ONTAP NAS drivers

Trident

NetApp
February 02, 2026

Table of Contents

ONTAP NAS drivers

ONTAP NAS driver overview

Learn about configuring an ONTAP backend with ONTAP and Cloud Volumes ONTAP NAS drivers.

ONTAP NAS driver details

Trident provides the following NAS storage drivers to communicate with the ONTAP cluster. Supported access modes are: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Driver	Protocol	volumeMode	Access modes supported	File systems supported
`ontap-nas`	NFS SMB	Filesystem	RWO, ROX, RWX, RWOP	"", `nfs`, `smb`
`ontap-nas-economy`	NFS SMB	Filesystem	RWO, ROX, RWX, RWOP	"", `nfs`, `smb`
`ontap-nas-flexgroup`	NFS SMB	Filesystem	RWO, ROX, RWX, RWOP	"", `nfs`, `smb`

- Use `ontap-san-economy` only if persistent volume usage count is expected to be higher than supported ONTAP volume limits.
- Use `ontap-nas-economy` only if persistent volume usage count is expected to be higher than supported ONTAP volume limits and the `ontap-san-economy` driver cannot be used.
- Do not use use `ontap-nas-economy` if you anticipate the need for data protection, disaster recovery, or mobility.
- NetApp does not recommend using Flexvol autogrow in all ONTAP drivers, except ontap-san. As a workaround, Trident supports the use of snapshot reserve and scales Flexvol volumes accordingly.

User permissions

Trident expects to be run as either an ONTAP or SVM administrator, typically using the `admin` cluster user or a `vsadmin` SVM user, or a user with a different name that has the same role.

For Amazon FSx for NetApp ONTAP deployments, Trident expects to be run as either an ONTAP or SVM administrator, using the cluster `fsxadmin` user or a `vsadmin` SVM user, or a user with a different name that has the same role. The `fsxadmin` user is a limited replacement for the cluster admin user.

If you use the `limitAggregateUsage` parameter, cluster admin permissions are required. When using Amazon FSx for NetApp ONTAP with Trident, the `limitAggregateUsage` parameter will not work with the `vsadmin` and `fsxadmin` user accounts. The configuration operation will fail if you specify this parameter.

While it is possible to create a more restrictive role within ONTAP that a Trident driver can use, we don't recommend it. Most new releases of Trident will call additional APIs that would have to be accounted for, making upgrades difficult and error-prone.

Prepare to configure a backend with ONTAP NAS drivers

Understand the requirements, authentication options, and export policies for configuring an ONTAP backend with ONTAP NAS drivers.

Beginning with the 25.10 release, NetApp Trident supports NetApp AFX storage system. NetApp AFX storage systems differ from other ONTAP systems (ASA, AFF, and FAS) in the implementation of their storage layer.

Only the `ontap-nas` driver (with NFS protocol) is supported for AFX systems; SMB protocol is not supported.

In the Trident backend configuration, you need not specify that your system is AFX. When you select `ontap-nas` as the `storageDriverName`, Trident detects automatically the AFX systems.

Requirements

- For all ONTAP backends, Trident requires at least one aggregate be assigned to the SVM.
- You can run more than one driver, and create storage classes that point to one or the other. For example, you could configure a Gold class that uses the `ontap-nas` driver and a Bronze class that uses the `ontap-nas-economy` one.
- All your Kubernetes worker nodes must have the appropriate NFS tools installed. Refer to here for more details.
- Trident supports SMB volumes mounted to pods running on Windows nodes only. Refer to Prepare to provision SMB volumes for details.

Authenticate the ONTAP backend

Trident offers two modes of authenticating an ONTAP backend.

- Credential-based: This mode requires sufficient permissions to the ONTAP backend. It is recommended to use an account associated with a pre-defined security login role, such as `admin` or `vsadmin` to ensure maximum compatibility with ONTAP versions.
- Certificate-based: This mode requires a certificate installed on the backend for Trident to communicate with an ONTAP cluster. Here, the backend definition must contain Base64-encoded values of the client certificate, key, and the trusted CA certificate if used (recommended).

You can update existing backends to move between credential-based and certificate-based methods. However, only one authentication method is supported at a time. To switch to a different authentication method, you must remove the existing method from the backend configuration.

If you attempt to provide **both credentials and certificates**, backend creation will fail with an error that more than one authentication method was provided in the configuration file.

Enable credential-based authentication

Trident requires the credentials to an SVM-scoped/cluster-scoped admin to communicate with the ONTAP backend. It is recommended to make use of standard, pre-defined roles such as `admin` or `vsadmin`. This ensures forward compatibility with future ONTAP releases that might expose feature APIs to be used by future Trident releases. A custom security login role can be created and used with Trident, but is not recommended.

A sample backend definition will look like this:

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
credentials:
  name: secret-backend-creds
```

JSON

```json
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "credentials": {
        "name": "secret-backend-creds"
    }
}
```

Keep in mind that the backend definition is the only place the credentials are stored in plain text. After the backend is created, usernames/passwords are encoded with Base64 and stored as Kubernetes secrets. The creation/updation of a backend is the only step that requires knowledge of the credentials. As such, it is an admin-only operation, to be performed by the Kubernetes/storage administrator.

Enable certificate-based Authentication

New and existing backends can use a certificate and communicate with the ONTAP backend. Three parameters are required in the backend definition.

- clientCertificate: Base64-encoded value of client certificate.
- clientPrivateKey: Base64-encoded value of associated private key.
- trustedCACertificate: Base64-encoded value of trusted CA certificate. If using a trusted CA, this parameter must be provided. This can be ignored if no trusted CA is used.

A typical workflow involves the following steps.

Steps

1. Generate a client certificate and key. When generating, set Common Name (CN) to the ONTAP user to authenticate as.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=vsadmin"
```

2. Add trusted CA certificate to the ONTAP cluster. This might be already handled by the storage administrator. Ignore if no trusted CA is used.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. Install the client certificate and key (from step 1) on the ONTAP cluster.

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. Confirm the ONTAP security login role supports `cert` authentication method.

```
security login create -user-or-group-name vsadmin -application ontapi
-authentication-method cert -vserver <vserver-name>
security login create -user-or-group-name vsadmin -application http
-authentication-method cert -vserver <vserver-name>
```

5. Test authentication using certificate generated. Replace <ONTAP Management LIF> and <vserver name> with Management LIF IP and SVM name. You must ensure the LIF has its service policy set to `default-data-management`.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Encode certificate, key and trusted CA certificate with Base64.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. Create backend using the values obtained from the previous step.

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Update authentication methods or rotate credentials

You can update an existing backend to use a different authentication method or to rotate their credentials. This works both ways: backends that make use of username/password can be updated to use certificates; backends that utilize certificates can be updated to username/password based. To do this, you must remove the existing authentication method and add the new authentication method. Then use the updated backend.json file containing the required parameters to execute `tridentctl update backend`.

```
cat cert-backend-updated.json
```

```
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}
```

```
#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

When rotating passwords, the storage administrator must first update the password for the user on ONTAP. This is followed by a backend update. When rotating certificates, multiple certificates can be added to the user. The backend is then updated to use the new certificate, following which the old certificate can be deleted from the ONTAP cluster.

Updating a backend does not disrupt access to volumes that have already been created, nor impact volume connections made after. A successful backend update indicates that Trident can communicate with the ONTAP backend and handle future volume operations.

Create custom ONTAP role for Trident

You can create an ONTAP cluster role with minimum privileges so that you do not have to use the ONTAP admin role to perform operations in Trident. When you include the username in a Trident backend configuration, Trident uses the ONTAP cluster role you created to perform the operations.

Refer to Trident custom-role generator for more information about creating Trident custom roles.

Using ONTAP CLI

1. Create a new role using the following command:

```
security login role create <role_name\> -cmddirname "command" -access all
–vserver <svm_name\>
```

2. Create a usename for the Trident user:

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> –vserver
<svm_name\> -comment "user_description"
```

3. Map the role to the user:

```
security login modify username <user_name\> –vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Using System Manager

Perform the following steps in ONTAP System Manager:

1. **Create a custom role**:
   a. To create a custom role at the cluster-level, select **Cluster > Settings**.

      (Or) To create a custom role at the SVM level, select **Storage > Storage VMs > `required SVM`> Settings > Users and Roles**.

   b. Select the arrow icon (→) next to **Users and Roles**.
   c. Select **+Add** under **Roles**.
   d. Define the rules for the role and click **Save**.
2. **Map the role to the Trident user**:
   + Perform the following steps on the **Users and Roles** page:
   a. Select Add icon **+** under **Users**.
   b. Select the required username, and select a role in the drop-down menu for **Role**.
   c. Click **Save**.

Refer to the following pages for more information:

- Custom roles for administration of ONTAP or Define custom roles
- Work with roles and users

Manage NFS export policies

Trident uses NFS export policies to control access to the volumes that it provisions.

Trident provides two options when working with export policies:

- Trident can dynamically manage the export policy itself; in this mode of operation, the storage administrator specifies a list of CIDR blocks that represent admissible IP addresses. Trident adds applicable node IPs that fall in these ranges to the export policy automatically at publish time. Alternatively, when no CIDRs are specified, all global-scoped unicast IPs found on the node that the volume being published to will be added to the export policy.
- Storage administrators can create an export policy and add rules manually. Trident uses the default export policy unless a different export policy name is specified in the configuration.

Dynamically manage export policies

Trident provides the ability to dynamically manage export policies for ONTAP backends. This provides the storage administrator the ability to specify a permissible address space for worker node IPs, rather than defining explicit rules manually. It greatly simplifies export policy management; modifications to the export policy no longer require manual intervention on the storage cluster. Moreover, this helps restrict access to the storage cluster only to worker nodes that are mounting volumes and have IPs in the range specified, supporting a fine-grained and automated management.

Do not use Network Address Translation (NAT) when using dynamic export policies. With NAT, the storage controller sees the frontend NAT address and not the actual IP host address, so access will be denied when no match is found in the export rules.

Example

There are two configuration options that must be used. Here's an example backend definition:

```
---
version: 1
storageDriverName: ontap-nas-economy
backendName: ontap_nas_auto_export
managementLIF: 192.168.0.135
svm: svm1
username: vsadmin
password: password
autoExportCIDRs:
  - 192.168.0.0/24
autoExportPolicy: true
```

When using this feature, you must ensure that the root junction in your SVM has a previously created export policy with an export rule that permits the node CIDR block (such as the default export policy). Always follow NetApp recommended best practice to dedicate an SVM for Trident.

Here is an explanation of how this feature works using the example above:

- `autoExportPolicy` is set to `true`. This indicates that Trident creates an export policy for each volume provisioned with this backend for the `svm1` SVM and handle the addition and deletion of rules using `autoexportCIDRs` address blocks. Until a volume is attached to a node, the volume uses an empty export policy with no rules to prevent unwanted access to that volume. When a volume is published to a node Trident creates an export policy with the same name as the underlying qtree containing the node IP within the specified CIDR block. These IPs will also be added to the export policy used by the parent FlexVol volume
  - For example:
    - backend UUID 403b5326-8482-40db-96d0-d83fb3f4daec
    - `autoExportPolicy` set to `true`
    - storage prefix `trident`
    - PVC UUID a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - qtree named trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c creates an export policy for the FlexVol named `trident-403b5326-8482-40db96d0-d83fb3f4daec` , an export policy for the qtree named `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c`, and an empty export policy named `trident_empty` on the SVM. The rules for the FlexVol export policy will be a superset of any rules contained in the qtree export policies. The empty export policy will be reused by any volumes that are not attached.
- `autoExportCIDRs` contains a list of address blocks. This field is optional and it defaults to ["0.0.0.0/0", "::/0"]. If not defined, Trident adds all globally-scoped unicast addresses found on the worker nodes with publications.

In this example, the `192.168.0.0/24` address space is provided. This indicates that Kubernetes node IPs that fall within this address range with publications will be added to the export policy that Trident creates. When Trident registers a node it runs on, it retrieves the IP addresses of the node and checks them against the address blocks provided in `autoExportCIDRs`. At publish time, after filtering the IPs, Trident creates the export policy rules for the client IPs for the node it is publishing to.

You can update `autoExportPolicy` and `autoExportCIDRs` for backends after you create them. You can append new CIDRs for a backend that is automatically managed or delete existing CIDRs. Exercise care when deleting CIDRs to ensure that existing connections are not dropped. You can also choose to disable `autoExportPolicy` for a backend and fall back to a manually created export policy. This will require setting the `exportPolicy` parameter in your backend config.

After Trident creates or updates a backend, you can check the backend using `tridentctl` or the corresponding `tridentbackend` CRD:

```
./tridentctl get backends ontap_nas_auto_export -n trident -o yaml
items:
- backendUUID: 403b5326-8482-40db-96d0-d83fb3f4daec
  config:
    aggregate: ""
    autoExportCIDRs:
    - 192.168.0.0/24
    autoExportPolicy: true
    backendName: ontap_nas_auto_export
    chapInitiatorSecret: ""
    chapTargetInitiatorSecret: ""
    chapTargetUsername: ""
    chapUsername: ""
    dataLIF: 192.168.0.135
    debug: false
    debugTraceFlags: null
    defaults:
      encryption: "false"
      exportPolicy: <automatic>
      fileSystemType: ext4
```

When a node is removed, Trident checks all export policies to remove the access rules corresponding to the node. By removing this node IP from the export policies of managed backends, Trident prevents rogue mounts, unless this IP is reused by a new node in the cluster.

For previously existing backends, updating the backend with `tridentctl update backend` ensures that Trident manages the export policies automatically. This creates two new export policies named after the backend's UUID and qtree name when they are needed. Volumes that are present on the backend will use the newly created export policies after they are unmounted and mounted again.

> Deleting a backend with auto-managed export policies will delete the dynamically created export policy. If the backend is re-created, it is treated as a new backend and will result in the creation of a new export policy.

If the IP address of a live node is updated, you must restart the Trident pod on the node. Trident will then update the export policy for backends it manages to reflect this IP change.

Prepare to provision SMB volumes

With a little additional preparation, you can provision SMB volumes using `ontap-nas` drivers.

> You must configure both NFS and SMB/CIFS protocols on the SVM to create an `ontap-nas-economy` SMB volume for ONTAP on-premises clusters. Failure to configure either of these protocols will cause SMB volume creation to fail.

> `autoExportPolicy` is not supported for SMB volumes.

Before you begin

Before you can provision SMB volumes, you must have the following.

- A Kubernetes cluster with a Linux controller node and at least one Windows worker node running Windows Server 2022. Trident supports SMB volumes mounted to pods running on Windows nodes only.
- At least one Trident secret containing your Active Directory credentials. To generate secret `smbcreds`:

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- A CSI proxy configured as a Windows service. To configure a `csi-proxy`, refer to GitHub: CSI Proxy or GitHub: CSI Proxy for Windows for Kubernetes nodes running on Windows.

Steps

1. For on-premises ONTAP, you can optionally create an SMB share or Trident can create one for you.

   SMB shares are required for Amazon FSx for ONTAP.

   You can create the SMB admin shares in one of two ways either using the Microsoft Management Console Shared Folders snap-in or using the ONTAP CLI. To create the SMB shares using the ONTAP CLI:

   a. If necessary, create the directory path structure for the share.

      The `vserver cifs share create` command checks the path specified in the -path option during share creation. If the specified path does not exist, the command fails.

   b. Create an SMB share associated with the specified SVM:

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

   c. Verify that the share was created:

```
vserver cifs share show -share-name share_name
```

      Refer to Create an SMB share for full details.

2. When creating the backend, you must configure the following to specify SMB volumes. For all FSx for ONTAP backend configuration options, refer to FSx for ONTAP configuration options and examples.

Parameter	Description	Example
`smbShare`	You can specify one of the following: the name of an SMB share created using the Microsoft Management Console or ONTAP CLI; a name to allow Trident to create the SMB share; or you can leave the parameter blank to prevent common share access to volumes. This parameter is optional for on-premises ONTAP. This parameter is required for Amazon FSx for ONTAP backends and cannot be blank.	`smb-share`
`nasType`	**Must set to `smb`.** If null, defaults to `nfs`.	`smb`
`securityStyle`	Security style for new volumes. **Must be set to `ntfs` or `mixed` for SMB volumes.**	`ntfs` or `mixed` for SMB volumes
`unixPermissions`	Mode for new volumes. **Must be left empty for SMB volumes.**	""

Enable secure SMB

Beginning with the 25.06 release, NetApp Trident supports secure provisioning of SMB volumes created using `ontap-nas` and `ontap-nas-economy` backends. When secure SMB is enabled, you can provide controlled access to SMB the shares for Active Directory (AD) users and user groups using Access Control Lists (ACLs).

Points to remember

- Importing `ontap-nas-economy` volumes is not supported.
- Only read-only clones are supported for `ontap-nas-economy` volumes.
- If Secure SMB is enabled, Trident will ignore the SMB share mentioned in the backend.
- Updating the PVC annotation, storage class annotation, and backend field does not update the SMB share ACL.
- The SMB share ACL specified in the annotation of the clone PVC will take precedence over those in the source PVC.
- Ensure that you provide valid AD users while enabling secure SMB. Invalid users will not be added to the ACL.
- If you provide the same AD user in the backend, storage class, and PVC with different permissions, the permission priority will be: PVC, storage class, and then backend.
- Secure SMB is supported for `ontap-nas` managed volume imports and not applicable to unmanaged volume imports.

Steps

1. Specify adAdminUser in TridentBackendConfig as shown in the following example:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.193.176.x
  svm: svm0
  useREST: true
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

2. Add the annotation in the storage class.

   Add the `trident.netapp.io/smbShareAdUser` annotation to the storage class to enable secure SMB without fail.
   The user value specified for the annotation `trident.netapp.io/smbShareAdUser` should be the same as the username specified in the `smbcreds` secret.
   You can choose one of the following for `smbShareAdUserPermission`: `full_control`, `change`, or `read`. The default permission is `full_control`.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

3. Create a PVC.

   The following example creates a PVC:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/snapshotDirectory: "true"
    trident.netapp.io/smbShareAccessControl: |
      read:
        - tridentADtest
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

ONTAP NAS configuration options and examples

Learn to create and use ONTAP NAS drivers with your Trident installation. This section provides backend configuration examples and details for mapping backends to StorageClasses.

Beginning with the 25.10 release, NetApp Trident supports NetApp AFX storage systems. NetApp AFX storage systems differ from other ONTAP-based systems (ASA, AFF, and FAS) in the implementation of their storage layer.

> Only the `ontap-nas` driver (with NFS protocol) is supported for NetApp AFX systems; SMB protocol is not supported.

In the Trident backend configuration, you need not specify that your system is an NetApp AFX storage system. When you select `ontap-nas` as the `storageDriverName`, Trident detects automatically the AFX storage system. Some backend configuration parameters are not applicable to AFX storage systems as noted in the table below.

Backend configuration options

See the following table for the backend configuration options:

Parameter	Description	Default
`version`		Always 1

Parameter	Description	Default
`storageDriverName`	Name of the storage driver For NetApp AFX systems, only `ontap-nas` is supported.	`ontap-nas`, `ontap-nas-economy`, or `ontap-nas-flexgroup`
`backendName`	Custom name or the storage backend	Driver name + "_" + dataLIF
`managementLIF`	IP address of a cluster or SVM management LIF A fully-qualified domain name (FQDN) can be specified. Can be set to use IPv6 addresses if Trident was installed using the IPv6 flag. IPv6 addresses must be defined in square brackets, such as `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]`. For seamless MetroCluster switchover, see the MetroCluster example.	"10.0.0.1", "[2001:1234:abcd::fefe]"
`dataLIF`	IP address of protocol LIF. NetApp recommends specifying `dataLIF`. If not provided, Trident fetches dataLIFs from the SVM. You can specify a fully-qualified domain name (FQDN) to be used for the NFS mount operations, allowing you to create a round-robin DNS to load-balance across multiple dataLIFs. Can be changed after initial setting. Refer to Update `dataLIF` after initial configuration. Can be set to use IPv6 addresses if Trident was installed using the IPv6 flag. IPv6 addresses must be defined in square brackets, such as `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]`. **Omit for Metrocluster.** See the MetroCluster example.	Specified address or derived from SVM, if not specified (not recommended)
`svm`	Storage virtual machine to use **Omit for Metrocluster.** See the MetroCluster example.	Derived if an SVM `managementLIF` is specified

Parameter	Description	Default
`autoExportPolicy`	Enable automatic export policy creation and updating [Boolean]. Using the `autoExportPolicy` and `autoExportCIDRs` options, Trident can manage export policies automatically.	false
`autoExportCIDRs`	List of CIDRs to filter Kubernetes' node IPs against when `autoExportPolicy` is enabled. Using the `autoExportPolicy` and `autoExportCIDRs` options, Trident can manage export policies automatically.	["0.0.0.0/0", "::/0"]`
`labels`	Set of arbitrary JSON-formatted labels to apply on volumes	""
`clientCertificate`	Base64-encoded value of client certificate. Used for certificate-based auth	""
`clientPrivateKey`	Base64-encoded value of client private key. Used for certificate-based auth	""
`trustedCACertificate`	Base64-encoded value of trusted CA certificate. Optional. Used for certificate-based auth	""
`username`	Username to connect to the cluster/SVM. Used for credential-based auth. For Active Directory authentication, see Authenticate Trident to a backend SVM using Active Directory credentials.	
`password`	Password to connect to the cluster/SVM. Used for credential-based auth. For Active Directory authentication, see Authenticate Trident to a backend SVM using Active Directory credentials.	
`storagePrefix`	Prefix used when provisioning new volumes in the SVM. Cannot be updated after you set it When using ontap-nas-economy and a storagePrefix that is 24 or more characters, the qtrees will not have the storage prefix embedded, though it will be in the volume name.	"trident"

Parameter	Description	Default
`aggregate`	Aggregate for provisioning (optional; if set, must be assigned to the SVM). For the `ontap-nas-flexgroup` driver, this option is ignored. If not assigned, any of the available aggregates can be used to provision a FlexGroup volume. When the aggregate is updated in SVM, it is updated in Trident automatically by polling SVM without having to restart the Trident Controller. When you have configured a specific aggregate in Trident to provision volumes, if the aggregate is renamed or moved out of the SVM, the backend will move to failed state in Trident while polling the SVM aggregate. You must either change the aggregate to one that is present on the SVM or remove it altogether to bring the backend back online. **Do not specify for AFX storage systems.**	""
`limitAggregateUsage`	Fail provisioning if usage is above this percentage. **Does not apply to Amazon FSx for ONTAP**. **Do not specify for AFX storage systems.**	"" (not enforced by default)
flexgroupAggregateList	List of aggregates for provisioning (optional; if set, must be assigned to the SVM). All aggregates assigned to the SVM are used to provision a FlexGroup volume. Supported for the **ontap-nas-flexgroup** storage driver. When the aggregate list is updated in SVM, the list is updated in Trident automatically by polling SVM without having to restart the Trident Controller. When you have configured a specific aggregate list in Trident to provision volumes, if the aggregate list is renamed or moved out of SVM, the backend will move to failed state in Trident while polling the SVM aggregate. You must either change the aggregate list to one that is present on the SVM or remove it altogether to bring the backend back online.	""

Parameter	Description	Default
`limitVolumeSize`	Fail provisioning if requested volume size is above this value.	"" (not enforced by default)
`debugTraceFlags`	Debug flags to use when troubleshooting. Example, {"api":false, "method":true} Do not use `debugTraceFlags` unless you are troubleshooting and require a detailed log dump.	null
`nasType`	Configure NFS or SMB volumes creation. Options are `nfs`, `smb` or null. Setting to null defaults to NFS volumes. **If specified, always set to `nfs` for AFX storage systems**.	`nfs`
`nfsMountOptions`	Comma-separated list of NFS mount options. The mount options for Kubernetes-persistent volumes are normally specified in storage classes, but if no mount options are specified in a storage class, Trident will fall back to using the mount options specified in the storage backend's configuration file. If no mount options are specified in the storage class or the configuration file, Trident will not set any mount options on an associated persistent volume.	""
`qtreesPerFlexvol`	Maximum Qtrees per FlexVol, must be in range [50, 300]	"200"
`smbShare`	You can specify one of the following: the name of an SMB share created using the Microsoft Management Console or ONTAP CLI; a name to allow Trident to create the SMB share; or you can leave the parameter blank to prevent common share access to volumes. This parameter is optional for on-premises ONTAP. This parameter is required for Amazon FSx for ONTAP backends and cannot be blank.	`smb-share`

Parameter	Description	Default
`useREST`	Boolean parameter to use ONTAP REST APIs. `useREST` When set to `true`, Trident uses ONTAP REST APIs to communicate with the backend; when set to `false`, Trident uses ONTAPI (ZAPI) calls to communicate with the backend. This feature requires ONTAP 9.11.1 and later. In addition, the ONTAP login role used must have access to the `ontapi` application. This is satisfied by the pre-defined `vsadmin` and `cluster-admin` roles. Beginning with the Trident 24.06 release and ONTAP 9.15.1 or later, `useREST` is set to `true` by default; change `useREST` to `false` to use ONTAPI (ZAPI) calls. **If specified, always set to `true` for AFX storage systems**.	`true` for ONTAP 9.15.1 or later, otherwise `false`.
`limitVolumePoolSize`	Maximum requestable FlexVol size when using Qtrees in ontap-nas-economy backend.	"" (not enforced by default)
`denyNewVolumePools`	Restricts `ontap-nas-economy` backends from creating new FlexVol volumes to contain their Qtrees. Only preexisting Flexvols are used for provisioning new PVs.	
`adAdminUser`	Active Directory admin user or user group with full access to SMB shares. Use this parameter to provide admin rights to the SMB share with full control.	

Backend configuration options for provisioning volumes

You can control default provisioning using these options in the `defaults` section of the configuration. For an example, see the configuration examples below.

Parameter	Description	Default
`spaceAllocation`	Space-allocation for Qtrees	"true"
`spaceReserve`	Space reservation mode; "none" (thin) or "volume" (thick)	"none"
`snapshotPolicy`	Snapshot policy to use	"none"
`qosPolicy`	QoS policy group to assign for volumes created. Choose one of qosPolicy or adaptiveQosPolicy per storage pool/backend	""

Parameter	Description	Default
`adaptiveQosPolicy`	Adaptive QoS policy group to assign for volumes created. Choose one of qosPolicy or adaptiveQosPolicy per storage pool/backend. Not supported by ontap-nas-economy.	""
`snapshotReserve`	Percentage of volume reserved for snapshots	"0" if `snapshotPolicy` is "none", otherwise ""
`splitOnClone`	Split a clone from its parent upon creation	"false"
`encryption`	Enable NetApp Volume Encryption (NVE) on the new volume; defaults to `false`. NVE must be licensed and enabled on the cluster to use this option. If NAE is enabled on the backend, any volume provisioned in Trident will be NAE enabled. For more information, refer to: How Trident works with NVE and NAE.	"false"
`tieringPolicy`	Tiering policy to use "none"	
`unixPermissions`	Mode for new volumes	"777" for NFS volumes; empty (not applicable) for SMB volumes
`snapshotDir`	Controls access to the `.snapshot` directory	"true" for NFSv4 "false" for NFSv3
`exportPolicy`	Export policy to use	"default"
`securityStyle`	Security style for new volumes. NFS supports `mixed` and `unix` security styles. SMB supports `mixed` and `ntfs` security styles.	NFS default is `unix`. SMB default is `ntfs`.
`nameTemplate`	Template to create custom volume names.	""

Using QoS policy groups with Trident requires ONTAP 9.8 or later. You should use a non-shared QoS policy group and ensure the policy group is applied to each constituent individually. A shared QoS policy group enforces the ceiling for the total throughput of all workloads.

Volume provisioning examples

Here's an example with defaults defined:

```
---
version: 1
storageDriverName: ontap-nas
backendName: customBackendName
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
labels:
  k8scluster: dev1
  backend: dev1-nasbackend
svm: trident_svm
username: cluster-admin
password: <password>
limitAggregateUsage: 80%
limitVolumeSize: 50Gi
nfsMountOptions: nfsvers=4
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: premium
  exportPolicy: myk8scluster
  snapshotPolicy: default
  snapshotReserve: "10"
```

For `ontap-nas` and `ontap-nas-flexgroups`, Trident now uses a new calculation to ensure that the FlexVol is sized correctly with the snapshotReserve percentage and PVC. When the user requests a PVC, Trident creates the original FlexVol with more space by using the new calculation. This calculation ensures that the user receives the writable space they requested for in the PVC, and not less space than what they requested. Before v21.07, when the user requests a PVC (for example, 5 GiB), with the snapshotReserve to 50 percent, they get only 2.5 GiB of writeable space. This is because what the user requested for is the whole volume and `snapshotReserve` is a percentage of that. With Trident 21.07, what the user requests for is the writeable space and Trident defines the `snapshotReserve` number as the percentage of the whole volume. This does not apply to `ontap-nas-economy`. See the following example to see how this works:

The calculation is as follows:

```
Total volume size = <PVC requested size> / (1 - (<snapshotReserve
percentage> / 100))
```

For snapshotReserve = 50%, and PVC request = 5 GiB, the total volume size is 5/.5 = 10 GiB and the available size is 5 GiB, which is what the user requested in the PVC request. The `volume show` command should show results similar to this example:

Existing backends from previous installs will provision volumes as explained above when upgrading Trident. For volumes that you created before upgrading, you should resize their volumes for the change to be observed. For example, a 2 GiB PVC with `snapshotReserve=50` earlier resulted in a volume that provides 1 GiB of writable space. Resizing the volume to 3 GiB, for example, provides the application with 3 GiB of writable space on a 6 GiB volume.

Minimal configuration examples

The following examples show basic configurations that leave most parameters to default. This is the easiest way to define a backend.

> If you are using Amazon FSx on NetApp ONTAP with Trident, the recommendation is to specify DNS names for LIFs instead of IP addresses.

ONTAP NAS economy example

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

ONTAP NAS Flexgroup example

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

MetroCluster example

You can configure the backend to avoid having to manually update the backend definition after switchover and switchback during SVM replication and recovery.

For seamless switchover and switchback, specify the SVM using `managementLIF` and omit the `dataLIF` and `svm` parameters. For example:

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

SMB volumes example

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
nasType: smb
securityStyle: ntfs
unixPermissions: ""
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Certificate-based authentication example

This is a minimal backend configuration example. `clientCertificate`, `clientPrivateKey`, and `trustedCACertificate` (optional, if using trusted CA) are populated in `backend.json` and take the base64-encoded values of the client certificate, private key, and trusted CA certificate, respectively.

```
---
version: 1
backendName: DefaultNASBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.15
svm: nfs_svm
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Auto export policy example

This example shows you how you can instruct Trident to use dynamic export policies to create and manage the export policy automatically. This works the same for the `ontap-nas-economy` and `ontap-nas-flexgroup` drivers.

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-nasbackend
autoExportPolicy: true
autoExportCIDRs:
- 10.0.0.0/24
username: admin
password: password
nfsMountOptions: nfsvers=4
```

IPv6 addresses example

This example shows `managementLIF` using an IPv6 address.

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas_ipv6_backend
managementLIF: "[5c5d:5edf:8f:7657:bef8:109b:1b41:d491]"
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-ontap-ipv6
svm: nas_ipv6_svm
username: vsadmin
password: password
```

Amazon FSx for ONTAP using SMB volumes example

The `smbShare` parameter is required for FSx for ONTAP using SMB volumes.

```
---
version: 1
backendName: SMBBackend
storageDriverName: ontap-nas
managementLIF: example.mgmt.fqdn.aws.com
nasType: smb
dataLIF: 10.0.0.15
svm: nfs_svm
smbShare: smb-share
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Backend configuration example with nameTemplate

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap-nas-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

Examples of backends with virtual pools

In the sample backend definition files shown below, specific defaults are set for all storage pools, such as `spaceReserve` at none, `spaceAllocation` at false, and `encryption` at false. The virtual pools are defined in the storage section.

Trident sets provisioning labels in the "Comments" field. Comments are set on FlexVol for `ontap-nas` or FlexGroup for `ontap-nas-flexgroup`. Trident copies all labels present on a virtual pool to the storage volume at provisioning. For convenience, storage administrators can define labels per virtual pool and group volumes by label.

In these examples, some of the storage pools set their own `spaceReserve`, `spaceAllocation`, and `encryption` values, and some pools override the default values.

ONTAP NAS example

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
svm: svm_nfs
username: admin
password: <password>
nfsMountOptions: nfsvers=4
defaults:
  spaceReserve: none
  encryption: "false"
  qosPolicy: standard
labels:
  store: nas_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      app: msoffice
      cost: "100"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
      adaptiveQosPolicy: adaptive-premium
  - labels:
      app: slack
      cost: "75"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      department: legal
      creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      app: wordpress
```

```
      cost: "50"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      app: mysqldb
      cost: "25"
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: "false"
      unixPermissions: "0775"
```

ONTAP NAS FlexGroup example

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: flexgroup_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      protection: gold
      creditpoints: "50000"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: gold
      creditpoints: "30000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: silver
      creditpoints: "20000"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      protection: bronze
      creditpoints: "10000"
    zone: us_east_1d
    defaults:
```

```
      spaceReserve: volume
      encryption: "false"
      unixPermissions: "0775"
```

ONTAP NAS economy example

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: nas_economy_store
region: us_east_1
storage:
  - labels:
      department: finance
      creditpoints: "6000"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: bronze
      creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      department: engineering
      creditpoints: "3000"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      department: humanresource
      creditpoints: "2000"
    zone: us_east_1d
    defaults:
      spaceReserve: volume
```

```
      encryption: "false"
      unixPermissions: "0775"
```

Map backends to StorageClasses

The following StorageClass definitions refer to Examples of backends with virtual pools. Using the `parameters.selector` field, each StorageClass calls out which virtual pools can be used to host a volume. The volume will have the aspects defined in the chosen virtual pool.

- The `protection-gold` StorageClass will map to the first and second virtual pool in the `ontap-nas-flexgroup` backend. These are the only pools offering gold level protection.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- The `protection-not-gold` StorageClass will map to the third and fourth virtual pool in the `ontap-nas-flexgroup` backend. These are the only pools offering protection level other than gold.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- The `app-mysqldb` StorageClass will map to the fourth virtual pool in the `ontap-nas` backend. This is the only pool offering storage pool configuration for mysqldb type app.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- TThe `protection-silver-creditpoints-20k` StorageClass will map to the third virtual pool in the `ontap-nas-flexgroup` backend. This is the only pool offering silver-level protection and 20000 creditpoints.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- The `creditpoints-5k` StorageClass will map to the third virtual pool in the `ontap-nas` backend and the second virtual pool in the `ontap-nas-economy` backend. These are the only pool offerings with 5000 creditpoints.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

Trident will decide which virtual pool is selected and ensures the storage requirement is met.

Update `dataLIF` after initial configuration

You can change the dataLIF after initial configuration by running the following command to provide the new backend JSON file with updated dataLIF.

```
tridentctl update backend <backend-name> -f <path-to-backend-json-file-
with-updated-dataLIF>
```

If PVCs are attached to one or multiple pods, you must bring down all corresponding pods and then bring them back up in order to for the new dataLIF to take effect.

Secure SMB examples

Backend configuration with ontap-nas driver

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Backend configuration with ontap-nas-economy driver

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas-economy
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Backend configuration with storage pool

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm0
  useREST: false
  storage:
  - labels:
      app: msoffice
    defaults:
      adAdminUser: tridentADuser
  nasType: smb
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Storage class example with ontap-nas driver

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADtest
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Ensure that you add `annotations` to enable secure SMB. Secure SMB does not work without the annotations, irrespective of configurations set in the Backend or PVC.

Storage class example with ontap-nas-economy driver

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser3
parameters:
  backendType: ontap-nas-economy
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

PVC example with a single AD user

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      change:
        - tridentADtest
      read:
        - tridentADuser
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

PVC example with multiple AD users

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-test-pvc
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      full_control:
        - tridentTestuser
        - tridentuser
        - tridentTestuser1
        - tridentuser1
      change:
        - tridentADuser
        - tridentADuser1
        - tridentADuser4
        - tridentTestuser2
      read:
        - tridentTestuser2
        - tridentTestuser3
        - tridentADuser2
        - tridentADuser3
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Copyright information

Trademark information