



## **Get started**

### **Astra Data Store**

NetApp  
May 31, 2022

This PDF was generated from <https://docs.netapp.com/us-en/astra-data-store-2112/get-started/requirements.html> on May 31, 2022. Always check docs.netapp.com for the latest.

# Table of Contents

- Get started ..... 1
  - Astra Data Store preview requirements ..... 1
  - Quick start for Astra Data Store preview ..... 4
  - Install Astra Data Store preview ..... 5
  - Set up Astra Data Store preview components ..... 22
  - Astra Data Store preview limits ..... 31
  - Frequently asked questions for Astra Data Store preview ..... 32

# Get started

## Astra Data Store preview requirements

Get started by verifying that your environment meets Astra Data Store preview requirements.

Astra Data Store preview supports both bare-metal and VM-based deployments. The Astra Data Store preview cluster can run on a Kubernetes cluster with four or more worker nodes. Astra Data Store preview software can co-exist with other applications running on the same Kubernetes cluster.

Astra Data Store preview supports provisioning only of persistent volumes for Kubernetes workloads using the Astra Trident CSI driver. VM workloads will be supported in a future release of Astra Data Store.



If you plan to manage your Astra Data Store preview cluster from Astra Control Center, make sure your Astra Data Store preview cluster meets the [requirements for clusters that will be managed by Astra Control Center](#) in addition to the requirements outlined here.

### Kubernetes worker node resource requirements

These are the resource requirements required to be assigned to the Astra Data Store preview software on each worker node in the Kubernetes cluster:

Resource	Minimum	Maximum
Number of data drives	<ul style="list-style-type: none"><li>• 3 (with a separate cache device present)</li><li>• 4 (if there is no cache device present)</li></ul>	14
Data drive size	100GiB	4TiB
Number of optional cache devices	1 (8GiB or greater in size)	N/A
Number of vCPUs	10	10
RAM	35GiB	35GiB



For the best write performance, you should configure a dedicated high-endurance, low-latency, low-capacity cache device.

Each worker node has the following additional requirements:

- 100GiB or greater free space on host disk (boot) for Astra Data Store preview log files to be stored.
- At least one 10GbE or faster network interface for cluster, data, and management traffic. Optionally, an additional 1GbE or faster interface can be used to separate out management traffic.

### Hardware and software requirements

Astra Data Store preview software is validated on the following hardware platforms, software, and storage configuration. Visit [NetApp community support](#) if your Kubernetes cluster configuration is different.

#### Hardware platforms

- HPE DL360
- HPE DL380
- Dell R640
- Dell R740

## Storage

Astra Data Store preview has been validated with the following drive types:

- **Bare-metal deployments:** Astra Data Store preview installed on a Kubernetes cluster directly on a Linux cluster without any hypervisor
  - SATA or NVMe TLC SSDs
- **VM-based deployments:** Astra Data Store preview installed on a Kubernetes cluster on Linux VMs hosted on an ESXi cluster
  - SATA, SAS, or NVMe TLC SSD-based datastores
  - Drives presented as virtual disks or passthrough drives



If your host uses SSDs behind a hardware RAID controller, configure the hardware RAID controller to use "passthrough" mode.



Each drive should have a unique serial number. Add the attribute `disk.enableuuid=TRUE` in the virtual machine advanced settings during VM creation.

## Software

- Hypervisor: Astra Data Store preview is validated with VMware-based VM deployments with ESXi 7.0. KVM-based deployments are not supported by Astra Data Store preview.
- Astra Data Store preview has been validated on the following host operating systems:
  - Red Hat Enterprise Linux 8.4
  - Red Hat Enterprise Linux 8.2
  - Red Hat Enterprise Linux 7.9
  - Red Hat Enterprise Linux CoreOS (RHCOS)
  - CentOS 8
  - Ubuntu 20.04
- Astra Data Store preview has been validated with the following Kubernetes distributions:
  - Red Hat OpenShift 4.7
  - Google Anthos 1.7
  - Kubernetes 1.21
  - Kubernetes 1.20



Astra Data Store preview requires Astra Trident version 21.10.1 for storage provisioning and orchestration. See the [Astra Trident installation instructions](#).

## Networking requirements

Astra Data Store preview requires one IP address per cluster for MVIP. It needs to be an unused or unconfigured IP address in same subnet as MIP. The Astra Data Store preview management interface should be same as the Kubernetes node's management interface.

In addition, each node can be configured as described in the following table:



The following abbreviations are used in this table:

MIP: Management IP address

CIP: Cluster IP address

MVIP: Management virtual IP address

Configuration	IP addresses needed
One network interface per node	<ul style="list-style-type: none"><li>• Two (2) per node:<ul style="list-style-type: none"><li>◦ MIP/CIP: One (1) pre-configured IP address on management interface per node</li><li>◦ Data IP: One (1) unused or unconfigured IP address per node in same subnet as MIP</li></ul></li></ul>
Two network interfaces per node	<ul style="list-style-type: none"><li>• Three per node:<ul style="list-style-type: none"><li>◦ MIP: One (1) pre-configured IP address on management interface per node</li><li>◦ CIP: One (1) pre-configured IP address on data interface per node in a different subnet from MIP</li><li>◦ Data IP: One (1) unused or unconfigured IP address per node in same subnet as CIP</li></ul></li></ul>



You should omit the data network gateway field in the cluster Custom Resource (CR) file, `astradscluster.yaml`, for both of these configurations. The existing routing configuration on each node accommodates all of the addresses.



No VLAN tags are used in these configurations.

## Astra Trident

Astra Data Store preview requires the application Kubernetes clusters to be running Astra Trident 21.10.1. Astra Data Store preview can be configured as a [storage backend](#) with Astra Trident to provision persistent volumes.

## CNI configuration

Astra Data Store preview has been validated with the following CNIs:

- Calico and Weave Net CNIs for vanilla Kubernetes clusters
- OpenShift SDN for Red Hat OpenShift Container Platform (OCP)

- Cilium for Google Anthos

These CNIs require the host firewall (firewalld) to be disabled.

## Persistent volume sharing requirements

Each Astra Data Store preview cluster supports using persistent volumes to address the storage needs of any apps installed on that cluster. Kubernetes apps access files using persistent volumes shared over NFSv4.1, which requires the AUTH\_SYS authentication method.

## Licensing

Astra Data Store preview requires an Astra Data Store preview license for full functionality. [Sign up here](#) to obtain the Astra Data Store preview license. Instructions to download the license will be sent to you after you sign up.

## AutoSupport configuration

Astra Data Store preview requires AutoSupport to be enabled and have connectivity to the AutoSupport backend. This may be through direct internet access or proxy configuration.

The [periodic settings that are used for sending mandatory telemetry AutoSupport bundles](#) should not be changed. If you disable the sending of periodic AutoSupport bundles, the cluster will be locked down and new volumes cannot be created until periodic settings are enabled again.

## What's next

View the [quick start](#) overview.

## For more information

[Astra Data Store preview limits](#)

# Quick start for Astra Data Store preview

This page provides a high-level overview of the steps needed to get started with Astra Data Store preview. The links within each step take you to a page that provides more details.

Try it out! If you want to try Astra Data Store preview, you can use a 90-day preview license.

[Sign up here](#) to obtain the Astra Data Store preview license.

1

### Review Kubernetes cluster requirements

- Clusters must be running in a healthy state, with at least four or more worker nodes.
- Each of the Kubernetes worker nodes that are part of the Astra Data Store preview deployment should have SSDs of the same interface type (SATA, SAS or NVMe) and the same number of drives that are assigned to the Astra Data Store preview cluster.
- Each SSD should have a unique serial number.

Learn more about [Astra Data Store preview requirements](#).

## 2

### Download and install Astra Data Store preview

- Download Astra Data Store preview from the [NetApp Support Site](#).
- Install Astra Data Store preview in your local environment.
- Apply the Astra Data Store preview license.
- Install the Astra Data Store preview cluster.
- Configure Astra Data Store preview monitoring.
- If you use Red Hat OpenShift, install Astra Data Store preview on Red Hat OpenShift Container Platform (OCP).

Learn more about [installing Astra Data Store preview](#).

## 3

### Complete some initial setup tasks

- Install Astra Trident.
- Install Kubernetes snapshot custom resource definitions (CRDs) and controller.
- Set up Astra Data Store preview as a storage backend.
- Create a default Astra Data Store preview storage class.

Learn more about the [initial setup process](#).

## 4

### Use Astra Data Store preview

After you finish setting up Astra Data Store preview, here's what you might do next:

- Use kubectl commands and kubectl astrads extension to manage the cluster, including tasks such as place a node in maintenance mode, replace a drive, or replace a node. Learn more about [how to use kubectl commands with Astra Data Store preview](#).
- Configure monitoring endpoints. Learn more about [configuring monitoring endpoints](#).

## 5

### Continue from this Quick Start

[Install Astra Data Store preview](#).

## Install Astra Data Store preview

To install Astra Data Store preview, download the installation bundle from the [NetApp Support Site](#) and complete the installation steps described in this procedure.

# Astra Data Store deployment



## What you'll need

- [Before you begin installation, prepare your environment for Astra Data Store preview deployment.](#)
- Access to the [NetApp Support Site](#). [Register](#) for a preview if you don't already have a full-access NetApp Support Site account.
- A [NetApp license file \(NLF\)](#) for Astra Data Store preview. Instructions to download the license will be sent to you after you [sign up](#).
- An active kubeconfig with cluster admin rights for the active context.
- An understanding of [the roles and privileges](#) used by Astra Data Store preview.
- Internet connectivity. Astra Data Store preview does not support air-gapped environments. Internet connectivity is needed to reach support.netapp.com either directly or via a proxy.

## About this task

The Astra Data Store preview installation process guides you through the following high-level steps.

- [Download the Astra Data Store preview bundle and extract the images](#)
- [Copy the binary and push images to your local registry](#)
- [OpenShift procedure](#)
- [Install the Astra Data Store preview operator](#)
- [Deploy the Astra Data Store preview version YAML](#)
- [Apply the Astra Data Store preview license](#)
- [Install the Astra Data Store preview cluster](#)
- [Understand deployment-related events](#)
- [Configure Astra Data Store preview monitoring](#)

If you want to enable Astra Data Store preview to work with image registries with secrets, see [this KB](#).



## Download the Astra Data Store preview bundle and extract the images

1. Log in to the [NetApp Support Site](#) and download the Astra Data Store preview bundle (2021.12.01\_ads.tar).
2. (Optional) Use the following command to verify the signature of the bundle:

```
openssl dgst -sha256 -verify 2021.12_ads.pub -signature 2021.12_ads.sig  
2021.12.01_ads.tar
```

3. Extract the images:

```
tar -xvf 2021.12.01_ads.tar
```

## Copy the binary and push images to your local registry

1. Copy the kubectl-astrads binary from the directory you used to extract images to the standard path where k8s kubectl binaries are installed; for example, /usr/bin/. kubectl-astrads is a custom kubectl extension that installs and manages Astra Data Store preview clusters.

```
cp -p ./bin/kubectl-astrads /usr/bin/.
```

2. Add the files in the Astra Data Store preview image directory to your local registry.



See a sample script for the automatic loading of images below.

- a. Log in to your registry:

```
docker login [your_registry_path]
```

- b. Set an environment variable to the registry path where you want to push the Astra Data Store preview images; for example, repo.company.com.

```
export REGISTRY=repo.company.com/astrads
```

- c. Run the script to load the images into Docker, tag the images, and push the images to your local registry:

```
for astraImageFile in $(ls images/*.tar) ; do
    astraImage=$(docker load --input ${astraImageFile} | sed 's~Loaded
image: ~~')
    astraImageShort=`echo $astraImage | sed 's~.*/~~'`
    docker tag ${astraImage} ${REGISTRY}/${astraImageShort}
    docker push ${REGISTRY}/${astraImageShort}
done
sed -i 's~\[YOUR_REGISTRY\]~'${REGISTRY}'~' ./manifests/*.yaml
```

## OpenShift procedure

The following procedure is needed only for deployment on the Red Hat OpenShift Container Platform (OCP). This procedure can be skipped for deployment on non-OCP Kubernetes clusters.

## Details

### Create a namespace to deploy Astra Data Store preview

Create a namespace `astrads-system` in which all Astra Data Store preview components will be installed.

The following steps are needed only for deployment on the Red Hat OpenShift Container Platform (OCP).

1. Create the namespace:

```
kubectl create -f ads_namespace.yaml
```

Sample: `ads_namespace.yaml`

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    control-plane: operator
  name: astrads-system
```

### Create a custom SCC

OpenShift uses security context constraints (SCC) that control the actions that a pod can perform. By default, the execution of any container will be granted the restricted SCC and only the capabilities defined by that SCC.

Restricted SCC does not provide permissions required by Astra Data Store preview cluster pods. Use this procedure to provide the required privileges (listed in the sample) to Astra Data Store preview.

Assign a custom SCC to the default service account for the Astra Data Store preview namespace.

The following steps are needed only for deployment on the Red Hat OpenShift Container Platform (OCP).

1. Create a custom SCC:

```
kubectl create -f ads_privileged_scc.yaml
```

Sample: `ads_privileged_scc.yaml`

```

allowHostDirVolumePlugin: true
allowHostIPC: true
allowHostNetwork: true
allowHostPID: true
allowHostPorts: true
allowPrivilegeEscalation: true
allowPrivilegedContainer: true
allowedCapabilities:
- '*'
allowedUnsafeSysctls:
- '*'
apiVersion: security.openshift.io/v1
defaultAddCapabilities: null
fsGroup:
  type: RunAsAny
groups: []
kind: SecurityContextConstraints
metadata:
  annotations:
    kubernetes.io/description: 'ADS privileged. Grant with caution.'
    release.openshift.io/create-only: "true"
  name: ads-privileged
priority: null
readOnlyRootFilesystem: false
requiredDropCapabilities: null
runAsUser:
  type: RunAsAny
seLinuxContext:
  type: RunAsAny
seccompProfiles:
- '*'
supplementalGroups:
  type: RunAsAny
users:
- system:serviceaccount:astrads-system:default
volumes:
- '*'

```

2. Display the newly added SCC using the `oc get scc` command:

```
# oc get scc/ads-privileged
NAME          PRIV  CAPS   SELINUX  RUNASUSER  FSGROUP
SUPGROUP     PRIORITY  READONLYROOTFS  VOLUMES
ads-privileged  true    ["*"]   RunAsAny  RunAsAny  RunAsAny
RunAsAny    <no value>  false           ["*"]
#
```

### Create the roles and role bindings

Create the required roles and role bindings to be used by the default service account for Astra Data Store preview.

The following yaml definition assigns various roles (via rolebindings) needed by the Astra Data Store preview resources in the `astrads.netapp.io` API group.

The following steps are needed only for deployment on the Red Hat OpenShift Container Platform (OCP).

1. Create the defined roles and role binding:

```
kubectl create -f oc_role_bindings.yaml
```

Sample: `oc_role_bindings.yaml`

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privcrole
rules:
- apiGroups:
  - security.openshift.io
  resourceNames:
  - ads-privileged
  resources:
  - securitycontextconstraints
  verbs:
  - use
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: default-scc-rolebinding
  namespace: astrads-system
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
```

```

    name: privcrole
  subjects:
  - kind: ServiceAccount
    name: default
    namespace: astrads-system
  ---

apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: ownerref
  namespace: astrads-system
rules:
- apiGroups:
  - astrads.netapp.io
  resources:
  - '*/finalizers'
  verbs:
  - update
  ---

apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: or-rb
  namespace: astrads-system
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: ownerref
subjects:
- kind: ServiceAccount
  name: default
  namespace: astrads-system

```

## Install the Astra Data Store preview operator

1. List the Astra Data Store preview manifests:

```
ls manifests/*.yaml
```

Response:

```
manifests/astradscluster.yaml
manifests/astradsoperator.yaml
manifests/astradsversion.yaml
manifests/monitoring_operator.yaml
```

## 2. Deploy the operator using kubectl apply:

```
kubectl apply -f ./manifests/astradsoperator.yaml
```

Response:



The namespace response might differ depending on whether you performed the standard installation or the OCP installation.

```
namespace/astrads-system created
customresourcedefinition.apiextensions.k8s.io/astradsautosupports.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradscLOUDsnapshots.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsclusters.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsdeployments.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsexportpolicies.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsfaileddrives.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradslICENSES.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsnfsOPTIONS.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsnodeinfoes.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsqOSPolicies.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsvolumeFILES.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsvolumes.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsvolumesnapshots.astrads.netapp.io created
role.rbac.authorization.k8s.io/astrads-leader-election-role created
clusterrole.rbac.authorization.k8s.io/astrads-astradscLOUDsnapshot-editor-role created
```

```
clusterrole.rbac.authorization.k8s.io/astrads-astradscloudsnapshot-  
viewer-role created  
clusterrole.rbac.authorization.k8s.io/astrads-astradscluster-editor-role  
created  
clusterrole.rbac.authorization.k8s.io/astrads-astradscluster-viewer-role  
created  
clusterrole.rbac.authorization.k8s.io/astrads-astradslicense-editor-role  
created  
clusterrole.rbac.authorization.k8s.io/astrads-astradslicense-viewer-role  
created  
clusterrole.rbac.authorization.k8s.io/astrads-astradsvolume-editor-role  
created  
clusterrole.rbac.authorization.k8s.io/astrads-astradsvolume-viewer-role  
created  
clusterrole.rbac.authorization.k8s.io/astrads-autosupport-editor-role  
created  
clusterrole.rbac.authorization.k8s.io/astrads-autosupport-viewer-role  
created  
clusterrole.rbac.authorization.k8s.io/astrads-manager-role created  
clusterrole.rbac.authorization.k8s.io/astrads-metrics-reader created  
clusterrole.rbac.authorization.k8s.io/astrads-netappexportpolicy-editor-  
role created  
clusterrole.rbac.authorization.k8s.io/astrads-netappexportpolicy-viewer-  
role created  
clusterrole.rbac.authorization.k8s.io/astrads-netappsdsdeployment-  
editor-role created  
clusterrole.rbac.authorization.k8s.io/astrads-netappsdsdeployment-  
viewer-role created  
clusterrole.rbac.authorization.k8s.io/astrads-netappsdsnfsoption-editor-  
role created  
clusterrole.rbac.authorization.k8s.io/astrads-netappsdsnfsoption-viewer-  
role created  
clusterrole.rbac.authorization.k8s.io/astrads-netappsdsnodeinfo-editor-  
role created  
clusterrole.rbac.authorization.k8s.io/astrads-netappsdsnodeinfo-viewer-  
role created  
clusterrole.rbac.authorization.k8s.io/astrads-proxy-role created  
rolebinding.rbac.authorization.k8s.io/astrads-leader-election-  
rolebinding created  
clusterrolebinding.rbac.authorization.k8s.io/astrads-manager-rolebinding  
created  
clusterrolebinding.rbac.authorization.k8s.io/astrads-proxy-rolebinding  
created  
configmap/astrads-autosupport-cm created  
configmap/astrads-firetap-cm created  
configmap/astrads-fluent-bit-cm created
```



```
configmap/astrads-kevents-asup created
configmap/astrads-metrics-cm created
service/astrads-operator-metrics-service created
deployment.apps/astrads-operator created
```

3. Verify that the Astra Data Store operator pod has started and is running:

```
kubectl get pods -n astrads-system
```

Response:

NAME	READY	STATUS	RESTARTS	AGE
astrads-operator-5ffb94fbf-7ln4h	1/1	Running	0	17m

## Deploy the Astra Data Store preview version YAML

1. Deploy using kubectl apply:

```
kubectl apply -f ./manifests/astradsversion.yaml
```

2. Verify that the pods are running:

```
kubectl get pods -n astrads-system
```

Response:

NAME	READY	STATUS	RESTARTS
AGE			
astrads-cluster-controller-7f6f884645-xxf2n	1/1	Running	0
117s			
astrads-ds-nodeinfo-astradsversion-2jqnk	1/1	Running	0
2m7s			
astrads-ds-nodeinfo-astradsversion-dbk7v	1/1	Running	0
2m7s			
astrads-ds-nodeinfo-astradsversion-rn9tt	1/1	Running	0
2m7s			
astrads-ds-nodeinfo-astradsversion-vsmhv	1/1	Running	0
2m7s			
astrads-license-controller-fb8fd56bc-bxq7j	1/1	Running	0
2m2s			
astrads-operator-5ffb94fbf-7ln4h	1/1	Running	0
2m10s			

## Apply the Astra Data Store preview license

1. Apply the NetApp License File (NLF) that you obtained when you signed up for the preview. Before you run the command, enter the name of the cluster (<Astra-Data-Store-cluster-name>) that you are [going to deploy](#) or have already deployed and the path to the license file (<file\_path/file.txt>):

```
kubectl astrads license add --license-file-path <file_path/file.txt>
--ads-cluster-name <Astra-Data-Store-cluster-name> -n astrads-system
```

2. Verify that the license has been added:

```
kubectl astrads license list
```

Response:

NAME	ADSCUSTER	VALID	PRODUCT
EVALUATION	ENDDATE	VALIDATED	
p100000006	astrads-example-cluster	true	Astra Data Store Preview
true	2022-01-23	2021-11-04T14:38:54Z	

## Install the Astra Data Store preview cluster

1. Open the YAML file:

```
vim ./manifests/astradscluster.yaml
```

2. Edit the following values in the YAML file.



A simplified example of the YAML file follows these steps.

- a. (Required) **Metadata:** In `metadata`, change the `name` string to the name of your cluster. This must be the same cluster name you use when you [apply the license](#).
- b. (Required) **Spec:** Change the following required values in `spec`:
  - Change the `mvip` string to the IP address of a floating management IP that is routable from any worker node in the cluster.
  - In `adsDataNetworks`, add a comma-separated list of floating IP addresses (`addresses`) that are routable from any host where you intend to mount a NetApp volume. Use one floating IP address per node. There should be at least as many data network IP addresses as there are Astra Data Store preview nodes. For Astra Data Store preview, this means at least 4 addresses, or 5 if you plan on expanding the cluster to 5 nodes later.
  - In `adsDataNetworks`, specify the netmask used by the data network.
  - In `adsNetworkInterfaces`, replace the `<mgmt_interface_name>` and `<cluster_and_storage_interface_name>` values with the network interface names you want to use for management, cluster, and storage. If no names are specified, the node's primary interface will be used for management, cluster, and storage networking.



Cluster and storage networks need to be on the same interface. The Astra Data Store preview management interface should be same as the Kubernetes node's management interface.

- c. (Optional) **monitoringConfig:** If you want to configure a [monitoring operator](#) (optional if you are not using Astra Control Center for monitoring), remove the commenting from the section, add the namespace in which the agent CR (monitoring operator resource) is applied (default is `netapp-monitoring`), and add the repo path for your registry (`your_registry_path`) that you used in previous steps.
- d. (Optional) **autoSupportConfig:** Retain the [AutoSupport](#) default values unless you need to configure a proxy:
  - For `proxyURL`, set the URL of the proxy with the port that will be used for AutoSupport bundle transfer.



Most comments have been removed from the YAML sample below.

```
apiVersion: astrads.netapp.io/v1alpha1
kind: AstraDSCluster
metadata:
  name: astrads-cluster-name
  namespace: astrads-system
spec:
  adsNodeConfig:
```

```

    cpu: 9
    memory: 34
    adsNodeCount: 4
    mvip: ""
    adsDataNetworks:
      - addresses: ""
        netmask:
# Specify the network interface names to use for management, cluster
and storage networks.
# If none are specified, the node's primary interface will be used for
management, cluster and storage networking.
# To move the cluster and storage networks to a different interface
than management, specify all three interfaces to use here.
# NOTE: The cluster and storage networks need to be on the same
interface.
    adsNetworkInterfaces:
      managementInterface: "<mgmt_interface_name>"
      clusterInterface: "<cluster_and_storage_interface_name>"
      storageInterface: "<cluster_and_storage_interface_name>"
# [Optional] Provide a k8s label key that defines which protection
domain a node belongs to.
# adsProtectionDomainKey: ""
# [Optional] Provide a monitoring config to be used to setup/configure
a monitoring agent.
# monitoringConfig:
#   namespace: "netapp-monitoring"
#   repo: "[YOUR_REGISTRY]"
autoSupportConfig:
  autoUpload: true
  enabled: true
  coredumpUpload: false
  historyRetentionCount: 25
  destinationURL: "https://support.netapp.com/put/AsupPut"
# ProxyURL defines the URL of the proxy with port to be used for
AutoSupport bundle transfer
# proxyURL:
periodic:
  - schedule: "0 0 * * *"
    periodicconfig:
      - component:
          name: storage
          event: dailyMonitoring
          userMessage: Daily Monitoring Storage AutoSupport bundle
          nodes: all
      - component:
          name: controlplane

```

```
event: daily
userMessage: Daily Control Plane AutoSupport bundle
```

3. Deploy the cluster using `kubectl apply`:

```
kubectl apply -f ./manifests/astradscluster.yaml
```

4. (OCP only) If SELinux is enabled, re-label the `selinux` context for the following directories on the nodes in the Astra Data Store preview cluster.

```
sudo chcon -R -t container_file_t
/var/opt/netapp/firetap/rootfs/var/asup/notification/firetap/
```

```
sudo chcon -R -t container_file_t /var/netapp/firetap/firegen/persist/
```



This step is needed because `selinux` prevents these directories from being writable, causing the support pods to enter a `CrashLoopBackoff` state. This step needs to be performed on all the nodes in the Astra Data Store preview cluster.

5. Wait a few minutes for the cluster creation operation to complete and then verify that the pods are running:

```
kubectl get pods -n astrads-system
```

Sample response:

NAME	READY	STATUS	RESTARTS	AGE
astrads-cluster-controller-7c67cc7f7b-2jww2	1/1	Running	0	7h31m
astrads-deployment-support-788b859c65-2qjkn	3/3	Running	19	12d
astrads-ds-astrads-cluster-lab0dbc-j9jzc	1/1	Running	0	5d2h
astrads-ds-astrads-cluster-lab0dbc-k9wp8	1/1	Running	0	5d1h
astrads-ds-astrads-cluster-lab0dbc-pwk42	1/1	Running	0	5d2h
astrads-ds-astrads-cluster-lab0dbc-qhvc6	1/1	Running	0	8h
astrads-ds-nodeinfo-astradsversion-gcmj8	1/1	Running	1	12d
astrads-ds-nodeinfo-astradsversion-j826x	1/1	Running	3	12d
astrads-ds-nodeinfo-astradsversion-vdthh	1/1	Running	3	12d
astrads-ds-nodeinfo-astradsversion-xwgsf	1/1	Running	0	12d
astrads-ds-support-828vw	2/2	Running	2	5d2h
astrads-ds-support-cfzts	2/2	Running	0	8h
astrads-ds-support-nzkkrr	2/2	Running	15	7h49m
astrads-ds-support-xxbnp	2/2	Running	1	5d2h
astrads-license-controller-86c69f76bb-s6fb7	1/1	Running	0	8h
astrads-operator-79ff8fbb6d-vpz9m	1/1	Running	0	8h

## 6. Verify the cluster deployment progress:

```
kubectl get astradscluster -n astrads-system
```

Sample response:

NAME	STATUS	VERSION	SERIAL NUMBER	MVIP
astrads-example-cluster	created	2021.10.0	p100000006	
10.x.x.x	10m			

## Understand deployment-related events

During cluster deployment, the operation status should change from blank to in progress to created. Cluster deployment will last approximately 8 to 10 minutes. To monitor cluster events during deployment, you can run either of the following commands:

```
kubectl get events --field-selector involvedObject.kind=AstraDScluster -n astrads-system
```

```
kubectl describe astradscluster <cluster name> -n astrads-system
```

The following are key events during deployment:

Event message	Meaning
Successfully selected 4 control plane nodes to join the ADS cluster	The Astra Data Store preview operator identified enough nodes with CPU, memory, storage, and networking to create an Astra Data Store preview cluster.
ADS cluster create in progress	The Astra Data Store preview cluster controller has started the cluster create operation.
ADS cluster created	The cluster was created successfully.

If the cluster's status doesn't change to `in progress`, check the operator logs for more details on node selection:

```
kubectl logs -n astrads-system <astrads operator pod name>
```

If the cluster's status is stuck at `in progress`, check the cluster controller's logs:

```
kubectl logs -n astrads-system <astrads cluster controller pod name>
```

## Configure Astra Data Store preview monitoring

You can configure Astra Data Store preview for Astra Control Center monitoring or for monitoring by another telemetry service.

Before setting up monitoring, you will need an active Astra data store cluster in the `astrads-system` namespace.

### Configure monitoring for Astra Control Center preview

Perform the following step only after Astra Data Store preview is managed as a backend in Astra Control Center.

1. Configure Astra Data Store preview for monitoring by Astra Control Center:

```
kubectl astrads monitoring -n netapp-monitoring -r [YOUR REGISTRY] setup
```

### Install the monitoring operator

(Optional) The monitoring operator is recommended if Astra Data Store preview will not be imported into Astra Control Center. You can install the monitoring operator if your Astra Data Store preview instance is a standalone deployment, uses Cloud Insights to monitor telemetry, or streams logs to a third-party endpoint such as Elastic.

1. Run this install command:

```
kubectl apply -f ./manifests/monitoring_operator.yaml
```

## 2. Configure Astra Data Store preview for monitoring:

```
kubectl astrads monitoring -n netapp-monitoring -r [YOUR_REGISTRY] setup
```

## What's next

Complete the deployment by performing [setup tasks](#).

## Set up Astra Data Store preview components

After you have installed Astra Data Store preview and addressed some environmental prerequisites, you'll install Astra Trident, configure Kubernetes snapshot capabilities, set up the storage backend, and create a default storage class:

- [Install Astra Trident](#)
- [Install Kubernetes snapshot CRDs and Controller](#)
- [Set up Astra Data Store as storage backend](#)
- [Create a default Astra Data Store storage class](#)

## Install Astra Trident

For Astra Data Store preview, you'll need to install Astra Trident 21.10.1. You can install Astra Trident using one of the following options:

- [Install Astra Trident using tridentctl](#).
- [Install Astra Trident using Trident operator](#).



You can deploy the Trident operator either manually or using Helm.

## Install Kubernetes snapshot CRDs and Controller

Kubernetes snapshot CRDs and controller are required to create persistent volume claim (PVC) snapshots. If you do not already have the CRD and controller installed for your environment, run the following commands to install them.



The following command examples assume `/trident` as the directory; however, the directory you use can be any directory that you used to download the YAML files.

### What you'll need

- [Before you begin installation, prepare your environment for Astra Data Store preview deployment](#).
- Download the [Kubernetes snapshot controller YAML files](#):
  - `setup-snapshot-controller.yaml`



- rbac-snapshot-controller.yaml
- Download the [YAML CRDs](#):
  - snapshot.storage.k8s.io\_volumesnapshotclasses.yaml
  - snapshot.storage.k8s.io\_volumesnapshotcontents.yaml
  - snapshot.storage.k8s.io\_volumesnapshots.yaml

## Steps

1. Apply snapshot.storage.k8s.io\_volumesnapshotclasses.yaml:

```
kubectl apply -f
trident/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
```

Response:

```
customresourcedefinition.apiextensions.k8s.io/volumesnapshotclasses.snapshot.storage.k8s.io configured
```

2. Apply snapshot.storage.k8s.io\_volumesnapshotcontents.yaml:

```
kubectl apply -f
trident/snapshot.storage.k8s.io_volumesnapshotcontents.yaml
```

Response:

```
customresourcedefinition.apiextensions.k8s.io/volumesnapshotcontents.snapshot.storage.k8s.io configured
```

3. Apply snapshot.storage.k8s.io\_volumesnapshots.yaml:

```
kubectl apply -f trident/snapshot.storage.k8s.io_volumesnapshots.yaml
```

Response:

```
customresourcedefinition.apiextensions.k8s.io/volumesnapshots.snapshot.storage.k8s.io configured
```

4. Apply setup-snapshot-controller.yaml:

```
kubectl apply -f trident/setup-snapshot-controller.yaml
```

Response:

```
deployment.apps/snapshot-controller configured
```

5. Apply rbac-snapshot-controller.yaml:

```
kubectl apply -f trident/rbac-snapshot-controller.yaml
```

Response:

```
serviceaccount/snapshot-controller configured
clusterrole.rbac.authorization.k8s.io/snapshot-controller-runner
configured
clusterrolebinding.rbac.authorization.k8s.io/snapshot-controller-role
configured
role.rbac.authorization.k8s.io/snapshot-controller-leaderelection
configured
rolebinding.rbac.authorization.k8s.io/snapshot-controller-leaderelection
configured
```

6. Verify that the CRD YAML files are applied:

```
kubectl get crd | grep volumesnapshot
```

Sample response:

```
astradsvolumesnapshots.astrads.netapp.io          2021-08-
04T17:48:21Z
volumesnapshotclasses.snapshot.storage.k8s.io     2021-08-
04T22:05:49Z
volumesnapshotcontents.snapshot.storage.k8s.io     2021-08-
04T22:05:59Z
volumesnapshots.snapshot.storage.k8s.io           2021-08-
04T22:06:17Z
```

7. Verify that the snapshot controller files are applied:

```
kubectl get pods -n kube-system | grep snapshot
```

Sample response:

```
snapshot-controller-7f58886ff4-cdh78
1/1      Running    0          13s
snapshot-controller-7f58886ff4-tmrd9
1/1      Running    0          32s
```

## Set up Astra Data Store as storage backend

Configure storage backend parameters in the `ads_backend.json` file and create the Astra Data Store storage backend.

### Steps

1. Create `ads_backend.json` using a secure terminal:

```
vi ads_backend.json
```

2. Configure the JSON file:

- a. Change the `"cluster"` value to the cluster name for the Astra Data Store cluster.
- b. Change the `"namespace"` value to the namespace you want to use with volume creation.
- c. Change the `"autoExportPolicy"` value to `true` unless you set up an `exportpolicy` CR instead for this backend.
- d. Populate the `"autoExportCIDRs"` list with IP addresses you want to grant access. Use `0.0.0.0/0` to allow all.
- e. For the `"kubeconfig"` value, do the following:
  - i. Convert and minimize the `.kube/config` YAML file to JSON format without spaces:

Example conversion:

```
python3 -c 'import sys, yaml, json;
json.dump(yaml.load(sys.stdin), sys.stdout, indent=None)' <
~/.kube/config > kubeconf.json
```

- ii. Encode as base64 and use the base64 output for the `"kubeconfig"` value:

Example encoding:

```
cat kubeconf.json | base64 | tr -d '\n'
```

```

{
  "version": 1,
  "storageDriverName": "astrads-nas",
  "storagePrefix": "",
  "cluster": "example-1234584",
  "namespace": "astrads-system",
  "autoExportPolicy": true,
  "autoExportCIDRs": ["0.0.0.0/0"],
  "kubeconfig": "<base64_output_of_kubeconf_json>",
  "debugTraceFlags": {"method": true, "api": true},
  "labels": {"cloud": "on-prem", "creator": "trident-dev"},
  "defaults": {
    "qosPolicy": "bronze"
  },
  "storage": [
    {
      "labels": {
        "performance": "extreme"
      },
      "defaults": {
        "qosPolicy": "bronze"
      }
    },
    {
      "labels": {
        "performance": "premium"
      },
      "defaults": {
        "qosPolicy": "bronze"
      }
    },
    {
      "labels": {
        "performance": "standard"
      },
      "defaults": {
        "qosPolicy": "bronze"
      }
    }
  ]
}

```

3. Change to the directory where you downloaded the Trident installer:

```
cd <trident-installer or path to folder containing tridentctl>
```

#### 4. Create the storage backend:

```
./tridentctl create backend -f ads_backend.json -n trident
```

Sample response:

```
+-----+-----+
+-----+-----+-----+-----+
|      NAME      | STORAGE DRIVER |              UUID              |
| STATE  | VOLUMES |              |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| example-1234584 | astrads-nas    | 2125fa7a-730e-43c8-873b-6012fcc3b527 |
| online  |          0 |              |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

## Create a default Astra Data Store storage class

Create the Astra Trident default storage class and apply it to the storage backend.

### Steps

1. Create the trident-csi storage class:
  - a. Create ads\_sc\_example.yaml:

```
vi ads_sc_example.yaml
```

Example:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: trident-csi
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
allowVolumeExpansion: true
mountOptions:
  - vers=4.1
```

b. Create trident-csi:

```
kubectl create -f ads_sc_example.yaml
```

Response:

```
storageclass.storage.k8s.io/trident-csi created
```

2. Verify that the storage class has been added:

```
kubectl get storageclass -A
```

Response:

NAME	PROVISIONER	RECLAIMPOLICY	VOLUMEBINDINGMODE
ALLOWVOLUMEEXPANSION	AGE		
trident-csi	csi.trident.netapp.io	Delete	Immediate
true	6h29m		

3. Change to the directory where you downloaded the Trident installer:

```
cd <trident-installer or path to folder containing tridentctl>
```

4. Verify that the Astra Trident backend has been updated with the default storage class parameters:

```
./tridentctl get backend -n trident -o yaml
```

Sample response:

```
items:
- backendUUID: 2125fa7a-730e-43c8-873b-6012fcc3b527
  config:
    autoExportCIDRs:
    - 0.0.0.0/0
    autoExportPolicy: true
    backendName: ""
    cluster: example-1234584
    credentials: null
    debug: false
    debugTraceFlags:
    api: true
```

```

    method: true
defaults:
  exportPolicy: default
  qosPolicy: bronze
  size: 1G
  snapshotDir: "false"
  snapshotPolicy: none
disableDelete: false
kubeconfig: <ID>
labels:
  cloud: on-prem
  creator: trident-dev
limitVolumeSize: ""
namespace: astrads-system
nfsMountOptions: ""
region: ""
serialNumbers: null
storage:
- defaults:
  exportPolicy: ""
  qosPolicy: bronze
  size: ""
  snapshotDir: ""
  snapshotPolicy: ""
  labels:
    performance: extreme
  region: ""
  supportedTopologies: null
  zone: ""
- defaults:
  exportPolicy: ""
  qosPolicy: bronze
  size: ""
  snapshotDir: ""
  snapshotPolicy: ""
  labels:
    performance: premium
  region: ""
  supportedTopologies: null
  zone: ""
- defaults:
  exportPolicy: ""
  qosPolicy: bronze
  size: ""
  snapshotDir: ""
  snapshotPolicy: ""

```

```

  labels:
    performance: standard
    region: ""
    supportedTopologies: null
    zone: ""
  storageDriverName: astrads-nas
  storagePrefix: ""
  supportedTopologies: null
  version: 1
  zone: ""
configRef: ""
name: example-1234584
online: true
protocol: file
state: online
storage:
  example-1234584_pool_0:
    name: example-1234584_pool_0
    storageAttributes:
      backendType:
        offer:
          - astrads-nas
      clones:
        offer: true
      encryption:
        offer: false
      labels:
        offer:
          cloud: on-prem
          creator: trident-dev
          performance: extreme
      snapshots:
        offer: true
    storageClasses:
      - trident-csi
    supportedTopologies: null
  example-1234584_pool_1:
    name: example-1234584_pool_1
    storageAttributes:
      backendType:
        offer:
          - astrads-nas
      clones:
        offer: true
      encryption:
        offer: false

```



```

labels:
  offer:
    cloud: on-prem
    creator: trident-dev
    performance: premium
  snapshots:
    offer: true
storageClasses:
- trident-csi
supportedTopologies: null
example-1234584_pool_2:
  name: example-1234584_pool_2
  storageAttributes:
    backendType:
      offer:
        - astrads-nas
    clones:
      offer: true
    encryption:
      offer: false
    labels:
      offer:
        cloud: on-prem
        creator: trident-dev
        performance: standard
    snapshots:
      offer: true
  storageClasses:
  - trident-csi
  supportedTopologies: null
volumes: []

```

## Astra Data Store preview limits

Astra Data Store is Kubernetes-native, shared file software-defined storage (SDS) solution for on-premises data centers that helps customers manage their cloud-native applications.

The Astra Data Store preview release has the following resource limits.

Resource	Minimum	Maximum
Number of nodes in Astra Data Store preview cluster	4	5
Number of persistent volumes per node	N/A	10

Resource	Minimum	Maximum
Total provisioned capacity of persistent volumes per node	N/A	1TiB
Volume size	20MiB	1TiB
Snapshots per volume	0	256
Clones per Volume	0	9



Astra Data Store preview does not support VM workloads. VMware vVol workload support will be available in a future release.



Astra Data Store preview is performance throttled and should not be used for performance characterization.

## Frequently asked questions for Astra Data Store preview

Find answers to the frequently asked questions about installing, configuring, upgrading, and troubleshooting Astra Data Store preview.

### General questions

#### Can I use Astra Data Store preview for production?

No. While Astra Data Store is designed and developed to deliver enterprise grade resiliency, as a preview release Astra Data Store preview is not targeted for production workloads.

#### Can I use Astra Data Store preview for virtual machine workloads?

Astra Data Store preview release is limited only to applications running on Kubernetes, whether on bare metal or virtual machines. Future releases will support applications on both Kubernetes and directly on ESXi virtual machines. See [Astra Data Store requirements](#).

#### Does Astra Data Store preview have any dependencies on other NetApp products for its functioning?

Yes. Astra Data Store preview requires NetApp CSI driver Astra Trident version 21.10.1 and later to be deployed on workload Kubernetes clusters. Learn about [Astra Data Store requirements](#).

Applications using Astra Data Store preview cluster as a storage backend can use [Astra Control Center](#) version 21.12 to leverage application-aware data management capabilities, including data protection, disaster recovery, and migration of Kubernetes workloads.

#### How can I manage Astra Data Store preview cluster?

You can manage Astra Data Store preview assets by using kubectl commands and by using the Kubernetes API extension.

The `kubectl astrads` command includes an `-h` switch that provides usage and flag documentation for your convenience.

#### How can I monitor Astra Data Store preview cluster metrics?

You can monitor Astra Data Store preview metrics using Cloud Insights. See [Monitor metrics with Cloud Insights](#).

You can also monitor logs. See [Configure and monitor event logs](#).

### **Can I use Astra Data Store preview along with ONTAP or other storage providers in my Kubernetes clusters?**

Yes. Astra Data Store preview can be used along with other storage providers in your application clusters.

### **Will Astra Trident be uninstalled if I remove a Kubernetes cluster from Astra Data Store preview?**

Astra Trident will not be uninstalled from the cluster if you uninstall Astra Data Store preview. If you require Astra Trident to be uninstalled, you will have to do it separately.

## **Licensing**

### **Does the Astra Data Store preview require a license?**

Yes, Astra Data Store preview requires a NetApp license file (NLF).

See [Astra Data Store requirements](#).

### **How long is the Astra Data Store preview license valid?**

The Astra Data Store preview license has a default term of 90 days from the download date.

## **Installation and use of Astra Data Store preview on a Kubernetes cluster**

### **Can I install Astra Data Store preview on a Kubernetes cluster running on bare metal or virtual machines?**

Yes. Astra Data Store preview can be installed on a Kubernetes cluster running on bare metal or on ESXi VMs. See [Astra Data Store preview requirements](#).

### **What are the supported versions of Kubernetes for Astra Data Store preview?**

Astra Data Store preview works with Kubernetes distributions that are compatible with v1.20 and later. However, it is not currently validated with all Kubernetes distributions. Learn about [Astra Data Store preview requirements](#).

### **My Kubernetes cluster is larger than 5 worker nodes. Can I install Astra Data Store preview on it?**

Yes. An Astra Data Store preview cluster can be deployed on 4 worker nodes in a Kubernetes cluster. After deployment, you can grow the cluster to 5 worker nodes.

### **Does Astra Data Store preview support an offline installation from a private registry?**

Yes. Astra Data Store preview can be installed offline from a local registry. See [Install Astra Data Store preview](#). However, Astra Data Store preview requires connectivity (direct or via a proxy) to the NetApp AutoSupport backend (support.netapp.com) for continued operations.

### **Do I need internet connectivity to use Astra Data Store preview?**

Astra Data Store preview requires connectivity to the NetApp AutoSupport backend to periodically send the mandatory telemetry AutoSupport bundles. This connectivity can be direct or via a proxy. If this connectivity is missing or AutoSupport is disabled, the cluster will lock down and new volume creation will be disabled until the periodic bundle uploads are resumed.

### **What are the roles and privileges used by Astra Data Store preview?**

You need to be a kube admin to deploy the Astra Data Store preview operator.

Astra Data Store preview has a privileged daemonset called `astrads-ds-nodeinfo-astradsversion` for discovering node resources used in selecting nodes.

In addition, the operator will use privileged Kubernetes jobs to install the storage cluster's containers on the selected worker nodes to build the Astra Data Store preview storage cluster.

## What manifest files do I need to update for Astra Data Store preview installation?

From the Astra Data Store preview bundle you download from the [NetApp Support Site](#), you will get the following manifests:

- `astradscluster.yaml`
- `astradsoperator.yaml`
- `astradsversion.yaml`
- `monitoring_operator.yaml`

You will need to update the `astradscluster.yaml` manifest with your deployment-specific configuration. See [Install Astra Data Store preview](#).

## Troubleshooting and support

With the Astra Data Store preview, you can access community support using the NetApp Containers Slack channel. This channel is monitored by NetApp Support and our Technical Marketing Engineers.

### [NetApp Container Slack channel](#)

The preview release requires that your system is connected to the cloud and integrated into the NetApp Active IQ and AutoSupport tools.

See [Astra Data Store support operations](#).

### How do I raise a support case OR ask clarification on a quick question?

To raise a support case or get clarification on a quick question, report your issue or question on the [NetApp Container Slack channel](#). NetApp Support will follow up with you to assist on a best-effort basis.

### How do I raise a request for a new feature?

If you have questions on supported configurations or capabilities, reach out to [astra.feedback@netapp.com](mailto:astra.feedback@netapp.com).

### How do I generate a support log bundle?

See [Generate support bundle](#) for instructions on setting up and downloading support log bundles for Astra Data Store preview.

### Astra Data Store preview cannot find my Kubernetes node. How do I fix this?

See [Install Astra Data Store preview](#).

### Can IPv6 addresses be used for management, data, and cluster networks?

No, Astra Data Store preview only supports IPv4 address. IPv6 support will be added in a future release of Astra Data Store preview.

### What NFS version is used while provisioning a volume on Astra Data Store preview?

By default, Astra Data Store preview supports NFS v4.1 for all volumes provisioned for Kubernetes applications.

### Why can't I get larger persistent volumes even though I have configured Astra Data Store preview with large capacity drives?

Astra Data Store preview limits the maximum capacity provisioned for all volumes on a node to 1 TiB and up to 5 TiB across all nodes in an Astra Data Store preview cluster.

See [Astra Data Store preview requirements](#) and [Astra Data Store preview limits](#).

## Upgrading Astra Data Store preview

### Can I upgrade from Astra Data Store preview release?

No. Astra Data Store preview is not for production workloads and new releases of Astra Data Store preview software will require a fresh installation.

## Copyright Information

Copyright © 2022 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means-graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system-without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

## Trademark Information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.