NetApp

Install using Trident operator

Astra Trident

NetApp
June 28, 2024

Table of Contents

Install using Trident operator

Manually deploy the Trident operator (Standard mode)

You can manually deploy the Trident operator to install Astra Trident. This process applies to installations where the container images required by Astra Trident are not stored in a private registry. If you do have a private image registry, use the process for offline deployment.

Critical information about Astra Trident 24.02

You must read the following critical information about Astra Trident.

Critical information about Astra Trident

- Kubernetes 1.27 is now supported in Trident. Upgrade Trident prior to upgrading Kubernetes.

- Astra Trident strictly enforces the use of multipathing configuration in SAN environments, with a recommended value of `find_multipaths: no` in multipath.conf file.

  Use of non-multipathing configuration or use of `find_multipaths: yes` or `find_multipaths: smart` value in multipath.conf file will result in mount failures. Trident has recommended the use of `find_multipaths: no` since the 21.07 release.

Manually deploy the Trident operator and install Trident

Review the installation overview to ensure you've met installation prerequisites and selected the correct installation option for your environment.

Before you begin

Before you begin installation, log in to the Linux host and verify it is managing a working, supported Kubernetes cluster and that you have the necessary privileges.

> (i) With OpenShift, use `oc` instead of `kubectl` in all of the examples that follow, and log in as **system:admin** first by running `oc login -u system:admin` or `oc login -u kube-admin`.

1. Verify your Kubernetes version:

```
kubectl version
```

2. Verify cluster administrator privileges:

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Verify you can launch a pod that uses an image from Docker Hub and reach your storage system over the pod network:

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Step 1: Download the Trident installer package

The Astra Trident installer package contains everything you need to deploy the Trident operator and install Astra Trident. Download and extract the latest version of the Trident installer from the *Assets* section on GitHub.

```
wget https://github.com/NetApp/trident/releases/download/v24.02.0/trident-
installer-24.02.0.tar.gz
tar -xf trident-installer-24.02.0.tar.gz
cd trident-installer
```

Step 2: Create the `TridentOrchestrator` CRD

Create the `TridentOrchestrator` Custom Resource Definition (CRD). You will create a `TridentOrchestrator` Custom Resources later. Use the appropriate CRD YAML version in `deploy/crds` to create the `TridentOrchestrator` CRD.

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

Step 3: Deploy the Trident operator

The Astra Trident installer provides a bundle file that can be used to install the operator and create associated objects. The bundle file is an easy way to deploy the operator and install Astra Trident using a default configuration.

- For clusters running Kubernetes 1.24 or earlier, use `bundle_pre_1_25.yaml`.

- For clusters running Kubernetes 1.25 or later, use `bundle_post_1_25.yaml`.

Before you begin

- By default, the Trident installer deploys the operator in the `trident` namespace. If the `trident` namespace does not exist, create it using:

```
kubectl apply -f deploy/namespace.yaml
```

- To deploy the operator in a namespace other than the `trident` namespace, update `serviceaccount.yaml`, `clusterrolebinding.yaml` and `operator.yaml` and generate your bundle file using the `kustomization.yaml`.

  1. Create the `kustomization.yaml` using the following command where *<bundle.yaml>* is `bundle_pre_1_25.yaml` or `bundle_post_1_25.yaml` based on your Kubernetes version.

  ```
  cp deploy/kustomization_<bundle.yaml> deploy/kustomization.yaml
  ```

  2. Compile the bundle using using the following command where *<bundle.yaml>* is `bundle_pre_1_25.yaml` or `bundle_post_1_25.yaml` based on your Kubernetes version.

  ```
  kubectl kustomize deploy/ > deploy/<bundle.yaml>
  ```

Steps

1. Create the resources and deploy the operator:

```
kubectl create -f deploy/<bundle.yaml>
```

2. Verify the operator, deployment, and replicasets were created.

```
kubectl get all -n <operator-namespace>
```

> (i) There should only be **one instance** of the operator in a Kubernetes cluster. Do not create multiple deployments of the Trident operator.

Step 4: Create the `TridentOrchestrator` and install Trident

You can now create the `TridentOrchestrator` and install Astra Trident. Optionally, you can customize your Trident installation using the attributes in the `TridentOrchestrator` spec.

```
kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubectl describe torc trident

Name:        trident
Namespace:
Labels:      <none>
Annotations: <none>
API Version: trident.netapp.io/v1
Kind:        TridentOrchestrator
...
Spec:
  Debug:     true
  Namespace: trident
Status:
  Current Installation Params:
    IPv6:                      false
    Autosupport Hostname:
    Autosupport Image:         netapp/trident-autosupport:24.02
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                     true
    Image Pull Secrets:
    Image Registry:
    k8sTimeout:          30
    Kubelet Dir:         /var/lib/kubelet
    Log Format:          text
    Silence Autosupport: false
    Trident Image:       netapp/trident:24.02.0
  Message:                 Trident installed  Namespace:
trident
  Status:                 Installed
  Version:                v24.02.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Verify the installation

There are several ways to verify your installation.

Using `TridentOrchestrator` status

The status of `TridentOrchestrator` indicates if the installation was successful and displays the version of Trident installed. During the installation, the status of `TridentOrchestrator` changes from `Installing` to `Installed`. If you observe the `Failed` status and the operator is unable to recover by itself, check the logs.

Status	Description
Installing	The operator is installing Astra Trident using this `TridentOrchestrator` CR.
Installed	Astra Trident has successfully installed.
Uninstalling	The operator is uninstalling Astra Trident, because `spec.uninstall=true`.
Uninstalled	Astra Trident is uninstalled.
Failed	The operator could not install, patch, update or uninstall Astra Trident; the operator will automatically try to recover from this state. If this state persists you will require troubleshooting.
Updating	The operator is updating an existing installation.
Error	The `TridentOrchestrator` is not used. Another one already exists.

Using pod creation status

You can confirm if the Astra Trident installation completed by reviewing the status of the created pods:

```
kubectl get pods -n trident

NAME                                      READY    STATUS     RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw       6/6      Running    0
1m
trident-node-linux-mr6zc                  2/2      Running    0
1m
trident-node-linux-xrp7w                  2/2      Running    0
1m
trident-node-linux-zh2jt                  2/2      Running    0
1m
trident-operator-766f7b8658-ldzsv         1/1      Running    0
3m
```

Using `tridentctl`

You can use `tridentctl` to check the version of Astra Trident installed.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.02.0        | 24.02.0        |
+----------------+----------------+
```

Manually deploy the Trident operator (Offline mode)

You can manually deploy the Trident operator to install Astra Trident. This process applies to installations where the container images required by Astra Trident are stored in a private registry. If you do not have a private image registry, use the process for standard deployment.

Critical information about Astra Trident 24.02

You must read the following critical information about Astra Trident.

Critical information about Astra Trident

- Kubernetes 1.27 is now supported in Trident. Upgrade Trident prior to upgrading Kubernetes.
- Astra Trident strictly enforces the use of multipathing configuration in SAN environments, with a recommended value of `find_multipaths: no` in multipath.conf file.

  Use of non-multipathing configuration or use of `find_multipaths: yes` or `find_multipaths: smart` value in multipath.conf file will result in mount failures. Trident has recommended the use of `find_multipaths: no` since the 21.07 release.

Manually deploy the Trident operator and install Trident

Review the installation overview to ensure you've met installation prerequisites and selected the correct installation option for your environment.

Before you begin

Log in to the Linux host and verify it is managing a working and supported Kubernetes cluster and that you have the necessary privileges.

> (i) With OpenShift, use `oc` instead of `kubectl` in all of the examples that follow, and log in as **system:admin** first by running `oc login -u system:admin` or `oc login -u kube-admin`.

1. Verify your Kubernetes version:

```
kubectl version
```

2. Verify cluster administrator privileges:

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Verify you can launch a pod that uses an image from Docker Hub and reach your storage system over the pod network:

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Step 1: Download the Trident installer package

The Astra Trident installer package contains everything you need to deploy the Trident operator and install Astra Trident. Download and extract the latest version of the Trident installer from the *Assets* section on GitHub.

```
wget https://github.com/NetApp/trident/releases/download/v24.02.0/trident-
installer-24.02.0.tar.gz
tar -xf trident-installer-24.02.0.tar.gz
cd trident-installer
```

Step 2: Create the `TridentOrchestrator` CRD

Create the `TridentOrchestrator` Custom Resource Definition (CRD). You will create a `TridentOrchestrator` Custom Resources later. Use the appropriate CRD YAML version in `deploy/crds` to create the `TridentOrchestrator` CRD:

```
kubectl create -f deploy/crds/<VERSION>.yaml
```

Step 3: Update the registry location in the operator

In `/deploy/operator.yaml`, update `image: docker.io/netapp/trident-operator:24.02.0` to reflect the location of your image registry. Your Trident and CSI images can be located in one registry or different registries, but all CSI images must be located in the same registry. For example:

- `image: <your-registry>/trident-operator:24.02.0` if your images are all located in one registry.

* `image: <your-registry>/netapp/trident-operator:24.02.0` if your Trident image is located in a different registry from your CSI images.

Step 4: Deploy the Trident operator

The Astra Trident installer provides a bundle file that can be used to install the operator and create associated objects. The bundle file is an easy way to deploy the operator and install Astra Trident using a default configuration.

* For clusters running Kubernetes 1.24 or earlier, use `bundle_pre_1_25.yaml`.
* For clusters running Kubernetes 1.25 or later, use `bundle_post_1_25.yaml`.

Before you begin

* By default, the Trident installer deploys the operator in the `trident` namespace. If the `trident` namespace does not exist, create it using:

```
kubectl apply -f deploy/namespace.yaml
```

* To deploy the operator in a namespace other than the `trident` namespace, update `serviceaccount.yaml`, `clusterrolebinding.yaml` and `operator.yaml` and generate your bundle file using the `kustomization.yaml`.

  1. Create the `kustomization.yaml` using the following command where *<bundle.yaml>* is `bundle_pre_1_25.yaml` or `bundle_post_1_25.yaml` based on your Kubernetes version.

     ```
     cp deploy/kustomization_<bundle.yaml> deploy/kustomization.yaml
     ```

  2. Compile the bundle using using the following command where *<bundle.yaml>* is `bundle_pre_1_25.yaml` or `bundle_post_1_25.yaml` based on your Kubernetes version.

     ```
     kubectl kustomize deploy/ > deploy/<bundle.yaml>
     ```

Steps

1. Create the resources and deploy the operator:

   ```
   kubectl create -f deploy/<bundle.yaml>
   ```

2. Verify the operator, deployment, and replicasets were created.

   ```
   kubectl get all -n <operator-namespace>
   ```

> ⓘ There should only be **one instance** of the operator in a Kubernetes cluster. Do not create multiple deployments of the Trident operator.

Step 5: Update the image registry location in the `TridentOrchestrator`

Your Trident and CSI images can be located in one registry or different registries, but all CSI images must be located in the same registry. Update `deploy/crds/tridentorchestrator_cr.yaml` to add the additional location specs based on your registry configuration.

Images in one registry

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:24.02"
tridentImage: "<your-registry>/trident:24.02.0"
```

Images in different registries

You must append `sig-storage` to the `imageRegistry` to use different registry locations.

```
imageRegistry: "<your-registry>/sig-storage"
autosupportImage: "<your-registry>/netapp/trident-autosupport:24.02"
tridentImage: "<your-registry>/netapp/trident:24.02.0"
```

Step 6: Create the `TridentOrchestrator` **and install Trident**

You can now create the `TridentOrchestrator` and install Astra Trident. Optionally, you can further customize your Trident installation using the attributes in the `TridentOrchestrator` spec. The following example shows an installation where Trident and CSI images are located in different registries.

```
kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubectl describe torc trident

Name:        trident
Namespace:
Labels:      <none>
Annotations: <none>
API Version: trident.netapp.io/v1
Kind:        TridentOrchestrator
...
Spec:
  Autosupport Image:  <your-registry>/netapp/trident-autosupport:24.02
  Debug:              true
  Image Registry:     <your-registry>/sig-storage
  Namespace:          trident
  Trident Image:      <your-registry>/netapp/trident:24.02.0
Status:
  Current Installation Params:
    IPv6:                     false
    Autosupport Hostname:
    Autosupport Image:        <your-registry>/netapp/trident-
autosupport:24.02
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                    true
    Http Request Timeout:     90s
    Image Pull Secrets:
    Image Registry:     <your-registry>/sig-storage
    k8sTimeout:         30
    Kubelet Dir:        /var/lib/kubelet
    Log Format:         text
    Probe Port:         17546
    Silence Autosupport: false
    Trident Image:      <your-registry>/netapp/trident:24.02.0
  Message:              Trident installed
  Namespace:            trident
  Status:               Installed
  Version:              v24.02.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Verify the installation

There are several ways to verify your installation.

Using `TridentOrchestrator` status

The status of `TridentOrchestrator` indicates if the installation was successful and displays the version of Trident installed. During the installation, the status of `TridentOrchestrator` changes from `Installing` to `Installed`. If you observe the `Failed` status and the operator is unable to recover by itself, check the logs.

Status	Description
Installing	The operator is installing Astra Trident using this `TridentOrchestrator` CR.
Installed	Astra Trident has successfully installed.
Uninstalling	The operator is uninstalling Astra Trident, because `spec.uninstall=true`.
Uninstalled	Astra Trident is uninstalled.
Failed	The operator could not install, patch, update or uninstall Astra Trident; the operator will automatically try to recover from this state. If this state persists you will require troubleshooting.
Updating	The operator is updating an existing installation.
Error	The `TridentOrchestrator` is not used. Another one already exists.

Using pod creation status

You can confirm if the Astra Trident installation completed by reviewing the status of the created pods:

```
kubectl get pods -n trident

NAME                                    READY    STATUS     RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw     6/6      Running    0
1m
trident-node-linux-mr6zc                2/2      Running    0
1m
trident-node-linux-xrp7w                2/2      Running    0
1m
trident-node-linux-zh2jt                2/2      Running    0
1m
trident-operator-766f7b8658-ldzsv       1/1      Running    0
3m
```

Using `tridentctl`

You can use `tridentctl` to check the version of Astra Trident installed.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.02.0        | 24.02.0        |
+----------------+----------------+
```

Deploy Trident operator using Helm (Standard mode)

You can deploy the Trident operator and install Astra Trident using Helm. This process applies to installations where the container images required by Astra Trident are not stored in a private registry. If you do have a private image registry, use the process for offline deployment.

Critical information about Astra Trident 24.02

You must read the following critical information about Astra Trident.

Critical information about Astra Trident

- Kubernetes 1.27 is now supported in Trident. Upgrade Trident prior to upgrading Kubernetes.
- Astra Trident strictly enforces the use of multipathing configuration in SAN environments, with a recommended value of `find_multipaths: no` in multipath.conf file.

  Use of non-multipathing configuration or use of `find_multipaths: yes` or `find_multipaths: smart` value in multipath.conf file will result in mount failures. Trident has recommended the use of `find_multipaths: no` since the 21.07 release.

Deploy the Trident operator and install Astra Trident using Helm

Using the Trident Helm Chart you can deploy the Trident operator and install Trident in one step.

Review the installation overview to ensure you've met installation prerequisites and selected the correct installation option for your environment.

Before you begin

In addition to the deployment prerequisites you need Helm version 3.

Steps

1. Add the Astra Trident Helm repository:

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. Use `helm install` and specify a name for your deployment as in the following example where `100.2402.0` is the version of Astra Trident you are installing.

```
helm install <name> netapp-trident/trident-operator --version 100.2402.0
--create-namespace --namespace <trident-namespace>
```

> (i) If you already created a namespace for Trident, the `--create-namespace` parameter will
>   not create an additional namespace.

You can use `helm list` to review installation details such as name, namespace, chart, status, app version, and revision number.

Pass configuration data during install

There are two ways to pass configuration data during the install:

Option	Description
`--values` (or `-f`)	Specify a YAML file with overrides. This can be specified multiple times and the rightmost file will take precedence.
`--set`	Specify overrides on the command line.

For example, to change the default value of `debug`, run the following `--set` command where `100.2402.0` is the version of Astra Trident you are installing:

```
helm install <name> netapp-trident/trident-operator --version 100.2402.0
--create-namespace --namespace trident --set tridentDebug=true
```

Configuration options

This table and the `values.yaml` file, which is part of the Helm chart, provide the list of keys and their default values.

Option	Description	Default
`nodeSelector`	Node labels for pod assignment	
`podAnnotations`	Pod annotations	
`deploymentAnnotatio ns`	Deployment annotations	

Option	Description	Default
`tolerations`	Tolerations for pod assignment	
`affinity`	Affinity for pod assignment	
`tridentControllerPluginNodeSelector`	Additional node selectors for pods. Refer to Understanding controller pods and node pods for details.	
`tridentControllerPluginTolerations`	Overrides Kubernetes tolerations for pods. Refer to Understanding controller pods and node pods for details.	
`tridentNodePluginNodeSelector`	Additional node selectors for pods. Refer to Understanding controller pods and node pods for details.	
`tridentNodePluginTolerations`	Overrides Kubernetes tolerations for pods. Refer to Understanding controller pods and node pods for details.	
`imageRegistry`	Identifies the registry for the `trident-operator`, `trident`, and other images. Leave empty to accept the default.	`""`
`imagePullPolicy`	Sets the image pull policy for the `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Sets the image pull secrets for the `trident-operator`, `trident`, and other images.	
`kubeletDir`	Allows overriding the host location of kubelet's internal state.	`"/var/lib/kubelet"`
`operatorLogLevel`	Allows the log level of the Trident operator to be set to: `trace`, `debug`, `info`, `warn`, `error`, or `fatal`.	`"info"`
`operatorDebug`	Allows the log level of the Trident operator to be set to debug.	`true`
`operatorImage`	Allows the complete override of the image for `trident-operator`.	`""`
`operatorImageTag`	Allows overriding the tag of the `trident-operator` image.	`""`
`tridentIPv6`	Allows enabling Astra Trident to work in IPv6 clusters.	`false`
`tridentK8sTimeout`	Overrides the default 30-second timeout for most Kubernetes API operations (if non-zero, in seconds).	`0`
`tridentHttpRequestTimeout`	Overrides the default 90-second timeout for the HTTP requests, with `0s` being an infinite duration for the timeout. Negative values are not allowed.	`"90s"`
`tridentSilenceAutosupport`	Allows disabling Astra Trident periodic AutoSupport reporting.	`false`
`tridentAutosupportImageTag`	Allows overriding the tag of the image for Astra Trident AutoSupport container.	`<version>`

Option	Description	Default
`tridentAutosupportProxy`	Allows Astra Trident AutoSupport container to phone home via an HTTP proxy.	`""`
`tridentLogFormat`	Sets the Astra Trident logging format (`text` or `json`).	`"text"`
`tridentDisableAuditLog`	Disables Astra Trident audit logger.	`true`
`tridentLogLevel`	Allows the log level of Astra Trident to be set to: `trace`, `debug`, `info`, `warn`, `error`, or `fatal`.	`"info"`
`tridentDebug`	Allows the log level of Astra Trident to be set to `debug`.	`false`
`tridentLogWorkflows`	Allows specific Astra Trident workflows to be enabled for trace logging or log suppression.	`""`
`tridentLogLayers`	Allows specific Astra Trident layers to be enabled for trace logging or log suppression.	`""`
`tridentImage`	Allows the complete override of the image for Astra Trident.	`""`
`tridentImageTag`	Allows overriding the tag of the image for Astra Trident.	`""`
`tridentProbePort`	Allows overriding the default port used for Kubernetes liveness/readiness probes.	`""`
`windows`	Allows Astra Trident to be installed on Windows worker node.	`false`
`enableForceDetach`	Allows enabling the force detach feature.	`false`
`excludePodSecurityPolicy`	Excludes the operator pod security policy from creation.	`false`
`cloudProvider`	Set to `"Azure"` when using managed identities or a cloud identity on an AKS cluster. Set to "AWS" when using a cloud identity on an EKS cluster.	`""`
`cloudIdentity`	Set to workload identity ("azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx") when using cloud identity on an AKS cluster. Set to AWS IAM role ("'eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/astratrident-role'") when using cloud identity on an EKS cluster.	`""`
`iscsiSelfHealingInterval`	The interval at which the iSCSI self-healing is invoked.	`5m0s`
`iscsiSelfHealingWaitTime`	The duration after which iSCSI self-healing initiates an attempt to resolve a stale session by performing a logout and subsequent login.	`7m0s`

Understanding controller pods and node pods

Astra Trident runs as a single controller pod, plus a node pod on each worker node in the cluster. The node pod must be running on any host where you want to potentially mount an Astra Trident volume.

Kubernetes node selectors and tolerations and taints are used to constrain a pod to run on a specific or preferred node. Using the`ControllerPlugin` and `NodePlugin`, you can specify constraints and overrides.

- The controller plugin handles volume provisioning and management, such as snapshots and resizing.

- The node plugin handles attaching the storage to the node.

Deploy Trident operator using Helm (Offline mode)

You can deploy the Trident operator and install Astra Trident using Helm. This process applies to installations where the container images required by Astra Trident are stored in a private registry. If you do not have a private image registry, use the process for standard deployment.

Critical information about Astra Trident 24.02

You must read the following critical information about Astra Trident.

Critical information about Astra Trident

- Kubernetes 1.27 is now supported in Trident. Upgrade Trident prior to upgrading Kubernetes.

- Astra Trident strictly enforces the use of multipathing configuration in SAN environments, with a recommended value of `find_multipaths: no` in multipath.conf file.

  Use of non-multipathing configuration or use of `find_multipaths: yes` or `find_multipaths: smart` value in multipath.conf file will result in mount failures. Trident has recommended the use of `find_multipaths: no` since the 21.07 release.

Deploy the Trident operator and install Astra Trident using Helm

Using the Trident Helm Chart you can deploy the Trident operator and install Trident in one step.

Review the installation overview to ensure you've met installation prerequisites and selected the correct installation option for your environment.

Before you begin

In addition to the deployment prerequisites you need Helm version 3.

Steps

1. Add the Astra Trident Helm repository:

   ```
   helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
   ```

2. Use `helm install` and specify a name for your deployment and image registry location. Your Trident

and CSI images can be located in one registry or different registries, but all CSI images must be located in the same registry. In the examples, `100.2402.0` is the version of Astra Trident you are installing.

Images in one registry

```
helm install <name> netapp-trident/trident-operator --version
100.2402.0 --set imageRegistry=<your-registry> --create-namespace
--namespace <trident-namespace>
```

Images in different registries

You must append `sig-storage` to the `imageRegistry` to use different registry locations.

```
helm install <name> netapp-trident/trident-operator --version
100.2402.0 --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=<your-registry>/netapp/trident-operator:24.02.0 --set
tridentAutosupportImage=<your-registry>/netapp/trident-
autosupport:24.02 --set tridentImage=<your-
registry>/netapp/trident:24.02.0 --create-namespace --namespace
<trident-namespace>
```

ⓘ If you already created a namespace for Trident, the `--create-namespace` parameter will not create an additional namespace.

You can use `helm list` to review installation details such as name, namespace, chart, status, app version, and revision number.

Pass configuration data during install

There are two ways to pass configuration data during the install:

Option	Description
`--values` (or `-f`)	Specify a YAML file with overrides. This can be specified multiple times and the rightmost file will take precedence.
`--set`	Specify overrides on the command line.

For example, to change the default value of `debug`, run the following `--set` command where `100.2402.0` is the version of Astra Trident you are installing:

```
helm install <name> netapp-trident/trident-operator --version 100.2402.0
--create-namespace --namespace trident --set tridentDebug=true
```

Configuration options

This table and the `values.yaml` file, which is part of the Helm chart, provide the list of keys and their default values.

Option	Description	Default
`nodeSelector`	Node labels for pod assignment	
`podAnnotations`	Pod annotations	
`deploymentAnnotations`	Deployment annotations	
`tolerations`	Tolerations for pod assignment	
`affinity`	Affinity for pod assignment	
`tridentControllerPluginNodeSelector`	Additional node selectors for pods. Refer to Understanding controller pods and node pods for details.	
`tridentControllerPluginTolerations`	Overrides Kubernetes tolerations for pods. Refer to Understanding controller pods and node pods for details.	
`tridentNodePluginNodeSelector`	Additional node selectors for pods. Refer to Understanding controller pods and node pods for details.	
`tridentNodePluginTolerations`	Overrides Kubernetes tolerations for pods. Refer to Understanding controller pods and node pods for details.	
`imageRegistry`	Identifies the registry for the `trident-operator, trident,` and other images. Leave empty to accept the default.	`""`
`imagePullPolicy`	Sets the image pull policy for the `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Sets the image pull secrets for the `trident-operator, trident,` and other images.	
`kubeletDir`	Allows overriding the host location of kubelet's internal state.	`"/var/lib/kubelet"`
`operatorLogLevel`	Allows the log level of the Trident operator to be set to: `trace, debug, info, warn, error,` or `fatal`.	`"info"`
`operatorDebug`	Allows the log level of the Trident operator to be set to debug.	`true`
`operatorImage`	Allows the complete override of the image for `trident-operator`.	`""`

Option	Description	Default
`operatorImageTag`	Allows overriding the tag of the `trident-operator` image.	`""`
`tridentIPv6`	Allows enabling Astra Trident to work in IPv6 clusters.	`false`
`tridentK8sTimeout`	Overrides the default 30-second timeout for most Kubernetes API operations (if non-zero, in seconds).	`0`
`tridentHttpRequestTimeout`	Overrides the default 90-second timeout for the HTTP requests, with `0s` being an infinite duration for the timeout. Negative values are not allowed.	`"90s"`
`tridentSilenceAutosupport`	Allows disabling Astra Trident periodic AutoSupport reporting.	`false`
`tridentAutosupportImageTag`	Allows overriding the tag of the image for Astra Trident AutoSupport container.	`<version>`
`tridentAutosupportProxy`	Allows Astra Trident AutoSupport container to phone home via an HTTP proxy.	`""`
`tridentLogFormat`	Sets the Astra Trident logging format (`text` or `json`).	`"text"`
`tridentDisableAuditLog`	Disables Astra Trident audit logger.	`true`
`tridentLogLevel`	Allows the log level of Astra Trident to be set to: `trace`, `debug`, `info`, `warn`, `error`, or `fatal`.	`"info"`
`tridentDebug`	Allows the log level of Astra Trident to be set to `debug`.	`false`
`tridentLogWorkflows`	Allows specific Astra Trident workflows to be enabled for trace logging or log suppression.	`""`
`tridentLogLayers`	Allows specific Astra Trident layers to be enabled for trace logging or log suppression.	`""`
`tridentImage`	Allows the complete override of the image for Astra Trident.	`""`
`tridentImageTag`	Allows overriding the tag of the image for Astra Trident.	`""`
`tridentProbePort`	Allows overriding the default port used for Kubernetes liveness/readiness probes.	`""`

Option	Description	Default
`windows`	Allows Astra Trident to be installed on Windows worker node.	`false`
`enableForceDetach`	Allows enabling the force detach feature.	`false`
`excludePodSecurityPolicy`	Excludes the operator pod security policy from creation.	`false`

Customize Trident operator installation

The Trident operator allows you to customize Astra Trident installation using the attributes in the `TridentOrchestrator` spec. If you want to customize the installation beyond what `TridentOrchestrator` arguments allow, consider using `tridentctl` to generate custom YAML manifests to modify as needed.

Understanding controller pods and node pods

Astra Trident runs as a single controller pod, plus a node pod on each worker node in the cluster. The node pod must be running on any host where you want to potentially mount an Astra Trident volume.

Kubernetes node selectors and tolerations and taints are used to constrain a pod to run on a specific or preferred node. Using the`ControllerPlugin` and `NodePlugin`, you can specify constraints and overrides.

- The controller plugin handles volume provisioning and management, such as snapshots and resizing.

- The node plugin handles attaching the storage to the node.

Configuration options

> ⚠️ `spec.namespace` is specified in `TridentOrchestrator` to signify the namespace where Astra Trident is installed. This parameter **cannot be updated after Astra Trident is installed**. Attempting to do so causes the `TridentOrchestrator` status to change to `Failed`. Astra Trident is not intended to be migrated across namespaces.

This table details `TridentOrchestrator` attributes.

Parameter	Description	Default
`namespace`	Namespace to install Astra Trident in	`"default"`
`debug`	Enable debugging for Astra Trident	`false`
`enableForceDetach`	`ontap-san` and `ontap-san-economy` only. Works with Kubernetes Non-Graceful Node Shutdown (NGNS) to grant cluster administrators ability to safely migrate workloads with mounted volumes to new nodes should a node become unhealthy.	`false`

Parameter	Description	Default
`windows`	Setting to `true` enables installation on Windows worker nodes.	`false`
`cloudProvider`	Set to `"Azure"` when using managed identities or a cloud identity on an AKS cluster. Set to "AWS" when using a cloud identity on an EKS cluster.	`""`
`cloudIdentity`	Set to workload identity ("azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx") when using cloud identity on an AKS cluster. Set to AWS IAM role ("'eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/astratrident-role") when using cloud identity on an EKS cluster.	`""`
`IPv6`	Install Astra Trident over IPv6	false
`k8sTimeout`	Timeout for Kubernetes operations	`30sec`
`silenceAutosupport`	Don't send autosupport bundles to NetApp automatically	`false`
`autosupportImage`	The container image for Autosupport Telemetry	`"netapp/trident-autosupport:24.02"`
`autosupportProxy`	The address/port of a proxy for sending Autosupport Telemetry	`"http://proxy.example.com:8888"`
`uninstall`	A flag used to uninstall Astra Trident	`false`
`logFormat`	Astra Trident logging format to be used [text,json]	`"text"`
`tridentImage`	Astra Trident image to install	`"netapp/trident:24.02"`
`imageRegistry`	Path to internal registry, of the format `<registry FQDN>[:port][/subpath]`	`"k8s.gcr.io/sig-storage"` (Kubernetes 1.19+) or `"quay.io/k8scsi"`
`kubeletDir`	Path to the kubelet directory on the host	`"/var/lib/kubelet"`
`wipeout`	A list of resources to delete to perform a complete removal of Astra Trident	
`imagePullSecrets`	Secrets to pull images from an internal registry	
`imagePullPolicy`	Sets the image pull policy for the the Trident operator. Valid values are: `Always` to always pull the image. `IfNotPresent` to pull the image only if it does not already exist on the node. `Never` to never pull the image.	`IfNotPresent`

Parameter	Description	Default
`controllerPluginNodeSelector`	Additional node selectors for pods. Follows same format as `pod.spec.nodeSelector`.	No default; optional
`controllerPluginTolerations`	Overrides Kubernetes tolerations for pods. Follows the same format as `pod.spec.Tolerations`.	No default; optional
`nodePluginNodeSelector`	Additional node selectors for pods. Follows same format as `pod.spec.nodeSelector`.	No default; optional
`nodePluginTolerations`	Overrides Kubernetes tolerations for pods. Follows the same format as `pod.spec.Tolerations`.	No default; optional

> ℹ️ For more information on formatting pod parameters, refer to Assigning Pods to Nodes.

Details about force detach

Force detach is available for `ontap-san` and `ontap-san-economy` only. Before enabling force detach, non-graceful node shutdown (NGNS) must be enabled on the Kubernetes cluster. For more information, refer to Kubernetes: Non Graceful node shutdown.

> ⚠️ Because Astra Trident relies on Kubernetes NGNS, do not remove `out-of-service` taints from an unhealthy node until all non-tolerable workloads are rescheduled. Recklessly applying or removing the taint can jeopardize backend data protection.

When the Kubernetes cluster administrator has applied the `node.kubernetes.io/out-of-service=nodeshutdown:NoExecute` taint to the node and `enableForceDetach` is set to `true`, Astra Trident will determine the node status and:

1. Cease backend I/O access for volumes mounted to that node.

2. Mark the Astra Trident node object as `dirty` (not safe for new publications).

> ℹ️ The Trident controller will reject new publish volume requests until the node is re-qualified (after having been marked as `dirty`) by the Trident node pod. Any workloads scheduled with a mounted PVC (even after the cluster node is healthy and ready) will be not be accepted until Astra Trident can verify the node `clean` (safe for new publications).

When node health is restored and the taint is removed, Astra Trident will:

1. Identify and clean stale published paths on the node.

2. If the node is in a `cleanable` state (the out-of-service taint has been removed and the node is in `Ready` state) and all stale, published paths are clean, Astra Trident will readmit the node as `clean` and allow new published volumes to the node.

Sample configurations

You can use the attributes in Configuration options when defining `TridentOrchestrator` to customize your installation.

Basic custom configuration

This is an example for a basic custom installation.

```
cat deploy/crds/tridentorchestrator_cr_imagepullsecrets.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullSecrets:
  - thisisasecret
```

Node selectors

This example installs Astra Trident with node selectors.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  controllerPluginNodeSelector:
    nodetype: master
  nodePluginNodeSelector:
    storage: netapp
```

Windows worker nodes

This example installs Astra Trident on a Windows worker node.

```
cat deploy/crds/tridentorchestrator_cr.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  windows: true
```

Managed identities on an AKS cluster

This example installs Astra Trident to enable managed identities on an AKS cluster.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  cloudProvider: "Azure"
```

Cloud identity on an AKS cluster

This example installs Astra Trident for use with a cloud identity on an AKS cluster.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  cloudProvider: "Azure"
  cloudIdentity: 'azure.workload.identity/client-id: xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxxxx'
```

Cloud identity on an EKS cluster

This example installs Astra Trident for use with a cloud identity on an AKS cluster.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  cloudProvider: "AWS"
  cloudIdentity: "'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```