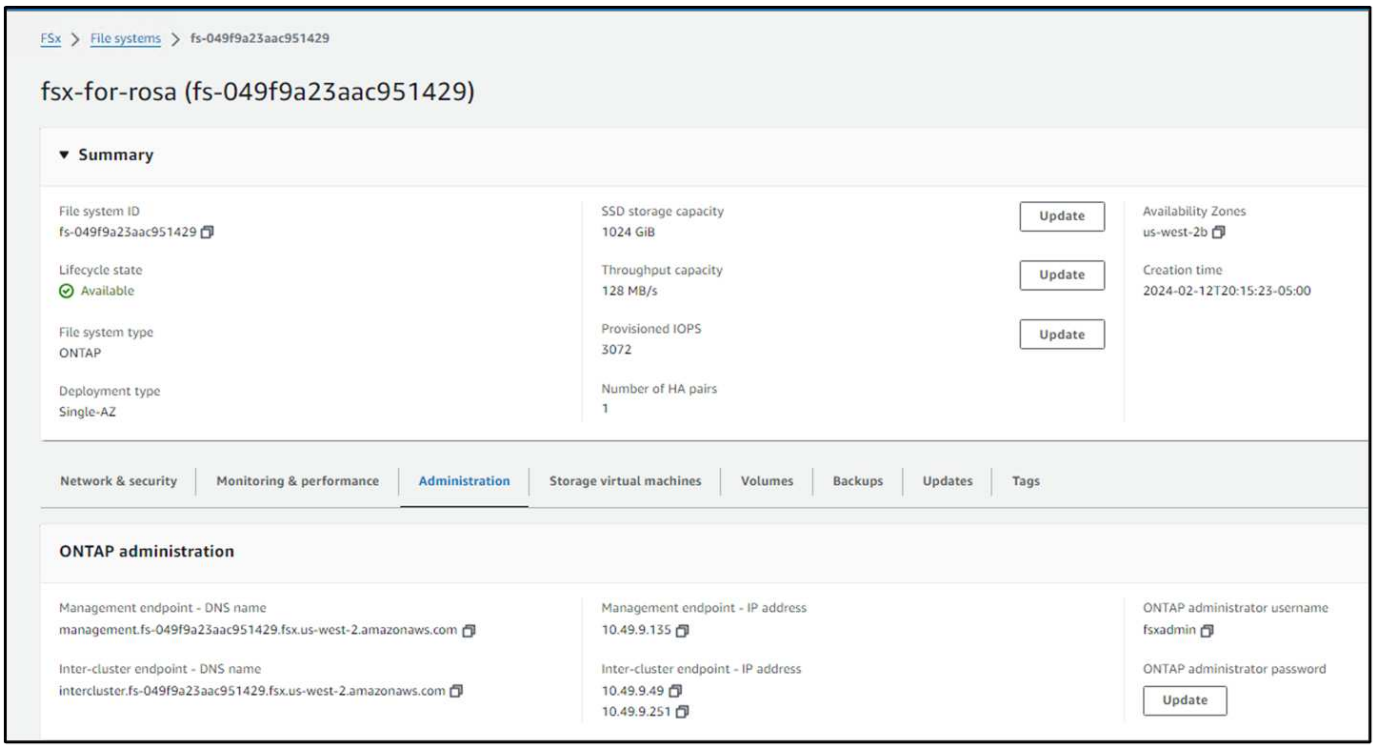

Verificare che tutti i pod di trident siano in esecuzione tornando alla vista Amministratore sulla console e selezionando i pod nel progetto trident.




**4. Creare una configurazione backend, di classe storage e di classe snapshot utilizzando il provisioner Trident CSI**

Utilizzare i file yaml illustrati di seguito per creare un oggetto backend tridente, un oggetto di classe di archiviazione e l'oggetto Volumesnapshot. Assicurati di fornire le credenziali al file system Amazon FSX per NetApp ONTAP che hai creato, la LIF di gestione e il nome del vserver del tuo file system nella configurazione yaml per il back-end. Per visualizzare questi dettagli, vai alla console AWS per Amazon FSX e seleziona il file system, quindi accedi alla scheda Administration (Amministrazione). Inoltre, fare clic su Update (Aggiorna) per impostare la password di `fsxadmin` utente.

> È possibile utilizzare la riga di comando per creare gli oggetti o con i file yaml dalla console del cloud ibrido.




### Configurazione del backend Trident

```
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
```

### Classe di stoccaggio

```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
```

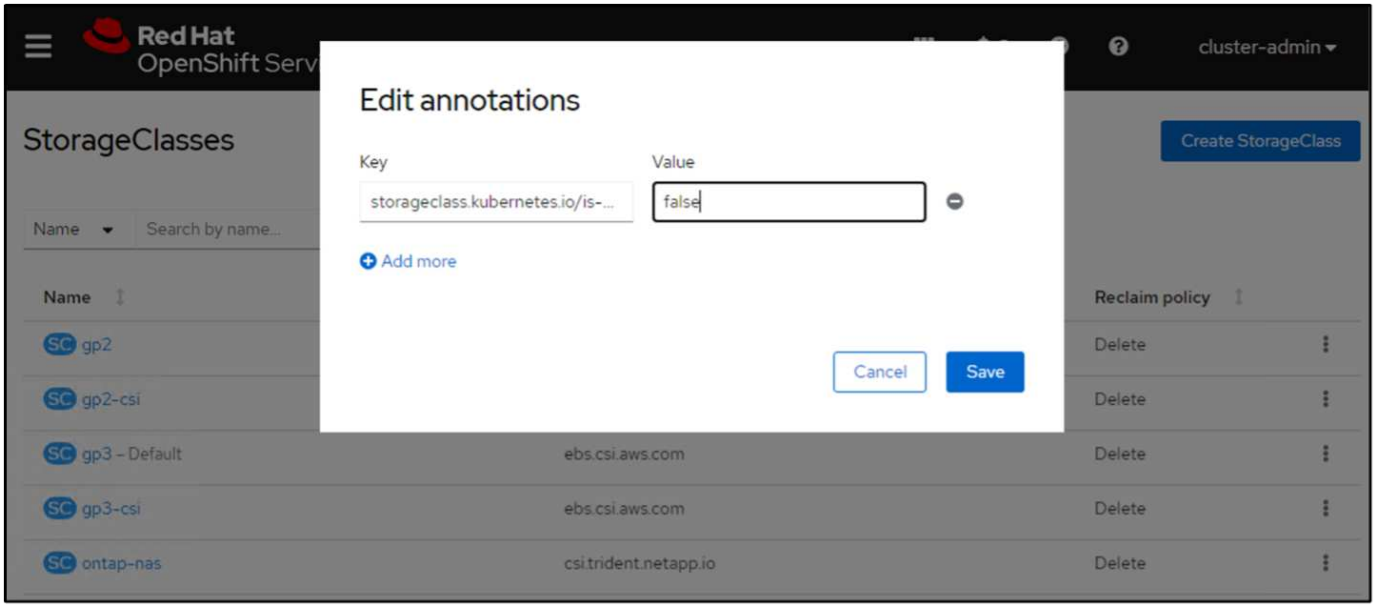
**classe istantanea**

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

Verificare che gli oggetti backend, di storage e trident-snapshotclass siano creati inviando i comandi indicati di seguito.

```
[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]$
```

In questo momento, un'importante modifica da apportare è impostare ontap-nas come classe di storage predefinita invece di GP3, in modo che l'app postgresql implementata in seguito possa utilizzare la classe di storage predefinita. Nella console OpenShift del cluster, in Storage selezionare StorageClasses. Modificare l'annotazione della classe predefinita corrente in modo che sia false e aggiungere l'impostazione della classe annotation storageclass.kuPensioni.io/is-default-class su true per la classe storage ontap-nas.





**5. Distribuire un'applicazione postgresql sul cluster**

È possibile distribuire l'applicazione dalla riga di comando nel modo seguente:

```
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]$
```

Se i pod delle applicazioni non sono in esecuzione, potrebbe essersi verificato un errore dovuto ai vincoli del contesto di protezione.

```
[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]$
```

Correggere l'errore modificando `runAsUser` e. `fsGroup` campi in `statefuleset.apps/postgresql` oggetto con l'uid che si trova nell'output di `oc get project` comando come illustrato di seguito.

```
[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]$
```

L'app postgresql deve essere in esecuzione e utilizzare volumi persistenti supportati da Amazon FSX per lo storage NetApp ONTAP.

```
[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]$
```

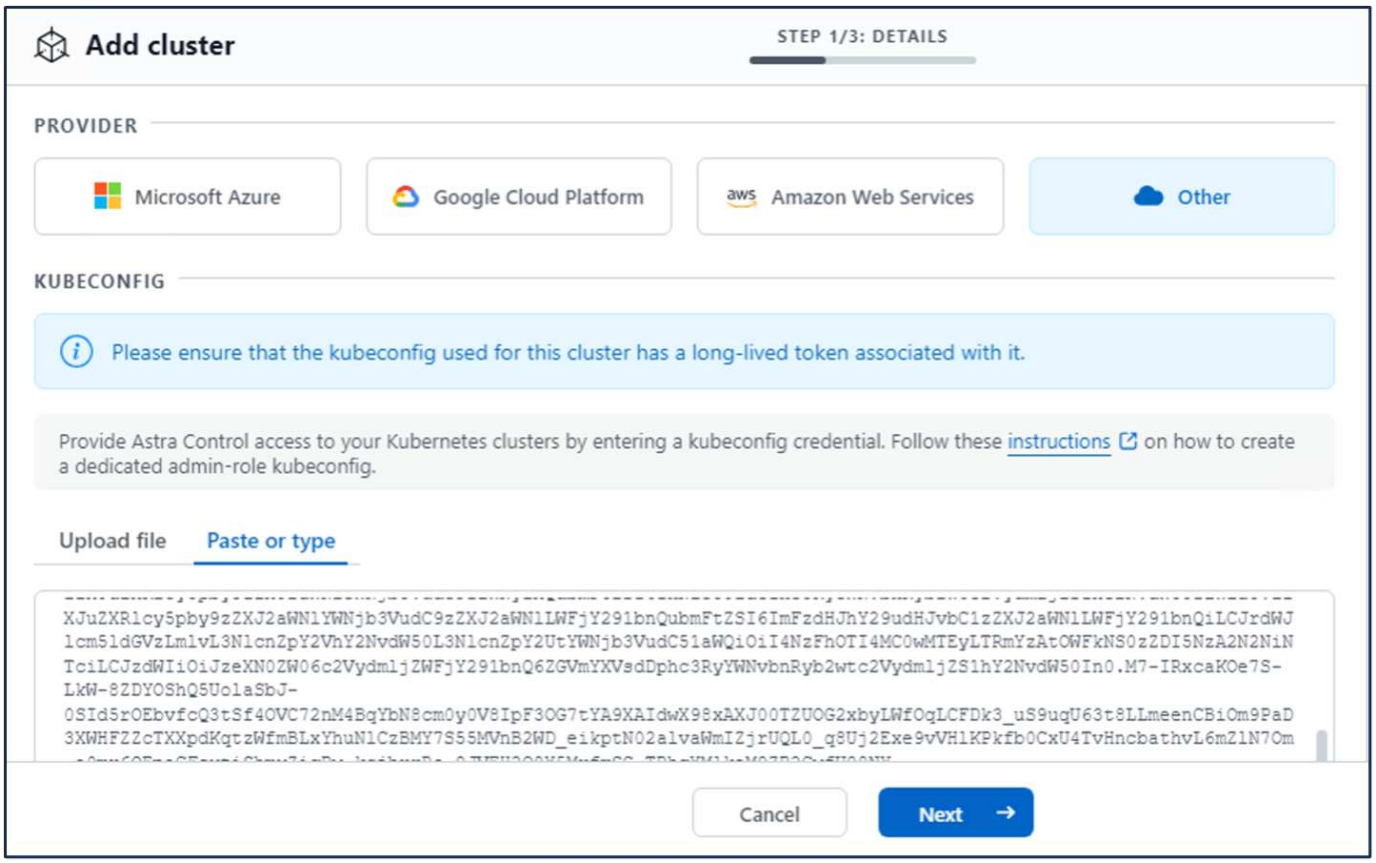
```
[ec2-user@ip-10-49-11-132 storage]$ kubectl get pvc -n postgresql
NAME                STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
data-postgresql-0   Bound    pvc-dd09524a-de75-4825-9424-03a9b91195ca   8Gi        RWO            ontap-nas      4m2s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

**6. Creare un database e aggiungere un record**

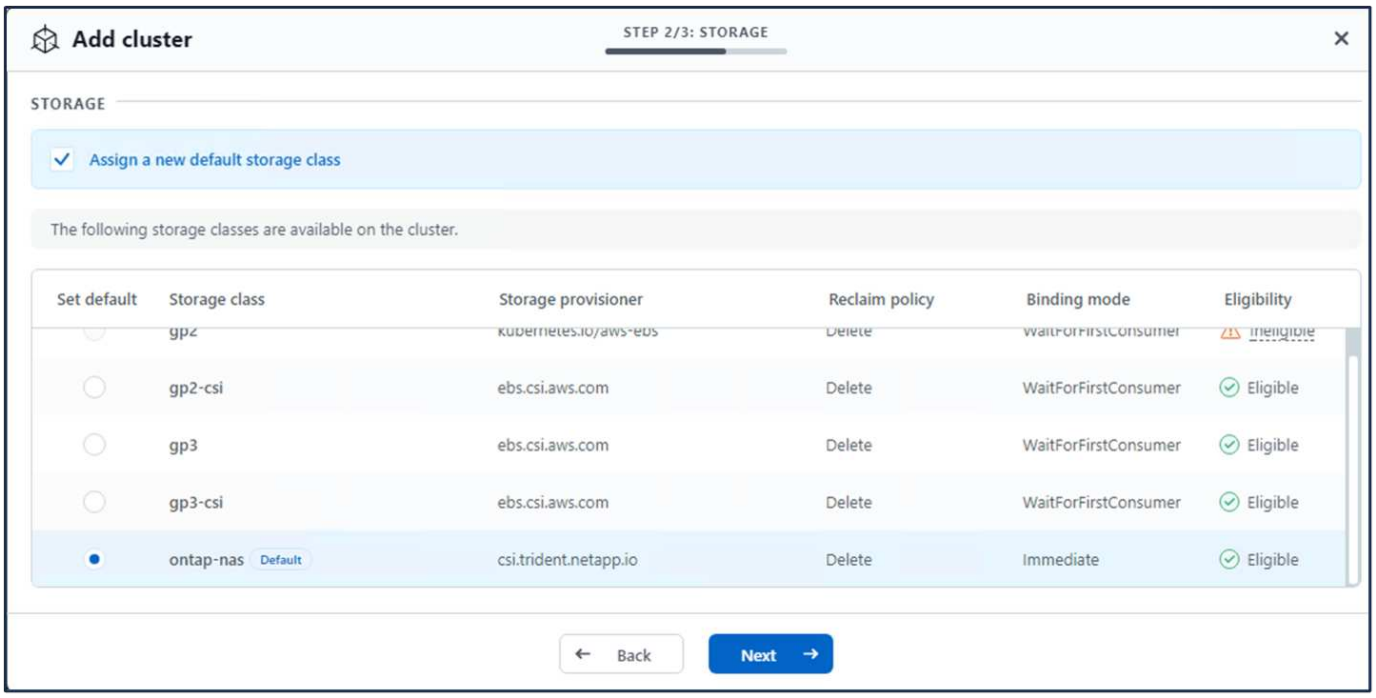
```
[ec2-user@ip-10-49-11-132 astra]$ export POSTGRES_PASSWORD=$(kubectl get secret --namespace postgresql postgresql -o jsonpath="
{.data.postgres-password}" | base64 -d)
[ec2-user@ip-10-49-11-132 astra]$ kubectl run postgresql-client --rm --tty -i --restart='Never' --namespace postgresql --image
docker.io/bitnami/postgresql:16.2.0-debian-11-r1 --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 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. Aggiungere il cluster in ACS**

Accedere a ACS. Selezionare cluster e fare clic su Add. Selezionare Altro e caricare o incollare il file kubeconfig.



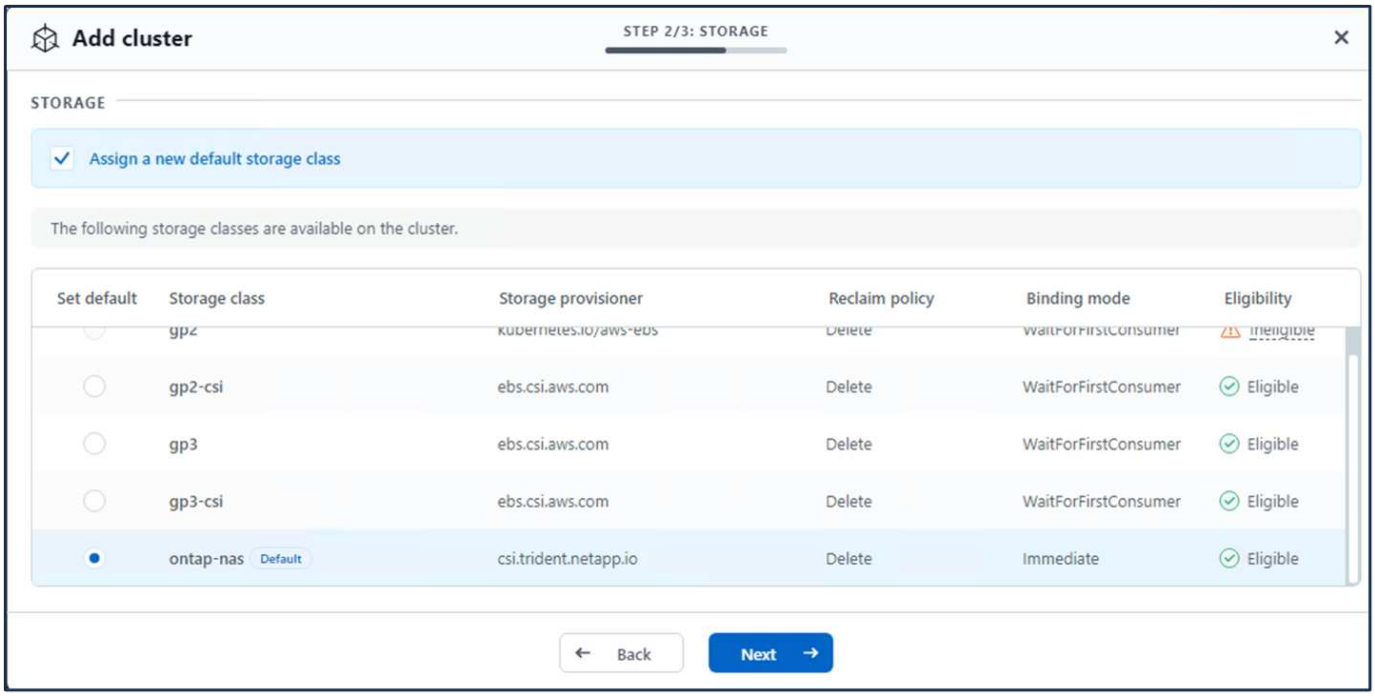

Fare clic su **Avanti** e selezionare ontap-nas come classe di storage predefinita per ACS. Fare clic su **Avanti**, rivedere i dettagli e **Aggiungi** il cluster.




#### 8. Definire l'applicazione in ACS

Definire l'applicazione postgresql in ACS. Dalla pagina di destinazione, selezionare **applicazioni**, **Definisci** e inserire i dettagli appropriati. Fare clic su **Avanti** un paio di volte, rivedere i dettagli e fare clic su **Definisci**.

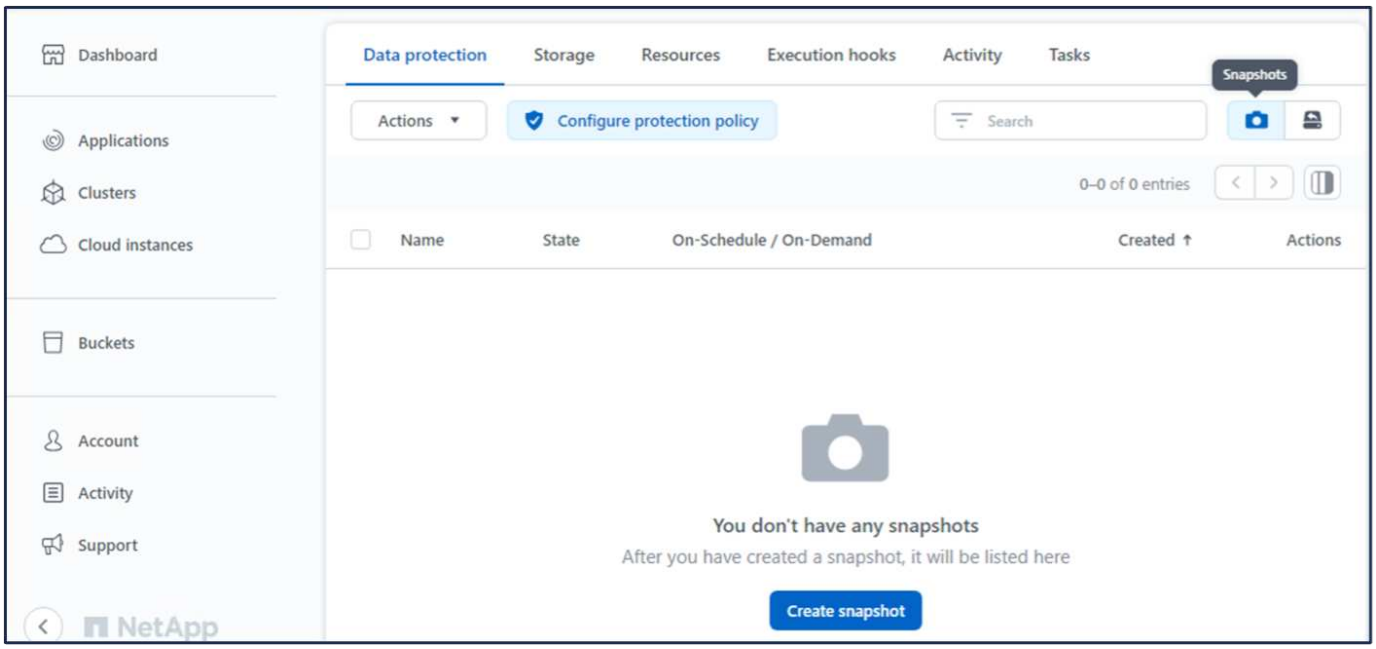
L'applicazione viene aggiunta a ACS.

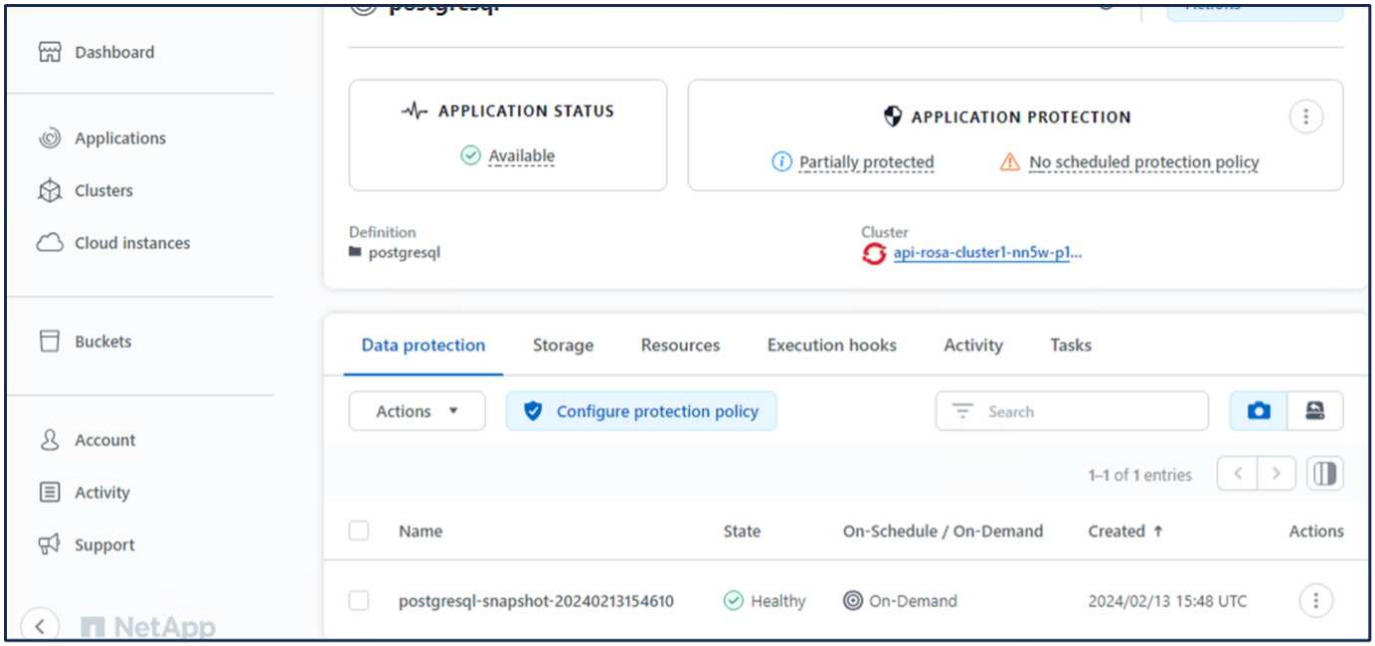



**9. Creare un'istantanea utilizzando ACS**

Esistono molti modi per creare uno snapshot in ACS. È possibile selezionare l'applicazione e creare un'istantanea dalla pagina che mostra i dettagli dell'applicazione. È possibile fare clic su Create Snapshot (Crea snapshot) per creare uno snapshot on-demand o configurare una policy di protezione.

Per creare un'istantanea su richiesta, è sufficiente fare clic su **Crea istantanea**, fornire un nome, rivedere i dettagli e fare clic su **istantanea**. Lo stato dell'istantanea diventa sano al termine dell'operazione.






**10. Eliminare il database nell'applicazione postgresql**

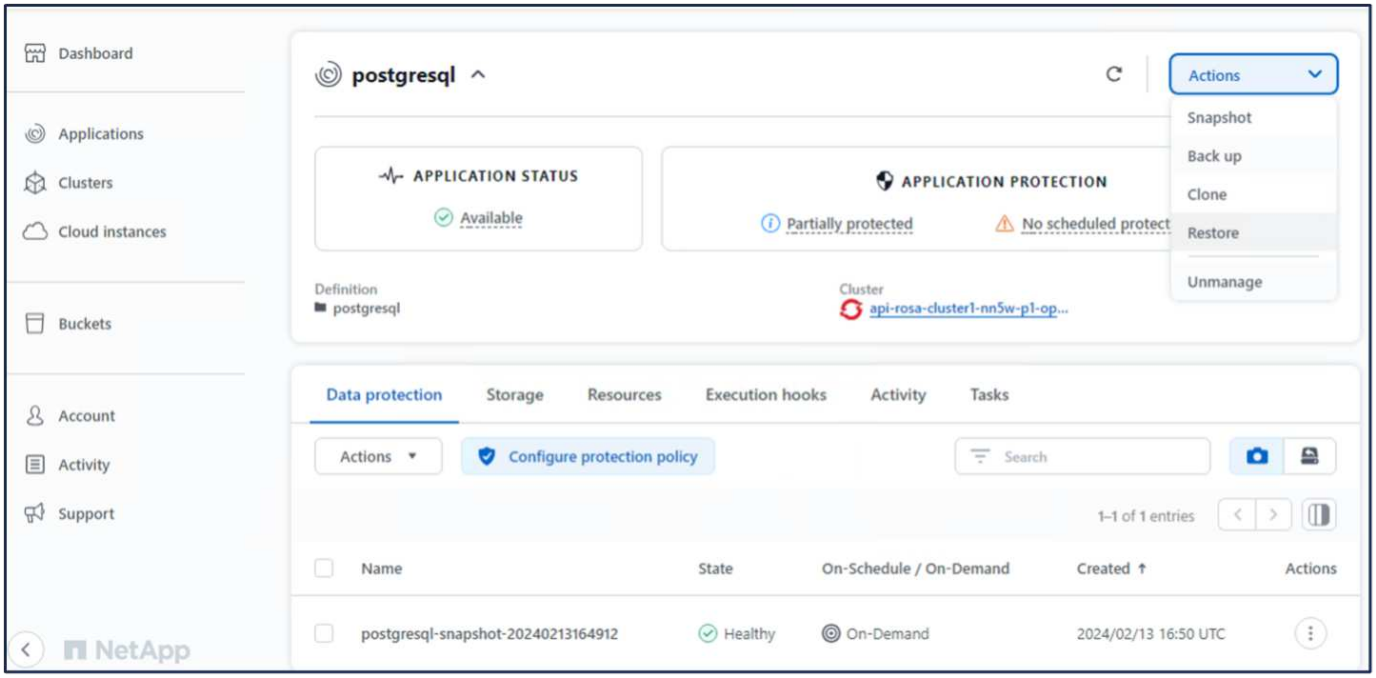
Accedere nuovamente a postgresql, elencare i database disponibili, eliminare quello creato in precedenza ed elencare nuovamente per assicurarsi che il database sia stato eliminato.

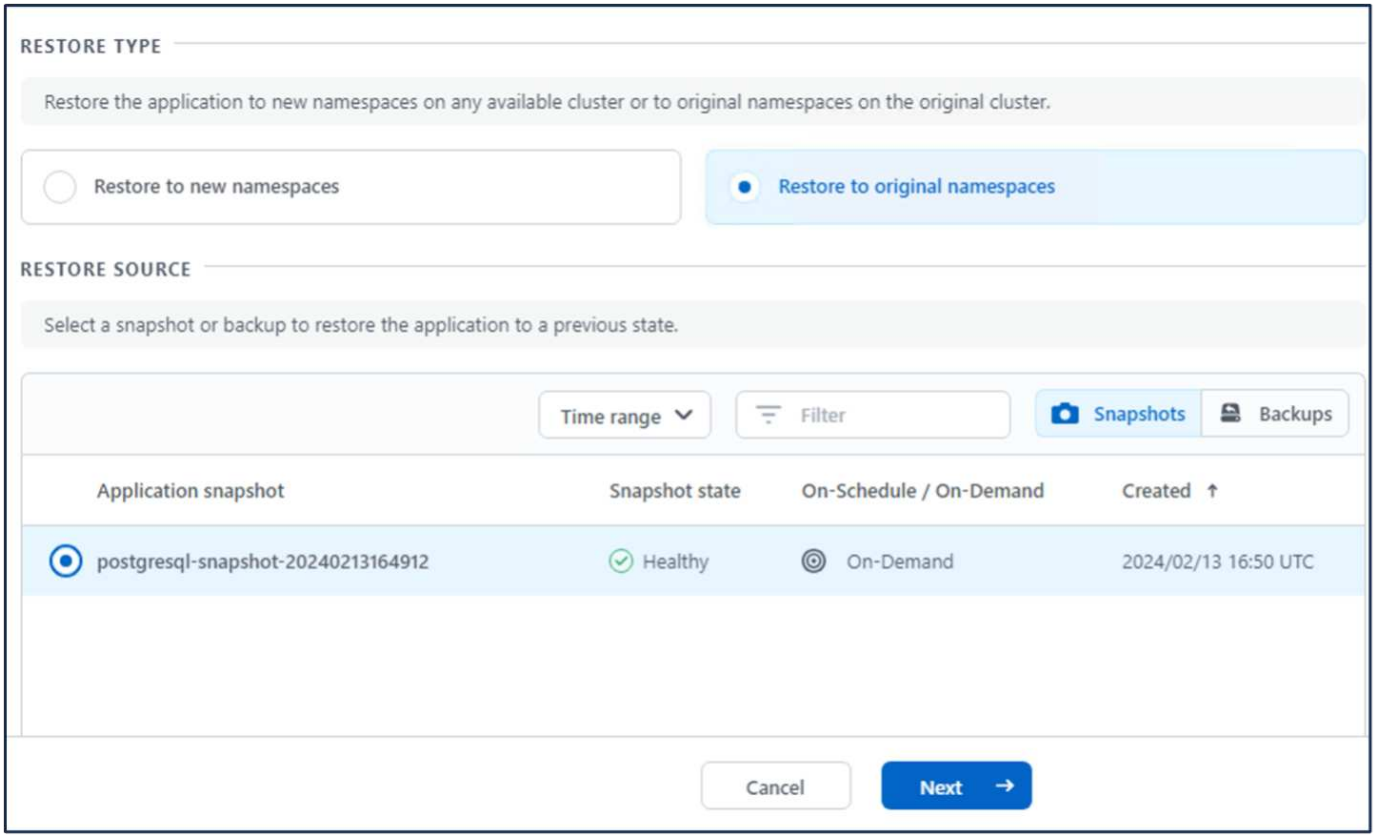
```
postgres=# \l
                                                    List of databases
   Name    |  Owner   | Encoding | Locale Provider |   Collate   |    Ctype    | ICU Locale | ICU Rules |   Access priv
-----------+----------+----------+-----------------+-------------+-------------+------------+-----------+---------------
 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)
```

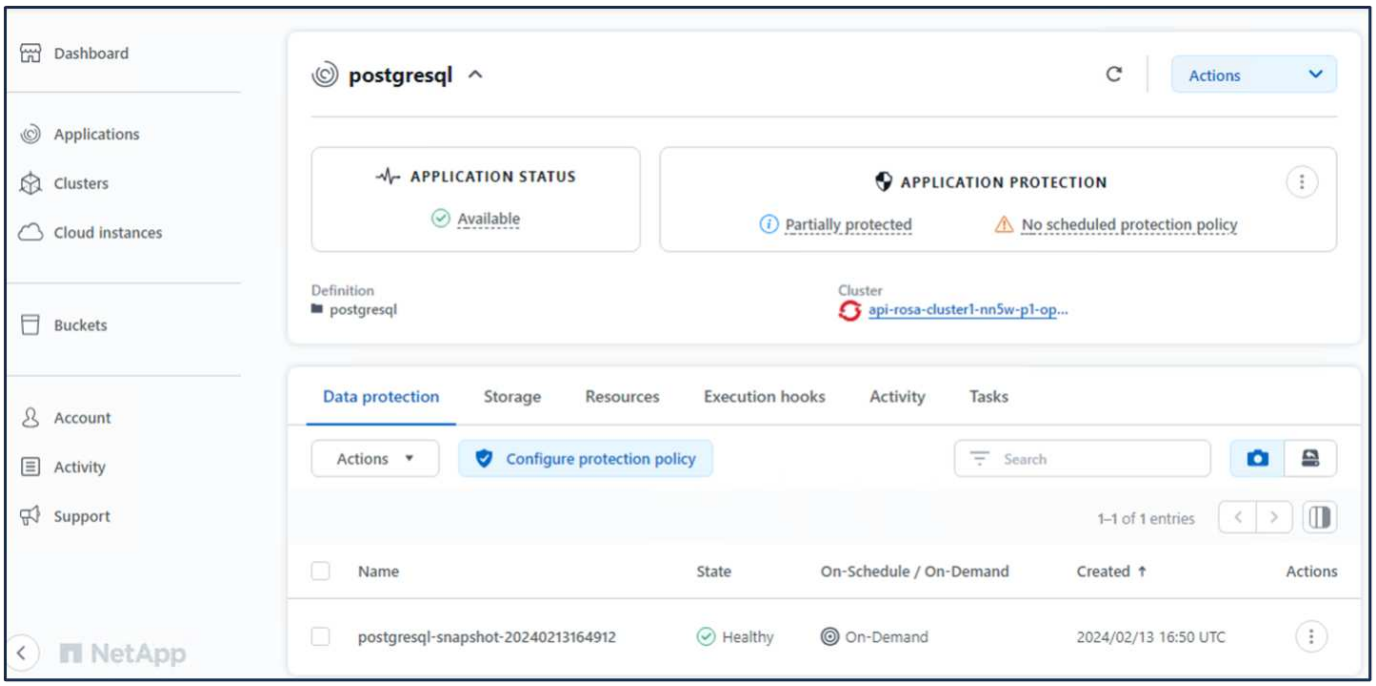
**11. Ripristino da uno snapshot utilizzando ACS**

Per ripristinare l'applicazione da uno snapshot, andare alla pagina di destinazione dell'interfaccia utente ACS, selezionare l'applicazione e selezionare Ripristina. È necessario scegliere uno snapshot o un backup da cui

eseguire il ripristino. (In genere, si creerebbero più criteri in base a un criterio configurato). Effettuare le scelte appropriate nelle due schermate successive, quindi fare clic su **Ripristina**. Lo stato dell'applicazione passa da Ripristino a disponibile dopo il ripristino dallo snapshot.






**12. Verifica che l'app sia stata ripristinata dall'istantanea**

Accedere al client postgresql e si dovrebbe ora vedere la tabella e il record nella tabella che si aveva in precedenza. Tutto qui. Basta fare clic su un pulsante per ripristinare lo stato precedente dell'applicazione. Con Astra Control, possiamo renderla semplice per i nostri clienti.

```
[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)
```

# Migrazione dei dati

Questa pagina mostra le opzioni di migrazione dei dati per i carichi di lavoro container sui cluster gestiti Red Hat OpenShift che utilizzano FSX per NetApp ONTAP per lo storage persistente.

## Migrazione dei dati

Il servizio Red Hat OpenShift su AWS e FSX per NetApp ONTAP (FSxN) fanno parte del loro portfolio di servizi di AWS. FSxN è disponibile nelle opzioni AZ singolo o AZ multiplo. L'opzione Multi-Az offre la protezione dei dati dai guasti della zona di disponibilità. FSxN può essere integrato con Astra Trident per fornire storage persistente per le applicazioni sui cluster ROSA.

### Integrazione di FSxN con Trident utilizzando Helm Chart

Integrazione cluster ROSA con Amazon FSX per ONTAP

La migrazione delle applicazioni container comporta:

- Volumi persistenti: Questa operazione può essere eseguita utilizzando BlueXP. Un'altra opzione consiste nell'utilizzare Astra Control Center per gestire le migrazioni delle applicazioni container dall'ambiente on-premise a quello cloud. L'automazione può essere utilizzata per lo stesso scopo.
- Metadati dell'applicazione: È possibile eseguire questa operazione utilizzando OpenShift GitOps (Argo CD).

### Failover e fail-back delle applicazioni sul cluster ROSA utilizzando FSxN per lo storage persistente

Il seguente video è una dimostrazione degli scenari di failover e fail-back delle applicazioni che utilizzano BlueXP e il CD Argo.

Failover e failback delle applicazioni sul cluster ROSA

### Soluzione per la migrazione e la protezione dei dati per i carichi di lavoro dei container OpenShift